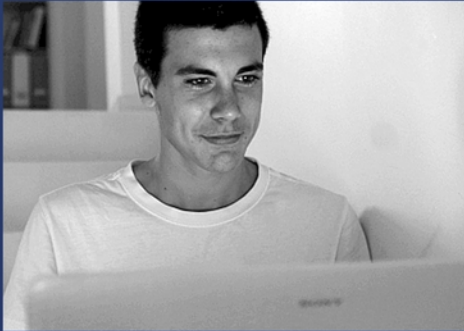


# Hacker HighSchool

## SECURITY AWARENESS FOR TEENS



## LESSON 1 BEING A HACKER



HACKING IS LEARNING  
www.hackerhighschool.org

ISECOM

Creative Commons 3.3 Attribution-Non-Commercial-NoDerivs ISECOM

WWW.ISECOM.ORG - WWW.OSSTMM.ORG - WWW.HACKERHIGHSCHOOL.ORG - WWW.BADPEOPLEPROJECT.ORG - WWW.OSSTMMTRAINING.ORG



## WARNING

The Hacker Highschool Project is a learning tool and as with any learning tool there are dangers. Some lessons, if abused, may result in physical injury. Some additional dangers may also exist where there is not enough research on possible effects of emanations from particular technologies. Students using these lessons should be supervised yet encouraged to learn, try, and do. However ISECOM cannot accept responsibility for how any information herein is abused.

The following lessons and workbooks are open and publicly available under the following terms and conditions of ISECOM:

All works in the Hacker Highschool Project are provided for non-commercial use with elementary school students, junior highschool students, and highschool students whether in a public institution, private institution, or a part of home-schooling. These materials may not be reproduced for sale in any form. The provision of any class, course, training, or camp with these materials for which a fee is charged is expressly forbidden without a license, including college classes, university classes, trade-school classes, summer or computer camps, and similar. To purchase a license, visit the LICENSE section of the HHS web page at <http://www.hackerhighschool.org/licensing.html>.

The Hacker Highschool Project is an open community effort and if you find value in this project, we ask that you support us through the purchase of a license, a donation, or sponsorship.



## Table of Contents

For the Love of Hacking.....	5
Why Be a Hacker?.....	7
Feed Your Head: Handles.....	9
How to Hack.....	10
Two Ways to Get What You Want.....	11
Feed Your Head: Espionage.....	12
Hacking to Take Over Your World.....	12
Game On: A Hacker's First Toy.....	15
The Four Point Process.....	18
The Echo Process.....	18
What to Hack.....	19
Feed Your Head: Classes and Channels.....	20
Feed Your Head: Porosity.....	21
Resources.....	22
Books.....	22
Magazines and Newspapers.....	23
Feed Your Head: Speculation.....	25
Search Engines.....	25
Websites and Web Applications.....	27
Zines.....	27
Blogs.....	28
Forums and Mailing Lists.....	28
Newsgroups.....	29
Wikis.....	29
Social Media.....	30
Chat.....	31
P2P.....	31
Certifications.....	32
Seminars.....	32
Microcomputer Universe: Introduction.....	33
Further Study.....	35



## Contributors

---

Pete Herzog, ISECOM  
Marta Barceló, ISECOM  
Chuck Truett, ISECOM  
Kim Truett, ISECOM  
Marco Ivaldi, ISECOM  
Shaun Coplestone, ISECOM  
Greg Playle, ISECOM  
Jeff Cleveland, ISECOM  
Simone Onofri, ISECOM  
Tom Thomas, ISECOM  
Dzen Hacks

**ISECOM**

## For the Love of Hacking

### Introduction by Pete Herzog

Whatever you may have heard about hackers, the truth is they do something really, really well: discover. Hackers are motivated, resourceful, and creative. They get deeply into how things work, to the point that they know how to take control of them and change them into something else. This lets them re-think even big ideas because they can really dig to the bottom of how things function. Furthermore, they aren't afraid to make the same mistake twice just out of a kind of scientific curiosity, to see if that mistake always has the same results. That's why hackers don't see failure as a mistake or a waste of time because every failure means something and something new to be learned. And these are all traits any society needs in order to make progress.

Many people who have been called hackers, especially by the media, or who have gotten in trouble for "hacking" were not, in fact, hackers.

**A hacker is** a type of hands-on, experimenting scientist, although perhaps sometimes the term "mad scientist" fits better since unlike professional scientists, they dive right in following a feeling rather than a formal hypothesis. That's not necessarily a bad thing. Many interesting things have been designed or invented by people who didn't follow standard conventions of what was known or believed to be true at the time.

The mathematician, *Georg Cantor*, proposed new ideas about infinity and set theory that caused outrage amongst many fellow mathematicians to the point that one called his ideas a "grave disease" infecting mathematics.

*Nikola Tesla* is another person considered a "mad scientist" in his day, but he knew more about how electricity behaved than anyone else. He designed possibly the first brushless motor that ran on AC electricity but is mostly known for the Tesla effect and the Tesla coil.

Then there was *Ignaz Philipp Semmelweis* who figured out that doctors need to wash their hands between treating patients to keep diseases from spreading. He wondered if the diseases following him around between patients were his fault, so he decided to try washing hands between his patient visits and sure enough, the transmissions disappeared. His ideas went against both the scientific conventions of what was known at the time about germs (nothing) as well as the convenience of the doctors who felt it was too much hassle to keep washing their hands.

**What you may think you know about hackers** is that they can break into other computers and take over other people's accounts. They can read your email without you knowing. They can look through your web cam without your permission and can see you and hear you in the supposed privacy of your own home. That's not untrue.

Some hackers see network security as just another challenge, so they tinker with ways to trick or fool the system, but really what they're trying to do is out-think the network installers or designers. They discover as much about the network as they can, where it gets its instructions, the rules it uses, and how it interacts with operating systems, the other systems around it, the users who have access to it and the administrators who manage it. Then they use that to try different ways of getting what they want. This kind of hacking can be greatly beneficial to the world for understanding how to be safer and for building even better technology.

Unfortunately though, sometimes the hacking is done by criminals and what they want is illegal, invasive, and destructive. And those are usually the only hackers you read about



in the news.

**A hacker is not** someone who posts to someone's account when they leave some social media page open or **shoulder-surfs** passwords and then logs into their account later. That's not hacking. A hacker also is not someone who downloads a **script kiddie** tool to break into someone's email. Those aren't hackers; those are just thieves and vandals.

Hacking is research. Have you ever tried something again and again in different ways to get it to do what you wanted? Have you ever opened up a machine or a device to see how it works, research what the components are, and then make adjustments to see what now worked differently? That's hacking. You are hacking whenever you deeply examine how something really works in order to creatively manipulate it into doing what you want.

It just so happens that the way the Internet is designed and the huge number of different applications, systems, devices, and processes it has makes it the most common place to find hackers. You could say it was built by hackers so it's the best playground for hackers. But it's not the only place. You can find great hackers in almost every field and industry and they all have one thing in common: they spend time learning how things work, so they can make them work in a new way. They didn't look at something as the original designers did, but instead saw bigger or better potential for it and hacked it to be something new.

Don't think you can just be a great hacker. Only by doing great hacks with great humility can you be great.

**Hacking itself is not illegal.** At least not any more than throwing a rock is illegal. It all comes down to intent. If you throw a rock and your intent is to injure someone, that's a crime. If your intent is not to hurt someone, but someone does get hurt, that may not be a crime, but you are responsible for your actions and will have to pay restitution. An ISECOM project called the **Hacker Profiling Project** found that the most damage from hacking comes from young, inexperienced hackers damaging other people's property by accident. That's like throwing rocks in the street just for fun but denting cars and smashing windows in the process. Maybe the damage is unintentional, but you can expect to be held responsible and pay for it. So do be careful when hacking around other people's property. Stick to hacking your own stuff.

**It may be illegal to hack something you bought and own.** There are hackers who have been punished for hacking their own devices and computers. There are hackers who hacked programs, music and movies they bought – and were prosecuted for it. In particular, you may not be allowed legally to hack software that you've purchased, even if it's just to check for yourself that it's secure enough to run on your own computer. This is because many of the things that you purchase may come with a contract or **End User License Agreement (EULA)** that says you can't. And you agree to it when you open or install the product, even if you can't read it or even know about it until after you've opened or installed the product. Keep this in mind when you are practicing your hacking skills on the things you purchased in the privacy of your own home.

-Pete Herzog



## Why Be a Hacker?

Consider how scientists mapped the human genome: they used a method developed for decoding passwords. Passwords are usually stored in an encrypted form, so they're hard to steal. Hierarchical shotgun **brute-forcing** is a method of decrypting passwords by **cracking** their encrypted form. It breaks down the encrypted **hash** of the password, solves a few characters at a time, then stitches it all back together. Genome researchers adapted the same technique to map the entire 3.3 billion base pairs of the human genome.

Hacking has shown up in kitchens as chefs use liquid nitrogen as the cooling agent to make perfect ice cream or when they hack food to make tomato fries with potato sauce as the ketchup or just need to make something they don't have the right equipment for....

Chemists have been hacking elements and compounds for centuries. By nature molecules are finicky when it comes to how they behave in different environments (hot weather, cold weather, on mountains, or deep beneath the ocean), so chemists need to deeply understand the properties of the chemicals they have, so they can try to hack together the one they need. Nowhere is this more evident than in the invention of new pharmaceuticals, where hundreds of plants in a region are studied for their chemical properties from roots to fruits, and extracted and combined with others to make new medicines. Then they try again and again, sometimes for years, to get the combinations right and make it do what they want it to do.

Hacking is used in business to understand a market or the buying behavior of certain types of consumers. They research deeply into the forces that drive the area of business they're concerned with, and then they try to change or influence it to make it do what they want. Sometimes they're hacking the product, and sometimes they're hacking you (with advertising and **priming**, something you'll work with in the Social Engineering lesson).

Hacking has also become an increasingly critical part of warfare. Highly skilled soldiers are resourceful and creative in accomplishing their goals, which is exactly what hackers are. Code breakers, intelligence analysts and field officers use what are basically hacking skills to figure out what the enemy has, what they are doing, and how to take advantage of any weaknesses in their equipment. As more nations rely on computers and networks, the use of hacking in cyber attacks and defense has become a valuable part of a nation's armed forces and intelligence operations. National and international security agencies are even going to hacker conventions to recruit hackers!

The real reason to be a hacker is because it's really powerful. You can do some very cool things when you have strong hacking skills. Any deep knowledge gives you great power. If you know how something works to the point that you can take control of it, you have serious power in your hands. Most of all, you have the power to protect yourself and those you care about.



More and more of people's lives are online as relationships form, people find jobs, and money is made on the Internet. Information can be valuable – or threatening – and hackers can protect themselves better than anyone else. They can research what's happening to their data. They can make sure to reveal only what they want and just generally keep themselves safer and more private. That's a huge competitive advantage in school, at work, and in life, because the smallest negative perception will eventually be used against you. Count on it.

**Hack everything but harm none.**



## Feed Your Head: Handles

What's a handle all about?

From the earliest days, hackers had handles. A handle is a made up name that a hacker goes by. This was mostly for fun because hackers in the early days were mostly kids. But later, this game of "playing someone else" became a tool that hackers could use to protect themselves by allowing them to be anonymous.

Early on, hackers chose names like "Erik Bloodaxe," "Mentor" and "Captain Crunch." Some hackers played on the spelling of words like Phiber Optik. Those are the names of some of the most notable, and each is now known publicly.

But hacker handles were meant to be like a super hero's secret identity. No one, or only close trusted friends should know the handle you use online. Why the need for this secrecy? Because sometimes, people don't understand how hackers help, and who we are.

For example, in the past many students who happened to be hackers discovered vulnerabilities in the very networks that they used everyday at school. Wanting to be helpful they reported these findings to school administration. Sometimes the administration was grateful for the information, and maybe even for help with fixing the flaw. But there are several cases when the student was expelled, or even prosecuted by the school system. That reaction comes from ignorance, but it's something that we have to understand and deal with.

Handles also promote free speech. As hackers, we need to say what's true to a world that sometimes doesn't want to hear it. It's a sad fact that some people that hold positions of authority may try to put pressure on a person to bend the truth or hide it when it doesn't agree with their personal views.

The utility of a hacker handle is that it allows us to be ourselves, without the worry of retribution from a well-meaning, but misunderstanding society.

So how does one get started with choosing a handle and establishing an online identity? Well, first things first. Choose a handle that means something to you, and try to make it unique. For example, "Batman" is not very unique, and means something to lots of people. But if you like Windows batch scripts, then BAT.mn is a step closer.

Many handles incorporate parts of **leet speak**. An example is the name used by the group **l0pht**. They chose that name for their group because they met in a rented loft. So you could decide to incorporate that into your chosen handle. If you've become pretty good at playing with Linux's iptables (a host based firewall) you might choose "fyr3w@ll". Just remember that whatever you use, you'll have to type it over and over, so don't go too overboard.

Once you have your handle all picked out, it's time to start establishing your online identity. The first step is the easiest. Go to one of the online providers of free email accounts and see if you can create an account with your new handle. You may find that someone else is using it already. If you do, you can tweak it to be different, change the handle altogether, or simply try the same handle with a different email service.

Once you have your email account established, you need to think about the forums, IRC channels and mailing lists that you want to subscribe to. Forums are a good way for you to establish your new handle. When you are new and inexperienced, you can get help from more seasoned hackers by posting questions to a forum. As you get more experience yourself, you can help others by answering posts that you have experience with. A word of caution here: always research thoroughly before posting questions. If you ask a question that has an obvious answer, you may get terse responses. Always try to be gracious if someone



is rude. No one likes a **flame war**.

You can now hang out in IRC, or sign in to instant messaging clients with your new handle.

You can also start a blog, or web page, listing your new handle as the author/administrator. But remember that this will mean a commitment on your part to keep the content updated; so think hard before you commit to something that could be time consuming.

Having established your online secret identity, or handle, you should enjoy keeping it a secret. Don't ever mention yourself, or anyone that you know in your posts, comments or messages. It's easy to keep things anonymous in most cases. If you do accidentally discover that misconfigured webpage, server or computer, then you can send an email from your handle's account, and not worry about the fact that the person that owns it is your principal, club leader, or dad.

You may find, as many hackers do, that as you get older and respected in computer security circles, that you don't need your handle anymore. That you can use your real name and be taken seriously. If you work hard, become proficient and well regarded then you don't need to hide behind anonymity to be taken seriously any longer. And that day can be just as liberating as it was the day that you chose your handle.

-Dzen Hacks

## Exercise

- 1.1 You may or may not choose to use a hacker handle, but you should think about it. As Dzen notes above, you may have a legitimate reason to maintain an alternate identity. So what would your handle be?

How about something dark and mysterious: Phantom Blade or Lightning Fury or Dark Summoner? Can you really live up to this persona? Do you even want to?

Maybe something light and non-threatening? Sk8ter? PonyGirl? Would you be worried about being taken seriously?

Okay, maybe a friendly name like FluffyBunny? Unfortunately, you might be giving the wrong impression about your interests.

The ideal handle is one that never gives you away, but says something about your interests and personality.

Choose a handle for yourself.

## How to Hack

Telling you how to hack is like explaining to you how to do a backward flip on a balance beam: no matter how detailed the explanation is you won't be able to do it on your own the first time. You need to develop the skills, feeling, and intuition through practice or else you'll fall flat on your face. But there are some things we can tell you to help you along and encourage you to keep practicing.

First, you should know some little secrets about how hacking actually works. We're going to take these from the **OSSTMM** ([www.osstmm.org](http://www.osstmm.org)). Hackers sound it out and pronounce it "aw-stem." The OSSTMM is the **Open Source Security Testing Methodology Manual**, and while it may read like DVD player setup instructions, it's the main document that many hacking professionals use to plan and execute their attacks and defenses. Deep in that manual are some real gems that will open your eyes.



## Two Ways to Get What You Want

For example, you should know that there are really only two ways to take anything: you take it or you have someone else take it and give it to you. That means all the taking in the world requires **interactions** between the person and the thing. Obvious, right? But think about it. That means that all protection mechanisms have to try to stop someone from interacting with the thing they are protecting. Unless you lock everything in a huge safe, you can't stop all interaction. Stores need to put stuff on shelves that shoppers can touch. Businesses need to send information through email clients that attach to mail servers and send messages to other mail servers.

All of these are interactions. Some of these interactions are between people and things that are familiar with each other, so we call those interactions **Trusts**. When the interactions happen between unknown people or systems we call these interactions **Accesses**. You can either use an access to take what you want yourself, or you can trick someone who has a trust with the target to take what you want for you and give it to you. If you think about that for a moment, it means that security means protecting something from both those it doesn't know and those it knows and trusts.

### Exercises

- 1.2 What kind of interaction is using a search engine? Think carefully: is anyone giving Access? Is anyone giving Trust?
- 1.3 Give a simple example of using Access and Trust to take a bicycle locked to a bike rack.
- 1.4 Give a simple example of how you can use Access and Trust to log into another person's web-mail account.



## Feed Your Head: Espionage

When hacking is used against a foreign government to commit criminal acts of breaking and entering, trespassing, theft, and destruction to get the edge in political or military information it's called **espionage**. But when the hacking is done from a foreign business against another business in a different country to get an edge in business, it's called **economic espionage**.

When hacking is used to get private and personal information on individual people to embarrass them publicly it's called **DoXing**. If public information is dug out to target a person or company for an attack, but no criminal acts are made to get the information, it's referred to as **document grinding** or **OSInt (Open Source Intelligence)**.

When hacking is used to understand a company network, systems, applications, and devices as the target of an attack without actually intruding or trespassing into the systems it's known as **network surveying**.

When hacking is used to deeply understand a competitor without breaking any laws (although what they do may be considered just plain mean or rude) it's called **competitive intelligence**.

You're probably dying to hear now what kind of mean and rude things they do that are still legal. Consider the example of inflicting stress and worry on someone to get information from them. As long as you don't kill them, telling them lies is still legal (although there are laws against causing panic in public places like yelling "Fire!" in a crowded movie theater when there is none).

Say the hacker wants to know where a company is planning to set up their new factory. They use document grinding to find out which people are in the position to make that decision. Then the hacker calls their offices to find out which cities they've been to and perhaps which factories they've visited. But of course that's private company information and nobody is just going to tell them that without raising red flags. So the hacker needs to trick the information from them. It's not hard to imagine the process.

Hacker: Hi, I'm Dr. Jones, and I'm calling from the school about your daughter Nancy.

Target: Oh really? What has she done now?

Hacker: Well, she's got a persistent nosebleed we can't get to stop. I'd like to ask you about any chemicals she's been exposed to, manufacturing chemicals and such. These symptoms are rare except in people exposed to these chemicals. Can you tell me anything?

Target: (spills their guts)

This is not really illegal in most places but it causes unneeded stress. Not to mention it's just mean to make a parent worry like that.

## Hacking to Take Over Your World

Hacking isn't just about interactions. You know that. Some people say politics is about interactions. Maybe. But you probably thought hacking is about breaking security. Sometimes it is. What it's really about is taking control of something or changing it as well. Understanding interactions and what they mean in the real world, using the basic terms we've discussed, is useful when you're trying to infiltrate, discover, or even invent. Why would you do this? To have the freedom to make something you own do what you want. And to keep others from changing something you own in the name of what some people might call security (but we're not those people).



Sometimes you buy something and the company you bought it from will attempt to forcefully or sneakily make sure you can't customize it or change it beyond their rules. And you can agree to that, as long as you accept the fact that if you break it then you can't expect them to fix it or replace it. So hacking something you own does more than make it yours, it makes it irrevocably and undeniably yours. As scary as that may sound to some, it certainly has its advantages. Especially if you want to keep others out of your stuff.

For many, many people (we could put many more "manys" here to get the point across that we really mean "way way too many"), security is about putting a product in place, whether that's a lock or an alarm or a firewall or anything that theoretically keeps them secure. But sometimes those products don't work as well they should, or come with their own problems that just increase your **Attack Surface**, when a security product should be shrinking it. (The Attack Surface is all the ways, all the interactions, that allow for something or someone to be attacked.) And good luck getting that product improved in a mass-marketing, pay-as-you-go, crowd-sourcing, "you bought it as-is and that's what you have to live with" kind of world. That's why you hack your security. You need to analyze the product and figure out where it fails and how to change it so it works better. Then you might have to hack it some more to keep that company you bought it from, from changing it back to the default!

So when you think of hacking in terms of breaking security, remember that's just one area that hacking is useful for, because without being able to do that, you may have to give up some freedom or some privacy that you don't want to give up. (And yes we get it that you may not care right now about certain things you do or say or post, but the Internet has a long memory and it's getting better and better at helping others recall those memories of you. What goes on the net stays on the net. So consider this for the future you even if the you of today doesn't care.)

Now that you get the idea about interactions, let's get into them into more detail. You know the basic interactions as Access and Trust but have you heard of **Visibility**? That's the third type of interaction. It's just as powerful as the other two. In police language, it's simplified as *opportunity* but in hacking it's more about knowing if there is something to interact with or not. This interaction brings along a whole lot of new security techniques like deception, illusion, and camouflage, as well as all-new hacking techniques for avoiding and getting around security measures like deception, illusion, and camouflage!

When famous bank robber Jesse James was asked why he robbed banks, he said it's because that's where the money is. What he meant is that through Visibility he knew that the banks had money where other things he could rob might not. Banks have Visibility: people know what assets they hold. But not everything has Visibility. As a matter of fact Privacy is the opposite of Visibility and it's a powerful way to avoid being a target. Whether on dangerous streets, in the jungle, or on the Internet, keeping a low **Exposure** and avoiding Visibility is a way to keep from getting attacked in the first place.

## Exercises

1.5 The Internet is so popular for creating myths and perpetuating false stories that it's hard to know what's real information and what is just a hoax. So if you want to learn to be a good hacker, get in the habit of checking your facts and learning the truth about things. That's why you're going to search and find out if Jesse James really did say that. And don't go easy on the answer by just going to the first web page you find, dig a little.

Now that you're getting used to looking things up, find the truth about these common things:

1.6 In the Inuit language where the word igloo comes from, what does it really mean? What kind of interactions did you use now to find out?



- 1.7 Many parents are quick to point out that sugar makes little kids hyper-active but does it really? What interactions are really occurring in their little bellies when children eat a lot of candy or sugary foods that make them act silly and hyper?
- 1.8 You might have heard that sugar causes cavities (caries) in your teeth but what is the real interaction that takes place – what really causes it? Is it sugar or not? Bonus points if you can say what brushing is as an interaction to fight the real cause and find the name of at least one of the chemicals that addresses the root of the problem (\*hint: fluoride is wrong\*).



## Game On: A Hacker's First Toy

It was a cool afternoon at Jace's grandparents' apartment. Rain slammed down onto the outside world but didn't bother anyone in the building. The ten-year-old always enjoyed spending time with her grandfather because he enjoyed spending time with her. Her mother had woven her fine dark hair into a single ponytail with a pink bow at the end. Three hours into playing and the pink bow was sideways and barely holding on to a few strands of hair. Jace was busy finding things to play with around the small home when she came across one of her favorite toys.

This particular toy was a small handmade box slimmer than a shoe box made of painted wood. There was a crank handle on one end. On the top of the box there was a small wooden wall with a tiny hole, a sleeping cat and a sleeping wooden old lady sitting in a chair. When Jace turned the box handle a mouse appeared from a hole in the wall and moved towards the sleeping cat. Once the mouse arrived at the cat, the cat leaped into the air and landed on top the sleeping ladies head. The surprised wooden lady would open her eyes and kick out her legs as the mouse watched in amusement.

Jace would wind the handle backwards to reset the automata machine and as soon as the mouse was hidden back in the wall, turn the handle forward to create the whole comical scene again. The ten-year-old would usually do this a few times before getting bored and turn her attention to something else. However, this time Grandpa was watching Jace and paying close attention to her expression. Jace didn't seem satisfied with just watching the mouse spook the cat and watching the cat land on the little woman's head. *She wonders how it works*, he thought, and decided to give her a surprise.

Crouching next to Jace in his gray work pants, he pushed his glasses back. He said, "I see you like the mouse causing all the trouble here. It is because the mouse is the smallest of the creatures in the cast or because the mouse is sneaky enough to surprise the cat and the woman in the chair?"

Jace didn't look behind her because she knew the smell of her grandfather and he always asked questions like this. His closeness to her made her feel comfortable but she didn't know the answer to his question. She had her own question but didn't know what that question was or how to ask it. She continued to turn the handle in both directions to see how each piece was moving. She answered into the air, "I don't understand how all the parts move. I like this but I wish I could see what's happening inside, what makes everything work."

Grandpa clapped his hands in delight. This was the moment he had been waiting years for, the moment Jace pushed her curiosity into his realm. Without getting up, Grandpa pivoted around on the orange carpet to sit next to the child. He gently pulled the toy away from her small hands and asked, "What if you can't open the box to look inside? What will you do?"

Jace studied the wooded container and simply said, "I'll break it open, then."

Grandpa's eyebrows knitted and he stared at Jace, who went wide-eyed. "This toy doesn't belong to you. You can play with it but you can't just go around breaking things that don't belong to you. If you want to see how it works, you need to find another way to do it. Criminals break things on purpose. You, my dear, are not a criminal. You are curious about things so I'm going to show you how to be a hacker."

Jace knew what a criminal was but had never heard of a hacker before. She squared her shoulders and sat up straight since this was turning into one of Grandpa's teaching talks. These were always fun. She asked her grandfather, "What's a hacker?"

"A hacker is someone who wants to learn how machines work. They read about them, build them, play with them. Hacking ... at least when I was growing up, hacking didn't mean anything illegal. There are people who call themselves hackers but they're really criminals because they steal from other people, destroy things that don't belong to them



and hurt other people. Real hackers don't."

Child eyes looking into Grandpa's, Jace said, "Like when I said I would break the box open, that's something a bad person would do." She nodded: no remorse for her previous statement, just taking in the difference between one type of action versus another. The gray haired gentleman moved the box in his hands until he felt a spot on the bottom of the toy. Using his thumbnail, he carefully slide the side panel upwards to reveal in the inside of the box. Turning the box over, he did the same thing to open the other side panel.

The ten-year-old stared into the inner workings of this machine. Grandpa smiled, he watching her tilting the box and looking in from every angle. Inside were wooden gears, cams, a worm drive and small fishing rod weights: no longer a toy but a spectacular invention of moving parts. It reminded Jace of Grandpa's old pocket watch, with all the delicate gears and springs you could see in the back.

Grandpa said, "Go on, touch it. Move the pieces to see what happens. Watch how each piece makes other parts move. Check out the cat and mouse. What makes them move?"

Jace was baffled by the puzzle of motion in her hands. She reached into the exposed box and touched the pin wheel. A pin fell off the wheel and landed somewhere inside the box. When Jace tried to turn the crank, the mouse moved but the cat didn't. She looked up at Grandpa, stricken: the toy was broken!

Grandpa looked at Jace and softly said, "No, no, no. It isn't broken. We just have to put the pin back on the wheel. Well, we first have to find the pin. Shake the box until you hear something loose rattle around."

The child gently jiggled the box, then titled it up and down until the pin fell on to the carpet in front of her. Grandpa continued, "See, if we do accidentally break something it is our duty as hackers to fix it. If we come across something that's already broken, we either let the owner know it's broken or we fix it and tell the owner how we fixed it. We have to be responsible citizens, as hackers and as good people."

Jace handed the pin to her grandfather who handed the pin right back to Jace. He said, "You broke it, you fix it. I'll show you how."

He picked up the machine and took it over to the kitchen table where the light was better. Jace followed him and pulled up a chair alongside her mentor. Grandpa pulled a pencil out of his breast pocket and laid the box on its side so they could see the insides.

"See the crank handle on the outside of the box," he began. "When you turn that handle in one direction the mouse comes out of the wall and moves towards the cat. The mouse is moved because below the box surface, the mouse sits on a hidden track that is kind of like a corkscrew. The handle you turn causes the corkscrew track, what is called a worm drive, to rotate the mouse forward and also turn a pin wheel at the end of the track. The pin wheel has four pins on it."

Grandpa pointed his pencil at the pinwheel that was missing one pin. Jace opened up her small hand to reveal the lone pin. He continued, "You turn the crank four times and each time the mouse moves forward and the pin wheel makes one turn of a pin. Once the pinwheel moves four rotations, there is a cam connected to the fourth pin. It's called a drop cam even though it's shaped like an egg or a pear. The fourth rotation causes the drop cam to move to its highest point, which is connected to the cat."

Jace was looking at each mechanism but wasn't sure she understood how they worked together. Her grandfather moved his pencil to the cat, which rested on a wire hook. "See, when the drop cam hits its highest point on the fourth rotation, the wire is released and the cat jumps into the air on a curved path. The cat lands on the little woman's head. Now, if you look closely, you'll see the sitting lady has two buttons on top of her head. When the cat lands on her head, the cat pushed down on the two buttons in her hair. One button cause the woman's eyes to open and the other button causes her legs to pop forward in surprise."





Grandpa started the box at the first stage to show how a single turn on the crank made the worm drive push the mouse towards the cat and also move the pin wheel one rotation. This one rotation caused the drop cam to turn a quarter of a rotation. A second turn of the crank caused the mouse to move closer to the cat and the pin wheel rotated one more time, which made the drop cam complete a half rotation. The third crank turn did the same thing as the previous two turns. It wasn't until the fourth crank turn that the mouse reached the cat, which turned the pin wheel to the fourth position. This also caused the drop cam to move to its highest point and released the cat. The cat flew on the wire and landed on the little woman's head as she sat sleeping in her chair.

The cat landed on the woman's head, the buttons pushed, that opened her eyes and made her legs swing out in surprise. Turning the crank in the opposite direction reset all the movements. Jace watched intently as she turned the handle slowly in one direction and then the other direction. Grandpa rested back in his kitchen chair and watched as Jace's mind absorbed all this mechanical movement information.

With her teacher looking on, Jace removed a pin or a part to see how that changed the toy's mechanics. Each change either had disastrous results or no effect at all. Several hours into the toy tinkering Jace began to ask a series of questions.

"Why not use a magnet instead?"

"How come the cat hangs on a wire when a spring would work better?"

"What happens if I turn the crank five or six times?"

"Why is a worm drive thing here, when you could use a curved track instead?"

"Does this work upside down?"

Grandpa interrupted Jace by raising his index finger to his lips. He looked with wonder in his eyes and repeated, "Why is a worm drive used when a curved track could be used instead? Is that what you just said?"

Jace nodded as she explored the pinwheel closer. Grandpa grabbed a piece of paper and started to draw on it. When he was done drawing a few drafts he slid the paper over to his granddaughter and asked, "Is that what you mean by a curved track?"

Jace took a quick look and said, "Yeah but you need to add a small piece to the mouse to keep it on the track. It would be smoother that way and easier to build."

Grandpa's mouth worked but he said nothing.

Jace added, "Plus you can get rid of the pin wheel by cutting a slot at the end of the track. One less thing to go wrong. You won't need to turn the crank four times, either. When the mouse gets to the end of the track, the slot will trigger the wire to release. You can get rid of the drop cam too. Less parts to build. And why have two buttons on the top of the woman's head. Ya only need one to open her eyes and make her legs pop out."

Grandpa's pencil hit paper and scribbled fast as he thought, *I just got schooled.*

## Game Over



## The Four Point Process

When you take the three types of interactions together, you have **Porosity**, the basis of an Attack Surface. And like the word implies, it's the pores or "holes" in any defenses you have to have in order for any necessary interactions to take place (as well as any unknown or unnecessary interactions taking place). For instance, a store still needs to put products on the shelves so people can touch them, put them in a cart and buy them. These are the interactions they need to sell things. But they might not be aware of the employees who are sneaking stuff out of the loading dock, which is an interaction that they don't want.

Porosity is something you need to know about to protect yourself or attack some target. But it's not enough to analyze something to hack it. To do that you need to know something deeper about the three types of interactions you just learned. This is another little secret from the OSSTMM and it's called the **Four Point Process (FPP)**. It outlines four ways these interactions are used to analyze something as deeply as possible, and by analyze we mean to mess with it so we can watch it and see what happens.

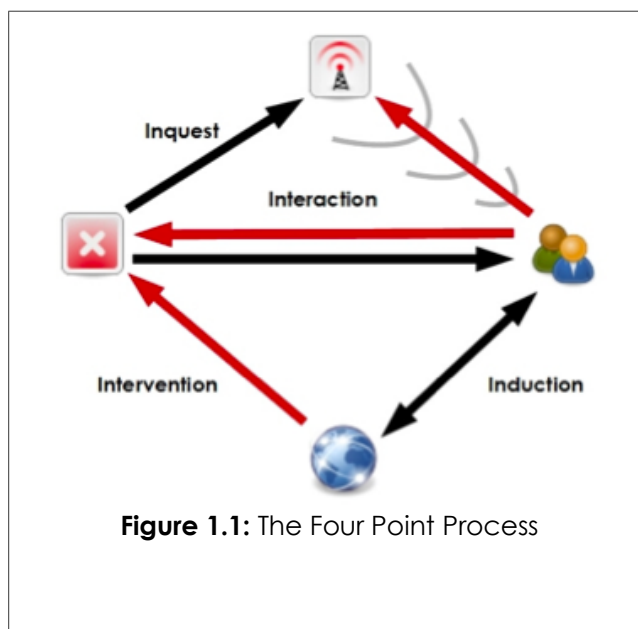
## The Echo Process

We grow up discovering things and learning things by interacting with them directly. Little kids poke the dried-up squirrel with a stick to see if it's dead. This is called the **echo process**. It's the most basic and immature form of analysis. It's like yelling into a cave and listening for the response. The echo process requires throwing different types of Access interactions at a target and then monitoring its reactions to figure out what ways you can interact with it. The echo process is a cause-and-effect type of verification.

This is an odd way to test something, because although it makes for a very fast test, it also isn't very accurate. For instance, when using the echo process in testing security, a target that does not respond is considered secure. That's the same as not having Visibility. But we also know that just because something is non-responsive to a particular type of interaction that doesn't mean it's "secure." If this were true then opossums would never get killed by other animals when they played dead and everyone would be safe from bear attacks just by passing out in fear. But it's just not true. Avoiding Visibility might help you survive some types of interactions but certainly not all.

Unfortunately, the majority of ways people investigate things in their everyday life is through the echo process alone. There is so much information lost in this kind of one-dimensional analysis that we should be thankful the health care industry has evolved past the "Does it hurt if I do this?" method of diagnosis. If hospitals only used the echo process to determine the health of a person they would rarely truly help people. On the bright side the waiting room times would be very short. That's why some doctors, most scientists, and especially hackers use the Four Point Process to make sure they don't miss anything.

The Four Point Process has you look at interactions in the following ways:



1. **Induction:** What can we tell about the target from its environment? How does it behave in that environment? If the target is not influenced by its environment, that's interesting too.
2. **Inquest:** What signals (emanations) does the target give off? Investigate any tracks or indicators of those emanations. A system or process generally leaves a signature of interactions with its environment.
3. **Interaction:** What happens when you poke it? This point calls for echo tests, including expected and unexpected interactions with the target, to trigger responses.
4. **Intervention:** How far will it bend before it breaks? Intervene with the resources the target needs, like electricity, or meddle with its interactions with other systems to understand the extremes under which it can continue operating.

**Back to our hospital example...**the four stages of the FPP would look like this:

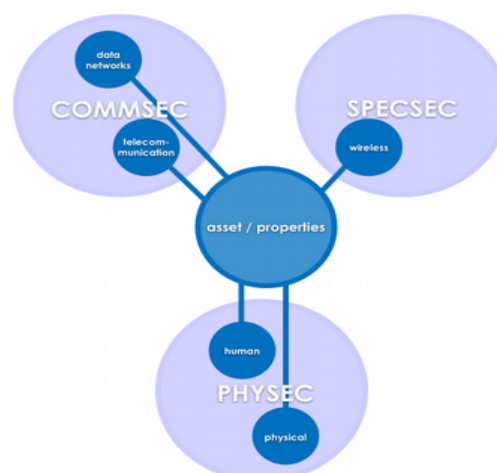
1. The **interaction** function is the echo process where the doctors poke the patients, talk to them, and test their reflexes on the elbows and knees and use other tools of the "Does it hurt if I do this?" method.
2. The **inquest** is reading **emanations** from the patient like pulse, blood pressure, and brain waves.
3. The **intervention** is changing or stressing the patient's homeostasis, behavior, routine, or comfort level to see what happens.
4. And finally **induction**, which is examining the environment, the places where the person visited before they got ill and how they may have affected the patient, such as what they may have touched, ingested, or breathed.

### Exercise

- 1.9 As you can see, the Four Point Process lets you more deeply investigate interactions. Now you can try it. Explain how you would use the Four Point Process to know if a clock is working – and then if it's working correctly by keeping the right time.

## What to Hack

When you're hacking anything you need to set up some ground rules. You need the language and the concepts to know what you're actually hacking. The **Scope** is a word we use to describe the total possible operating environment, which is every interaction that the thing you want to hack has.



**Figure 1.2:** Scope

## Feed Your Head: Classes and Channels

In professional terminology (also useful to hackers), the Scope is made up of three **Classes** that subdivide to five **Channels**:

Class	Channel
Physical Security (PHYSSEC)	Human
	Physical
Spectrum Security (SPECSEC)	Wireless
Communications Security (COMSEC)	Telecommunications
	Data Networks

**Classes** are not something you have to be too worried about but they are the official labels used currently in the security industry, government, and the military. Classes define an area of study, investigation, or operation. So if you're looking up more information on any topic, it's really good to know what the pros call it.

**Channels** are the common terms for the ways you interact with assets. It's not uncommon to hack a gadget by using the Four Point Process over each Channel. Yes it seems like a lot of work, but think of how thrilling it is when you discover a way to make it work that's not listed in any manual, or better yet not even known to the manufacturer!

An **Asset** can be anything that has value to the owner. It can be physical property like gold, people, blueprints, laptops, the typical 900 MHz frequency phone signal, and money; or intellectual property such as personnel data, a relationship, a brand, business processes, passwords, and something said over the 900 MHz phone signal.

**Dependencies** are the things beyond the asset owner's ability to provide independently. Not many computer owners generate their own electricity, for instance. Even if it's not likely someone will cut off your electricity, it is still within your scope.

The goal of security is **Separation** between an asset as well as its dependencies, and any threat to them.

We say **security is a function of separation**. There are four ways we can create this separation:

- Move the asset to create a barrier between it and the threats.
- Change the threat to a harmless state.
- Destroy the threat.
- Destroy the asset. (Not recommended!)

When we're hacking we look for places where interactions with the target are possible, and where they are not. Think of doors into a building. Some are necessary for workers;



others are necessary for customers. Some may be needed to escape fire. And some may not be necessary at all.

Every door, though, is a point of interaction, one that aids both necessary operations and unwanted ones like theft. When we come on the scene as hackers, we don't start out knowing the reasons for all these interactive points, so we analyze them with the Four Point Process.

Consider the example of a guy who wants to be totally safe from lightning. The only way to do this (while still on Earth) is to get into a mountain where it is completely impossible for lightning to get inside through all that dirt and rock. Assuming he never needs to go out again we can say his security is absolutely 100%. But if we start drilling holes in the mountain the lightning has one more point of access with every hole, and the porosity increases. The OSSTMM differentiates between being **Safe** from lightning and being **Secure** from it. The simple fact is that the more porosity there is, the more likely a hacker can make changes and control what they want.

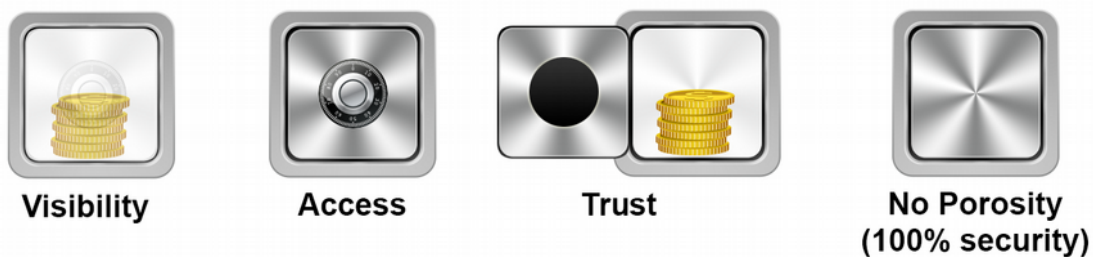


Figure 1.3: Porosity

**Feed Your Head: Porosity**

Here are some examples that describe how pores can be located, classified, and determined in the hacking process.

Term	Definition
Visibility	When police investigate a crime, they look for <i>means, motive</i> and <i>opportunity</i> . If an asset is visible, it can be attacked, but if it's not visible, it can't be targeted – though it could be discovered. Some security professionals like to say that <b>obfuscation</b> is poor security because it doesn't protect anything, it only hides it. But that's not a bad thing especially since you don't always need a permanent security answer. To that effect the <b>OSSTMM</b> offers this little gem: " <i>Security doesn't have to last forever, just longer than anything else that might notice it's gone.</i> "
Access	Access is the number of different places where interactions can occur to the outside of the scope. For a building this could be doors to the street or windows and for a server on the Internet it could be the number of open network ports or services available on that computer.
Trust	Trust is when one entity accepts free interaction with another entity within the scope. It's why you don't ask your mother for ID when she comes to hug you. It's also why you don't suspect she's poisoned your food. You learn to trust the things inside your scope. Then one day if she gets taken over by



an alien race and replaced (*a lá Invasion of the Body Snatchers*) and does poison your food you'll eat it unsuspectingly. Thus trust is both a security hole and a common replacement for authentication, the way we can validate if someone is who we think they are. Trust is a strange topic because it's such a human thing to do and pretty valuable in society. Without trust, we'd never be able to interact freely. But because of trust, we are easily tricked, fooled, robbed, and lied to. OSSTMM research on trust shows that there are 10 reasons to trust someone called **Trust Properties** and if all ten reasons are satisfied then we can trust safely without any worry. But that same research shows that most people need only one trust reason to be satisfied and the really paranoid or cynical are okay with just three reasons to trust.

## Resources

Effective research, learning and critical thinking are key skills for a hacker. Hacking, in reality, is a creative process that is based more on lifestyle than lesson. We can't teach you everything that you need to know, but we can help you recognize what you need to learn. Because science advances so quickly, what we teach today may not be relevant tomorrow. It is much better for you to embrace hacker learning habits, which are probably the most vital part of hacking, and which will separate you from the **script kiddie** (a hacker word that means a person who uses tools without really knowing how or why they work).

If you run into a word or concept you don't understand in this lesson, it's essential you look it up. Ignoring new words will only make it difficult for you to understand concepts in the coming lessons. You'll be asked to investigate a topic and then expected to use the information that you find to complete the exercises in that lesson – but those lessons won't explain to you how to do this research. So be sure to spend as much time as you need learning to use the various resources available to you.

## Books

You might be surprised that we don't point you straight at the Internet, but books are a great way to learn the foundation and factual science of everything you want to explore. Want to know something about computer science, like the hardware details of your PC? Nothing will help you more than reading a current book on the subject. The main problem with books for computers is that they become out-of-date very quickly. The secret is to learn to see the fundamental structure underneath the thin skin of details. MS-DOS and Windows are clearly different, but both are based on principles of Boolean logic that have driven computers since Ada, Countess of Lovelace, wrote the first computer programs in the nineteenth century. Security and privacy concerns may have changed in the last 2,500 years, but **The Art of War** by Sun Tzu covers fundamental principles that still apply today. (By the way, there is no faster way to look like a **n00b** than by quoting Sun Tzu. Some things you should know how to apply but not say. And quoting The Art of War proves that you didn't really read The Art of War because Sun Tzu even says to keep your real knowledge a secret.)

Even though information found in books may not be as up to date as information that comes from other sources, the information you find in books is more likely to be better written than most other sources. Sometimes they are more accurate too. A writer who spends a year writing a book is more likely to check facts than someone who is updating a blog six times a day. (See the Sections on Zines and Blogs for more information.)

But remember – accurate does not mean unbiased. The sources of the author's information themselves might be biased. "History books are written by the winners" (look up that quote), and the same holds true when the politics and social norms of the time may prevent certain information from being published. This commonly happens with



school textbooks that are chosen through a political process and contain only the information considered socially acceptable to know. Don't think you've found a golden truth just because you read it in a book. The truth is that anyone can write a book and any book can contain anyone's version of the truth.

Don't look at a book and give up before you even start just because it's so big. Nobody reads most of these massive books that you see sitting around from cover to cover. **Think of them as prehistoric web pages.** Open one up to a random page and begin to read. If you don't understand something, go backward and look for the explanation (or skip forward to something that does make sense). Jump through the book, backwards and forwards, just as you would bounce from link to link in a web page. This type of non-linear research is often much more interesting and satisfying for hackers, since it's about satisfying your curiosity more than it is about reading.

Finally, one thing that book readers gain as a valuable skill is the ability to write well. This is a huge advantage whenever you are trying to understand and participate in a new field. It also makes what you have to say more credible to other readers, especially those who are in positions of authority.

## Magazines and Newspapers

Magazines and newspapers are highly useful for providing concise, timely information. Both types of publications can be very short on details, though. Also be aware that every newspaper or magazine has its own audience and its own agenda or theme, regardless of any claims of being "fair and unbiased." Know the theme of the publication: a Linux magazine isn't necessarily a good source of information about Microsoft Windows, because Windows is a conflicting theme (a competing operating system), and frankly a Linux magazine's readers want to read about the superiority of Linux. Many of the specialty magazines employ **cherry picking**, the technique of highlighting only the positive aspects of something fitting the magazine's theme or highlighting only the negative aspects of the things that don't.

Be aware of a publication's possible biases. That's where they give you opinions instead of facts or leave out facts from a story to fit their own opinions or so you can't form your own opinion. Consider the source! Even "neutral" appearing periodicals can be full of biases and speculation, a nice way of saying "educated guesses" but is more often just "guesses" on the part of the journalist.

There's a huge movement in the medical field to have all medical and pharmaceutical trials published (or at least all publicly-funded trials) even if they failed so that doctors can make even more informed choices about what medicine and procedures to apply. While current medical journals may be publishing "facts" from research trials, the details and circumstances behind those facts are still cloudy. That's really important when you deal with subjects that rely on having root causes. Causation requires that the cause precedes and is the reason for the effect.

Other tricks used by periodicals (both inadvertently and purposely) are **anecdotal evidence**, which is opinions from people published as evidence regardless of whether they are experts or not; **authoritative evidence**, in which industry employees represented as experts give their opinions, or people who are authorities in one area offer their opinion in another area in which they have no expertise; and finally, **speculation**, dressing up something as true because "everyone" believes it is true, though there's no actual attribution to anyone specific.



The best way to deal with the issues of accuracy and agenda is to be well and widely read. If you read about an interesting issue in a magazine, look into it further. Take one side of the issue, and look for confirmations; then take the other side, and look for rebuttals. Some cultures do this by default. It's part of their social habits to seek other sides to the story. That's a really powerful cultural trait, especially if you're trying to assure a successful democracy.

### Exercises

- 1.10 Search the Internet for three online magazines on the subject of hacking. How did you find these magazines?
- 1.11 Are all three magazines specifically about computer hacking? What else do they offer that might be helpful in other fields or other businesses?





## Feed Your Head: Speculation

The following paragraph is from a newspaper article about a robbery. Can you find the Speculation? Note the areas that you suspect:

The Lake Meadow Bank and Mortgage Lender was robbed Tuesday afternoon when masked gunmen walked in just moments before closing and held the employees hostage for an hour before what appears to be making their get-away in a late model SUV. None of the hostages were reportedly injured.

No one could identify the gunmen which leads the police to believe that this may have been a professional job as moments after the robbery, the car was spotted behind the bank and heading south towards the dense woods of the Bluegreen Mountains. The police are now likely to be looking for experienced robbers who may have prison records and who also have relationships to people living in that area.

With an average of 57 thefts at banks being reported every day within this country and the Bluegreen county population reportedly to reach a population of over 50,000 by next year, this could be the start of a rash of bank robberies from that region. "This looks like the start of a trend." said the police commissioner, Smith.

As we become more desensitized to speculation and remain functionally ignorant of the bias in statistical data and results, the future of all our news might as well just come from a single journalist speculating on news stories as they happen. In the short example above, there is only one real fact - a bank had been robbed on Tuesday afternoon. Now for the sake of obviousness, this is what it would look like if we changed all the speculation to make it more ridiculous:

The Righteous Bank and Mortgage Lender was robbed Tuesday afternoon when what appears to be masked chickens walked in just moments before closing which means they could have held the employees hostage for as long as a decade before what appears to be making their get-away in a hot air balloon shaped like a chicken coop. None of the hostages were reportedly covered in feathers.

No one could identify the chickens which leads the police to believe that they may have had a professional disguise artist among them in addition to an accomplished balloonist as moments after the robbery, a hot air balloon was spotted above the bank and flying south towards the tundra of Antarctica. The police are now likely to be looking for accomplished make-up artists who also have ties to balloon hobbyists.

With an average of 57 thefts at banks being reported every day within this country and the ballooning industry reporting sales to reach \$47 gazillion by some future date, this could be the start of a rash of bank robberies using hot air balloons. "This looks like the start of a trend." said the police commissioner, Gordon.

With the overwhelming use of speculation and statistics across all industries it is no wonder that it has entered the security industry with such force. The commonly used term in this industry is **FUD** which is an acronym for **Fear, Uncertainty, and Doubt**. It is how speculation and subjective risk analysis are used in security to gain attention for one's interests and sell security solutions. Unfortunately it plays very well with the primitive paranoia in the human psyche and a growing numbness to speculation. This has led to inappropriate security solutions, inappropriately applied security, reactive security controls, and false confidence in authorities. There is clearly a failure of critical thinking skills within the population and this is being exploited both by the commercial sector and the criminals.

## Search Engines

Google is a well known search engine but it's not the only search engine. Bing is very good with searches in the form of simple questions and Yahoo is good for doing thorough research. Be very aware that all these web services want to know everything they can about you, and probably know more than they should. They will record your searches and which websites you visit after you search.



There are engines like AltaVista and DuckDuckGo.com that may give you some – or a lot – of anonymity, which might be a good thing when you are looking into dark corners.

Websites are searchable while they're online and usually long after that. Typically they're preserved in the form of **cached pages**. An Internet cache is an online record of past versions of websites, or even of websites that have gone dark. Search engines and archive sites keep this information indefinitely, which in Internet terms is "forever." That's a valuable thing to remember before you ever put anything on the Internet: it isn't going away. Ever. You may have to look for a link to the cached copy of a page. Google, for instance, used to put a simple link labeled "Cache" alongside the regular link to a result. That has changed to a fly-out menu on the right, and may have changed again by the time you read this.

Besides the search engines, there are also useful public caches at places like the **Internet Archive at <http://www.archive.org>**. You can find cached versions of whole websites from over the years, which can be very useful for finding information that has "disappeared."

One final note on websites: don't assume you can trust a website just because it shows up in a search engine. Many hacker attacks and viruses are spread just by visiting a website or downloading innocent-looking programs, screen savers or any shared files. You can safeguard yourself by not downloading programs from untrusted websites, and by making sure your browser runs in a **sandbox**. But this may not be enough. A browser is a window to the Internet and like any window, bad stuff can float in just because it's open. Sometimes you won't even know until it's too late.

## Exercises

- 1.12 There are many search engines out there. Some are good for getting to the **Invisible Web**, areas of the Internet that are hard for most search engines to dig through, like certain private databases. A good researcher knows how to use them all. Some websites specialize in tracking search engines. So find five search engines you haven't used or maybe even heard of before.
- 1.13 There are also search engines that search other search engines. These are called **meta search engines**. Find one of these meta search engines.
- 1.14 Search for "security and hacking" (including the quotation marks) and note the top three answers. How do the results differ when you DON'T use quotes?
- 1.15 It is very different to search for a topic than it is to search for a word or phrase. In the previous exercise, you searched for a phrase. Now you will search for an idea.

To do this, **think of phrases that might be on the page you're looking for**. If you want the search engine to give you a list of online magazines about hacking, you won't get far by searching for "a list of online magazines about hacking." Not many web pages will contain that phrase! You'll get some hits but not many.

Instead, you need to think, "If I was setting up a hacking magazine, what would a typical sentence in that magazine look like?" Put the following words and phrases into a search engine and find out which provides the best results for your search:

1. my list of favorite magazines on hacking
2. list of professional hacking magazines
3. resources for hackers
4. hacking magazine
5. magazines hacking security list resources



- 1.16 Find the oldest website from Mozilla in the Internet Archive. To do this you need to search on "www.mozilla.org" at the <http://www.archive.org> website.
- 1.17 Now to put it all together, let's say you want to download version 1 of the Netscape web browser. Using search engines and the Internet Archives, see if you can locate and download version 1.

## Websites and Web Applications

The *de facto* standard for sharing information is currently through a web browser. While we classify everything we see as "the web," more and more what we really use are "web applications," since not everything on the web is a website. If you check email using a web browser, or get music through a web-connected service, you are using a web application.

Sometimes web applications require privileges. This means you need a login name and password to get access. Having access when you have a legal right to access is called having **privileges**. Hacking into a website to change a page may mean you have access, but since you have no legal right to be there, you don't have privileged access. As you continue to use the web, you'll find that many places give access to privileged areas by accident.

When you find something like this, it's a good habit to report it to the website administrator. However, beware of potential legal repercussions. Unfortunately, many administrators frown upon unsolicited vulnerability reports.

To contribute to making the Internet a safer place while also protecting yourself, you should consider using an **anonymizer** service (e.g., Tor or anonymous remailers, etc.) for sending out vulnerability reports to these administrators. But be aware: all of these anonymous technologies have their weak points and you may not be as anonymous as you think you are! (More than one hacker has learned this the hard way.)

## Exercises

- 1.18 Use a search engine to find sites that have made the mistake of giving privileged access to everyone. To do this, we'll look for folders that let us list the contents (a "directory listing"), something that usually shouldn't be allowed. For this we will use some Google command tricks at <http://www.google.com>. Enter this into the search box:

```
allintitle:"index of" .js
```

Scan the results and you may find one that looks like a directory listing. This type of searching is known as Google Hacking.

- 1.19 Can you find other types of documents in this way? Find three more directory listings that contain .xls files, .doc files, and .avi files.
- 1.20 Are there other search options like "allintitle:"? How can you find them?

## Zines

**Zines**, also known as **e-zines**, are the descendants of **fanzines**: small, usually free magazines with a very small distribution (less than 10,000 readers) and often produced by hobbyists and amateur journalists. Fanzines were printed on paper. Zines on the Internet, like the famous **2600** or the **Phrack** web zine, are written by volunteers; often that means the producers don't edit content for non-technical errors. Sometimes strong language can be surprising for those who aren't familiar with this genre.



Zines have very strong themes or agendas, and tend to be very opinionated. However, they are also more likely to show and argue both sides of issues, since they usually don't care about or have to please advertisers and subscribers.

### Exercises

- 1.21 Search the Internet for three zines on the subject of hacking. How did you find these zines?
- 1.22 Why do you classify these as zines? Remember, just because they market it as a zine or put "zine" in the title doesn't mean it is one.

### Blogs

A **blog** can be considered an evolution of the zine, usually with a writing staff of one. Blogs are updated more often than most print publications or zines, and create communities tied to very strong themes. It's as important to read the commentary as the postings. Even more so than zines, on blogs the response is often immediate and opinionated, with comments from every side. This is one of their special values.

There are millions of blogs on the Internet, but only a small percentage of them are active. The information on almost all, however, is still available.

### Exercises

- 1.23 Search the Internet for three blogs on the subject of hacking.
- 1.24 What groups or communities are these associated with?
- 1.25 Is there a security, law enforcement or academic theme to the blog?

### Forums and Mailing Lists

**Forums** and **mailing lists** are communally developed media, a lot like a recording of conversations at a party. Be a little skeptical about everything you read there. The conversations shift focus often, a lot of what is said is rumor, some people are **trolling**, a **flame war** might erupt, and, when the party is over, no one is certain who said what. Forums and mailing lists are similar, because there are many ways for people to contribute inaccurate information – sometimes intentionally – and there are ways for people to contribute anonymously or as someone else. Since topics and themes change quickly, to get all the information it's important to read the whole thread of comments and not just the first few.

You can find forums on almost any topic, and many online magazines and newspapers offer forums for readers to write responses to the articles they publish. Because of this, forums are invaluable for getting more than one opinion on an article; no matter how much one person liked it, there is certain to be someone who didn't.

There are many mailing lists on special topics, but they can be hard to find. Sometimes the best technique is to search for information on a particular subject to find a mailing list community that deals with it.

As a hacker, what is most important for you to know is that many forums and mailing lists are not searchable through major search engines. While you might find a forum or list through a search engine, you may not find information on individual posts. This information



is part of the invisible web because it contains data that is only searchable directly on the website or forum.

### Exercises

1.26 Find two hacker forums. How did you find these forums?

Can you determine the themes or topics of specialty of these websites?

Do the topics in the forums reflect the theme of the website hosting them?

1.27 Find two hacking or security mailing lists.

Who is the "owner" of these lists? Can you see the member list? (You might need to figure out the application the list has been developed on and then search the web for the hidden commands to see the list of members on that kind of mailing list.)

On which lists would you expect the information to be more factual and less opinionated? Why?

### Newsgroups

**Newsgroups** have been around a long time. There were newsgroups long before the World Wide Web existed. Google purchased the entire archive of newsgroups and put them online at <http://groups.google.com>. Newsgroups are like mailing list archives but without the mail. People posted there directly like they do when commenting to a website. You will find posts in there from the early 1990s onward.

Like the web archives, the group archives can be important for finding who really originated an idea or created a product. They're also useful for finding obscure information that never may have made it to a web page.

Newsgroups aren't used any less today than they were years ago, before the web became the mainstream for sharing information. However, they also haven't grown as their popularity is replaced by new web services like blogs and forums.

### Exercises

1.28 Using Google's groups, find the oldest newsgroup posting you can on the subject of hacking.

1.29 Find other ways to use newsgroups. Are there applications you can use to read newsgroups?

1.30 How many newsgroups can you find that talk about hacking?

1.31 Can you find a current list of all the different newsgroups currently in existence?

### Wikis

**Wikis** are a newer phenomenon on the Internet. Wikipedia ([www.wikipedia.org](http://www.wikipedia.org)) is probably the most famous one, but there are many others. Like many other sites wikis are put together by communities. Reports often claim that wikis are not accurate because they are contributed to by amateurs and fanatics. But that's true of books, mailing lists, magazines, and everything else too. What's important to know is that experts aren't the only source of great ideas or factual information. As the OSSTMM points out, facts come from the small steps of verifying ideas and not great leaps of discovery. That's why wikis are great sources of both professional and amateur ideas slowly and incrementally verifying each other.



Wikis will often discuss many sides of a topic and allow you to follow how information is argued, refuted, refined and changed through a list of edits. So they are great places to dig up information, but often you have to go to the wiki's website to do searches.

### Exercises

- 1.32 Search for "Ada Lovelace." Do you see results from wikis?
- 1.33 Go to Wikipedia and repeat this search. Take a look at the article on her. Was it included in your search results?
- 1.34 Check out the edits of that Wikipedia page and look at the kinds of things that were corrected and changed. What kinds of things were changed? Was there anything changed and then changed back? Now pick a popular movie star or singer you like and go to that page in Wikipedia and check the edits. Do you notice a difference?
- 1.35 Find another wiki site and do the search again. Did any of the results show up in your original search engine search?

### Social Media

Do you use a social media site? Or more than one? As a hacker you're well aware of the popular social media sites of the moment. How about the ones that aren't as hot as they used to be? They still exist, and all their data is still available, in most cases.

This means there is a huge repository of information about us, most of which we have freely given away. And it's going to be there pretty much forever.

Social media sites often have sub-groups or communities of interest. Sites with a professional theme frequently have cybersecurity groups, and sites with an "underground" theme frequently have hacker groups. On the professional sites you are (and everyone else is) expected to use your real name. On the hacker sites, not so much.

Most important of all, do you use your real name on social media sites, or do you use a "handle?" Is there any way your handle could be traced to your real name? Most people don't realize it when they use handles, but it's not uncommon for them to accidentally or sometimes on purpose post their real names, address, city, school, jobs, and so on when using the handle. If another hacker DoXes your handle then they can usually pretty quickly figure out who you really are because of such small mistakes. If you use a handle to be anonymous to those who don't know you, then make sure you take measures to keep it that way. And NEVER confuse your handles if you have more than one.

### Exercises

- 1.36 Search for yourself. Do you get any results (that are actually you)? Are any of them from social media sites?
- 1.37 Go to a social media site you use. Don't log in, but repeat the search as if you were an outsider. How much can you find out about yourself?
- 1.38 Go to a social media site a friend uses. Again, don't log in if you have an account. Search for your friend. How much can you find out about your friend?



## Chat

**Chat**, which comes in the forms of **Internet Relay Chat (IRC)** and **Instant Messaging (IM)**, is a very popular way to communicate.

As a research source, chat is extremely inconsistent because you're dealing with individuals in real time. Some will be friendly and some will be rude. Some will be harmless pranksters, but some will be malicious liars. Some will be intelligent and willing to share information, and some will be completely uninformed, but no less willing to share. It can be difficult to know which is which.

However, once you get comfortable with certain groups and channels, you may be accepted into the community. You will be allowed to ask more and more questions, and you will learn whom you can trust. Eventually you can get access to the very newest hacking exploits (also known as **zero day**, meaning it was just discovered right now) and advance your own knowledge.

### Exercises

- 1.39 Find three instant messaging programs. What makes them different? Can they all be used to talk to each other?
- 1.40 Find out what IRC is and how you can connect to it. Can you discover what network holds ISECOM's channel? Once you join the network, how do you attach to the isecom-discuss channel?
- 1.41 How do you know what channels exist on an IRC network? Find three security channels and three hacker channels. Can you enter these channels? Are people talking or are they bots?

## P2P

**Peer to Peer**, also known as **P2P**, is a network inside the Internet. Unlike the usual client/server network, where every computer communicates through a central server, the computers in a P2P network communicate directly with each other. Most people associate P2P with downloading MP3s and pirated movies on the infamous original Napster, but there are many other P2P networks – both for the purpose of exchanging information, and as a means of conducting research on distributed information sharing.

The problem with P2P networks is that, while you can find just about anything on them, some things are on the network illegally. And other things are there legally but the companies that created them still feel that they should not be there and are happy to demand money from the owner of any **Internet gateway** where it's downloaded.

At the moment there is not much agreement on whether the person whose Internet access was used to download content is responsible or if the police actually have to catch the person who did it. That's like saying that if your car is used to commit a crime the owner of the car, not the driver, goes to jail. Internet laws are currently not just and not fair so be extra careful!

Whether or not you are the kind of person who risks downloading intellectual property, there is no question that P2P networks can be a vital resource for finding information. Remember: there is nothing illegal about P2P networks – there are a lot of files that are available to be freely distributed under a wide variety of licenses – but there are also a lot of files on these networks that shouldn't be there. Don't be afraid to use P2P networks, but be aware of the dangers, and of what you are downloading.



## Exercises

- 1.42 What are the three most popular and most used P2P networks? How does each one work? What program do you need to use it?
- 1.43 Research the protocol of one of those P2P networks. What does it do, and how does it make downloading faster?
- 1.44 Search for the words "download Linux." Can you download a distribution (or distro) of Linux using P2P?

## Certifications

There are OSSTMM Security Tester and Security Analyst certifications, various colors of "hacker" certifications, certifications based on one version of "best practices" or another and certifications with all kinds of crazy initials or pieces of punctuation.

Why do you care about certifications? Because you can get some of them at any age, because you don't have to have a college degree to get them, and because they can put you in the position of the sought-after person, rather than the person asking for a gig.

The problem with best-practices based certifications is that best practices change often, because *best practices* is just another way of saying "what everyone else is doing right now." Often what everyone else is doing is wrong this week, and will still be wrong when they update it next week.

Then there are research-based certifications, based on valid and repeatable research into human and system behavior. Needless to say our parent organization, [ISECOM](#), falls squarely into the realm of research-based certification authorities. Whether from ISECOM or elsewhere, look for skills-based certifications and analysis-based or **applied knowledge** certifications that make you prove you can do what you say you've learned. That will be handy when you actually have to do it.

## Seminars

Attending seminars is a great way to hear theory explained in detail and to watch skills in action. Even product-focused seminars are good to attend to see how a product is intended to be used, as long as you're aware that the event is a marketing pitch and their real job is to sell.

We would be remiss if we didn't mention that we can bring [Hacker Highschool Seminars](#) to many locations, and we can cover any of the available lessons. Seminars consist of professional hackers talking to students about hacking and being a hacker, both the bad and the good. These seminars take a sharp look at what real hackers are from research in the **Hacker Profiling Project**, a collaboration project with the United Nations to explore who hackers are and why they hack. Then you will go on to discover the bright side of hacking and how it's not always a dark thing.

One of the most powerful things we can help you find is the means to be as intellectually curious and resourceful as a hacker. Hackers succeed at what they do because they know how to teach themselves, go beyond the lessons available and learn the skills they need to go further.

You're also welcome to ask your parents and educators to learn how to help out and how to start a Hacker Highschool chapter at your school. Contact ISECOM for more information.



## Microcomputer Universe: Introduction

Have you heard of the Raspberry Pi? It's a tiny single-board computer (**SBC**) about the size of a credit card, and about the price of a couple of pizzas. It's not a wimpy little thing, either: the P1 2 has a quad-core CPU, USB and HDMI video. A tiny microSD card serves as the hard drive, and swaps so easily you can use several operating systems.



Figure 1.4: a boxed Raspberry Pi, photo by Nico Kaiser

The Raspberry Pi isn't the only microcomputer out there. Other manufacturers sell the Banana Pi, the Orange Pi, the BeagleBone, the Radxa Rock – each with a unique set of features. Want built-in Bluetooth and wifi, for instance? One of the above offers exactly that. (Start researching.)

There are basically three types of microcomputers; the first is the Linux-based board that can act as a full computer or server. The next is an Arduino device that is more of a controller for sensors, robotic equipment, lighting control and basic items. The last type of microcomputer is the hybrid device that is built for specific purposes like controlling sensors but can also be a computer to a point.

The Raspberry Pi (RPI) is in the first category since it runs Linux. This computer has a 1GHZ Cortex Arm quad core processor with 1 gig of ram. The board includes a GPU, four USB ports, an ethernet port, HDMI video output, audio output and a 40-pin GPIO connector to add additional parts onto it. (The attachments have a fun name: because they are Hardware Attached on Top, they're called "HATs" - but that's only with Pies.) You can find them for well under \$50, even lower for older models.

You can learn programming (python, scratch, java, php and others) on this device. It can also be used as a server for email, web, multimedia, VPN, or even a NAT firewall. One very popular use is as a Minecraft server. There is a huge collection of information available on the Internet for this device.

Arduino devices are made in Italy and were primarily built to teach coding, control sensors and robotic equipment. These devices are a bit more expensive and are more customized for specific jobs. These require some expertise to work with.

Hybrid devices include Udoo, Remix, Radxa Rock and others. The prices for these vary greatly but often include wifi and Bluetooth built in. Neither Arduino nor Raspberry Pi have these connections but it doesn't cost much to add them on (except the Arduino Yun which has wifi built in). The hybrids are built to act as controllers and computers on a much higher level than the RPI or Arduino. The Remix is a full blown computer that runs a special version of Android. The Udoo tries to replace the RPI and Arduino with a single device.

If you are looking to start out, we would suggest the RPI as your first step. It is cheap but requires a learning curve to set working. All of these devices use a microSD card to hold the operating system. Luckily, most devices allow you to choose which operating system you want to work with. Changing operating systems is as simple as swapping out the microSD card.



Most of the devices connect to a TV or monitor using HDMI. You can connect a wireless or USB keyboard to control your device or puTTY in based on the local IP address of the device.

We'll be working more with these microcomputer devices in Hacker Highschool in upcoming **Microcomputer Universe** sections.



## Further Study

---

Now you should practice until you're a master of researching. The better you get at it, the more information you will find quickly, and the faster you will learn. But be careful also to develop a critical eye. Not all information is truth.

Always remember to ask yourself, why would someone lie? Is there money involved in being dishonest or perpetuating a rumor or story? Where do the facts come from? And most importantly, what is the scope?

Like all hacking, research includes a scope. That's really important when you see statistics, like math that uses percentages and fractions and odds. Always look to see where the scope took place and recognize that the scope has to apply. A common place you see this is national crime or health statistics taken from a small sample in just one part of the country. Just because something affects 10% of people out of 200 studied in a single city doesn't mean that 10% of the whole nation has this same problem. So be smart about how you read information as well as how you find it. Figuring out the scope of the information always makes a huge difference!

To help you become a better researcher for the Hacker Highschool program, here are some additional topics and terms for you to investigate:

Meta Search

The Invisible Web

Google Hacking

How Search Engines Work

The Open Source Search Engine

The Jargon File

OSSTMM

ISECOM Certifications:

OPST (OSSTMM Professional Security Tester)

OPSA (OSSTMM Professional Security Analyst)

OPSE (OSSTMM Professional Security Expert)

OWSE (OSSTMM Wireless Security Expert)

CTA (Certified Trust Analyst)

SAI (Security Awareness Instructor)

Today's teens are in a world with major communication and productivity channels open to them and they don't have the knowledge to defend themselves against the fraud, identity theft, privacy leaks and other attacks made against them just for using the Internet. This is the reason for Hacker Highschool.

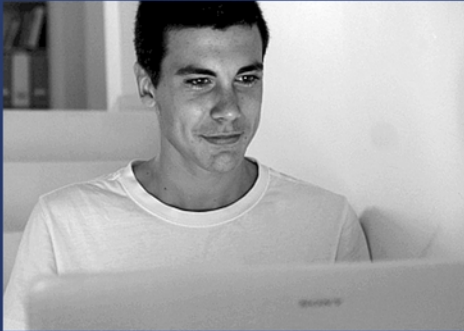
**The Hacker Highschool project is the development of security and privacy awareness learning materials for junior high and high school students.**

Hacker Highschool is a set of lessons and a practical means of making hackers. Beyond just providing cybersecurity awareness and critical Internet skills, we need to teach the young people of today how to be resourceful, creative, and logical, traits synonymous with hackers. The program contains free security and privacy awareness teaching materials and back-end support for teachers of accredited junior high, high schools, and home schooling. There are multiple workbooks available in multiple languages. These are lessons that challenge teens to be as resourceful as hackers, including safe Internet use, web privacy, researching on the internet, avoiding viruses and Trojans, legalities and ethics, and more.

**The HHS program is developed by ISECOM, a non-profit, open-source research group focused on security awareness and professional security development and accreditation.**

# Hacker HighSchool

## SECURITY AWARENESS FOR TEENS



## LESSON 2 ESSENTIAL COMMANDS



HACKING IS LEARNING  
[www.hackerhighschool.org](http://www.hackerhighschool.org)

ISECOM

Creative Commons 3.3 Attribution-Non-Commercial-NoDerivs ISECOM

[WWW.ISECOM.ORG](http://WWW.ISECOM.ORG) - [WWW.OSSTMM.ORG](http://WWW.OSSTMM.ORG) - [WWW.HACKERHIGHSCHOOL.ORG](http://WWW.HACKERHIGHSCHOOL.ORG) - [WWW.BADPEOPLEPROJECT.ORG](http://WWW.BADPEOPLEPROJECT.ORG) - [WWW.OSSTMMTRAINING.ORG](http://WWW.OSSTMMTRAINING.ORG)



## WARNING

---

The Hacker Highschool Project is a learning tool and as with any learning tool there are dangers. Some lessons, if abused, may result in physical injury. Some additional dangers may also exist where there is not enough research on possible effects of emanations from particular technologies. Students using these lessons should be supervised yet encouraged to learn, try, and do. However ISECOM cannot accept responsibility for how any information herein is abused.

The following lessons and workbooks are open and publicly available under the following terms and conditions of ISECOM:

All works in the Hacker Highschool Project are provided for non-commercial use with elementary school students, junior high school students, and high school students whether in a public institution, private institution, or a part of home-schooling. These materials may not be reproduced for sale in any form. The provision of any class, course, training, or camp with these materials for which a fee is charged is expressly forbidden without a license, including college classes, university classes, trade-school classes, summer or computer camps, and similar. To purchase a license, visit the LICENSE section of the HHS web page at <http://www.hackerhighschool.org/licensing.html>.

The Hacker Highschool Project Project is an open community effort and if you find value in this project, we ask that you support us through the purchase of a license, a donation, or sponsorship.



## Table of Contents

Introduction and Objectives.....	5
Requirements and Setup.....	6
Requirements.....	6
Setup.....	6
Operating System: Windows.....	7
How to open a CLI window.....	7
Commands and Tools (Windows/DOS).....	7
Commands.....	8
Tools.....	9
Game On: Taking Command.....	12
Operating System: Linux.....	13
Feed Your Head: Console, Terminal or Shell?.....	13
How to open a terminal window.....	14
Linux Commands and Tools.....	14
Commands.....	14
Tools.....	17
Operating System: OSX.....	18
How to open a Terminal window.....	18
Commands and Tools (OSX).....	19
Commands.....	19
Tools.....	21
Basic Command Equivalences for Windows, OSX and Linux.....	24



## Contributors

---

Pete Herzog, ISECOM  
Marta Barceló, ISECOM  
Bob Monroe, ISECOM  
Marco Ivaldi, ISECOM  
Greg Playle, ISECOM  
Simone Onofri, ISECOM  
Kim Truett, ISECOM  
Jaume Abella, ISECOM  
Tom Thomas, ISECOM  
Jairo Hernández  
Aneesh Dogra

**ISECOM**





## Introduction and Objectives

Whether you've seen "hacking" in the 1995 movie *Hackers* or Trinity hacking into a UNIX system in *The Matrix Reloaded*, when you picture a hacker they're working at the command line. For good reason.

You can do very big, very powerful things in the command line interface (**CLI**). You don't have to be a master at the command line but you should be comfortable working with it.

Once you've mastered the basics of the CLI, you can start using these commands in text files (called **scripts**); it's the easiest programming ever.

We will discuss commands and basic tools for Windows, OSX and Linux operating systems. You'll need to know them for exercises in the following lessons. At the end of this lesson, you should be familiar with:

- General Windows, Linux and OSX commands
- Basic network commands and tools, including

```
ping
tracert/traceroute
netstat
ipconfig/ifconfig
route
```



## Requirements and Setup

### Requirements

To complete this lesson you will need:

- A PC running Windows
- A PC running Linux
- Optionally a Mac running OSX
- Access to the Internet

### Setup



**Figure 2.1:** General Network Setup

This is the network in which we'll do most of our work. It consists of your PC, the Internet, and the ISECOM Hacker Highschool test network, which you will access through the Internet.

Note that access to the ISECOM test network is restricted. In order to gain access to it, your instructor must contact the system administrator, as detailed on the <http://www.hackerhighschool.org> web site.

However, you can also substitute any test network for these exercises. **NEVER** run tests against computers you don't own! That may be a criminal offense, and can be dangerous in lots of other ways.

If you want to set up your own test network, it can be as easy as testing another computer in your classroom or home. No special set-up is needed! Of course if you want something more robust or something that lets you experience the challenges and flaws of accessing another computer over the Internet, then you'll need an Internet-based test network. This can also be done by making alliances with other schools or homes and letting them access certain computers of yours remotely and you access theirs. But make sure you know what you're doing in setting it up because what you don't want is for those open computers to get hijacked by some random person on the Internet who does damage for which you will be responsible.

## Operating System: Windows

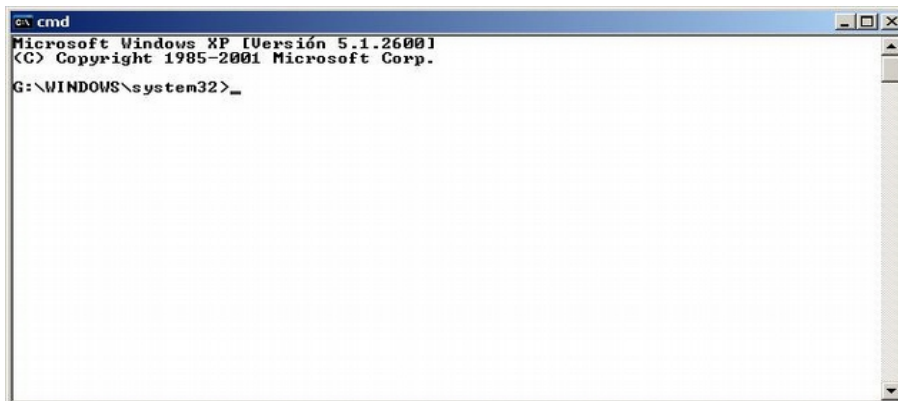
In the days of yore, if we weren't working in UNIX, we all worked in DOS. We didn't need to open a CLI; we lived in one. Then UNIX developed "window" interfaces, an idea that eventually came to the PC with Microsoft Windows.

Once Windows arrived, we opened DOS in a window on our desktop and called that a **command prompt**. Long after Windows had moved beyond being DOS-based, Windows still has a CLI – and many people still call it a **DOS box**. It's not really DOS any more, but for our purposes, it doesn't matter. Here's how you open one.

### How to open a CLI window

The procedure is similar for all versions of Windows.

1. Click the START button.
2. Choose the RUN option (skip this in Vista and later).
3. Type **command** if you are using Windows 95/98 or **cmd** for all other versions of Windows and press Enter or click OK.
4. A window similar to the following will appear:



5. Now you can use the commands and tools listed below.

### Commands and Tools (Windows/DOS)

Commands provide built-in operating system functions. Tools do more: they probe networks, search for **hosts** (which is, by the way, what we call computers attached to a network), and let you see or set your host's routing information.



## Commands

Words in italics are options that you must enter.  
 Some commands have both long and short versions.

Command	Purpose
<b>date</b>	Display or set the date
<b>time</b>	Display or set the time
<b>ver</b>	Display the MS-DOS or Windows version
<b>dir</b>	Display the list of subdirectories and files in a directory.
<b>cls</b>	Clear the screen.
<b>mkdir <i>directory</i></b> or <b>md <i>directory</i></b>	Make a directory with the name <i>directory</i> :  md tools
<b>chdir <i>directory</i></b> or <b>cd <i>directory</i></b>	Change the current directory to another directory:  cd tools
<b>rmdir <i>directory</i></b> or <b>rd <i>directory</i></b>	Delete the directory:  rd tools
<b>tree <i>directory</i></b>	Display the structure of folders and files in text-graphic format:  tree c:\tools
<b>chkdsk</b>	Check a disk and show a status report.
<b>mem</b>	Show the amount of memory used and free in the system.
<b>rename <i>source</i></b> <b><i>dest</i></b> or <b>ren <i>source dest</i></b>	Change the name of files:  ren pictures MyPics
<b>copy <i>source dest</i></b>	Copy one or more files to another location:  copy c:\tools\myfile.txt c:\tmp\
<b>move <i>source dest</i></b>	Move files and change the name of files and directories:  move c:\tools c:\tmp
<b>type <i>file</i></b>	Display the content of one or more text files:  type c:\tools\myfile.txt
<b>more <i>file</i></b>	Display the information screen by screen:  more c:\tools\myfile.txt
<b>delete <i>file</i></b> or <b>del <i>file</i></b>	Delete one or more files:  del c:\tools\myfile.txt



## Tools

Words in italics are options that you must enter.

Tool	Purpose
<p><b>ping</b> <i>host</i></p>	<p>Verify contact with the machine <i>host</i>.</p> <p>This command sends ICMP (Internet Control Message Protocol) ping packets to another computer to see how long it takes to respond, or if it responds at all. You can use a hostname or an IP address:</p> <pre>ping hackerhighschool.org ping 216.92.116.13</pre> <p>Options include:</p> <pre>ping -n 100 hackerhighschool.org</pre> <p>which sends 100 ping packets, and</p> <pre>ping -t 216.92.116.13</pre> <p>which pings the host until stopped with CTRL+C.</p> <p>To see more options:</p> <pre>ping /h</pre>
<p><b>tracert</b> <i>host</i></p>	<p>Show the route that packets follow to reach the machine <i>host</i>.</p> <p>The DOS <b>tracert</b> command is an adaptation of the UNIX <b>traceroute</b>. (DOS commands could only be eight characters long, back in the day.) Both allow you to find the route that a packet follows from your host to the destination host, tracert also tracks how long each hop takes and travels, at the most, 30 hops. Often you can see the hostnames of the machines through which the packets travel:</p> <pre>tracert hackerhighschool.org tracert 216.92.116.13</pre> <p>Some options are:</p> <pre>tracert -n 25 hackerhighschool.org</pre> <p>to specify N, at the most, jumps, and</p> <pre>tracert -d 216.92.116.13</pre> <p>to hide hostnames.</p> <p>To see more options:</p> <pre>tracert /?</pre>



Tool	Purpose
<b>ipconfig</b>	<p>Used alone, displays information on your host's active network interfaces (ethernet, ppp, etc.). It is similar to Linux <b>ifconfig</b>.</p> <p>Some options are:</p> <pre>ipconfig /all</pre> <p>to show more details</p> <pre>ipconfig /renew</pre> <p>to renew the network connection when automatic configuration with DHCP is used, and</p> <pre>ipconfig /release</pre> <p>to deactivate networking when DHCP is used.</p> <p>More options:</p> <pre>ipconfig /?</pre>
<b>route print</b>	<p>Displays the routing table. <b>route</b> can also be used to set up or erase static routes.</p> <p>Some options:</p> <pre>route print</pre> <p>to show the list of routes,</p> <pre>route delete</pre> <p>to delete a route, and</p> <pre>route add</pre> <p>to add a route.</p> <p>More options:</p> <pre>route/?</pre>

Tool	Purpose
<b>netstat</b>	Displays information on the status of the network and established connections with remote machines.  Some options:  <pre>netstat -a</pre> to check all the connections and listening ports,  <pre>netstat -n</pre> to display addresses and port numbers in numeric form, and  <pre>netstat -e</pre> to sample Ethernet statistics.  Options can be used together:  <pre>netstat -an</pre> To see more options:  <pre>netstat/?</pre>

For additional information on these commands and tools try these options:

```
command /h
```

```
command /?
```

```
help command
```

from a CLI window.

For example, for additional information on the tool **netstat**, you have three possibilities:

```
netstat /h
```

```
netstat /?
```

```
help netstat
```

## Exercises

- 2.1 Open a CLI window.
- 2.2 Identify the version of DOS or Windows that you are using.
- 2.3 Identify the date and time of the system. If they are incorrect, correct them.
- 2.4 Identify all the directories and files that are in c:\.
- 2.5 Create the directory c:\hhs\lesson2. Copy to this directory the files with the extension .sys that are in c:\. What files have you found?
- 2.6 Identify the IP address of your host.
- 2.7 Trace the route to [www.hackerhighschool.org](http://www.hackerhighschool.org). Identify IP addresses of the intermediate routers.



### Game On: Taking Command

"Microsoft Fenestra is neither an operating system nor an interface. It is a graphical system built around Solitaire," announced the technology teacher with bits of food stuck to the corners of his wet mouth. Mr. Tri was satisfied that the students bought that line of trash so he moved on. "Fenestra has a command interface, where you speak to the monitor and the computer does whatever it is that you want it to do. If you want a cup of coffee, just tell the monitor and a nice fresh cup of Joe appears."

Jace was so close to strangling this man, wondering if the police and judge would be sympathetic to her murder plea considering how he was butchering computer technology.

"Wait, hold on Mr. Tri." Jace hadn't let a breath out in the last ten minutes so her face was a funny color. "Sir, Fenestra is a graphical user interface, GUI, like the used gum you keep in that jar." Kids wrinkled their noses and giggled.

She got up and slid around him, getting behind the keyboard like a professional basketball player slipping past the defense. "Click Windows, type CMD, hit Enter. Check out the CLI. See that blinking line? That's where you type. See how it says what folder you're in?" Like a Formula 1 driver she never looked back; she just picked up speed.

"Now you can type CD C: and you're into the system root." Jace buried the throttle. "With a new system, you'll want to know as much as you can about your environment. Start by typing in VER, that's short for version. Now we can tell exactly what version of the operating system is running. See?" Students were staring. Mr. Tri was paralyzed.

Jace felt herself connecting with the computer, typing faster, becoming effortless. She mused out loud, "You can get a computer to spill its guts and tell you everything that's happening inside of it." Her fingers flew across the keyboard, dislodging a key and sending it into the air where it landed in the moldy jar of old gum on the teacher's desk. Three girls in the front swallowed their gum.

Jace took this as her cue to stop. She stood up abruptly, giving the keyboard back to her teacher. His face was white and there was spit on his lips. She pulled a laser pointer from her inside jacket pocket quickly as if she were drawing a gun and shone it on Mr. Tri's forehead. A boy in the back of the class peed himself. Then turning it to the pathetic presentation slide on the screen at the front of the class, she calmly said, "These slides are so wrong they have to go."

"Maybe it's you who should go," the teacher said, handing her a pass to the big man's office, the Vice Principal, aka the Principal of Vice. Her third pass this week. Technology was going to be the end of her, or at least the loss of her free time with another night of detention.

### Game Over





## Operating System: Linux



Just like in Windows, when you're using Linux, you run commands in a CLI window. You'll see these called **consoles**, **terminals** and **shells**.

### Feed Your Head: Console, Terminal or Shell?

Amaze your friends by knowing the difference.

- The **console** was actually the screen and keyboard attached directly to a computer back when the old people today used **dumb terminals** to access the computer remotely.
- You actually have your choice of **shell** in Linux, including **bash**, **tcsh** and **zsh**, among others. Different shells let you do very different things, and which one you like is almost a political issue. In most cases, you'll use bash. When you connect to the Hacker Highschool test network, you'll get an **empty shell**.
- When you open a **console window** what you're technically opening is a **terminal emulator** or **terminal window**, that is, a "fake" dumb terminal running in a window on your desktop.

What can you do at the Linux command line? Everything you could possibly do in any GUI tool, plus vastly more. Race your Windows friends to set your IP address: they will have to drill through all kinds of interfaces to do it. In Linux you could do it with:

```
ifconfig eth0 192.168.1.205
```

Bet you can type that faster than they can click!



## How to open a terminal window

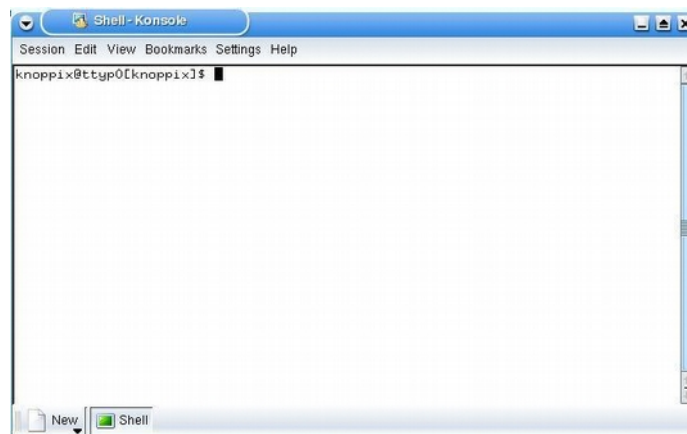
Because there are many versions of Linux, there are several ways to start a console window.

1. Click the Start Application button.
2. If you see a "Run Command" option, click it and enter "konsole", then Return.
3. Or look for Accessories, then choose Terminal.
4. Or on many systems you can press CTL-ALT-T.
5. A window similar to this will appear.
6. Now you can use the commands and tools listed below.

## Linux Commands and Tools

### Commands

Words in *italics* are options that you must enter.



Command	Purpose
<b>date</b>	Display or set the date.
<b>time</b>	Display or set the time.
<b>fsck</b>	Check a file system and show a status report.
<b>cat <i>file</i></b>	Display the content of one or more text files:  <b>cat /etc/passwd</b>
<b>pwd</b>	Display the name of the current directory.
<b>hostname</b>	Display the name of the computer you are currently using.
<b>finger <i>user</i></b>	Display information on a user:  <b>finger root</b>

Command	Purpose
<b>ls</b>	List the contents of the current directory:  <pre>ls -la</pre> List the contents of another directory:  <pre>ls -la /etc</pre>
<b>cd <i>directory</i></b>	Change from current directory to <i>directory</i> . If no directory name is specified it changes to the home directory.  For the login name "fred" the command  <pre>\$cd</pre> changes the directory to /home/fred, and  <pre>\$cd -</pre> changes to the last visited directory (think of "subtracting" one directory), and  <pre>\$cd /tmp</pre> changes to the /tmp directory.
<b>cp <i>source dest</i></b>	Copy the file <i>source</i> to the file <i>dest</i> .  Example:  <pre>cp /etc/passwd /tmp/bunnies</pre>
<b>rm <i>file</i></b>	Delete files. Only users with proper access permissions (or root) can delete specific files.  <pre>rm letter.txt</pre>
<b>mv <i>source dest</i></b>	Move or rename files and directories.  Example:  <pre>mv secrets.zip innocent.zip</pre>
<b>mkdir <i>directory</i></b>	Make a directory with the name <i>directory</i> .  Example:  <pre>mkdir tools</pre>
<b>rmdir <i>directory</i></b>	Delete the directory with the name <i>directory</i> but only if it is empty:  <pre>rmdir tools</pre> Bonus question: how do you delete a directory with files in it?
<b>find / -name <i>file</i></b>	Look for files, starting at /, with the name <i>file</i> :  <pre>find / -name myfile</pre>
<b>echo <i>string</i></b>	Write <i>string</i> to the screen:  <pre>echo hello</pre>



Command	Purpose
<b><i>command &gt; file</i></b>	<b>Redirect</b> the normal screen output of <i>command</i> to <i>file</i> :  <pre>ls &gt; listing.txt</pre> If this file already exist, it will get <b>clobbered</b> , meaning overwritten!
<b><i>command &gt;&gt; file</i></b>	Redirect the normal screen output of <i>command</i> to <i>file</i> . If the file already exists, it <b>appends</b> the output to the end of the file.  Example:  <pre>ls &gt;&gt; listing.txt</pre>
<b><i>man command</i></b>	Show the pages of the online manual about <i>command</i> :  <pre>man ls</pre>

For additional information on these commands and tools try these options:

```
command -h
command --help
man command
help command
info command
```

For example, for additional information on the *ls* command, type in either of these two possibilities:

```
ls --help
man ls
```



## Tools

Words in italics are options that you must enter.

Tool	Purpose
<b>ping</b> <i>host</i>	Verify the contact with the machine <i>host</i> :  <code>ping www.google.com</code>
<b>tracert</b> <i>host</i>	Show the route that the packets follow to reach the machine <i>host</i> :  <code>tracert www.google.com</code>
<b>ifconfig</b>	Display information on active network interfaces (ethernet, ppp, etc.).
<b>route</b>	Display the routing table.
<b>netstat</b>	Display information on your network connections.  <code>netstat -an</code>

## Exercises

- 2.8 Identify the owner of the file **passwd**. (Note: first locate where this file is.)
- 2.9 Create the directory **work** in your own home directory (for example, if your login is **fred**, create the directory in /home/fred), and copy the file passwd to the directory work that you just created. Identify the owner of the passwd copy.
- 2.10 Create the directory **.hide** in the work directory (notice that the file name begins with a dot). List the contents of this directory. What did you have to do to see the contents of directory .hide?
- 2.11 Create the file **test1** with the content, "This is the content of the file test1" in the work directory. Create the file test2 with the content, "This is the content of the file test2" in the work directory. Copy into a file with the name **test** the contents of both previous files.

## Operating System: OSX

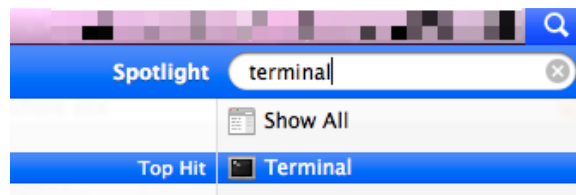
Just like in Linux, when you're using OSX, you run commands in a CLI window. In OSX this application is called **Terminal**.

OSX is based on NetBSD and FreeBSD UNIX, ancestors of Linux. Its GUI and CLI approach is similar to Linux: you can do everything you could possibly do in any GUI tool, plus vastly more.

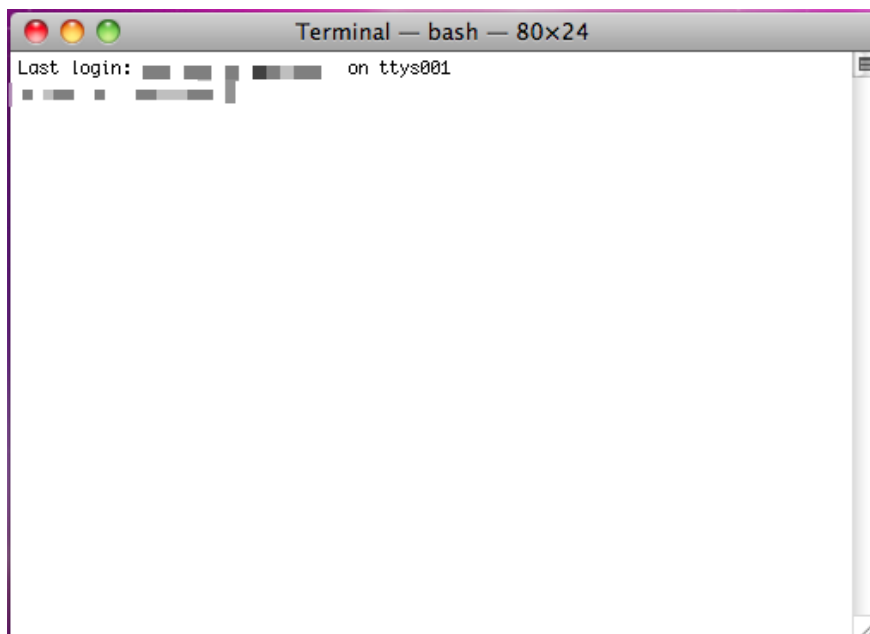
Some people think Windows stole the whole idea of a GUI from Mac. Actually, GUI interfaces and mouse pointers were used in a much older OS. You can know more than practically everybody by finding out which.

### How to open a Terminal window

1. Click on the **Spotlight** icon, an icon of a magnifying glass typically located on the top right of your screen, and search for **Terminal**.



2. Then press Enter or click on it. You will see the Terminal window.



Typically Terminal is located under **Applications > Utilities**. Impress your friends by changing the Terminal style depending on your preferences. Press both the Command and comma

keys to get the Preference dialog for Terminal and choose your preferred colors. Usually this keyboard shortcut gives you access to program preferences in OSX.

## Commands and Tools (OSX)

Mac ships with a bash shell, so almost all typical Linux commands work on OSX as well.

### Commands

Words in italics are options that you must enter.

Command	Purpose
<b>date</b>	Display or set the date.
<b>time <i>command</i></b>	Display how long it takes for <i>command</i> to execute.
<b>fsck</b>	Check a file system and show a status report. If you use an OSX journaled volume such as Mac OSX 10.3 or later, in which journaling is enabled by default, you probably won't need to run this command.
<b>cat <i>file</i></b>	Display the content of one or more text files:  <code>cat /etc/passwd</code>
<b>pwd</b>	Display the name of the current directory.
<b>hostname</b>	Display the name of the computer you are currently using.
<b>finger <i>user</i></b>	Display information on a user:  <code>finger root</code>
<b>ls</b>	List the contents of the current directory:  <code>ls -la</code>  List the contents of another directory:  <code>ls -la /etc</code>
<b>cd <i>directory</i></b>	Change from current directory to <i>directory</i> . If no directory name is specified it changes to the user's home directory.  For the login name "fred" the command  <code>cd</code>  changes the directory to /Users/fred, and  <code>cd -</code>  changes to the last visited directory (think of "subtracting" one directory), and  <code>cd /tmp</code>  changes to the /tmp directory.
<b>cp <i>source dest</i></b>	Copy the file <i>source</i> to the file <i>dest</i> .  <code>cp /etc/passwd /tmp/bunnies</code>



Command	Purpose
<b>rm file</b>	Delete files. Only users with proper access permissions (or root) can delete certain files.  <code>rm letter.txt</code>
<b>mv source dest</b>	Move or rename files and directories.  <code>mv secrets.zip innocent.zip</code>
<b>mkdir directory</b>	Make a directory with the name <i>directory</i> .  <code>mkdir tools</code>
<b>rmdir directory</b>	Delete the directory with the name <i>directory</i> but only if it is empty:  <code>rmdir tools</code> Bonus question: how do you delete a directory with files in it?
<b>find / -name file</b>	Look for files, starting at /, with the name <i>file</i> :  <code>find / -name myfile</code>
<b>echo string</b>	Write <i>string</i> to the screen:  <code>echo hello</code>
<b>command &gt; file</b>	<b>Redirect</b> the normal screen output of <i>command</i> to <i>file</i> :  <code>ls &gt; listing.txt</code> If this file already exist, it will get <b>clobbered</b> , meaning overwritten!
<b>command &gt;&gt; file</b>	Redirect the normal screen output of <i>command</i> to <i>file</i> . If the file already exists, it <b>appends</b> the output to the end of the file.  Example:  <code>ls &gt;&gt; listing.txt</code>
<b>man command</b>	Show the pages of the online manual about <i>command</i> :  <code>man ls</code>

For additional information on these commands and tools try these options:

```
command -h
command --help
man command
help command
info command
```

For example, for additional information on the `ls` command, type in either of these two possibilities:

```
ls --help
man ls
```





## Tools

Words in italics are options that you must enter.

Tool	Purpose
<p><b>ping</b> <i>host</i></p>	<p>Verify contact with the machine <i>host</i>.</p> <p>This command sends ping packets using ICMP (Internet Control Message Protocol) to another computer to see how long it takes to respond, or if it responds at all. You can use a hostname or an IP address:</p> <pre>ping www.hackerhighschool.org ping 216.92.116.13</pre> <p>Options include:</p> <pre>ping -c 100 www.hackerhighschool.org</pre> <p>which sends 100 ping packets, and</p> <pre>ping -t 216.92.116.13</pre> <p>which pings the host until stopped with CTRL+C.</p> <p>More options:</p> <pre>man ping</pre>
<p><b>tracert</b> <b>tracert</b> <i>host</i></p>	<p>Show the route that packets follow to reach the machine <i>host</i>.</p> <p><b>tracert</b> has the same scope as Windows <b>tracert</b> but uses different network protocols: <b>tracert</b> uses UDP (User Datagram Protocol) and <b>tracert</b> uses ICMP (Internet Control Message Protocol). You may obtain different results using <b>tracert</b> and <b>tracert</b> from same network source and destination.</p> <p>Both allow you to find the route that a packet follows from your host to the destination host. Each also tracks how long each hop takes and travels for, at the most, 30 hops. Often you can see the hostnames of the machines through which the packets travel:</p> <pre>tracert www.hackerhighschool.org tracert 216.92.116.13</pre> <p>To specify the maximum (-m) number of hops:</p> <pre>tracert -m 25 www.hackerhighschool.org</pre> <p>To save DNS lookups by showing the IP address rather than a hostname:</p> <pre>tracert -n 216.92.116.13</pre> <p>To see more options:</p> <pre>man tracert</pre>



Tool	Purpose
<b>ifconfig</b>	<p>Used alone, displays information on your host's active network interfaces (ethernet, ppp, etc.). It is similar to Windows <b>ipconfig</b>.</p> <p>To show more details, meaning to be <b>verbose</b>:</p> <pre>ifconfig -v</pre> <p>To show only the <i>en1</i> network interface information:</p> <pre>ipconfig en1</pre> <p>To deactivate the network interface:</p> <pre>ifconfig en1 down</pre> <p>To bring it up:</p> <pre>ifconfig en1 up</pre> <p>Note: you must have permission to use this command, so you may need to put <b>sudo</b> in front of these commands. Then you will have to enter your password. <b>Use sudo carefully!</b></p> <pre>sudo ifconfig en1 up</pre> <p>More options:</p> <pre>man ifconfig</pre>
<b>netstat</b>	<p>Displays information on the status of the network and established connections with remote machines. On BSD-like systems, <b>netstat</b> is also used to see your routing table.</p> <p>To sample all the connections and listening ports:</p> <pre>netstat -a</pre> <p>To display the routing table:</p> <pre>netstat -r</pre> <p>Used with <b>-n</b> to show addresses numerically:</p> <pre>netstat -nr</pre> <p>To show information for <i>en1</i> network interface.</p> <pre>netstat -r -ii en1</pre> <p>To see more options:</p> <pre>man netstat</pre>



## Exercises

- 2.12 Identify the name and the IP address of your machine.
- 2.13 Trace the route to `www.hackerhighschool.org`. Identify IP addresses of the intermediate routers and find your path.
- 2.14 In Windows use **tracert** to see the path between you and `www.hackerhighschool.org` as seen by Windows, and send the output to a file named **output.txt** for further analysis.
- 2.15 Then run the equivalent `tracert` command on OSX and Linux from the same network, putting the output in files named **output2OSX.txt** and **output2Linux.txt**. Look at the output files carefully.
  1. Are the paths the same or are there differences?
  2. Did you find any lines containing the string:  
\* \* \*  
What does it mean?
  3. Repeat this test at least an hour later. Are the results always the same?



## Basic Command Equivalences for Windows, OSX and Linux

Words in italics are options that you must enter.

<b>Linux</b>	<b>OSX</b>	<b>Windows</b>
command --help	command --help	<i>command /h,</i> <i>command /?</i>
man <i>command</i>	man <i>command</i>	help <i>command</i>
cp	cp	copy
rm	rm	del
mv	mv	move
mv	mv	ren
more, less, cat	more, less, cat	type
lpr	lpr	print
rm -R	rm -R	deltree
ls	ls	dir
cd	cd	cd
mkdir	mkdir	md
rmdir	rmdir	rd
netstat -r	netstat -r	route print
tracert	tracert	tracert
ping	ping	ping
ifconfig	ifconfig	ipconfig

Today's teens are in a world with major communication and productivity channels open to them and they don't have the knowledge to defend themselves against the fraud, identity theft, privacy leaks and other attacks made against them just for using the Internet. This is the reason for Hacker Highschool.

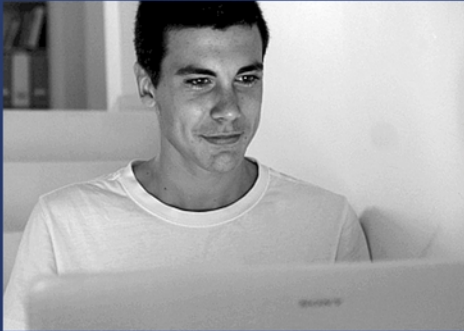
**The Hacker Highschool project is the development of security and privacy awareness learning materials for junior high and high school students.**

Hacker Highschool is a set of lessons and a practical means of making hackers. Beyond just providing cybersecurity awareness and critical Internet skills, we need to teach the young people of today how to be resourceful, creative, and logical, traits synonymous with hackers. The program contains free security and privacy awareness teaching materials and back-end support for teachers of accredited junior high, high schools, and home schooling. There are multiple workbooks available in multiple languages. These are lessons that challenge teens to be as resourceful as hackers, including safe Internet use, web privacy, researching on the internet, avoiding viruses and Trojans, legalities and ethics, and more.

**The HHS program is developed by ISECOM, a non-profit, open-source research group focused on security awareness and professional security development and accreditation.**

# Hacker HighSchool

## SECURITY AWARENESS FOR TEENS



### LESSON 3 BENEATH THE INTERNET



HACKING IS LEARNING  
[www.hackerhighschool.org](http://www.hackerhighschool.org)

ISECOM

Creative Commons 3.3 Attribution-Non-Commercial-NoDerivs ISECOM

[WWW.ISECOM.ORG](http://WWW.ISECOM.ORG) - [WWW.OSSTMM.ORG](http://WWW.OSSTMM.ORG) - [WWW.HACKERHIGHSCHOOL.ORG](http://WWW.HACKERHIGHSCHOOL.ORG) - [WWW.BADPEOPLEPROJECT.ORG](http://WWW.BADPEOPLEPROJECT.ORG) - [WWW.OSSTMMTRAINING.ORG](http://WWW.OSSTMMTRAINING.ORG)



## WARNING

The Hacker Highschool Project is a learning tool and as with any learning tool there are dangers. Some lessons, if abused, may result in physical injury. Some additional dangers may also exist where there is not enough research on possible effects of emanations from particular technologies. Students using these lessons should be supervised yet encouraged to learn, try, and do. However ISECOM cannot accept responsibility for how any information herein is abused.

The following lessons and workbooks are open and publicly available under the following terms and conditions of ISECOM:

All works in the Hacker Highschool Project are provided for non-commercial use with elementary school students, junior high school students, and high school students whether in a public institution, private institution, or a part of home-schooling. These materials may not be reproduced for sale in any form. The provision of any class, course, training, or camp with these materials for which a fee is charged is expressly forbidden without a license, including college classes, university classes, trade-school classes, summer or computer camps, and similar. To purchase a license, visit the LICENSE section of the HHS web page at <http://www.hackerhighschool.org/licensing.html>.

The Hacker Highschool Project is an open community effort and if you find value in this project, we ask that you support us through the purchase of a license, a donation, or sponsorship.



## Table of Contents

WARNING.....	2
Contributors.....	4
Introduction and Objectives.....	5
Basic Concepts of Networking.....	6
Devices.....	6
Topologies.....	6
Game On: Leaving the Back Door Open.....	7
The TCP/IP (DoD) Model.....	9
Layers.....	9
Application.....	9
Transport.....	10
Internetwork.....	10
Network Access.....	10
Feed Your Head: See "The OSI Model" .....	10
Protocols.....	10
Application layer protocols.....	11
Transport layer protocols.....	11
Internet layer protocols.....	11
Internet Control and Management Protocol (ICMP).....	11
IPv4 Addresses.....	12
Classes.....	13
Loopback Addresses.....	15
Network Addresses.....	15
Broadcast Addresses.....	15
Ports.....	15
Encapsulation.....	17
Feed Your Head: The OSI Model.....	21





## Contributors

---

Marta Barceló, ISECOM

Pete Herzog, ISECOM

Chuck Truett, ISECOM

Bob Monroe, ISECOM

Kim Truett, ISECOM

Gary Axten, ISECOM

Marco Ivaldi, ISECOM

Simone Onofri, ISECOM

Greg Playle, ISECOM

Tom Thomas, ISECOM

Mario Platt

Ryan Oberto, Johannesburg South Africa

Vadim Chakryan, Ukraine

Peter Houppermans

**ISECOM**



## Introduction and Objectives

---

In the far depths of the past, before there was an Internet, electronic communication was pure voodoo. Every computer manufacturer had their own idea about how machines should talk over a wire. And nobody even considered the possibility that a Wang computer might communicate with a Burroughs machine.

The world changed when scientists and students experienced the joy of using a terminal to access a mainframe computer. The famous IBM PC arrived, and quickly owners wanted to access that mainframe from their personal computer. Soon modems were making dial-up connections and users were working in terminal emulators. Networking had graduated to a Black Art, and the insiders were called (really) **gurus**.

The world shifted again dramatically when the Internet, which started as a military project, was opened to the public. Networking had always been local, meaning confined to one office or at most one campus. How were all these different systems going to talk?

The answer was “wedge” a universal address system into existing networks, a system we generally call **Internet Protocol (IP)**. Think about it this way: imagine your friend overseas sends you a package. That package may travel by plane, train or automobile, but you don't really need to know the airline schedule or the location of the nearest train station. Your package will eventually arrive at your street address, which is ultimately the only thing that matters. Your **IP address** is a lot like this: packets may travel as electrons, beams of light or radio waves, but those systems don't matter to you. All that matters is your IP address, and the IP address of the system with which you're talking.

One thing that complicates this idea in the real world is that more than one person may be living at a single address. In the networking world, that's what's happening when one server provides for instance both regular HTTP and secure HTTPS, as well as FTP. Notice the P at or near the end of those acronyms? That's always a dead giveaway for **protocol**, which is just another way of saying “a type of communication.”

This lesson will help you understand how protocols and their ports work in Windows, Linux, and OSX. You'll also become familiar with several utilities (some of which have already been introduced in the previous lesson) that explore your system's networking capabilities.

At the end of the lesson you should have a basic knowledge of:

- the concepts of networks and how communication takes place
- IP addresses
- ports and protocols

## Basic Concepts of Networking

The starting point for networking is the local area network (**LAN**). LANs let computers in a common physical location share resources like printers and drive space, and **administrators** control that access. Sections below describe common network devices and topologies.

### Devices

Going forward in your career as a hacker, you're going to see a lot of network diagrams. It's useful to recognize the most common symbols:



**Figure 3.1:** Common Network Symbols

A **hub** is like an old-fashioned telephone party line: everybody is on the same wire, and can hear everyone else's conversations. This can make a LAN noisy fast.

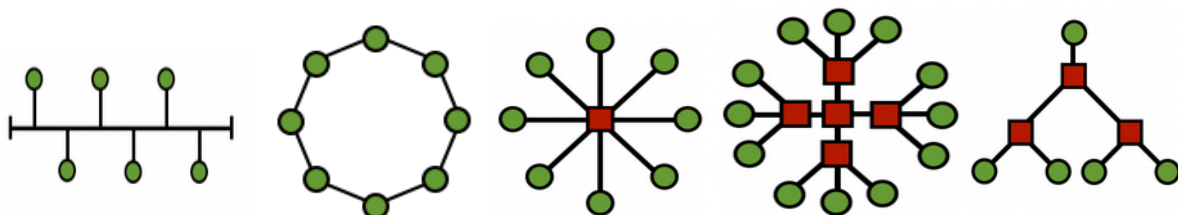
A **switch** is better: it filters traffic so that only the two computers talking to each other can hear the conversation. But like a hub, it's used only on a LAN.

A **router** sits between LANs; it's used to access other networks and the Internet, and it uses IP addresses. It looks at the packets being sent and decides which network those packets belong on. If the packet belongs on the "other" network, like a traffic police, it sends the packet where it belongs.

### Topologies

A **topology** is just another way of saying "the way we connect it". The kind of decisions we make regarding our topology can affect positively as well as negatively in the future, depending on technologies being used, technological and physical constraints, performance and security requirements, size and nature of the organization, etc.

A LAN's physical structure can look like any of the following physical topologies:



**Bus****Ring****Star****Extended Star****Hierarchic****Figure 3.2:** Topologies

In a **bus** topology, all the computers are connected to a single cable, and each computer can communicate directly with any of the others. But break any part of the bus, and everyone's off the network.

In the **ring** configuration, each computer is connected to the following one, and the last one to the first, and each computer can only communicate directly with the two adjacent computers.

Bus topologies are rarely used nowadays. Ring technologies are often used at the inter-state level, usually with two counter rotating rings, sending traffic in opposite directions, for reliability and fault tolerance.

In the **star** topology, none of the computers are directly connected with others. Instead they are connected through a hub or switch that relays information from computer to computer.

If several hubs or switches are connected to each other, you get an **extended star** topology.

In a star or extended star topology, all the central points are **peers**, that is, they're essentially equals. This is the most common LAN topology today.

However, if you connect two star or extended star networks together using a central point that controls or limits traffic between the two networks, then you have a **hierarchical** network topology. This is the topology usually deployed in bigger enterprises.

### Game On: Leaving the Back Door Open

In the sun-baked heat of the summer, Jace was happy to help the local air-conditioned police department set up their small network. They paid her with cookies, time away from the heat, conversation and the opportunity to install backdoors. Crawling under steel work desks that hadn't been moved in decades, Jace had found the cruddiest hidden spot to hide a wifi access point. She'd just plugged it in and sprinkled trash on top of it, and was dragging a roll of Ethernet cable to the wall ports she'd installed earlier.

A heavy hand slapped the desk above her. Jace kicked metal and yelled "Ow! My head!" then added, "you sure you don't want me to set up your server?"

The cop cleared his throat and tried to put on a Dork Professor voice. "Well I would, but I'm not sure how the flux ray resistor would hold up to the micro-channel cross feeds. Especially when the full moon falls on the last Tuesday of the month."

Jace flapped her feet in mock teen irritation. "Apparently you have no problem achieving quantum levels of baloney. And when do I get my cookies, Officer Kickam?"

"Jace please, please call me Hank. You make me feel like an old man when you call me Officer Kickam." He tried to sound hurt but she knew social engineering when she heard it: he was really trying to distract her from the cookies.

"Hank, hate to break the news to you, but you *are* an old man."

"Ouch, that hurt. I am not old, I am distinguished," he countered, considering his highly polished black police shoes as Jace's tattered sneaks disappeared under the



heavy desk. Then cinnamon brown eyes and a face covered in spider webs emerged. Jace still had a reel of cable under one arm. Hank helped her up and brushed the bug webs off her face and shoulders.

"Help, police brutality," Jace teased.

"Hostile criminal," Hank returned. "So educate me on your diabolical plan here," the hairy muscled lawman asked in what almost sounded to Jace like a pleading tone.

That felt good, so she asked, "Are you sure you want to know about this networking stuff?" He nodded eagerly. Jace thought: *bobble head*.

"Okay, what I did was design a network topography, like a map that shows where all the equipment, computers, hubs, jacks, switches, routers and firewalls will go. You can't start a project like this without a map," she said, glancing up at the cop. "It's all about making sure every node can talk to every other node, with no single point of failure. So, like, a bus architecture stinks, because if one node in the bus goes down, everyone else does too." Hank nodded so Jace continued.

"Think like networking is this cop shop, uh, police station, and someone just brought in a suspect. Every cop deserves their fair turn to beat up on the guy without hogging someone else's time. If the victim, I mean suspect, is moved to another cell, all the cops who still need to beat on the dude have to know where he went."

"Oh Jace, you're gonna start looking like you need a good beating too if you keep talking like that about us peace officers." Hank pulled up his gun belt and sucked in his slight belly.

Jace snorted a laugh. "So the suspect is a data packet and you police thugs are the network devices. And every device, a switch, a router, firewall, another server or whatever, needs to know that the data packet gets dealt with. You know, pummeled with police batons. I think you called it giving someone a wood shampoo."

Hank rolled his eyes and groped for the baton that he didn't have on him.

Giggling, Jace brought up the reel of cable like a shield. "Hey, I've got a spool of wire and I'm not afraid to use it. Put down the cup of coffee and nobody gets hurt." Off balance and laughing, Jace flopped onto Hank, who didn't budge. *Wow, this guy is a total rock*, she realized. The hand he laid on her shoulder reminded her of ... something.

She stood a little too quickly, reddening. "So there's two kinds of devices. Smart devices and dumb ones. Just like cops." Four approaching uniforms appeared at exactly the wrong time to hear the "dumb ones, just like cops." Lamely Jace continued, "Smart devices remember everything they do. They keep logs of their activities."

"And the dumb ones? Like cops?" asked the Chief of Police.

### Game Over

## The TCP/IP (DoD) Model

TCP/IP was developed by the **DoD (Department of Defense)** of the United States and **DARPA (Defense Advanced Research Project Agency)** in the 1970s. TCP/IP was designed to be an open standard that anyone could use to connect computers together and exchange information between them. Ultimately, it became the basis for the Internet.

Generally, the simplest form of the TCP/IP model is called the **DoD Model**, and that's where we'll start.

### Layers

The simple DoD model defines four totally independent layers into which it divides the process of communication between two devices. The layers that pass information are:

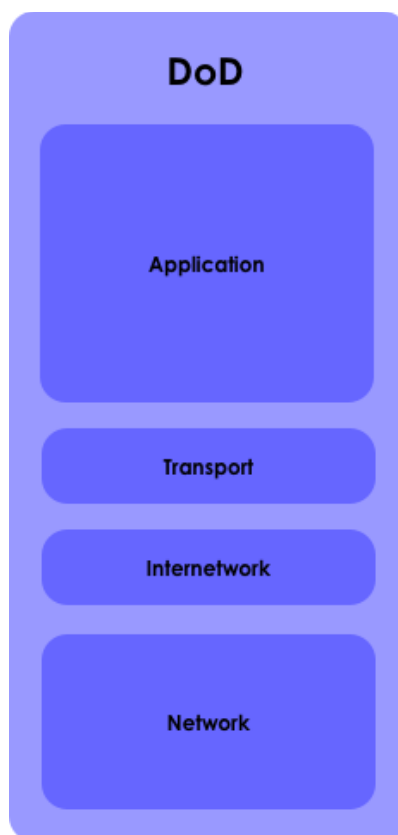


Figure 3.3: The DoD Model

### Application

The application layer is exactly what you probably think it is: the layer where applications like Firefox, Opera, email clients, social networking sites, instant messaging and chat applications work. Actually quite a few applications access the Internet: some office applications, for instance, connect to online collections of clip art. The application layer creates the payload that all the other layers carry. A good analogy is a postal system. The application creates the package that it wraps with instructions on how the package should be used. Then it hands off the package to the mail room: the Transport layer.



## Transport

The transport layer sets up network connections, which are called **sessions**. In the world of the Internet, the primary protocol at the Transport layer is **TCP, the Transmission Control Protocol**. TCP adds another “wrapping” to the outside of the package, with instructions about which package it is (say, 1 of 3), how to make sure the package got there, and whether the package is intact.

Let's say you're going to email a letter to your mother. The letter may be tiny or huge, but it's too big to send over the Internet in one piece. Instead, TCP breaks up that letter into **segments**, small chunks that are consecutively numbered, with a little bit of error-checking code at the end. If a packet gets corrupted in transit, TCP requests a retransmission. At the receiving end, TCP puts the pieces back together in the correct order and your mother gets the letter in her email.

But don't forget that TCP isn't the only game in town: **UDP** also functions at this layer, and in particular it does NOT create sessions. It just shoots a stream of **datagrams**, which are similar to segments, but UDP never checks if you've received it.

Whether TCP or UDP, all traffic is assigned to specific **port numbers** at this layer.

## Internetwork

This layer adds information about the source and destination addresses, and where the **packet** begins and ends. It's like a delivery company that gets packages to the correct address. It doesn't care if all the packets make it, or if they are intact; that's the Transport layer's job. The major protocol at this level is, appropriately, **IP (Internet Protocol)**. And this is the layer that uses IP addresses to get packets to the right place by the best route.

## Network Access

This layer is the low-level physical network that you use to connect to the Internet. If you're dialing up, we're sorry, and you're using a simple **PPP** connection. If you have **DSL** you may be using **ATM** or **Metro Ethernet**. And if you have cable Internet you're using a **DOCSIS** physical network. It doesn't matter what kind you use, because TCP/IP makes everything work together. The network access layer consists of the Ethernet cable and **network interface card (NIC)**, or the wireless card and access point. It handles the lowest level ones and zeroes (bits) as they go from one point to another.

### Feed Your Head: See “The OSI Model”

*See “The OSI Model” at the end of this lesson for an alternative take on network modeling.*

## Protocols

So now you're connected to the Internet. That seems simple enough, but consider the usual situation you're in: you are conducting innocent, important research on the Internet, while your dear brother or sister is wasting time streaming a movie. Why don't these two streams of traffic get mixed up? How does the network tell them apart?

The answer is **protocols**, which are like languages that different kinds of traffic speaks. Web traffic uses one protocol, file transfers another one, and email a different one still. Like all things digital, protocols don't really use names down at the network level; they use IP addresses and **port numbers**.



### Application layer protocols

**FTP** or *File Transfer Protocol* is used for the transmission of files between two devices. It uses one port to deliver data, and another port to send control signals ("I got the file! Thanks!"). The most commonly used ports are 20 and 21 (TCP).

**HTTP** or *Hyper-Text Transfer Protocol* is used for web pages. This traffic usually uses TCP port 80. **HTTPS** is a secure variant that encrypts network traffic, usually on TCP port 443.

**SMTP** or *Simple Mail Transfer Protocol* is the protocol that sends email. Its TCP port is 25.

**DNS** or *Domain Name Service* is how a domain like ISECOM.org gets mapped to an IP address like 216.92.116.13. It uses port is 53 (UDP).

### Transport layer protocols

**TCP** and **UDP** are the two main protocols used by the transport layer to transfer data.

**TCP or Transmission Control Protocol** establishes a logical connection (a **session**) between two hosts on a network. It sets up this connection using the three-way handshake.

1. When my computer wants to connect with yours, it sends a **SYN** packet, which is basically saying, "Let's synchronize clocks so we can exchange traffic with timestamps."
2. Your computer (if it's going to accept the connection) responds with a **SYN/ACK** acknowledgment packet.
3. My computer seals the deal with an **ACK** packet, and we're connected.

But this happens only with TCP. Instead, **UDP or User Datagram Protocol** is a transport protocol that doesn't even care if you have a connection. It's like a fire hose: if you catch the stream you catch it, and if you don't, you don't. This makes UDP very fast, so it's useful for things like streaming voice and video, where missing a single frame doesn't matter much or online gaming, where missing a single frame doesn't matter much (depending on which side of the bullet you are).

### Internet layer protocols

**IP or Internet Protocol** serves as a universal protocol to allow any two computers to communicate through any network at any time. It's like the postal carrier who delivers mail; all it does is get packets to their destination address.

### Internet Control and Management Protocol (ICMP)

**ICMP** is the protocol that the network devices and network administrators use to troubleshoot and maintain the network. It includes things like **ping** (Packet InterNet Groper) and similar commands that test the network and report errors. Because people have used things like ping floods to bring down hosts and networks, most systems limit ICMP to one response per second.

To summarize, ports and protocols come together like this:



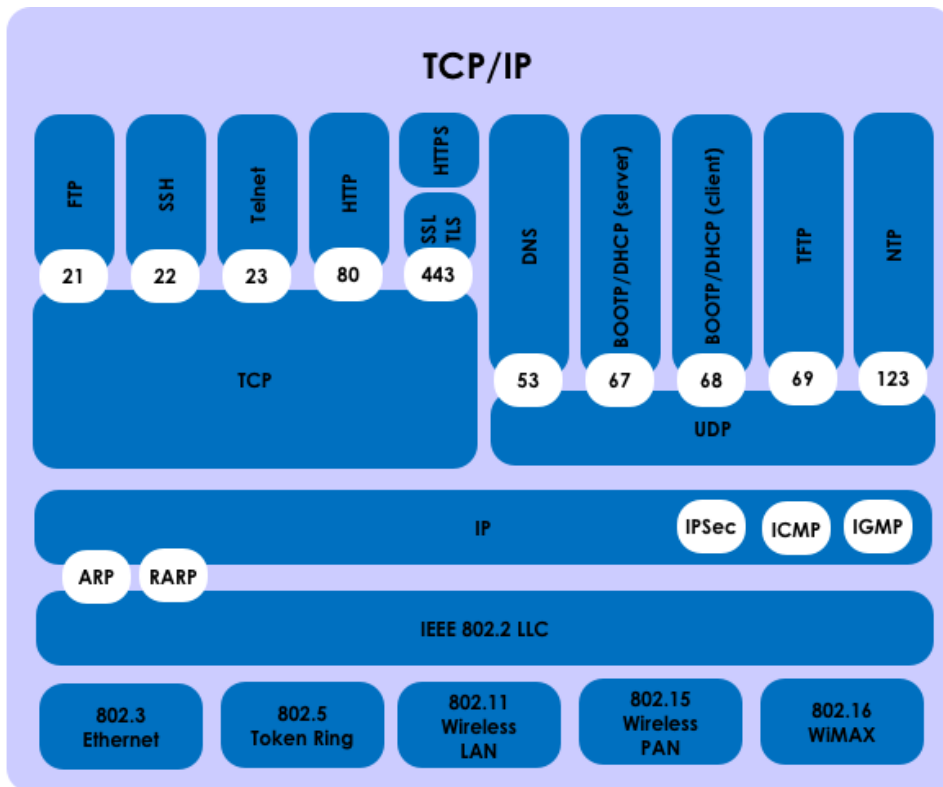
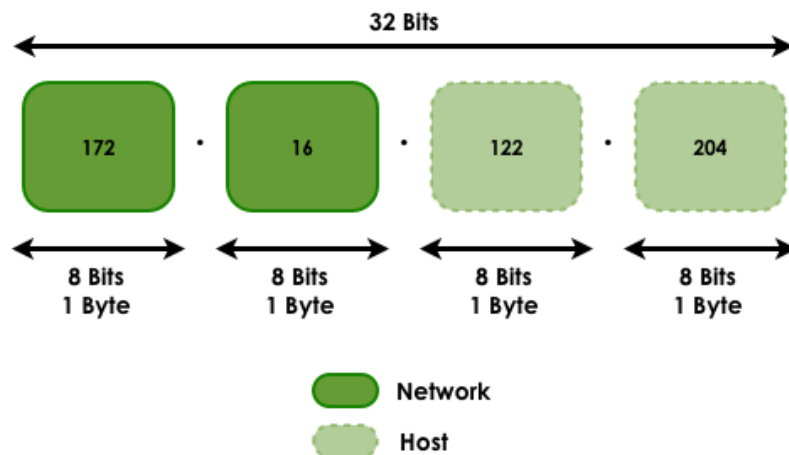


Figure 3.4: The TCP/IP Stack

## IPv4 Addresses

Domain names are handy for humans, because we're good at remembering names like ISECOM.org. But networks don't actually understand them; they only understand numeric IP addresses. So when you ask for ISECOM.org, your computer does a quick lookup using **DNS (Domain Name Service)** to find the corresponding IP address.

IP addresses are like street addresses. If you want mail, you have to have one. **IPv4** addresses consist of 32 bits that are divided in four 8-bit **octets**, which are separated by dots. This means that there are  $2^{32}$  (or 4,294,967,296) unique addresses on the Internet under IPv4. Part of the IP address identifies the network, and the remainder of the IP address identifies the individual computers on the network. Think of these parts as the country/city (network) portion of the address and the street (host) portion of the address.



**Figure 3.5:** Network Number and Host ID

Returning to the postal service analogy: IP is the delivery truck that gets the packet to the correct post office. TCP is the outer wrapper with the list of how many packages are in a shipment, and which one this is (say, number 3 of 65). The host-level addresses are the particular house (computer) for which the packet is destined.

There are both public and **private (non-routable) IP addresses**. Private IP addresses are used by private networks; routers won't allow these addresses onto the Internet.

IP addresses within a private network should not be duplicated within that network, but computers on two different – but unconnected – private networks could have the same IP addresses. The IP addresses that are defined by IANA, the Internet Assigned Numbers Authority, as being available for private networks (see RFC 1918) are:

10.0.0.0 through 10.255.255.255 (Class A)

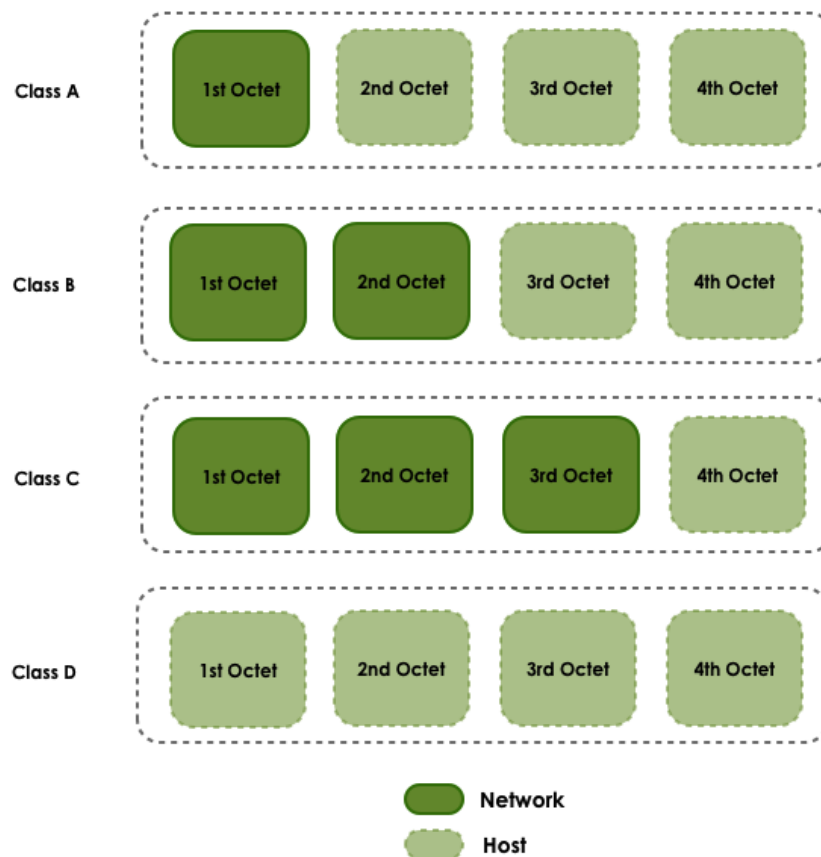
172.16.0.0 through 172.31.255.255 (Class B)

192.168.0.0 through 192.168.255.255 (Class C)

## Classes

IP addresses are divided into classes based on what portion of the address is used to identify the network and what portion is used to identify the individual computers.

Depending on the size assigned to each part, more devices will be allowed within the network, or more networks will be allowed. The existing classes are:



**Figure 3.5:** IP Class Divisions

**Class A:** The first bit is always zero, so this class includes the addresses between 0.0.0.0 (which, by convention, is never used) and 126.255.255.255. Note: the addresses of 127.x.x.x are reserved for the services of loopback or localhost (see below).

**Class B:** The first two bits of the first octet are '10', so this class includes the addresses between 128.0.0.0 and 191.255.255.255.

**Class C:** The first three bits of the first octet are '110', so this class includes the addresses between 192.0.0.0 and 223.255.255.255.

**Class D:** The first four bits of the first octet are '1110', so this class includes the addresses between 224.0.0.0 and 239.255.255.255. These addresses are reserved for group multicast implementations.

The remaining addresses are used for experimentation or for possible future allocations.

The **mask (or netmask)** is used to mark these class splits. Down in binary, a '1' bit shows the part containing the network identification and a '0' bit represents the part that identifies the individual host. The default netmasks for the first three classes are:

255.0.0.0 (Class A)

255.255.0.0 (Class B)

255.255.255.0 (Class C)



This is actually pretty slick, since networks that use default classes will mask one octet if they're Class A, two octets for Class B and three octets for Class C. Using default classes is handy – but not everyone's doing it.

What all this means is that to identify a host, you'll need both the IP address and a network mask:

IP: 172.16.1.20
Mask: 255.255.255.0

### Loopback Addresses

IP addresses 127.0.0.1 through 127.255.255.254 are reserved to be used as **loopback** or localhost addresses, that is, they refer directly back to the local computer. Every computer has a localhost address of 127.0.0.1, therefore that address cannot be used to identify different devices.

There are also other addresses that cannot be used. These are the **network address** and the **broadcast address**.

### Network Addresses

The network address is basically the network part of an IP address, **with zeroes where the host part would be**. This address cannot be given to a host, because it identifies the whole network, not just one host.

IP: 172.16.1.0
Mask: 255.255.255.0

### Broadcast Addresses

The broadcast address is basically the network part of an IP address, **with ones where the host part would be**. This address can't be used to identify a specific host, because it's the address that all hosts listen to (of course that's what broadcast means: everybody listens).

IP: 172.16.1.255
Mask: 255.255.255.0

### Ports

Both TCP and UDP use **ports** to exchange information with applications. A port is an extension of an address, like adding an apartment or room number to a street address. A letter with a street address will arrive at the correct apartment building, but without the apartment number, it won't get to the correct recipient.



Ports work in the same way. A packet can be delivered to the correct IP address, but without the associated port, there is no way to determine which application should act on the packet. A port number is also a 16 bit number, which means it can have decimal values between 0 and 65535 (2 to the power of 16).

Another way to think about this would be: every computer is a post office. Each application has its own post office box; no two applications should share the same post office box. The port number is that post office box number.

Port numbers make it possible to have multiple streams of information going to one IP address, where each one is sent to the appropriate application. The port number lets a service running on a remote computer know what type of information a local client is requesting and what protocol is used to send that information, all while maintaining simultaneous communication with a number of different clients.

For example, if a local computer attempts to connect to the website [www.osstmm.org](http://www.osstmm.org), whose IP address is 62.80.122.203, with a web server running on port 80, the local computer would connect to the remote computer using the **socket address**:

**62.80.122.203:80**

In order to maintain a level of standardization among the most commonly used ports, IANA has established that the ports numbered from 0 to 1024 are to be used for common, **privileged** or **well-known services**. The remaining ports – up through 65535 – are used for dynamic allocations or particular services.

The most commonly used (well-known) ports – as assigned by the **IANA** – are listed here:

Port Assignments		
Number	Keywords	Description
5	rje	Remote Job Entry
0		Reserved
1-4		Unassigned
7	echo	Echo
9	discard	Discard
11	sysstat	Active Users
13	daytime	Daytime
15	netstat	Who is Up or NETSTAT
17	qotd	Quote of the Day
19	chargen	Character Generator
20	ftp-data	File Transfer [Default Data]
21	ftp	File Transfer [Control]

Port Assignments		
22	ssh	SSH Remote Login Protocol
23	telnet	Telnet
25	smtp	Simple Mail Transfer
37	time	Time
39	rlp	Resource Location Protocol
42	nameserver	Host Name Server
43	nickname	Who Is
53	domain	Domain Name Server
67	bootps	Bootstrap Protocol Server
68	bootpc	Bootstrap Protocol Client
69	fftp	Trivial File Transfer
70	gopher	Gopher
75		any private dial out service
77		any private RJE service
79	finger	Finger
80	www-http	World Wide Web HTTP
95	supdup	SUPDUP
101	hostname	NIC Host Name Server
102	iso-tsap	ISO-TSAP Class 0
110	pop3	Post Office Protocol - Version 3
113	auth	Authentication Service
117	uucp-path	UUCP Path Service
119	nntp	Network News Transfer Protocol
123	ntp	Network Time Protocol
137	netbios-ns	NETBIOS Name Service
138	netbios-dgm	NETBIOS Datagram Service
139	netbios-ssn	NETBIOS Session Service
140-159		Unassigned
160-223		Reserved

## Encapsulation

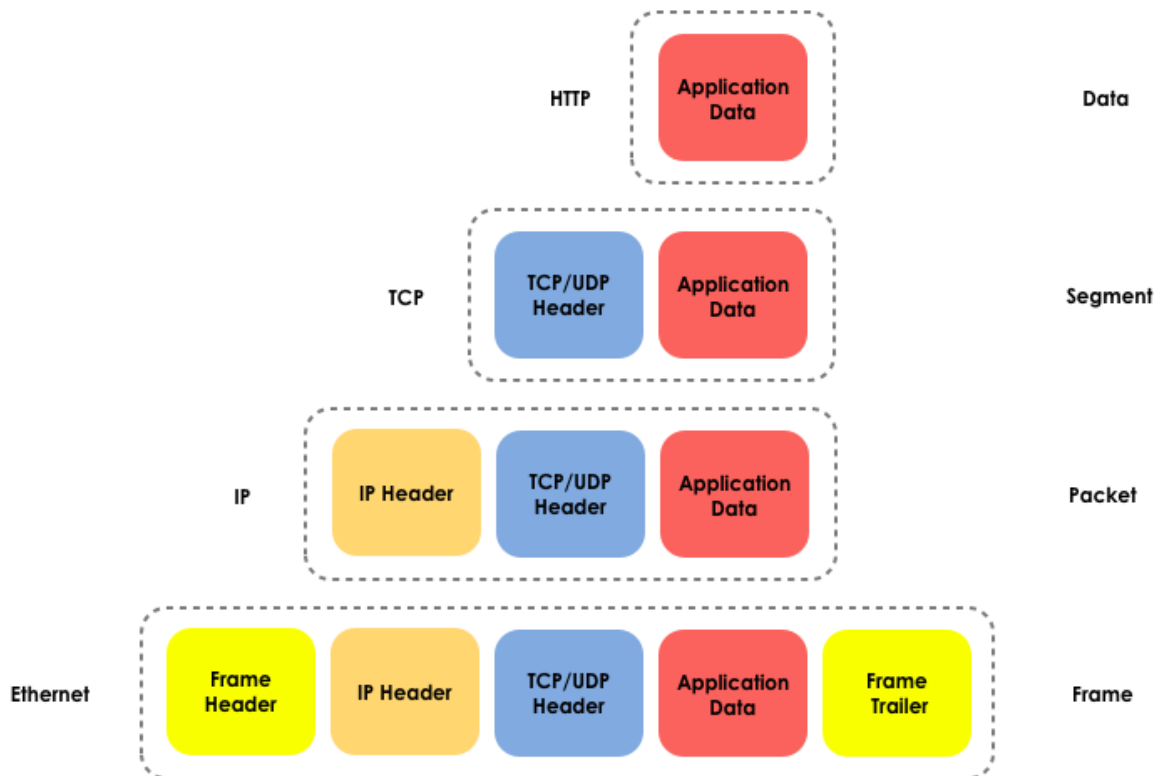
When a piece of information – an e-mail message, for example – is sent from one computer to another, it is subject to a series of transformations. The application layer generates the data, which is then sent to the transport layer.

The transport layer takes this information, breaks it into segments and adds a header to each one, which contains ports, unique number of the segment and other session information.

Then the segment is passed to Network layer where another header is added, containing the source and destination IP addresses and more meta-information.

The next layer, which in most local networks is supplied by Ethernet, adds yet another header, and so on. This procedure is known as **encapsulation**.

Each layer after the first makes its data an encapsulation of the previous layer's data, until you arrive at the final layer, in which the actual transmission of data occurs. So encapsulation looks like this:



**Figure 3.6:** Encapsulation

When the encapsulated information arrives at its destination, it must then be de-encapsulated. As each layer passes information to the next layer up the stack, it removes the information contained in the header placed there by that lower layer.

The final bit of information in this great addressing scheme is the absolutely unique address of the computer's NIC: the **Media Access Controller (MAC) address**. This address is usually displayed as six two-character **hexadecimal** numbers separated by colons or hyphens (dashes). It is the physical address of the network card and supposedly can't be changed (actually, there are ways to change it, but exactly how is for you to figure out). A MAC address looks like this:

00-15-00-06-E6-BF



## Exercises

3.1. Using commands you learned in Lessons 1 and 2, get your IP address, netmask, DNS hostname, and MAC address. Compare these with your partners. What seems similar and what is different? Given the IP address scheme the network is using, is this a private or a public network?

3.2. netstat

The **netstat** command tells you your network statistics: with whom you're connected, how long networking has been up, and so forth. In Linux, Windows or OSX you can open a command line interface and type:

```
netstat
```

In the CLI window, you'll see a list of established connections. If you want to see the connections displayed in numeric form, type:

```
netstat -n
```

To see the connections and the active (listening, open) ports, type:

```
netstat -an
```

To see a list of other options, type:

```
netstat -h
```

In the netstat output, look for the columns listing the local and remote IP addresses and the ports they're using:

```
Proto Recv-Q Send-Q Local Address          Foreign Address        (state)
tcp4      0      0 192.168.2.136.1043    66.220.149.94.443     ESTABLISHED
```

The ports are the numbers after the regular IP address; they may be separated by dots or colons. Why are the ports used by the remote address different from the ports used by the local address?

Open several browser windows or tabs to various websites, then run netstat again.

If there are several tabs open, how does the browser know which information goes to which tab?

Why is it that when a web browser is used, no listening port is specified?

What protocols are used?

What happens when one protocol gets used in more than one instance?

3.3. My First Server

To perform this exercise, you must have the **netcat (nc)** program. BackTrack includes it by default, as does OSX, but you can download installers for various operating systems.

**1.** In a CLI window, type:

```
nc -h
```





This displays the options that are available in netcat.

To create a simple server, In Linux or Windows type:

```
nc -l -p 1234
```

or in OSX type:

```
nc -l 1234
```

You just started a server listening to port 1234.

**2.** Open a second CLI window and type:

```
netstat -a
```

This should verify that there is a new service listening on port 1234.

To communicate with a server, you have to have a client! In your second CLI window type:

```
nc localhost 1234
```

This command makes a connection with the server that's listening to port 1234. Now, anything that is written in either of the two open CLI windows can be seen in the other window.

Consider the implications. How could someone abuse this capability to exploit your machine?

Netcat sends all its traffic in the clear. Is there a secure alternative?

**3.** Stop your server by going back to the first CLI window and typing Control-C.

**4.** Now create a simple text file named *test* containing the text, "Welcome to my server!"

Once you've done that, look at the following command and translate it for the instructor: what does each part do? Then, in your first CLI window, type:

```
nc -l -p 1234 < test
```

From another CLI window, connect to the server by typing:

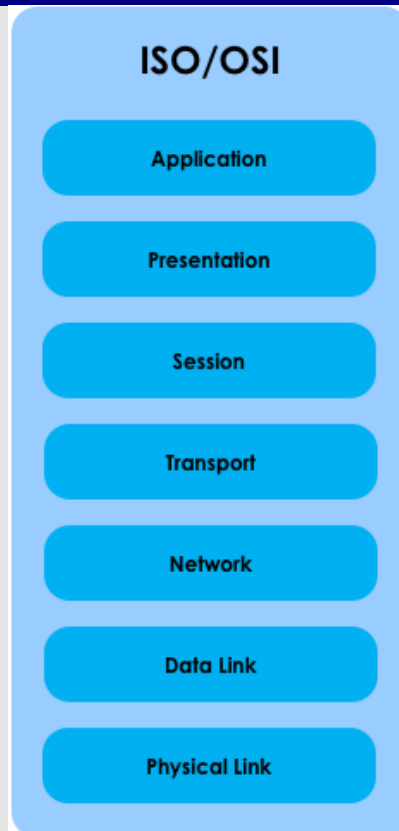
```
nc localhost 1234
```

When the client connects to the server, you should see the output of the file *test*.

What protocol has been used to connect with the server?

Does netcat allow you to change this? If so, how?

## Feed Your Head: The OSI Model



**Figure 3.7:** The ISO/OSI Model

The **OSI Model** was developed in the 1980s (about ten years after the TCP/IP Model) by **ISO**, the International Standards Organization. OSI stands for **Open Systems Interconnection**, and it was an attempt to standardize networking architecture that came from an organization that wasn't really involved in the development of networking.

The OSI Model is a layered model with a handful of simple rules. Similar functions are grouped together in the same layer, and (please do not forget this) every layer is serviced by the layer **beneath** it, and serves the layer **above** it.

This layered model is a good idea, because since every layer (in theory) does its own communication, new developments in any one layer don't break any of the other ones. This feature alone may explain the Internet boom we've had since 2000, with new applications and services appearing almost each day.

Besides the two rules of this OSI model we've already discussed (similar functions are grouped, and every layer is serviced by the layer beneath it and serves the layer above it) this standard has one more strict rule. Every layer involved in communication from one computer communicates directly with the same layer on the other computer. This means that when you type `www.google.com` in your browser, there is a direct interaction between your computer's Layer 7 interface (your web browser) and Google.com's web servers (also a Layer 7 interface), and that the same can be said of any other layer.

So let's first define what are the OSI model layers and their respective responsibilities.

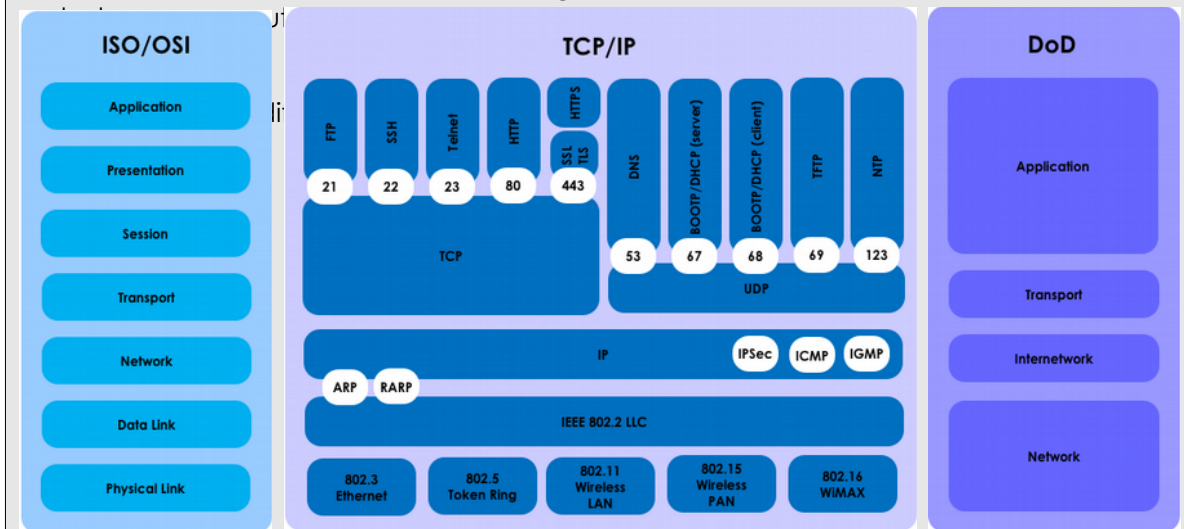


Application Layer	Responsible for direct interaction between applications and the user interface to the application, for instance the use of a web browser like IE or Firefox.
-------------------	--



Presentation Layer	Responsible for guaranteeing that data is exchanged in a way that is comprehensible between both parties. In some services that use a form of encryption, the encryption happens at the presentation layer.
Session Layer	Responsible for dialogue control between two computers. Basically it establishes, manages and terminates all connections that happen between the computers.
Transport Layer	Provides transparent transfer of data between computers, providing reliable data transfer services to the upper layers. This means that it is responsible for assembling all data in smaller portions that can be carried reliably on a data network. If a packet is lost or not received, it is the transport layer's job to make sure that that single packet is retransmitted and then reassembled in the correct order.
Network Layer	This layer is responsible for the addressing part of the connection. Not only on ensuring that each address is <i>unique</i> on the network, but also on making sure that whatever path is available (whether a good or a bad one), it always delivers the information where it needs to go, and that our information will be sent from hop to hop until it reaches its final destination.
Data Link layer	The data link layer was designed to deal with ensuring the physical layer can recover from errors that might happen and to deal with different connecting mediums. Basically it prepares (encapsulates) data so that it can be transmitted over whatever physical means are necessary (radio waves, fiber-optic cable, copper).
Physical layer	This layer defines the physical specifications of the devices and what needs to be done in order for the information to be transmitted over the selected medium. For a WiFi connection, this is a radio signal; for a fiber connection it's the light being sent; or for a copper connection the electronic signal being sent on the wire.

These seven layers comprise everything that is needed for the reliable communication





**Figure 3.8:** Networking Models Compared

Today's teens are in a world with major communication and productivity channels open to them and they don't have the knowledge to defend themselves against the fraud, identity theft, privacy leaks and other attacks made against them just for using the Internet. This is the reason for Hacker Highschool.

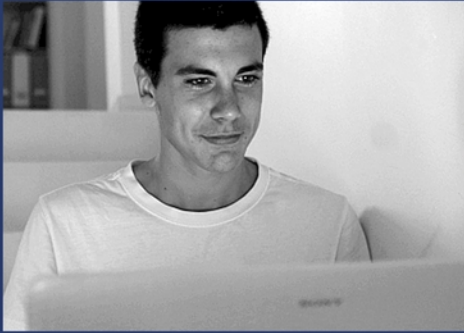
**The Hacker Highschool project is the development of security and privacy awareness learning materials for junior high and high school students.**

Hacker Highschool is a set of lessons and a practical means of making hackers. Beyond just providing cybersecurity awareness and critical Internet skills, we need to teach the young people of today how to be resourceful, creative, and logical, traits synonymous with hackers. The program contains free security and privacy awareness teaching materials and back-end support for teachers of accredited junior high, high schools, and home schooling. There are multiple workbooks available in multiple languages. These are lessons that challenge teens to be as resourceful as hackers, including safe Internet use, web privacy, researching on the internet, avoiding viruses and Trojans, legalities and ethics, and more.

**The HHS program is developed by ISECOM, a non-profit, open-source research group focused on security awareness and professional security development and accreditation.**

# Hacker HighSchool

## SECURITY AWARENESS FOR TEENS



### LESSON 4 PLAYING WITH DAEMONS



HACKING IS LEARNING  
[www.hackerhighschool.org](http://www.hackerhighschool.org)

ISECOM

Creative Commons 3.3 Attribution-Non-Commercial-NoDerivs ISECOM

[WWW.ISECOM.ORG](http://WWW.ISECOM.ORG) - [WWW.OSSTMM.ORG](http://WWW.OSSTMM.ORG) - [WWW.HACKERHIGHSCHOOL.ORG](http://WWW.HACKERHIGHSCHOOL.ORG) - [WWW.BADPEOPLEPROJECT.ORG](http://WWW.BADPEOPLEPROJECT.ORG) - [WWW.OSSTMMTRAINING.ORG](http://WWW.OSSTMMTRAINING.ORG)



## WARNING

The Hacker Highschool Project is a learning tool and as with any learning tool there are dangers. Some lessons, if abused, may result in physical injury. Some additional dangers may also exist where there is not enough research on possible effects of emanations from particular technologies. Students using these lessons should be supervised yet encouraged to learn, try, and do. However ISECOM cannot accept responsibility for how any information herein is abused.

The following lessons and workbooks are open and publicly available under the following terms and conditions of ISECOM:

All works in the Hacker Highschool Project are provided for non-commercial use with elementary school students, junior high school students, and high school students whether in a public institution, private institution, or a part of home-schooling. These materials may not be reproduced for sale in any form. The provision of any class, course, training, or camp with these materials for which a fee is charged is expressly forbidden without a license, including college classes, university classes, trade-school classes, summer or computer camps, and similar. To purchase a license, visit the LICENSE section of the HHS web page at <http://www.hackerhighschool.org/licensing.html>.

The Hacker Highschool Project is an open community effort and if you find value in this project, we ask that you support us through the purchase of a license, a donation, or sponsorship.





## Table of Contents

WARNING.....	2
Contributors.....	4
Introduction.....	5
Services.....	6
HTTP and the Web.....	6
Email – SMTP, POP and IMAP.....	8
IRC.....	10
FTP.....	11
Telnet and SSH.....	13
Game On: Command Me.....	14
DNS.....	15
DHCP.....	16
Connections.....	16
ISPs.....	17
Plain Old Telephone Service.....	17
DSL.....	17
Cable Modems.....	17
Wimax.....	18
Wifi.....	18
Feed Your Head: Playing With HTTP.....	19
Sniffing the Connection Between You and the HHS HTTP Server.....	19
Your First Manual Connection.....	20
The Request Method.....	22
Scripting HTTP requests with curl.....	24
References and Further Reading.....	25
Conclusion.....	27



## Contributors

---

Pete Herzog, ISECOM  
Marta Barceló, ISECOM  
Chuck Truett, ISECOM  
Kim Truett, ISECOM  
Marco Ivaldi, ISECOM  
Bob Monroe, ISECOM  
Jaume Abella, ISECOM  
Greg Playle, ISECOM  
Guiomar Corral, Barcelona  
Ashar Iqbal

**ISECOM**



## Introduction

There are thousands of different human languages and dozens of dialects within some of them. You may know several languages yourself, but the odds of you being able to travel across the world and speak to everyone you meet are slim to none.

Yes, you could argue that mathematics is a universal language or music speaks to everyone, but let's be realistic. Try and order a glass of soda with just a little bit of lemon and a scoop of ice cream using those "universal languages" and see how far you get.

If you happen to be in a country where you do not speak the language, please send ISECOM a video of yourself using bagpipes or a saxophone to order a soda. We really want to see that! We may not want to hear it, but we sure do want to see it.

But every day millions of people communicate with one another using a single common language over the Internet. Humans may not all speak the same language; however, our computers and networks do.

The model we use in contemporary networks is the **client-server model**. Physical computers (**hosts** or **servers**) offer **services** (in UNIX they're called **daemons: disk access and execution monitors** – now go impress someone). Think of a web server: it serves up a web page when you ask for it. No mystery.

But actually "you" don't request that page; your web browser does, meaning it's a **client** (or more formally, your computer is). And your computer can be a server, too, at the same time. That's the beauty of networking: you do this for me; I do that for you.

Multiply this model a million times, and you have the Internet. Consider this: millions of computers are offering some kind of service. What does it take to be a client? And is it possible to **subvert** all of this? (Go look up that word if you're not dead certain what it means. This is a hacker course, after all.)

Ready or not, let's dive in.



## Services

You have a computer, and you know that there's useful information on it, or you may participate in the common hallucination that you have nothing of digital value. You also know that other people, millions of other people, also have computers and their computers also may have useful information, not to mention handy resources like processors and RAM and disk space and bandwidth.

Now, you can assume that these other people, and these other computers, may very likely have information of interest to someone. The only problem is how to get at all that useful information.

Computers communicate with each other easily through ports, using the protocols we discussed in Lesson 3, but that doesn't really let you read the streams of binary data that computers exchange (unless you have some serious extra time on your hands). You need a way for your computer to get data, interpret it for you, and present it in some form you can use.

The way computers transfer data is through **network services**, or simply **services**. These services allow you to view web pages, exchange email, chat, and interact with remote computers. These services are mapped to port numbers.

Your computer, the **local computer**, uses programs called **clients** to interpret the information that you receive. You might receive information from a **server** (providing a service/running a daemon), over a **Tor** network, from **Torrent seeders** or over **peer-to-peer** networks.

Of course, your computer can provide services to other remote computers as well, meaning it's a data server or service provider. If you happen to have malware on your computer, you may be providing quite a few services that you don't know about.

Examples of clients include web browsers, email clients, chat programs, Skype, Tor clients, Torrent clients, RSS and so on. These are the applications in the **application layer** of the TCP/IP protocol stack. At the application layer, all the data that is transmitted, encapsulated, encrypted, decrypted, addressed and so forth by the lower layers is turned into something that you, the user, can read and understand.

## HTTP and the Web

When we talk about "the Internet," most people are actually thinking of the **World Wide Web**. The World Wide Web, or just the **Web**, isn't the Internet, it's just a small portion of the services available. Usually it just involves viewing web pages through a browser.

The actual Internet, by the way, consists of all the computers, routers, wires, cable and wireless systems that move all kinds of data around. Only a fraction of this is web traffic.



The web uses **HTTP** or **HyperText Transfer Protocol** and applications (clients) called **web browsers** to access documents on **web servers**. Information from the remote computer is sent to your local computer using the HTTP protocol, usually over port 80. Your web browser interprets that information and renders it on your local computer.

All browsers are not created equal. Each offers different tools and renders HTML content in slightly (or very) different ways. Security and privacy issues may be handled with various degrees of success. This means that you should know what your browser can and can't do, and what settings and plugins give you that perfect balance of security and privacy (unless you like malware, advertisements, spam and your neighborhood knowing that you like to watch kittens play in green jello).

The **hypertext** part of the HTTP protocol refers to the non-linear way you read it. You normally read in a linear fashion: page 1 then page 2; chapter 1 then chapter 2; lesson 1 then lesson 2, and so on. Hypertext lets you look at information in a non-linear way. You can jump around from topic to topic, learning as you go, then retrace your steps and maybe go see other information before you finish the parent article. That's the difference between hypertext and plain text.

In hypertext, words and ideas connect not only with the words that directly surround them, but also with other words, images, videos, and music. Hypertext isn't restricted to the Web. Most full-featured word processors let you create locally stored pages in web, or HTML, format. You read these pages in your web browser and they act like any other web page, only they are stored on your local computer, not a remote computer.

It's easy to create your own web page. The easiest way to do this is to use one of the common word processors like OpenOffice/LibreOffice Writer, Microsoft Word, or WordPerfect. These programs will allow you to produce simple web pages, combining text, hypertext and images. Plenty of people have made functional web pages using these (or even simple text editors like vi, found on most Unix platforms). Other text editors include Microsoft Notepad, Notepad++, SciTe, emacs and so on.

But these pages aren't flashy. Flashy means **CSS** and **scripts** and animations. You can spend lots of money on fancy web page design applications. These apps allow you to create interesting effects on your web page, but they're more complex to use. Still, they usually make the overall job easier. The lower-cost alternative is to get one of the text editors tailored for working with HTML and scripting languages, learn the syntax of HTML and scripting and code your own web pages from scratch.

Once you have the pages designed, you'll need a computer to put them onto, if you want other people to see them. **Internet Service Providers (ISPs)** provide **web hosting** on their web servers.

You can run a web server from your own home, using your own computer, but there are a whole handful of issues. Information stored on a web server is only available when that server's powered up, operating properly and has an open connection. So if you want to run a web server from your bedroom, you have to leave your computer on all the time; you have to make sure that the web server program is operating properly all the time (this includes troubleshooting hardware problems, controlling viruses, worms and other attacks, and dealing with the inevitable bugs and flaws within the program itself); and you have to keep a connection to the Internet open, which must be stable and fast. ISPs charge extra for a fast upload connection and a fixed IP address, which is why most people pay someone else to do all this.



A web hosting company stores your web pages on their computer. It's nice to let their servers be attacked, instead of yours. A good web hosting company will have multiple, redundant servers and a regular backup policy, so that your site doesn't disappear just because of hardware problems; a support staff keeps servers running despite attacks and program bugs; and a number of open connections to the Internet give some guarantee against outages. So all you have to do is design your web page, upload it to the hosting company's server, turn off your computer and go to sleep. Your web page will be available to the entire world, as long as you keep paying the bill.

It's also possible to find organizations that offer free web hosting. Some of these organizations are funded by paid advertising, which means that anyone who wants to view your web page will first have to view someone else's advertisement. But they won't have to buy anything, and you won't have to pay anything.

### Exercises

- 4.1 A web page is just text that tells the browser where images, videos, and other things should be. You can see what it looks like by viewing the Page Source. Go to your favorite browser, point it at ISECOM.ORG and load the page. Now view the source. You'll see some tags with the word "meta" in them. For example, the first is `meta-charset="utf-8"`. What does that mean? What is the point of it?
- 4.2 Find 3 more meta tags and explain what the point of them is. You may need to search around the web to figure it out so think carefully about what key words you will use in your search to make sure you get the right answers.
- 4.3 Save the ISECOM.ORG Page Source to your computer. Drag it to the browser. What changed? Why do you think it changed?
- 4.4 Open the ISECOM.ORG Page Source in a text editor and you'll see it's just words and numbers. Whatever you change or type in there will affect how the page looks when you save it and drag it back into the web browser. Delete things and you'll see it deleted. Change words and they will appear as you typed them. Now strip the page of anything else and add your name so it shows up larger and bolder than the other words. Try it. Save it. And drag it to the web browser and see if you were successful. No? Then keep trying!

See the **Feed Your Head: Playing With HTTP** at the end of this lesson for an opportunity to dig deeper.

### Email – SMTP, POP and IMAP

The second most visible aspect of the Internet is probably email. On your computer you use an email client, which connects to a mail server. When you set up your email account, you get a unique name in the form of **user@domain** and you have to create a password.

There are two parts to email: **SMTP (Simple Mail Transfer Protocol)**, which *sends* mail, and the mail server, either **POP (Post Office Protocol)** or **IMAP (Internet Message Access Protocol)**, which *retrieves* your email.



The SMTP protocol (we'll remind you again) is used to *send* email. SMTP defines the **fields** in an email message, including the FROM, TO, SUBJECT, CC and BODY fields. Plain old SMTP does not require a password and sends everything in clear text; everybody gets to read your mail. This may not have been bad when the protocol was designed and the Internet was a small world inhabited by like-minded people. But it left a loophole that allowed any user to send **spam** and do other nasty things like **email spoofing**, which basically means lying about (spoofing) the sending address. Almost all contemporary mail servers use Secure SMTP, which means you must prove your identity before you can send an email.

In later lessons we'll show you how spoofing works and how to look for it in email headers. This bit of knowledge can turn you from the sheep to the wolf awfully quickly.

**POP3 (Post Office Protocol version 3)** is a "store and dump" protocol. The email server receives your email and stores it for you, until you connect and download (dump) your email. Then your outgoing mail is sent using SMTP. This is a good way to approach email if you have a dial-up connection, since it takes less time to send and receive email, and you can read your email offline.

**IMAP**, on the other hand, by default stores your mail on the server. Many corporate email solutions use a form of IMAP, depending on their software vendors. In IMAP, you can create folders in your mailbox and move messages between these folders. When you connect to the IMAP server, your mailboxes and the server synchronize the folders, contents, incoming email and deleted email. This has quite an advantage: you can get to all your mail from any computer or device you use: laptop, kiosk, phone or tablet. Plus you can download and store email in personal data files on your own computer.

However, there are also two drawbacks: first, obviously, is that you need to exchange more information, so you need a faster connection and more time. The second is that space is limited. Your mail server will assign you a mailbox size that you can't exceed. If you run out of space, you won't be able to receive messages unless you delete emails (or pay for more space). Ultimately this means that corporate IMAP email requires data management. You have to move mail to local storage and clean out your sent mail, spam, and trash on a regular basis to conserve space. Mail with attachments will destroy you. In this age of free Internet email accounts with huge free data storage, all this upkeep may seem stupid. Until you get sued. Or someone compromises the mail server and gets ALL your email.

Both POP and IMAP servers require a password to access your account. But both protocols send *everything* in the clear, including passwords, so anyone can potentially read them. You have to use some form of encryption to mask the login process (like SSL) and the contents of the mail. That's why many email clients have a *Use SSL* check box.

When you click the Send button in your email client, two things happen: first your client forces you to log in to the SMTP server (even though you've already logged in to the POP server, dang it!), and then it sends your outgoing mail (via the SMTP protocol).

This got annoying by the mid-1990s, when servers began to use a protocol called **POP-before-SMTP**: you first send the POP server your user name and password, then your incoming mail downloads, then the SMTP server checks you against the POP server ("Is this guy okay?" "Yeah, I authenticated him.") and sends your messages. It's a nice time-saver.

One important item to remember is, despite being password protected, email is not a way to send secure information. Most POP clients and servers require that your password be communicated – unencrypted – to your mail server. This doesn't mean that anyone who receives an email from you also receives your password; but it does mean that someone with the right knowledge and tools can sniff out your password – as well as the contents of your emails. (For ideas on making your email more secure, see **Lesson 9: Email Security**.)



## Exercises

- 4.5 Send yourself an email from your main account, to your main account. Send the same email to your same main account from another account, for instance a free online account (come on, we know you have them). How long do the two messages take to arrive? If there is a difference, why?
- 4.6 Look at one of the billions of spam emails that clog your inbox. Can you determine who actually sent you a particular email? Is there any kind of hidden information in emails, for instance? If there is, how can a clever hacker see it?
- 4.7 Can you delay the sending of an email until a certain time or day (which can really screw up Deniability)? Can you think of cool way to use email sending delays to mess with your friends?

## IRC

**IRC (Internet Relay Chat)** is a great place to see the unregulated nature of the Internet at its best. Or worst. On IRC, anyone with anything to say gets a chance to say it. IRC is also known as **Usenet** or **news groups**. Each news group has its own name, sub-name, sub-sub-name and so forth.

You may be familiar with chat rooms. IRC is just like a chat room, only there are no rules beyond basic **netiquette**, and quite often there are no chaperones. You may find exactly what you are looking for on an IRC channel, or you may find something you never knew existed.

All the rules that you've heard about chat rooms are applicable to IRC channels. Don't tell anyone your real name. Don't give out your phone number, your address, or your bank account numbers. But have fun! If you are roaming around, be careful about the content available. Not everything on the Internet is malware-free, and not everyone on the Internet is nice.

IRC is not secure and everything you type is passed in clear text from IRC server to IRC server. You can set up private conversations between yourself and another IRC member but those are transmitted in the clear as well. Using a nickname will only get you a little privacy. If you're planning on conducting any malicious or unsavory actions, don't use the same nickname for every account. Using the same nickname is an excellent way of getting tracked down by the police. Or much less savory people.

Topics are called "channels." Since there are thousands of channels, we'll give you a URL that lists many of them for you to pursue until you lose your mind:

`http://www.nic.funet.fi/~irc/channels.html`

If you are having problems with the comments made by another member, you can either report them to the moderator (if there is one), or have that person **bumped** from that channel. If you don't want to hear what someone has to say, you can always block them or ignore their messages. Maybe that thread's not for you anyway.





## Exercises

- 4.8 Find three IRC channels that focus on security topics. How do you join in the public conversation? What do you have to do to have a private conversation with a person?
- 4.9 What port does IRC use?
- 4.10 It is possible to exchange files through IRC. How do you do this? Would you want to exchange files through IRC?
- 4.11 What's the major difference between MIME and SMIME? When you see an "S" in an acronym, does that mean anything special to you as a *Secure* (hint, hint) minded person?

## FTP

Old-school **File Transfer Protocol (FTP)** typically runs over ports 20 and 21. Guess what: it lets you transfer files between two computers. While it can be used for private file transfers, because it doesn't use encryption it's more commonly used for free, anonymous FTP servers that offer public access to collections of files, like the ISO for that cool new Linux distro.

Anonymous FTP was once the main way for computer users to exchange files over the Internet. While there are plenty of anonymous FTP servers used to distribute files illegally (which is a nice way to spread binary disease), more are legally used to distribute programs and files. You can find servers that offer anonymous FTP services through the usual methods, for instance search engines. But remember: FTP logins are sent in clear-text. Yes, even though we're talking about a user name and password. (Is that lame or what?) There is secure FTP (SFTP) but it's not universally used.

Most anonymous FTP servers allow you to access their files using the FTP protocol through a web browser. There are also some really great FTP clients that work like a file management program. Once you're logged into the FTP server, you can move files onto your computer in much the same way you move files on your own computer. FTP just takes a bit longer to download each file over your computer, mainly because the FTP server may be located on the other side of the planet.

## Exercises

- 4.12 Windows, OSX and Linux come with a basic command line FTP client; to access it, open a command prompt or terminal window and type:

```
ftp
```

At the `ftp>` prompt, you can type `help`, to get a list of available commands.

```
ftp> help
```

Commands may be abbreviated. Commands are:

!	delete	literal	prompt	send
?	debug	ls	put	status
append	dir	mdelete	pwd	trace
ascii	disconnect	mmdir	quit	type
bell	get	mget	quote	user
binary	glob	mkdir	recv	verbose
bye	hash	mls	remotehelp	



```
cd                help                mput              rename
close            lcd                  open              rmdir
```

### The basic commands are:

Connect to the FTP server named *ftp.domain.name*:

```
ftp> open ftp.domain.name
```

List the contents of the remote working directory:

```
ftp> ls
```

or

```
ftp> dir
```

Change the remote working directory to a directory named *newdir*:

```
ftp> cd newdir
```

Download a file named *filename* from the remote computer to the local computer:

```
ftp> get filename
```

Download multiple files named *file1*, *file2*, and *file3* from the remote computer to the local computer (you can also use wildcards to download many files with the same suffix, or all the files in a directory):

```
ftp> mget file1 file2 file3
```

Uploads a file named *filename* from the local computer to the remote computer:

```
ftp> put filename
```

Disconnect from the remote FTP server:

```
ftp> close
```

Shut down your local FTP client:

```
ftp> quit
```

### An FTP session, step by step

To connect to an anonymous ftp service, first open your local FTP client:

```
ftp
```

Use the open command to connect to the server. The command

```
ftp> open anon.server
```

connects your FTP client with the anonymous FTP server named *anon.server*. Substitute the name of a real server, of course.

When the remote FTP server accepts your connection, it will identify itself to your local client, then ask for a user name.

```
Connected to anon.server.
```

```
220 ProFTPD Server (Welcome . . . )
```

```
User (anon.server:(none)):
```

For most anonymous FTP servers, you should enter in the word *anonymous* (or *ftp*) as the user name. The remote FTP server will acknowledge that you are connecting as an anonymous user, and will give you instructions on what to use as a password.



```
331 Anonymous login ok, send your complete email address as your
password.
```

```
Password:
```

In most cases, the remote server does not check the validity of the email address entered as a password, so it will not stop you from accessing the service if you enter an invalid address. This is considered to be a breach of netiquette, but it's actually necessary: do not give away your real email address! After you have entered a password, the remote server will send a welcome message to your local computer.

```
230-
```

```
Welcome to ftp.anon.server, the public ftp server of anon.server. We
hope you find what you're looking for.
```

```
If you have any problems or questions, please send email to
ftpadmin@anon.server
```

```
Thanks!
```

```
230 Anonymous access granted, restrictions apply.
```

From here, you can use the `ls`, `dir`, `cd` and `get` commands to download files from the remote server to your local computer.

### Exercises

- 4.13 Using these examples, find and download a file from an anonymous FTP server.
- 4.14 Use your web browser and a search engine to find an anonymous FTP server that has a copy of *Alice in Wonderland*, then, using the command line FTP client – not your web browser – download the file.
- 4.15 What are the better FTP clients out there? Can they automate all the command-line stuff and provide you a nice graphical interface? Do you lose any functionality you have at the command line?
- 4.16 Could your computer become an FTP server?

### Telnet and SSH

**Telnet** allows a local user to send a wide variety of commands to a remote computer. This allows the local user to instruct the remote computer to perform functions and return data to the local computer, almost as if you were sitting at a keyboard in front of the remote computer. **Secure Shell (SSH)** is intended as a secure, encrypted replacement for clear-text telnet.

Again, most Windows, OSX and Linux versions come with a basic, command line telnet client; to access it, open a command prompt or terminal window and type:

```
telnet
```

To access a telnet server, you will need to have an account and password set up for you by the administrator of the server, because the telnet program lets you do lot of things, some of which could severely compromise the remote computer.

Telnet was used back in the day to allow computer administrators to remotely control servers and to provide user support from a distance. This service is part of the old Internet and isn't used much anymore.



Telnet can also be used for a number of other tasks, such as sending and receiving email and viewing the source code for web pages (although telnet is probably the most difficult way to do these things). Many of these things are legitimate, but they can be abused for illegal or immoral reasons. You can use telnet to check your email, and view, not just the subject line, but the first few lines of an email, which will allow you to decide whether or not to delete the email without downloading the entire message.

If you are going to use SSH, be sure you're using a current version, because older versions had various vulnerabilities, and many automatic vulnerability scanners constantly search for these on the Internet.

### Game On: Command Me

The dark screen twinkled across the front of Grandpa's thick glasses as the cursor blinked impatiently, expecting a command. With his gray thin hair lapped lazily over his wrinkled head, Grandpa tapped the keyboard. Jace watched the silent piano player play the keys of his computer, tap, tap, tap, tap. He smiled at Jace with his head turned to look into her young eyes. "Jace, I'm going to show you a new world out there. Buckle your seatbelt," he winked at the eight year old.

Jace's feet barely reached the floor in the computer chair with her grandfather behind the computer screen. She heard the telephone dial tone coming from a small box close by. The white box lit up with green and red lights as the tone changed to a sound like a duck being swallowed by a garbage disposal. Grandpa raised his excited eyebrows and stared with all of his might at the black screen in front of him. The duck stopped wailing and all of the lights turned green on the telephone box.

Grandpa said, "Watch this."

Usually when Grandpa said, "watch this" something would explode or black smoke would come from something. Either way, "watch this" meant Grandpa was going to mad about something he did wrong. Jace loved to hear those words though, because it was exciting anticipation for some wonderful event.

The computer screen woke up from its dark slumber with a banner of ASCII text surrounding the words "Welcome to Cline's Bulletin Board System (BBS)." "We're in," Grandpa clapped and attempted to high-five eight year old Jace. His hands missed her hand by several inches and he almost smacked the little girl in her face. She laughed, and Grandpa did too.

They both looked back and forth at the keyboard and the computer screen. Grandpa rubbed his fingers together while Jace rubbed her mind, trying to figure out what was going on. Grandpa began typing commands on the silent piano, head bowed down over the keys as a vulture might eat roadkill. Head up, head down, head up, head. Oops. He sat back. Grandpa had forgotten something very important.

He paused and spoke like a teacher. "Jace, I'm sorry but I forgot to tell you what's going on here. Right now, I am connected to another computer over our telephone line. That noisy thing over there is called a "Modem" and its job is to convert digital signals to analog and vice versa." Jace already knew way too much about telephone systems due to Grandpa's desire to play around with it every chance he could. 48 volts during normal use and 90 volts during a telephone ring notification, she knew more than any telephone technician needed to know. Plain old telephone system or POTS was a joke between Grandpa and herself. Grandma didn't seem to



get the POTS joke which made it that much funnier.

Telephone lines can be tapped by an interested party, but that could be detected by using a voltage regulator. The telephone line voltage would spike momentarily and keep slightly elevated if someone tried to tap the line. Jace thought that Grandpa loved his voltmeter more than he loved Grandma; he never left home without it. Grandpa even went so far as to name his device "Valerie." Valerie the voltmeter. That was his best friend, besides Jace.

Jace rolled her smarty-pants eyes at the Grandpa, more so at the lecture on analog modulation to digital modulation, converting sound to a digital signal. That's pretty much what a modem does. Grandpa continued his lecture to the reluctant student, "The computer I'm connected to allows me to connect to other computers and play around with whatever services they provide." Her ears picked up a word that she hadn't heard before, "services."

"Grandpa what do you mean 'services?'" the curious girl asked expecting an answer involving fast food. "Excellent question, my dear," Grandpa was expecting Jace to ask a question like that. "My computer is connected to a network of computers or I have the ability to connect to other computers across the world," he gladly replied. "This modem lets me talk to these other computers that offer access to files, information, people to talk with, and wonderful stuff like that. These computers offer services like File Transport Protocol, Usenet, IRC, Telnet, and Email."

Jace wasn't satisfied with the answer she was given and this usually led to many more questions fired at Grandpa in rapid succession. She loaded up her question ammo belt and began the carnage: "What is File Transfer proto thing? What is MIC? Where is Telnet? Does Imail need special stamps? What color is it in the digital world? Who invented Usenet? Why do they call it Imail? Does Grandma know about your services? Why do they call it services? Where do babies come from? Where does Jello come from?"

Grandpa had to cover his ears to shelter his brain from the onslaught of questions. "Wait, wait, wait, slow down."

## Game Over

## DNS

When you want to call a friend on the phone, you need to know the correct phone number; when you want to connect to a remote computer, you also need to know its number. You may remember from previous lessons that, for computers on the Internet, this number is the IP address.

IP addresses are very easily managed by computers, but we humans prefer to use names, in this case **domain names**. For example, to connect to the Hacker Highschool website, type `www.hackerhighschool.org` into the address bar of a web browser. However, the web browser can't use this name to connect to the server that hosts the Hacker Highschool website – it needs the IP address. This means that your local computer must have some means of translating domain names into IP addresses. If there were only hundreds, or even thousands of computers on the Internet, then it might be possible for you to have a simple table (a **hosts file**) stored on your computer to use to look up these addresses. However, for better or worse, not only are there millions of computers on the Internet, but the correlations between domain names and IP addresses change constantly.



**Domain Name Service (DNS)** is used to dynamically translate domain names into IP addresses (and vice-versa). When you type the domain name `www.domainname.com` into your web browser, your web browser contacts the DNS server chosen by your ISP. If that DNS server has `www.domainname.com` in its database, then it returns the IP address to your computer, allowing you to connect.

If your DNS server doesn't have `www.domainname.com` in its database, then it sends a request to another DNS server, and they will keep sending requests to other DNS servers until one finds the correct IP address, or establishes that the domain name is invalid.

### Exercises

- 4.17 Open a command line window and identify the IP address of your computer. What command have you used? What IP address do you have?
- 4.18 Identify the IP address of your DNS server. What command have you used? What is the IP address of the DNS server?
- 4.19 Ping `www.isecom.org`. Do you receive an answer? What IP address answers the ping?
- 4.20 Can you direct your computer to use a different DNS server? If so, change the configuration of your computer so that it uses a different DNS server. Ping `www.isecom.org` again. Do you receive the same response? Why?

### DHCP

**DHCP** or **Dynamic Host Configuration Protocol** allows a local network server to hand out IP addresses within the network. The server is given a block of IP addresses to use. When a computer joins the network, it gets an IP address. When a computer leaves, its IP address becomes available for use by another computer.

This is useful for large networks of computers, since it's not necessary for each computer to have an individually assigned, static IP address. Instead, you use a DHCP server. When a new computer connects to the network, the first thing that it does is request an IP address from the DHCP server. Once it has been assigned an IP address, the computer then has access to all the services of the network.

Now think about that. Most wifi networks offer DHCP, meaning that *anyone* can get an IP address on that subnet. If you're running a coffee shop that's exactly what you want, but if you're running a secure office, you might consider using fixed IP addresses instead. It depends....

### Connections

---

In the bad old days, computers connected to the Internet through a modem. Modems translate bits into sounds and back, **modulating** and **demodulating**, thus the name. Modem speeds are measured in **baud** (a rating number) and **bps**, or bits per second. Higher baud rates usually mean higher bps, but you must also consider what you are planning to do. There are certain applications – such as telnetting into **Multi-User Dungeons (MUDs)** – for which a twenty year old 300 baud modem would still be acceptable (provided your typing speed wasn't so good), while high bandwidth applications such as streaming video can often strain even the most powerful cable or DSL connections.



## ISPs

You don't just call up the Internet. You need to access a server that will connect your computer to the Internet. The server does all the heavy work, and is on all the time. The server is run by an **Internet Service Provider (ISP)**.

An ISP has a point-of-presence on the Internet that is constant, and it has servers that run services you can use. But you can run these services on your own, too. For example, you can run a mail server on your local computer, but it will require you to have your computer powered up and connected to a network all the time, just waiting for those brief moments when information has to be exchanged. An ISP, however, consolidates the efforts of a large number of users, so the mail server is working all the time, instead of sitting around, doing nothing. The ISP's computers use a high speed connection to connect to a **Network Access Point (NAP)**. These NAPs then interconnect with each other through ultra-high speed connections called **backbones**. All these things together are the Internet.

## Plain Old Telephone Service

**Plain old telephone service (POTS)** was once the most widely used method of accessing the Internet. Its primary disadvantage is its low speed, but in many cases this is made up for by its wide availability. Most national Internet service providers have a large number of local access numbers, and almost everyone still has a phone with a land line. In theory, if you had an acoustic modem and a pocket full of change, you could connect from almost any public pay phone (if you can find one). Not that you would really want to do that.

POTS is slow. The fastest telephone modems are rated at a speed of 56,600 bits per second (bps). That, however, as they explain in the small print, is a lie. Power constraints limit the actual download speed to about 53,000 bps and the effective rate is usually much lower. This doesn't compare very well with DSL or cable modems.

That said, telephone service is widely available, and POTS based ISPs are relatively cheap (and sometimes free). You wouldn't want to trade pirated movies over POTS, because it's immoral, illegal and ties up your phone line all night and maybe all weekend, but you could certainly send friendly, text based emails to Granny. And if you used telnet, you could even do it with a dusty DOS based machine that you pulled out of the basement.

## DSL

A **Digital Subscriber Line (DSL)** is a method of sending large amounts of information over the wires that already exist for POTS. Its main advantage over POTS is that it is much faster than analog modems, and it provides a permanent connection. In addition, it allows you to make and receive regular telephone calls while you are connected to the Internet. Its main disadvantage is that its availability is limited by how close you are to the telephone company's switching equipment – if you live too far down the line, you're out of luck.

## Cable Modems

Cable gateways don't use traditional telephone lines to connect to the Internet. Instead they use coaxial cable (or fiber-optic lines, if you're really lucky) provided by cable companies. Like DSL, cable gateways can allow you to make and receive regular telephone calls while you are connected to the Internet, and they provide a permanent connection, but cable gateways are generally faster than DSL.

Cable gateways have some basic flaws. The first is that cable gateway access is a shared resource, so your connection speed will be decreased when there are other users on the same cable with you. The second is that cable access is only available in areas where



cable companies have installed the necessary wiring. And the most serious one is that any traffic you put on the cable can be viewed by any other user on that cable! This means that if you connect your computer to the cable gateway and do not use a firewall, everyone else in the neighborhood can see your computer and all its files. Do you really want to share your bank account information like that?

## Wimax

Wimax is a wireless connection method that generally competes with DSL. It is used in places where a wired infrastructure would be too expensive or difficult to setup. Signal strength can be affected by buildings, trees or other large objects. Some versions use a fixed access point, but others give you mobile access over truly large areas.

## Wifi

Wifi is not a method to connect to your ISP but it is a common networking method for connecting to the Internet at home or at commercial establishments like malls or coffee shops. Most smartphones and all laptops now use Wifi so it's a favorite target for attackers. Consider yourself naked in a crowded room when you use public Wifi: cover yourself, make sure nobody's looking at you but assume everybody wants to. Of course you will read the Wireless Security lesson too, right?

## Exercises

- 4.21 What kind of Internet connection do you have at home, if you have one? How can you tell? And most important:
- 4.22 Who can you see on that network? (How can you find out?)
- 4.23 How fast is your connection? Can you improve your speed without calling up your ISP?
- 4.24 What additional services does your ISP provide? We talked about services already; your ISP may support several.
- 4.25 What services can you provide from your own computer?



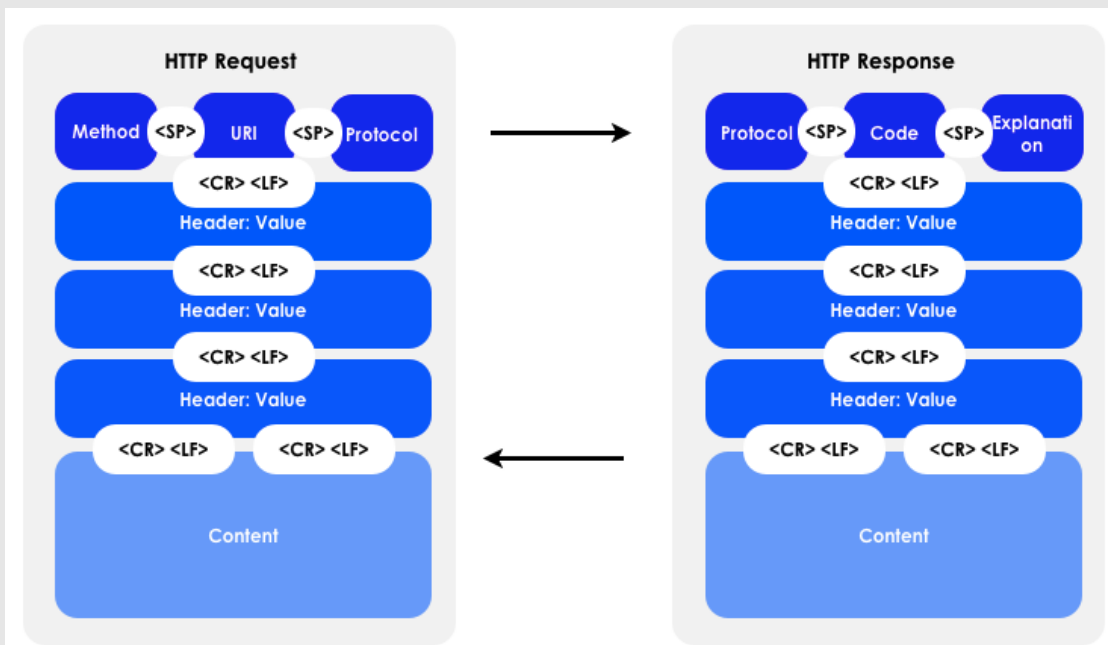
## Feed Your Head: Playing With HTTP

HTTP, the acronym for Hypertext Transfer Protocol, is located on the top of TCP/IP stack as is defined in two main RFC:

- 1945 for 1.0 (based from 0.9).
- 2616 for 1.1.

There are some substantial upgrades and differences from 1.0 to 1.1 for Extensibility, Caching, Bandwidth optimization, Network connection management, Message transmission, Internet address conservation, Error notification, Security, integrity, and authentication, Content negotiation [3]. Differences between 1.0 and 1.1 are useful to obtain information about a web server.

Basically HTTP is a stateless protocol in which Client sends an HTTP Request to Server, which sends an HTTP Response: the Request/Response paradigm.



As you may know we can obtain a lot of information sending commands to an HTTP server. We will use some basic network tools:

- netcat: the TCP/IP tool kit
- curl: the HTTP tool kit
- proxy: like OWASP ZAP or Burpsuite free

### Sniffing the Connection Between You and the HHS HTTP Server

Use a proxy to connect your browser. Go to <http://www.hackerhighschool.org> and intercept your request:

```
GET / HTTP/1.1
```

```

Host: www.hackerhighschool.org
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.8; rv:11.0)
Gecko/20100101 Firefox/11.0

Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip, deflate
Proxy-Connection: keep-alive

and response:
HTTP/1.1 200 OK
Content-Length: 10376
Date: Fri, 03 Feb 2013 09:11:17 GMT
Server: Apache/2.2.22
Last-Modified: Mon, 06 Feb 2013 09:31:18 GMT
ETag: "2f42-4b8485316c580"
Accept-Ranges: bytes
Identity: The Institute for Security and Open Methodologies, The
Institute for Security and Open Methodologies
P3P: Not supported at this time, Not supported at this time
Content-Type: text/html
Connection: keep-alive

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"[]><html
xmlns="http://www.w3.org/1999/xhtml" dir="ltr" lang="en-US"
xml:lang="en"><head><meta http-equiv="Content-Type"
content="text/html; charset=UTF-8" /><title>Hacker Highschool -
Security Awareness for Teens</title>

[...]
```

### Exercises

- 4.26 Identify parts of requests from proxy using the diagrams.
- 4.27 Is there interesting information in the headers?

### Your First Manual Connection

Netcat can be used to connect to a web server using host port settings.

Start by typing:

```
nc www.hackerhighschool.org 80
```

Then press Enter two times.

```
GET / HTTP/1.0
```

The server will reply:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
```

```
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd" []>
<html xmlns="http://www.w3.org/1999/xhtml" dir="ltr" lang="en-US"
xml:lang="en"><head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>ISECOM - Institute for Security and Open Methodologies</title>
<meta name="description" content="Description" />
```

As you can see the page appears to be from isecom.org and not from hackerhighschool.org. Why?

One hypothesis could be that the same host serves up both HHS the ISECOM site. Is this possible?

To find out, check the hackerhighschool.org IP address:

```
nslookup www.hackerhighschool.org
[...]
Non-authoritative answer:
www.hackerhighschool.org      canonical name = hackerhighschool.org.
Name: hackerhighschool.org
Address: 216.92.116.13
```

And now for www.isecom.org:

```
nslookup isecom.org
[...]
Non-authoritative answer:
Name: isecom.org
Address: 216.92.116.13
```

Same IP address! Using netcat it's possible to show the host by manually adding the Host header and using HTTP 1.1:

```
GET / HTTP/1.1
Host: www.hackerhighschool.org

HTTP/1.1 200 OK
Content-Length: 10376
Date: Fri, 03 Feb 2013 09:11:17 GMT
Server: Apache/2.2.22
Last-Modified: Mon, 06 Feb 2013 09:31:18 GMT
ETag: "2f42-4b8485316c580"
Accept-Ranges: bytes
Identity: The Institute for Security and Open Methodologies, The
```

```
Institute for Security and Open Methodologies
P3P: Not supported at this time, Not supported at this time
Content-Type: text/html
Connection: keep-alive
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd" []>
<html xmlns="http://www.w3.org/1999/xhtml" dir="ltr" lang="en-US"
xml:lang="en"><head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Hacker Highschool - Security Awareness for Teens</title>
```

### The Request Method

Another part of an HTTP request that can be modified is the Request Method. Typically web applications use GET and POST requests, but other request protocols may be active on a web server or application server. Common methods are:

- **OPTIONS** - used to ask what request options are supported.  
If you're running a web server, be aware that giving this information away could be a problem.
- **GET** - used to retrieve information directly via the URL, for example:  
`http://www.usairnet.com/cgi-bin/launch/code.cgi?Submit=Go&sta=KSAF&state=NM`  
See all that stuff after the question mark? That's the request data. Passing data this way is risky, because it's in plain sight, and it's easy to finker with.
- **HEAD** - used like GET but the server does not return an actual page.  
This can be used to identify Accesses, optimizing bandwidth consumption and – in some cases – bypassing access controls. In fact some ACL implementations check only GET requests. In this case you have found a Vulnerability.
- **POST** - used to send data to web applications – like GET – but data is transmitted in the Request Body, out of sight to at least a degree.
- **PUT** - used to allocate resources on a web server or to update it.  
In many contexts this method should have been disabled or protected by an Authentication Control. In other contexts this is a delightful find.
- **DELETE** - used to free resources on a web server.  
This method should be disabled or protected by an Authentication Control. See PUT above.
- **TRACE** - used as an application layer loopback that reflects messages.  
This debug method should be disabled, in particular on production environment because is a Confidentiality Concern and introduces a Vulnerability because it can be used for Cross Site Scripting exploits.
- **CONNECT** – to use the web server as a proxy.  
This should be disabled or protected by an Authentication Control because it permits others to connect to third party services using the proxy IP.

Also consider more protocols based on HTTP can adds more methods, as WebDAV. You can alter Request Method in order to look at server replies for interesting stuff, asking for known methods and also arbitrary words.



### Requesting OPTIONS

You could start the netcat session as usual:

```
# nc www.hackerhighschool.org 80
```

But don't press Enter twice this time. Instead, type the next line:

```
OPTIONS / HTTP/1.1
```

and you'll get a response like:

```
Host: www.hackerhighschool.org
HTTP/1.0 200 OK
Date: Tue, 07 Feb 2013 08:43:38 GMT
Server: Apache/2.2.22
Allow: GET,HEAD,POST,OPTIONS
Identity: The Institute for Security and Open Methodologies, The
Institute for Security and Open Methodologies
P3P: Not supported at this time, Not supported at this time
Content-Length: 0
Content-Type: text/html
```

### Requesting HEAD

This time, after starting your session, enter the HEAD option.

```
# nc www.hackerhighschool.org 80
```

```
HEAD / HTTP/1.1
```

```
Host: www.hackerhighschool.org

HTTP/1.0 200 OK
Date: Tue, 07 Feb 2013 08:41:14 GMT
Server: Apache/2.2.22
Last-Modified: Fri, 13 Feb 2013 15:48:14 GMT
ETag: "3e3a-4bd916679ab80"
Accept-Ranges: bytes
Content-Length: 15930
Identity: The Institute for Security and Open Methodologies
P3P: Not supported at this time
Content-Type: text/html
Age: 45
Connection: close
```

### Let Me Use You As A Proxy: the CONNECT Request

```
# nc www.hackerhighschool.org 80
```

```
CONNECT http://www.isecom.org/ HTTP/1.1
```

```
Host: www.hackerhighschool.org
```

### Exercise

- 4.28 Use netcat (nc) to try all of the Request methods listed above on the HHS network servers or a server set up for the purpose. What kind of interesting stuff can you dig up?

### Scripting HTTP requests with curl

Some Web Application Testing is based not only to Web Server response but on the (Web) Application Layer. Often you can find web application vulnerabilities by altering GET and POST parameters, altering cookies and tinkering with headers. A useful tool, for bash scripting is the **curl** command, a command-line tool for requesting a web page. But curl adds some logic over netcat.

Asking for:

```
# curl http://www.isecom.org
```

is not the same as

```
# nc www.isecom.org 80
```

```
GET / HTTP/1.1
```

To see this you can use the `-v` switch for verbose output:

```
# curl -v http://www.isecom.org/
* About to connect() to www.isecom.org port 80 (#0)
*   Trying 216.92.116.13...
*   connected
* Connected to www.isecom.org (216.92.116.13) port 80 (#0)
> GET / HTTP/1.1
> User-Agent: curl/7.26.0
> Host: www.isecom.org
> Accept: */*
>
* HTTP 1.0, assume close after body
< HTTP/1.0 200 OK
< Date: Tue, 07 Feb 2013 09:29:23 GMT
< Server: Apache/2.2.22
< Last-Modified: Fri, 13 Feb 2013 15:48:14 GMT
< ETag: "3e3a-4bd916679ab80"
< Accept-Ranges: bytes
< Content-Length: 15930
```

```

< Identity: The Institute for Security and Open Methodologies
< P3P: Not supported at this time
< Content-Type: text/html
< Age: 247
< Connection: close
<
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"[]>
<html xmlns="http://www.w3.org/1999/xhtml" dir="ltr" lang="en-US"
xml:lang="en">
[...]
```

As you can see curl selects automatically the HTTP version 1.1, adds host header, user agent and accept. Which points to an important rule for hackers: know your tools.

Luckily curl is a nice tool can be highly customized using switches.

To see all of them use `curl -help`

Some switches for functions similar to the netcat example above are:

- **-H** to add a header line
- **-X** to select a request method (also known as Command)
- **-d** to add POST data
- **-i** to include protocol headers in the output
- **-s** to enable silent mode, useful for scripting

With curl and some bash scripting you can automate web application testing. Looking for interesting HTTP headers from a server can be automated simply with curl and grep:

```
# curl -siX HEAD http://www.isecom.org/ | grep "Server:"
Server: Apache/2.2.22
```

### Exercise

- 4.29 Expand the script above to request more HTTP headers and potentially useful information.

### References and Further Reading

<http://www.ietf.org/rfc/rfc1945.txt>  
<http://www.ietf.org/rfc/rfc2616.txt>  
<http://www8.org/w8-papers/5c-protocols/key/key.html>  
<http://netcat.sourceforge.net/>  
<http://curl.haxx.se/>







## Conclusion

---

The World Wide Web is a whole lot more than the Internet: there are all kinds of services besides just HTTP. FTP, SSH, DNS, DHCP and a whole lot more all offer windows into other people's computers – and yours. Understanding how you connect to these services, whether “through the proper channels” or otherwise, is key to knowing how you or your computer can be attacked – or attack. Just remember the motto: Hack everything, but harm none.

Today's teens are in a world with major communication and productivity channels open to them and they don't have the knowledge to defend themselves against the fraud, identity theft, privacy leaks and other attacks made against them just for using the Internet. This is the reason for Hacker Highschool.

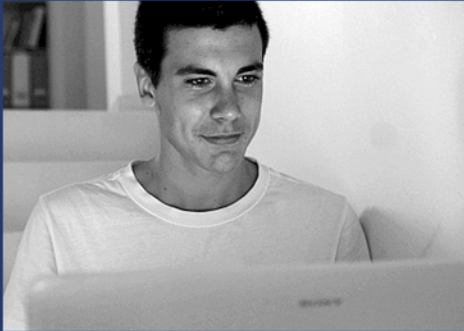
**The Hacker Highschool project is the development of security and privacy awareness learning materials for junior high and high school students.**

Hacker Highschool is a set of lessons and a practical means of making hackers. Beyond just providing cybersecurity awareness and critical Internet skills, we need to teach the young people of today how to be resourceful, creative, and logical, traits synonymous with hackers. The program contains free security and privacy awareness teaching materials and back-end support for teachers of accredited junior high, high schools, and home schooling. There are multiple workbooks available in multiple languages. These are lessons that challenge teens to be as resourceful as hackers, including safe Internet use, web privacy, researching on the internet, avoiding viruses and Trojans, legalities and ethics, and more.

**The HHS program is developed by ISECOM, a non-profit, open-source research group focused on security awareness and professional security development and accreditation.**

# Hacker HighSchool

## SECURITY AWARENESS FOR TEENS



## LESSON 5 SYSTEM IDENTIFICATION



HACKING IS LEARNING  
[www.hackerhighschool.org](http://www.hackerhighschool.org)

ISECOM

Creative Commons 3.3 Attribution-Non-Commercial-NoDerivs ISECOM

[WWW.ISECOM.ORG](http://WWW.ISECOM.ORG) - [WWW.OSSTMM.ORG](http://WWW.OSSTMM.ORG) - [WWW.HACKERHIGHSCHOOL.ORG](http://WWW.HACKERHIGHSCHOOL.ORG) - [WWW.BADPEOPLEPROJECT.ORG](http://WWW.BADPEOPLEPROJECT.ORG) - [WWW.OSSTMMTRAINING.ORG](http://WWW.OSSTMMTRAINING.ORG)



## WARNING

The Hacker Highschool Project is a learning tool and as with any learning tool there are dangers. Some lessons if abused may result in physical injury. Some additional dangers may also exist where there is not enough research on possible effects of emanations from particular technologies. Students using these lessons should be supervised yet encouraged to learn, try, and do. However ISECOM cannot accept responsibility for how any information herein is abused.

The following lessons and workbooks are open and publicly available under the following terms and conditions of ISECOM:

All works in the Hacker Highschool Project are provided for non-commercial use with elementary school students, junior high school students, and high school students whether in a public institution, private institution, or a part of home-schooling. These materials may not be reproduced for sale in any form. The provision of any class, course, training, or camp with these materials for which a fee is charged is expressly forbidden without a license including college classes, university classes, trade-school classes, summer or computer camps, and similar. To purchase a license, visit the LICENSE section of the HHS web page at <http://www.hackerhighschool.org/licensing.html>.

The HHS Project is an open community effort and if you find value in this project we ask that you support us through the purchase of a license, a donation, or sponsorship.



## Table of Contents

WARNING.....	2
Contributors.....	4
Introduction.....	5
Identifying a Server.....	7
Identifying the Owner of a Domain.....	7
Identifying the IP Address of a Domain.....	8
Game On: Slash and Burn.....	9
Identifying Services.....	10
Ping and Traceroute.....	10
Nmap.....	12
Banner Grabbing.....	12
Misleading Banners.....	14
Automated Banner Grabbing.....	14
Identifying Services from Ports and Protocols.....	15
System Fingerprinting.....	17
Scanning Remote Computers.....	18
Feed Your Head: Going Deep with Nmap.....	21
TCP Scan.....	22
SYN Scan.....	23
UDP scan.....	24
Service Scan (UDP).....	25
OS Detection.....	26
Using Scripts.....	29
Conclusion.....	31



## Contributors

---

Pete Herzog, ISECOM  
Marta Barceló, ISECOM  
Chuck Truett, ISECOM  
Kim Truett, ISECOM  
Marco Ivaldi, ISECOM  
Greg Playle, ISECOM  
Bob Monroe, ISECOM  
Simone Onofri, ISECOM  
Ryan Oberto, Johannesburg South Africa  
Dennis King  
Mario Platt  
Grigoris Chrysanthou

**ISECOM**



## Introduction

---

"I think my laptop has a virus," one of my students told me. "Can you take a look at it?"

I took the notebook computer from him, didn't open it, but tilted it every direction, looking closely. "Looks like a computer to me," I said, handing it back to him.

"But something's wrong with it," Aidan insisted. "I went to my friend's house and got on the Internet, and something got into my email and sent messages to all my friends."

"Okay, how do you get to your email? Did you install an application?" I asked.

"No, I do it on the web. I mean Internet."

"You mean in a web browser?" He nodded. "Then that means your email is online, not on your computer. So in this case I'd start with your email account. Have you changed the password?"

"Yeah. They shut down my account until I changed it." He looked down, like there was more to the story, but I didn't press him. My bet was that he'd already been yelled at. A lot.

"Have your friends gotten any more of the messages?" I asked instead.

"No." Staring firmly at his shoes.

"And did you choose a decent password? Not 12345?"

Now he smiled. "It's a really hard one. Nobody's ever gonna get it."

I had my doubts about that, but I nodded. "Okay, then, sounds like you've got it all sorted out."

"No," he insisted. "Why would somebody do that?"

Now I had the fish on the hook. "Why don't you find out. Do you have any of those emails that your friends got?"

"Yeah. A bunch of them. People sent them back to me." Ah: there it was. I'd bet his contact list numbered in the dozens. Or the hundreds. That had to have been fun.

"Then it sounds to me like you need to find out exactly where that link in the email goes."

Cameras flashed behind his eyes. "You mean we can do that?"

"Hah," I laughed. "It means YOU can do it. But I'll show you how."

Aidan stopped. "Is this what you mean by the sheep and the wolf you're always talking about?"

"Yes, exactly that. You can be one or the other. Choose now," I told him.

Suddenly he didn't look so much like a kid. "Wolf," he told me.

\* \* \*

System identification can easily be the most important step of any computer attack or defense. Everything you do afterward depends on the data you gather at this stage. What's the operating system of the host that's attacking you, or that you're defending? Can you – or others – see what applications or services are running? How about the administrator's personal details: are they in plain sight anywhere? These are the questions to ask at this stage. Depending on which side you're on, you might be delighted or horrified at what's easily available if you know where to look.



Knowing how an attack works is cool. Knowing how to protect against it or defeat it is even cooler. Here's where we start digging deep and learning how to identify a system and find its weaknesses – whether it's our own system or someone else's.

We'll be using tools that are publicly available and we'll even show you how to use them. It wouldn't make much sense to show you software but not teach you how to use it. As with any security program, they can be used for good or bad purposes. Our mission is to show you both uses so that you can fix your own security challenges, while protecting against similar attacks.

In this lesson, you'll be following two individuals as one teaches and the other person learns. The teacher doesn't always know what the answer will be so you as the reader will not be spoon-fed information either. Learn to break things and learn how to fix those things you broke. Repeat as necessary.

Pay close attention to attributes used in various programs. A slight change in an upper case to a lower case syntax letter may bring you entirely different data, more-so in different operating systems. These first few lessons are the foundation of networking and how the internet works. Each lesson builds on the previous knowledge so don't be in a hurry, but skipping around the paragraphs and pages is a good way to get familiar with this material before you go back and read in depth. Obviously you don't want to overlook a crucial piece of knowledge.





## Identifying a Server

“Okay, Aidan, what did you find out?” I was trying not to grit my teeth with the fear that he'd gone and clicked that stupid link in the email his hacked account had sent out.

“I didn't left-click it,” Aidan told me, smiling up like he'd read my mind. “I copied it and pasted it into a plain text file.”

“The text you could see? Or the actual link?”

He frowned. “I'm not stupid. I right-clicked and chose 'Copy link location.' Then I pasted it here. Look, link.txt.”

“Sorry. Just had to be sure. So okay. Where does it go?”

“This crazy domain. Chewmoogoo.com or something. There's a bunch of other stuff after that too,” he said, opening his laptop and showing me the link.

“Oh yeah,” I told him. “Now we've got 'em. Now let's see what information we can gather and the tools that can help us collect it. First let's talk about domain names and IP addresses.”

## Identifying the Owner of a Domain

The first step in identifying a remote system is to look at its host name, domain name or IP address. A **whois** lookup on a domain name turns up a bunch of information:

- The identity of the owner of the domain, usually a full name
- Contact information, which may include street addresses, phone numbers and email addresses
- The DNS servers where the domain is registered, which may also tell you the ISP that serves up the domain
- The IP address of the server, another potential clue to the ISP
- Domain name information, like the date it was created, when it was updated or when it will expire

Keep in mind that there are a lot of different domain name registrars, and not all whois databases contain the information for all domains. You may have to look at more than one whois database to find information about the domain that you are investigating.

Aidan soaked this up instantly. “Okay, what do I do?”

“Here's your assignment,” I said.

## Exercise

- 5.1 Get the domain name you're investigating. (If you're not Aidan, use [isecom.org](http://isecom.org).) Try the following command on Linux, Windows and OSX.

```
whois ise.com.org
```

Who owns the domain?

When was it created? When will it expire? (Does that expiration present an opportunity?)

When was it last updated?



Who are the different contacts listed?  
 What are its primary and secondary name servers?

- 5.2 Now do the same lookup in a browser (for instance, `http://www.whois.net -> "sample.com"`). Here's the critical question: Does it match what you got from your whois command?  
 Check at least two whois websites. Try `http://whois.domaintools.com`; can you find more?).

### Identifying the IP Address of a Domain

"So what have you got?" I asked Aidan.

"All this stuff. I pasted it in." He showed me his text file.

"That's good. Keep every single scrap of information. What's the domain IP?"

"This thing, isn't it?" Aidan pointed at a long number.

"Yes. You can get the domain's IP address with a whois command, or you can do a DNS lookup with a **ping** command:

```
ping isecom.org
```

"The first thing you'll see is the domain's IP address."

If you can capture email from the target, examine the **email headers** (see Lesson 9, Email Security); that will give you the IP address of the originating mail host. You can also use resources like search engines (Lesson 20, Social Engineering) or tools like **Maltego** or **FOCA**. Search on terms like the target organization's name, the domain registration point of contact, telephone numbers and addresses. Each of those can lead you to more information.

"Once you've got an IP – or more than one – you need to find out where it is. IP numbers get assigned to service providers all over the globe in big groups. Find out which group an IP address was issued in (and who has the rights to that group, if you can). That can help you find out what server or service provider the website uses and the real gold for you - what country houses that server," I told Aidan. "Bet it's not this one. So here's what you do next."

### Exercises

Now you're going to look at DNS records directly. Another way to find information about a domain and server(s) is to use information in DNS. There are three commands to get started.

- 5.3 Open a terminal window. Try this command:

```
dig isecom.org
```

Does this command work on your OS? Try it in Windows, Linux and OSX.



5.4 Now try this command:

```
host isecom.org
```

Does this command work on your OS? Try it in Windows, Linux and OSX again.

5.5 Finally try this command:

```
nslookup isecom.org
```

Does this command work on your OS? Once again, try it in Windows, Linux and OSX.

What's the DNS server for your target? Does the organization have a mail server? Does the mail server have the same IP address as the web server? What does this suggest? What else can you learn?

5.6 Once you have the IP address, you can access the records of the various members of the **Number Resource Organization** (<http://www.arin.net/>, <http://www.ripe.net/>, or <http://www.apnic.net/>), to gain insight about how IP addresses are distributed.

### Game On: Slash and Burn

It was a grudge match as far as Jace was concerned. The battle of the century, as she thought of it. No matter how much sweat, blood, pain, physical and intellectual force required, the ambitious teen was prepared to win this fight. She had to win since there was no plan B. Her cocoa colored hair swayed over her eyes like a bullfighter taunting with a red cape. A one last calming deep breath and the network killer was prepared to being.

With her nimble fingers floating over the grinning keyboard, she assessed the situation and took stock of here available resources. Jace had a copy of Nmap already loaded into the computer beast. Ping and Traceroute had already been run so the combative hacker was ready to start slashing away.

Down went the first of a rapid succession of keyboard taps. A machine gun couldn't fire as quickly as Jace did when it came to working the computer commands. Ping, down! Traceroute, down! The IP commands didn't stand a chance against her massive barrage of key pecking. Time to live, down! The bloodshed was horrendous, as bits and bytes tumbled across the monitor in blurs. The CLI seemed to direct the incoming blitz of powerful switches, with attack attributes flanking the main network.

Jace maneuvered her main assault to gain a foothold inside the network. Her scouts performed an intensive reconnaissance of forward deployed firewalls, servers, and routers. This data was compared against Common Vulnerabilities and Exposures (CVE) and cross-referenced with Nmap's own Network Scanning information. Each weakness, every vulnerability, and exploit was examined for tactical advantage and damage assessments. A truce was not an option for Jace. She was winning.

It wasn't over yet, she told herself. In fact, all she had done was captured a small part of the enemies resources but the intelligence was invaluable, nonetheless. Jace suffered little casualties on her end. Fingers and knuckles were a slightly sore. She had a small bruise near her forehead where she banged it against the monitor in frustration. TTLs were killing her.



In the end, battle banners gave up details without the need for interrogation or repeated torture using the “bread-boarding” technique. Raspberry pie was kept in reserve. Jace had enough information on the enemy to perform phase two of the network attack. Next phase required loaded emails and the unintentional insider help.

This was always the scariest part of any battle, obtaining turncoats. Jace needed users on the inside who would be sympathetic to her cause. Now was the time for all good security habits to be broken. Social engineering was the weapon of mass disruption in her arsenal. She would have to craft legitimate emails loaded with Trojan soldiers to penetrate the networks inner walls.

As Jace began constructing each malicious email, she knew that she was on the right side of this confrontation. No matter what it took, no matter how long it took, Jace was determined to know which secret flavor of ice cream the local dairy was working on next.

**Game continues...**

## Identifying Services

“So you saved all that stuff, right?” I grinned but tried not to, because I knew the answer even if my teacherly nature made me ask.

Aidan barely let a sideways glance slip: *bonehead* he was thinking, but he said, “Check it out” and handed me his notebook.

“Lots of info now, isn't there?” I scrolled down through the pages.

“Yeah. I need a better way to keep track of stuff,” Aidan said, taking the computer back.

“You sure do. What's your target's IP?” This time I smiled openly.

“Well ... there's about five. Maybe more than that. I'm trying to figure out why, because I can ping some and some I can't.”

*Good man* I thought. Once you've got the IPs for a domain you can start digging into services, and that means running hosts. *Oh fun.*

## Ping and Traceroute

“You're starting in the right place. You need to make sure there really are active machines. And you're right; ping is your friend. You did remember to ping the domain name, the IP addresses and the host names, didn't you?”

“Which ones are host names?” Aiden asked.

“They're the ones with letters and a dot before the domain name, like `www.isecom.org`,” I told him.

“I don't see any of those.”

“Check out your dig results. You didn't try the other ones. Did you try `www.isecom.org` and `ftp.isecom.org` and `mail.isecom.org`?”

“No...”

"Well, if you get a response, there's something alive at that address. And you're getting through the firewall. And they're letting ICMP through." I opened a CLI and entered a command.

```
C:\>ping isecom.org
```

```
Pinging isecom.org [216.92.116.13] with 32 bytes of data:
```

```
Reply from 216.92.116.13: bytes=32 time=186ms TTL=56
```

```
Reply from 216.92.116.13: bytes=32 time=186ms TTL=56
```

```
Reply from 216.92.116.13: bytes=32 time=186ms TTL=56
```

```
Reply from 216.92.116.13: bytes=32 time=186ms TTL=56
```

```
Ping statistics for 216.92.116.13:
```

```
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
```

```
Approximate round trip times in milli-seconds:
```

```
    Minimum = 186ms, Maximum = 186ms, Average = 186ms
```

"You can get an idea of how far the server is from you, both on the network and physically, by the round trip times. Divide in half, and you can get a feel for the distance to the server. I want you to try another tool, traceroute. It's spelled **tracert** in Windows and **traceroute** in Linux. It'll show you the steps packets take from your computer to your target. Like this," I said, and typed again.

```
C:\>tracert isecom.org
```

"Now, here's what I want you to do."

## Exercises

- 5.7 Use traceroute/tracert to put together all the information you can find about the computers and routers between your computer and your target.
- 5.8 Computers with similar IP addresses are often part of the same network. Ping a valid website or IP address (for example, ping [www.isecom.org](http://www.isecom.org) or ping 216.92.116.13). If you get a successful response, ping the next IP address. Did you get a response? Try more nearby addresses.
- 5.9 Use a search engine to find out how to estimate the distance to the server.
- 5.10 Look for a tool that can help you map the server to a physical location.
- 5.11 Look for a Visual Trace Route tool online. There are quite a few sites that provide tools like this. This ought to give you a better visualization of where your traffic is going.



## Nmap

"Got all that? Now let me introduce you to my little friend," I said, trying to do a Scarface voice. Aidan looked at me like I had two heads, so I cleared my throat, and, um, finished, "nmap."

"It can be really simple, or you can get really tricky. Run the nmap command with a host name or an IP address, and it'll scan that host. Or use a bunch of switches to do really tricky things. If you ask right, it'll try to tell you the OS of your target. We're going to use the 'scan TCP' option, which is -sT."

```
nmap -sT 216.92.116.13
```

```
Starting Nmap 5.51 ( http://nmap.org ) at 2012-05-28 10:58 GTB Daylight Time
```

```
Nmap scan report for 216.92.116.13
```

```
Host is up (1.1s latency).
```

```
Not shown: 969 closed ports
```

```
PORT      STATE SERVICE
```

```
25/tcp    open  smtp
```

```
80/tcp    open  http
```

```
110/tcp   open  pop3
```

```
119/tcp   open  nntp
```

```
135/tcp   open  msrpc
```

```
139/tcp   open  netbios-ssn
```

```
143/tcp   open  imap
```

```
445/tcp   open  microsoft-ds
```

```
465/tcp   open  smtps
```

```
554/tcp   open  rtsp
```

```
Nmap done: 1 IP address (1 host up) scanned in 215.42 seconds
```

It's important to remember that nmap isn't the only tool for doing these scans, and that's a good thing. Different tools can give you different results, and in fact any of them can be deliberately misled.

You can tell nmap, for instance, to guess the operating system – but you should not trust its guess! Verify its theory using other tools.

## Banner Grabbing

Aidan was gleeful. "Look what I've got now!" He had text documents and a spreadsheet on his laptop, drawings in a paper notebook and color printouts that had to have cost somebody a fortune in ink cartridges.



"Okay, now you know you've got some live machines, who runs it and roughly where it is. Next you want to know what kind of machine it is: what operating system is it running? What services is it running?" I asked him.

This made him less gleeful. "Um, how do I tell?"

"You don't have to. Get the machine to spill its guts: operating system, services and patch levels. When you're the attacker, that makes your job really easy; all you have to do is look up the exploits for that service, software and version. If you're the defender, you'll want to suppress that information. Or better yet, lie." This made him look thoughtful.

"So what you do next is called **banner grabbing**. Fancy word: it's an **enumeration technique** to get all kinds of information on your target active services and ports. I'm going to show you some more commands. You can use telnet, ftp or netcat to grab the banner. The banner is that old-school text message you'd get at the command line when you connected to tell you what server program is running. So check it out: when I connect to an anonymous FTP server, I get a banner." I typed into my terminal window:

```
ftp isecom.org
```

```
Connected to anon.server.
```

```
220 ProFTPD Server (Welcome . . . )
```

```
User (anon.server:(none)):
```

"That number 220 is a code that says the server's ready for a new user. And isn't this nice: ProFTPD Server is the FTP program running on that host. Now we hit the web to find out what OS ProFTPD runs on, what it can do ... what's messed up, if anything." I rattled away on the keyboard. "Here: your next assignment is to use the ftp command."

## Exercise

5.12 You can use FTP with either a host name or an IP address, like this:

```
ftp isecom.org
```

or

```
ftp 216.92.116.13
```

Try both to see what banner the FTP server returns. Your result may look like this:

```
Connected to isecom.org.
```

```
220 ftp316.pair.com NcFTPd Server (licensed copy) ready.
```

```
User (isecom.org:(none)):
```

5.13 You can use Telnet with either a host name or an IP address, too. With either one you can specify the port, which is 21 when we're connecting to FTP:



```
telnet iseecom.org 21
or
telnet 216.92.116.13 21
```

Again, see what banner the server returns – if anything. You may get something like this:

```
220 ftp316.pair.com NcFTPD Server (licensed copy) ready.
```

- 5.14 Use netcat with either a host name or an IP address, too. Just like with Telnet, you can specify the port, which is 21 for FTP:

```
nc iseecom.org 21
or
nc 216.92.116.13 21
```

Again, see what banner the server returns – if anything.

### Misleading Banners

“Here’s the trick,” I told Aidan. “You can change the banner. That’s one kind of **spoofing** – lying about who you are. So I can change my banner to read *NoneOfYourBusiness Server*, which is cute, but a Unix system with a banner that reads *WS\_FTP Server* is going to throw people off, because that’s a Windows FTP server.”

“Wait a minute – how do you change the banner?” he asked.

“Glad you asked,” I said.

### Exercise

- 5.15 Get on the web and find out how to change the banners for SMTP, FTP, SSH, HTTP and HTTPS. Is it hard to do? In other words, should you just trust what banners say?

### Automated Banner Grabbing

“Now check this out. We can go back to nmap and automate this; we have to use the `-sTV` switches to get banners.” I typed the first line and got this report:

```
nmap -sTV -Pn -n --top-ports 10 --reason -oA hhs_5_06 hackerhighschool.org
```

```
Starting Nmap 6.00 ( http://nmap.org ) at 2012-06-23 05:10 CEST
Nmap scan report for hackerhighschool.org (216.92.116.13)
Host is up, received user-set (0.30s latency).
PORT      STATE  SERVICE      REASON      VERSION
21/tcp    open  ftp          syn-ack     NcFTPD
```



```

22/tcp open  ssh      syn-ack  OpenSSH 5.9 (protocol 2.0)
23/tcp closed telnet   conn-refused
25/tcp filtered smtp     no-response
80/tcp open  http     syn-ack  Apache httpd 2.2.22
110/tcp open  pop3     syn-ack  Dovecot pop3d
139/tcp closed netbios-ssn conn-refused
443/tcp open  ssl/http syn-ack  Apache httpd 2.2.22
445/tcp closed microsoft-ds conn-refused
3389/tcp closed ms-wbt-server conn-refused
Service Info: OS: Unix

```

```

Service detection performed. Please report any incorrect results at
http://nmap.org/submit/ .

```

```

Nmap done: 1 IP address (1 host up) scanned in 17.32 seconds

```

"Nmap found NcFTPd, OpenSSH 5.9 (protocol 2.0) and Apache httpd 2.2.22. Bingo: the OS is Unix. Sometimes the banners give you the operating system version, but we're going to need a little bit more info to get specific," I continued. "Here's what I want you to do."

## Exercises

- 5.16 Use nmap on your target (hackerhighschool.org, if you're not Aidan).
- 5.17 Try it again with the option **--version-intensity number** using numbers from 0 to 9 to get more accurate results. What differences can you see in these reports?

## Identifying Services from Ports and Protocols

"Nmap did that last scan by looking for default services. But you can do it from the other direction too: look for open ports first, then see what service is actually behind them," I said.

"Wait a minute," Aidan demanded. "Aren't the ports always the same?"

"Yeah, in theory they are. But really, port numbers are sort of a gentlemen's agreement. I can put my services on different ports if I want."

"Okay, how do I do that?"

"Start by looking at your own local computer. Go to a command line and run the **netstat** command with the **-a** switch to scan all ports. Like this," I demonstrated.

```
netstat -a
```

The young hacker followed my example, then burst out, "Whoa! All of these are open?"

I looked at his screen. "Your computer is named Quasimodo?"

```
Active Connections
```



Proto	Local Address	Foreign Address	State
TCP	Quasimodo:microsoft-ds	Quasimodo:0	LISTENING
TCP	Quasimodo:1025	Quasimodo:0	LISTENING
TCP	Quasimodo:1030	Quasimodo:0	LISTENING
TCP	Quasimodo:5000	Quasimodo:0	LISTENING
TCP	Quasimodo:netbios-ssn	Quasimodo:0	LISTENING
TCP	Quasimodo:1110	216.239.57.147:http	TIME_WAIT
UDP	Quasimodo:microsoft-ds	*:*	
UDP	Quasimodo:isakmp	*:*	
UDP	Quasimodo:1027	*:*	
UDP	Quasimodo:1034	*:*	
UDP	Quasimodo:1036	*:*	
UDP	Quasimodo:ntp	*:*	
UDP	Quasimodo:netbios-ns	*:*	
UDP	Quasimodo:netbios-dgm	*:*	

"Yeah, Quasimodo," Aidan grinned. "The Hunchback."

"Okay then, Victor. Here's what I want you to do."

### Exercises

5.18 Run netstat on your local computer, using the -a switch.

```
netstat -a
```

Which ports are open?

5.19 Run netstat on your local computer, using the -o switch.

```
netstat -o
```

What services are listening behind the open ports?

5.20 Run netstat on your local computer, using the -aon switch combination.

```
netstat -aon
```

What does this combination get you?



- 5.21 Using a web search engine, match these ports with the services that run on them. Some of them you need for things like networking. But do you really want all the services you see running?
- 5.22 Run nmap, using the `-sS` (to do a SYN or so-called “stealth” scan) and `-O` (for guess operating system) switches and the IP address 127.0.0.1 as the target. The IP address 127.0.0.1 is called the **loopback** address. It always means localhost, your local computer.

```
nmap -sS -O 127.0.0.1
```

What open ports does nmap find? What services and programs are using these ports? Now try running nmap while you have a web browser or telnet client open. How does this change the results?

The “stealth” scan uses just the first part of the TCP three-way handshake – the SYN packet – to probe a port without fully setting up a connection. While this gets you past the system's logs (which won't log your probe unless you really make a connection), it is NOT undetectable. Any intrusion detection system is going to see your big, greasy fingerprints all over the network, so don't fool yourself that you're really being stealthy.

- 5.23 Nmap has additional command line switches. What do `-sV`, `-sU`, `-sP`, `-A`, `--top-ports` and `--reason` do? What other possibilities are there? If you were an attacker and you wanted to remain stealthy rather than banging on the server, which switches should you *not* use, or use?
- 5.24 Go to [www.foundstone.com](http://www.foundstone.com), and find, download and install **fport** on your Windows box. It's similar to netstat, but it also details which programs are using the open ports and protocols. Run it. How does it compare to netstat?

## System Fingerprinting

“You didn't go stumbling around and ringing bells, did you?” I asked.

Aidan replied slowly, really thinking about it, “No, I don't think so. But does it really matter? I mean their servers are way over in ....”

I interrupted. “I don't know where they are, I don't care, you're going to operate ethically – and carefully – as long as you're working with me.”

“Okay,” sheepishly.

“It's a good policy to leave no tracks. Which is almost impossible. But you should always be trying. Because tracks are exactly what you're going to work on next. Or actually, fingerprints....”

“Hey! Those aren't the same!”

"Okay, got me. But regardless, we're going to put everything together to **fingerprint** your target, find the OS and all its services."

## Scanning Remote Computers

"What did you finally get back on your stealth scans?" I asked. Aidan showed me a report he'd pasted into a text document.

```
nmap -sS -O 216.92.116.13
```

```
Starting Nmap 5.51 ( http://nmap.org ) at 2012-05-28 16:54 GTB Daylight Time
```

```
Nmap scan report for isecom.org (216.92.116.13)
```

```
Host is up (0.19s latency).
```

```
Not shown: 965 closed ports
```

PORT	STATE	SERVICE
21/tcp	open	ftp
22/tcp	open	ssh
25/tcp	filtered	smtp
26/tcp	open	rsftp
80/tcp	open	http
110/tcp	open	pop3
111/tcp	filtered	rpcbind
113/tcp	filtered	auth
135/tcp	filtered	msrpc
139/tcp	filtered	netbios-ssn
143/tcp	open	imap
161/tcp	filtered	snmp
179/tcp	filtered	bgp
306/tcp	open	unknown
443/tcp	open	https
445/tcp	filtered	microsoft-ds
465/tcp	open	smtps
514/tcp	filtered	shell
543/tcp	open	klogin
544/tcp	open	kshell
587/tcp	open	submission
646/tcp	filtered	ldp
800/tcp	filtered	mdb_s_daemon
993/tcp	open	imaps

```

995/tcp open  pop3s
1720/tcp filtered H.323/Q.931
2105/tcp open  eklogin
6667/tcp filtered irc
7000/tcp filtered afs3-fileserver
7001/tcp filtered afs3-callback
7007/tcp filtered afs3-bos
7777/tcp filtered cbt
9000/tcp filtered cslistener
12345/tcp filtered netbus
31337/tcp filtered Elite
Device type: general purpose|storage-misc
Running (JUST GUESSING): FreeBSD 7.X|6.X (88%)
Aggressive OS guesses: FreeBSD 7.0-BETA4 - 7.0 (88%), FreeBSD 7.0-RC1
(88%), FreeBSD 7.0-RELEASE - 8.0-STABLE (88%), FreeBSD 7.0-STABLE (88%),
FreeBSD
7.1-RELEASE (88%), FreeBSD 6.3-RELEASE (86%), FreeNAS 0.7 (FreeBSD 7.2-
RELEASE) (85%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 8 hops
OS detection performed. Please report any incorrect results at
http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 24.09 seconds

```

"See all those ports marked **filtered**? That means they're protected by a firewall. They're well-known and vulnerable, so they should always be blocked. But look: ports 21, 22 and 80 – that's FTP, Secure Shell and HTTP – are all open." I looked over at Aidan.

"Sheep?" he asked hopefully.

"Well, rightful prey, at least. Okay. The last thing that nmap does is try to figure out the operating system on your target. Lots of the time, like now, it only makes an 'aggressive guess,' but that's usually pretty good. Since the scan shows FTP and SSH open, the banners you grabbed would be the next piece of evidence.

"Hit the web, it tells us NcFTPd is a Unix program and that FreeBSD is a Unix-type operating system. SSH you'd usually find on Unix-like OSs. So it's likely the server's running some version of FreeBSD. You know those banners can be spoofed, but it's a reasonable guess.

"Now, depending on where your target is, your next step might be to find the ISP. The ISP itself might be famous for hosting spammers or malicious sites – do a search – but you might be able to whine to them and get your evil attacker shut down. In your case I think it's not going to be an ISP you can really deal with....

"Because it's in..." Aidan burst in, but I held up my finger.

"Stop. Your information is your information. I don't need it, as long as you are being ethical and safe. Which you are."

Aidan nodded.



"So watcha gonna do?" I asked.

"Well, they've got a web server running, right?" Aidan began, and all I could do was smile.

## Feed Your Head: Going Deep with Nmap

Say you've identified the hostname, the owner, the network and verified the host is up. Now in order to identify a system you need to find some open ports. Don't forget that the host can be up but have all ports closed (or even filtered).

You can use the famous Network Mapper (aka **nmap**) tool from Fyodor to do this task. Nmap is a port scanner and is able to remotely probe computers for open ports and related network services. When you execute nmap, depending on the command line switches that you use, you'll get a list of open ports and the services or protocols that use those ports. Nmap may also be able to determine what operating system your computer is using.

Nmap has many options and scan types. We will use a few nmap options but you can always use

```
nmap --help
```

or

```
man nmap
```

to see the details.

Before we begin, have you read Lesson 3? No? Now's the time to do it! Back already? No? Then go now!

Ok, explain the differences between TCP and UDP and describe the three-way handshake. Knowing how this works is important for understanding how nmap works.

Nmap syntax is:

```
nmap scan-techniques host-discovery options target
```

- **scan-techniques** specify what kind of packets will be used and how responses from target should be interpreted. The main available techniques are:
  - **-sS** SYN scan (yes, only the first part of three-handshake)
  - **-sT** TCP Connect scan (full three-way handshake)
  - **-sA** ACK scan (send only ACK packets)
  - **-sU** UDP Scan
  - **-O** OS Detection
  - **-A** All functionalities such as OS detection, plugins, traceroute
- **host-discovery** specifies the techniques used to define if a host is alive or not. If the host is alive it will be scanned, otherwise not.
  - **-PE** check if host responds to a ping
  - **-PS** check if host responds to a SYN
  - **-PA** check if host responds to an ACK
  - **-PU** check if host responds to a UDP datagram
  - **-PN** don't check, treat all hosts as active (we'll use this because we know that our target is alive, since we checked earlier)
- **options** specify further details for the selected scan type, such as



- **-p1-65535** port numbers to scan (in this example from 1 to 65535).
- **--top-ports <number>** nmap knows which are the most frequently used ports, and can scan only for the top <number> specified
- **-T0, -T1, -T2, -T3, -T4** for the scan speed, where 0 is slow and 4 is fast (slower means more stealthy and with less network congestion)
- **-oA <filename>** for the output in all three main nmap formats (we will always use it to keep track of our activities)
- **--reason** nmap writes about its interpreted results (recommended)
- **--packet-trace** similar to `-reason` but you will see the traffic traces (you use these to learn about a scan technique and to troubleshoot scans)
- **-n** do not resolve DNS (we will not use DNS because we have already analyzed it manually)

### TCP Scan

Our first scan starts with the following command:

```
nmap -sT -Pn -n --top-ports 10 -oA hhs_5_tcp hackerhighschool.org
```

Which gives us this output:

```
Starting Nmap 6.00 ( http://nmap.org ) at 2012-06-23 04:10 CEST
Nmap scan report for hackerhighschool.org (216.92.116.13)
Host is up (0.23s latency).
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    closed telnet
25/tcp    filtered smtp
80/tcp    open  http
110/tcp   open  pop3
139/tcp   closed netbios-ssn
443/tcp   open  https
445/tcp   closed microsoft-ds
3389/tcp  closed ms-wbt-server
```

```
Nmap done: 1 IP address (1 host up) scanned in 2.04 seconds
```

We found some ports open, some closed and one filtered. What does this mean? It depends on the scan type (in this case `-sT`). And we can use the `--reason` option to see why nmap has inferred a particular State.

```
nmap -sT -Pn -n --top-ports 10 --reason -oA hhs_5_tcp_02
hackerhighschool.org
```

```
Starting Nmap 6.00 ( http://nmap.org ) at 2012-06-23 04:17 CEST
```





```
Nmap scan report for hackerhighschool.org (216.92.116.13)
```

```
Host is up, received user-set (0.22s latency).
```

PORT	STATE	SERVICE	REASON
21/tcp	open	ftp	syn-ack
22/tcp	open	ssh	syn-ack
23/tcp	closed	telnet	conn-refused
25/tcp	filtered	smtp	no-response
80/tcp	open	http	syn-ack
110/tcp	open	pop3	syn-ack
139/tcp	closed	netbios-ssn	conn-refused
443/tcp	open	https	syn-ack
445/tcp	closed	microsoft-ds	conn-refused
3389/tcp	closed	ms-wbt-server	conn-refused

```
Nmap done: 1 IP address (1 host up) scanned in 2.26 seconds
```

Now we know how nmap “maps” replies to states for **TCP Scan**:

- **open**: target replies with a SYN ACK packet
- **closed**: TCP connection refused
- **filtered**: no reply from target

When you find open and filtered ports use other scan techniques to find out exactly why.

### SYN Scan

Another famous scanning technique is the SYN scan. When it's doing this type of scan, nmap sends only a SYN packet without completing the three-way handshake. This is also called a “half-open” or “stealth” scan because there TCP connections are not completed. (Be very clear that while a target may not log a connection, you are still making digital “noise” that can be detected.) Use the -sS scan type as follows:

```
nmap -sS -Pn -n --top-ports 10 --reason -oA hhs_5_syn hackerhighschool.org
```

```
Starting Nmap 6.00 ( http://nmap.org ) at 2012-06-24 12:58 CEST
```

```
Nmap scan report for hackerhighschool.org (216.92.116.13)
```

```
Host is up, received user-set (0.15s latency).
```

PORT	STATE	SERVICE	REASON
21/tcp	open	ftp	syn-ack
22/tcp	open	ssh	syn-ack
23/tcp	closed	telnet	reset
25/tcp	filtered	smtp	no-response
80/tcp	open	http	syn-ack

```

110/tcp open  pop3      syn-ack
139/tcp filtered netbios-ssn no-response
443/tcp open  https     syn-ack
445/tcp filtered microsoft-ds no-response
3389/tcp closed ms-wbt-server reset

```

Nmap done: 1 IP address (1 host up) scanned in 1.81 seconds

The results are similar to the TCP Scan but notice the differences between "full" TCP Scan and "half-open" SYN scan, comparing the results (with `-reason` and `-packet-trace`) using the same target with `-sT`, `-sS` and `-sA` (ACK scan).

### UDP scan

Another scan technique is the UDP scan (`-sU`): knowing the reason is fundamental to getting good results.

```
nmap -sU -Pn -n --top-ports 10 --reason -oA hhs_5_udp
hackerhighschool.org
```

Starting Nmap 6.00 ( <http://nmap.org> ) at 2012-06-23 04:28 CEST

Nmap scan report for hackerhighschool.org (216.92.116.13)

Host is up, received user-set (0.23s latency).

PORT	STATE	SERVICE	REASON
53/udp	closed	domain	port-unreach
67/udp	open filtered	dhcpc	no-response
123/udp	closed	ntp	port-unreach
135/udp	closed	msrpc	port-unreach
137/udp	closed	netbios-ns	port-unreach
138/udp	closed	netbios-dgm	port-unreach
161/udp	closed	snmp	port-unreach
445/udp	closed	microsoft-ds	port-unreach
631/udp	closed	ipp	port-unreach
1434/udp	closed	ms-sql-m	port-unreach

Nmap done: 1 IP address (1 host up) scanned in 2.05 seconds

It can be a little bit confusing. What happened? We see some of the reasons: port-unreach (unreachable, i.e. closed) and no-response (open|filtered). Why? We need more details. We can use the packet trace option and limit the scan to two ports, for example ports 53 and 67 UDP:

```
nmap -sU -Pn -n -p53,67 --reason --packet-trace -oA hhs_5_udp_02
hackerhighschool.org
```

```

Starting Nmap 6.00 ( http://nmap.org ) at 2012-06-23 04:32 CEST
SENT (0.0508s)  UDP  192.168.100.53:54940  >  216.92.116.13:67  ttl=46
id=54177 iplen=28
SENT (0.0509s)  UDP  192.168.100.53:54940  >  216.92.116.13:53  ttl=37
id=17751 iplen=40
RCVD (0.3583s)  ICMP 216.92.116.13 > 192.168.100.53 Port unreachable
(type=3/code=3) ttl=54 id=1724 iplen=56
SENT (2.5989s)  UDP  192.168.100.53:54941  >  216.92.116.13:67  ttl=49
id=33695 iplen=28

Nmap scan report for hackerhighschool.org (216.92.116.13)
Host is up, received user-set (0.31s latency).
PORT      STATE      SERVICE REASON
53/udp    closed     domain  port-unreach
67/udp    open|filtered dhcps   no-response

Nmap done: 1 IP address (1 host up) scanned in 4.15 seconds

```

We found out that 192.168.100.53 sent UDP packets to port 53 and 67 of hackerhighschool.org. What happened here? Port 67 is unresponsive and for 53 we received a Port Unreachable (T03C03).

Port Unreachable means the port is closed, and as for no-response – even if is a normal response for UDP – we don't know if the service is active or not because the UDP protocol can only reply if it receives the correct packets. Can we investigate it more? Yes, using the -sV Service Scan in which nmap tries to send well-known packets for UDP services.

### Service Scan (UDP)

```
nmap -sUV -Pn -n -p53,67 --reason --packet-trace -oA hhs_5_udp_03
hackerhighschool.org
```

```

Starting Nmap 6.00 ( http://nmap.org ) at 2012-06-23 04:44 CEST
SENT (0.1730s)  UDP  192.168.100.53:62664  >  216.92.116.13:53  ttl=48
id=23048 iplen=40
SENT (0.1731s)  UDP  192.168.100.53:62664  >  216.92.116.13:67  ttl=48
id=53183 iplen=28
RCVD (0.4227s)  ICMP 216.92.116.13 > 192.168.100.53 Port unreachable
(type=3/code=3) ttl=54 id=20172 iplen=56
SENT (2.4252s)  UDP  192.168.100.53:62665  >  216.92.116.13:67  ttl=50
id=39909 iplen=28

NSOCK (3.8460s) UDP connection requested to 216.92.116.13:67 (IOD #1)
EID 8
NSOCK (3.8460s) Callback: CONNECT SUCCESS for EID 8 [216.92.116.13:67]
Service scan sending probe RPCCheck to 216.92.116.13:67 (udp)
...and 80 more packets...

Nmap scan report for hackerhighschool.org (216.92.116.13)

```

```
Host is up, received user-set (0.25s latency).
```

```
PORT STATE SERVICE REASON VERSION
53/udp closed domain port-unreach
67/udp open|filtered dhcps no-response
```

We're not lucky this time, since we got the same results. A good hacker can also try specific UDP packets manually, or with the proper client on the standard port 67. We have already used the service scan, the next step for service identification. Learn the well known services on your local machine and do some exercises, then continue with banner grabbing.

## Exercises

- 5.25 Go to <http://nmap.org>, download and install the latest version of nmap for your operating system.
- 5.26 Repeat all the scans in this section using more ports. Have in mind that you need the sudo command on Linux systems, or local administrator rights on Windows.
- 5.27 Create a table reference for all scan techniques mapping state, reason and the real response from the target (packet-trace).

## OS Detection

Knowing services is important to fingerprinting the target machine. Nmap can help again using more options such as -A for all scans and -O for OS detection, using the default ports:

```
sudo nmap -A -Pn -n --reason -oA hhs_5_all hackerhighschool.org
```

```
Starting Nmap 6.00 ( http://nmap.org ) at 2012-06-23 05:38 CEST
```

```
Nmap scan report for hackerhighschool.org (216.92.116.13)
```

```
Host is up, received user-set (0.21s latency).
```

```
Not shown: 971 closed ports
```

```
Reason: 971 resets
```

```
PORT STATE SERVICE REASON VERSION
21/tcp open ftp syn-ack NcFTPD
22/tcp open ssh syn-ack OpenSSH 5.9 (protocol 2.0)
| ssh-hostkey: 1024 cd:27:c2:bf:ad:35:e5:67:e0:1b:cf:ef:ac:2b:18:9a
(DSA)
|_1024 17:83:c5:8a:7a:ac:6c:90:48:04:0b:e5:9c:e5:4d:ab (RSA)
25/tcp filtered smtp no-response
26/tcp open tcpwrapped syn-ack
80/tcp open http syn-ack Apache httpd 2.2.22
|_http-title: Hacker Highschool - Security Awareness for Teens
```

```

110/tcp open  pop3          syn-ack  Dovecot pop3d
|_pop3-capabilities: USER CAPA UIDL TOP OK(K) RESP-CODES PIPELINING
STLS SASL(PLAIN LOGIN)

111/tcp filtered rpcbind      no-response

113/tcp open  tcpwrapped  syn-ack

143/tcp open  imap        syn-ack  Dovecot imapd
|_imap-capabilities: LOGIN-REFERRALS QUOTA AUTH=PLAIN LIST-STATUS
CHILDREN CONTEXT=SEARCH THREAD=REFERENCES UIDPLUS SORT IDLE
MULTIAPPEND CONDSTORE ESEARCH Capability UNSELECT AUTH=LOGINA0001
IMAP4rev1 ID WITHIN QRESYNC LIST-EXTENDED SORT=DISPLAY THREAD=REFS
STARTTLS OK completed SEARCHRES ENABLE I18NLEVEL=1 LITERAL+ ESORT
SASL-IR NAMESPACE

161/tcp filtered snmp        no-response

179/tcp filtered bgp          no-response

306/tcp open  tcpwrapped  syn-ack

443/tcp open  ssl/http    syn-ack  Apache httpd 2.2.22
|_ssl-cert: Subject: commonName=www.isecom.org/organizationName=ISECOM
- The Institute for Security and Open
Methodologies/stateOrProvinceName=New York/countryName=US
|_Not valid before: 2010-12-11 00:00:00
|_Not valid after: 2013-12-10 23:59:59
|_http-title: Site doesn't have a title (text/html).
|_sslv2: server supports SSLv2 protocol, but no SSLv2 cyphers

465/tcp open  ssl/smtp    syn-ack  Postfix smtpd
|_smtp-commands: kunatri.pair.com, PIPELINING, SIZE 41943040, ETRN,
AUTH PLAIN LOGIN, AUTH=PLAIN LOGIN, ENHANCEDSTATUSCODES, 8BITMIME,
DSN,
|_ssl-cert: Subject: commonName=*.pair.com/organizationName=pair
Networks, Inc./stateOrProvinceName=Pennsylvania/countryName=US
|_Not valid before: 2012-01-10 00:00:00
|_Not valid after: 2015-01-09 23:59:59

543/tcp open  tcpwrapped  syn-ack

544/tcp open  tcpwrapped  syn-ack

587/tcp open  smtp        syn-ack  Postfix smtpd
|_smtp-commands: kunatri.pair.com, PIPELINING, SIZE 41943040, ETRN,
STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN,
|_ssl-cert: Subject: commonName=*.pair.com/organizationName=pair
Networks, Inc./stateOrProvinceName=Pennsylvania/countryName=US
|_Not valid before: 2012-01-10 00:00:00
|_Not valid after: 2015-01-09 23:59:59

646/tcp filtered ldap        no-response

800/tcp filtered mdbs_daemon  no-response

993/tcp open  ssl/imap    syn-ack  Dovecot imapd
|_ssl-cert: Subject: commonName=*.pair.com/organizationName=pair

```

```

Networks, Inc./stateOrProvinceName=Pennsylvania/countryName=US
| Not valid before: 2012-01-10 00:00:00
|_Not valid after: 2015-01-09 23:59:59
|_sslsv2: server supports SSLv2 protocol, but no SSLv2 cyphers
|_imap-capabilities: LOGIN-REFERRALS completed OK SORT=DISPLAY
Capability UNSELECT AUTH=PLAIN AUTH=LOGINA0001 IMAP4rev1 QUOTA
CONDSTORE LIST-STATUS ID SEARCHRES WITHIN CHILDREN LIST-EXTENDED ESORT
ESEARCH QRESYNC CONTEXT=SEARCH THREAD=REFS THREAD=REFERENCES
I18NLEVEL=1 UIDPLUS NAMESPACE ENABLE SORT LITERAL+ IDLE SASL-IR
MULTIAPPEND

995/tcp open ssl/pop3 syn-ack Dovecot pop3d
|_sslsv2: server supports SSLv2 protocol, but no SSLv2 cyphers
|_pop3-capabilities: OK(K) CAPA RESP-CODES UIDL PIPELINING USER TOP
SASL(PLAIN LOGIN)
| ssl-cert: Subject: commonName=*.pair.com/organizationName=pair
Networks, Inc./stateOrProvinceName=Pennsylvania/countryName=US
| Not valid before: 2012-01-10 00:00:00
|_Not valid after: 2015-01-09 23:59:59
2105/tcp open tcpwrapped syn-ack
6667/tcp filtered irc no-response
7000/tcp filtered afs3-fileserver no-response
7001/tcp filtered afs3-callback no-response
7007/tcp filtered afs3-bos no-response
7777/tcp filtered cbt no-response
9000/tcp filtered cslistener no-response
31337/tcp filtered Elite no-response

Device type: general purpose|firewall|specialized|router
Running (JUST GUESSING): FreeBSD 6.X|7.X|8.X (98%), m0n0wall FreeBSD
6.X (91%), OpenBSD 4.X (91%), VMware ESX Server 4.X (90%), AVtech
embedded (89%), Juniper JUNOS 9.X (89%)

OS CPE: cpe:/o:freebsd:freebsd:6.3 cpe:/o:freebsd:freebsd:7.0
cpe:/o:freebsd:freebsd:8.1 cpe:/o:m0n0wall:freebsd
cpe:/o:openbsd:openbsd:4.0 cpe:/o:vmware:esxi:4.1
cpe:/o:m0n0wall:freebsd:6 cpe:/o:juniper:junos:9

Aggressive OS guesses: FreeBSD 6.3-RELEASE (98%), FreeBSD 7.0-RELEASE
(95%), FreeBSD 8.1-RELEASE (94%), FreeBSD 7.1-PRERELEASE 7.2-STABLE
(94%), FreeBSD 7.0-RELEASE - 8.0-STABLE (92%), FreeBSD 7.1-RELEASE
(92%), FreeBSD 7.2-RELEASE - 8.0-RELEASE (91%), FreeBSD 7.0-RC1 (91%),
FreeBSD 7.0-STABLE (91%), m0n0wall 1.3b11 - 1.3b15 FreeBSD-based
firewall (91%)

No exact OS matches for host (test conditions non-ideal).

Network Distance: 12 hops

Service Info: Host: kunatri.pair.com; OS: Unix

TRACEROUTE (using port 1723/tcp)

```

```

HOP RTT    ADDRESS
[...]
8  94.98 ms 89.221.34.153
9  93.70 ms 89.221.34.110
10 211.60 ms 64.210.21.150
11 ...
12 209.28 ms 216.92.116.13

```

OS and Service detection performed. Please report any incorrect results at <http://nmap.org/submit/> .

Nmap done: 1 IP address (1 host up) scanned in 57.94 seconds

Using -A is possible to see more data. Specialized plugins fetch more information from a server, perform OS Guessing and use a variant of traceroute that uses different methods than regular traceroute or tracert. For OS guessing more ports are better.

### Exercises

5.28 Scan your own machine with nmap. Is the OS guessing valid?

5.29 Use the traceroute option on nmap using different ports:

```
nmap -n -Pn --traceroute --version-trace -p80 hackerhighschool.org
```

5.30 Are there some differences on nmap traceroute using different ports and tracert or traceroute from your OS?

5.31 Research TCP/IP stack fingerprinting. How do you do it? Is it spoof-proof?

### Using Scripts

Nmap also has a lot of useful scripts for scanning. You can use the `-script script-name` option to load scripts. One interesting script is `ipidseq`, which performs Incremental IP fingerprinting. This script can be used to find hosts for Idle Scan (-sI). This scan uses a problematic IP implementation on zombie hosts to scan other targets.

```
nmap --script ipidseq -oA hhs_5_ipidseq hackerhighschool.org
```

```
Starting Nmap 6.00 ( http://nmap.org ) at 2012-06-23 05:47 CEST
```

```
Nmap scan report for hackerhighschool.org (216.92.116.13)
```

```
Host is up (0.23s latency).
```

```
rDNS record for 216.92.116.13: isecom.org
```

```
Not shown: 971 closed ports
```

### Exercises



5.32 Research Idle Scan techniques. What is it and how do you do it?





## Conclusion

---

Knowing where to look and what to look for is only part of the security battle. Networks are constantly being surveyed, analyzed, poked and prodded. If the network you are protecting isn't being watched then you aren't using the right tools to detect that behavior. If the network you're cracking isn't being watched, you may (may) get away with scanning it. As a cyber security expert, you should know every inch of the systems you are protecting – or testing. You need to know where the weaknesses are and where the strengths are as well, regardless of which side you're on.

Simply gathering up intelligence on a server, such as the operating system and open ports, isn't enough these days. An Advanced Persistent Threat will try to learn as much about your network as it can. This information includes -

- Firewall brand, model, firmware version, and software patches that exists
- Remote connections authentication, access privileges, and processes
- Other servers that connect to the network, this includes Email, HTML, back-up, redundant, off-site, hired or out-sourced services, and even contractors that may have used your network or are using it now
- Printers, fax machines, photocopiers, wireless routers, and network connections in your company waiting room
- Portable devices such as tablets, smartphones, digital picture frames, and anything that might connect to the network.

Even though we have covered many topics in this lesson, system identification covers an even broader area. There is quite a bit of information that flows through networks that identify parts of each device. Each device on the network can be exploited and thus used as an entry point for an attacker. Approaching this daunting challenge requires more than just software. Research your own equipment and learn as much as you can. That knowledge will pay off.

Today's teens are in a world with major communication and productivity channels open to them and they don't have the knowledge to defend themselves against the fraud, identity theft, privacy leaks and other attacks made against them just for using the Internet. This is the reason for Hacker Highschool.

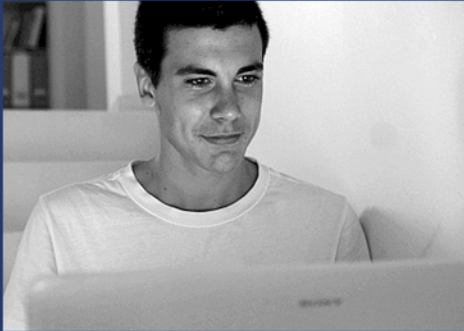
**The Hacker Highschool project is the development of security and privacy awareness learning materials for junior high and high school students.**

Hacker Highschool is a set of lessons and a practical means of making hackers. Beyond just providing cybersecurity awareness and critical Internet skills, we need to teach the young people of today how to be resourceful, creative, and logical, traits synonymous with hackers. The program contains free security and privacy awareness teaching materials and back-end support for teachers of accredited junior high, high schools, and home schooling. There are multiple workbooks available in multiple languages. These are lessons that challenge teens to be as resourceful as hackers, including safe Internet use, web privacy, researching on the internet, avoiding viruses and Trojans, legalities and ethics, and more.

**The HHS program is developed by ISECOM, a non-profit, open-source research group focused on security awareness and professional security development and accreditation.**

# Hacker HighSchool

## SECURITY AWARENESS FOR TEENS



## LESSON 6 HACKING MALWARE



HACKING IS LEARNING  
[www.hackerhighschool.org](http://www.hackerhighschool.org)

ISECOM

Creative Commons 3.3 Attribution-Non-Commercial-NoDerivs ISECOM

[WWW.ISECOM.ORG](http://WWW.ISECOM.ORG) - [WWW.OSSTMM.ORG](http://WWW.OSSTMM.ORG) - [WWW.HACKERHIGHSCHOOL.ORG](http://WWW.HACKERHIGHSCHOOL.ORG) - [WWW.BADPEOPLEPROJECT.ORG](http://WWW.BADPEOPLEPROJECT.ORG) - [WWW.OSSTMMTRAINING.ORG](http://WWW.OSSTMMTRAINING.ORG)

**WARNING**

The Hacker Highschool Project is a learning tool and as with any learning tool there are dangers. Some lessons if abused may result in physical injury. Some additional dangers may also exist where there is not enough research on possible effects of emanations from particular technologies. Students using these lessons should be supervised yet encouraged to learn, try, and do. However ISECOM cannot accept responsibility for how any information herein is abused.

The following lessons and workbooks are open and publicly available under the following terms and conditions of ISECOM:

All works in the Hacker Highschool Project are provided for non-commercial use with elementary school students, junior high school students, and high school students whether in a public institution, private institution, or a part of home-schooling. These materials may not be reproduced for sale in any form. The provision of any class, course, training, or camp with these materials for which a fee is charged is expressly forbidden without a license including college classes, university classes, trade-school classes, summer or computer camps, and similar. To purchase a license, visit the LICENSE section of the HHS web page at <http://www.hackerhighschool.org/licensing.html>.

The HHS Project is an open community effort and if you find value in this project we ask that you support us through the purchase of a license, a donation, or sponsorship.



# Table of Contents

- WARNING.....2
- Contributors.....4
- Introduction.....5
- Viruses (Virii).....6
  - The Polymorphic Virus.....7
  - The Macro Virus.....7
- Game On: The Malware Teacher.....9
- Worms.....10
- Trojans and Spyware.....11
- Rootkits and Backdoors.....12
- Logic Bombs and Time Bombs.....13
- Malware Today.....14
- Feed Your Head: Malware Flavors.....15
  - Mobile Malware.....16
  - An Apple a Day.....16
  - Botnets.....17
  - Free Stuff.....18
  - Delivery Techniques.....18
- Countermeasures.....19
  - Antivirus Software.....19
  - Removing Unwanted Guests.....20
  - Malware Analysis.....20
  - NIDS/NIPS.....22
  - HIDS/HIPS.....22
  - Firewalls.....22
  - Sandboxes.....22
  - Patch, Patch, Patch, Back-up.....22
  - Scramble.....23
- Conclusion.....25



## Contributors

---

Pete Herzog, ISECOM  
Marta Barceló, ISECOM  
Kim Truett, ISECOM  
Marco Ivaldi, ISECOM  
Greg Playle, ISECOM  
Bob Monroe, ISECOM  
Simon Biles  
Rachel Mahncke  
Stephan Chenette  
Fred Cohen  
Monique Castillo

**ISECOM**



## Introduction

---

It's amazing how easily **malware** exploits systems. Dr. Fred Cohen wrote his PhD dissertation on the idea of a virus back in 1984. It was published in 1985. The university found the dissertation profound and initially ridiculous until Dr. Cohen demonstrated his idea. This was around the time of the Morris worm. As soon as the academics saw the potential of a virus, it scared the hell out of them.

The school was afraid his dissertation might give bad guys some bad ideas. So they yanked the published product later in 1985, but the idea had already been planted and demonstrated even before the Cohen report.

The evolution of these products has led them to be made into weapons, for instance **Stuxnet**. The smallest replicating virus was only 90 lines long (and came out of MIT's Core Wars).

Malware can teach security professionals about social engineering, software exploits, stealth and advancements in technology that show the skills of some serious programmers. Newer forms of malicious programs can be extremely sophisticated and require teams of well-funded programmers to create. Other pieces are simple exploits cooked up in someone's bedroom that sneak past security controls and wreak havoc.

Malware didn't originally earn any money or bring any tangible gain (other than occasional ransomware) for the person who wrote the program. This changed over the years as malware creators learned to take advantage of data theft, using credit card information to get access into the global banking systems with the Zeus virus. Since then, this category has thrived.

Lots of new types of malware attempt to take advantage of you through scams, spam, bot-nets and spying. And it has created a billion dollar a year antivirus industry. Hmm, suppose there's any connection here?

When you dissect a virus, you get to see the inner workings of truly marvelous programs. Polymorphic, what an awesome idea! That's intelligent design, in our book. Why don't we have polymorphic Intrusion Detection Systems? It's tough to understand why malware writers use these wonderful techniques that giant software makers don't. Just like a real virus, we can learn how users think and how this software exploits human behavior to survive (and thrive).

Most Malware (or **malicious software**) is a computer program, or part of a program, that has damaging or unwanted effects on your computer. When people think of malware they think of a **virus**, but the term malware is about much more than just viruses. Our fun online friends have created **worms** and **Trojans**, **rootkits**, **logic bombs**, **spyware** and **botnets**. Malware can be any of these, or it can package several of these together. It's difficult to label malware today as simply a virus, worm or even worm/trojan. And that is why the generic term malware is more appropriate for our discussion.

Are you ready to dive in?



The AV-TEST Institute registers over 180,000,000 malware programs beginning from 1984. They add an additional 20,000 new samples per day. Check it out for yourself at <http://www.av-test.org/en/statistics/malware/>.

The problem is that we don't know how they categorize malware. For instance, polymorphic malware could look like lots of different virii, or the same one, or not be detectable at all. And intrusion detection systems will see different things than antivirus programs. Take all of these numbers with a big grain of salt.

## Viruses (Virii)

This is what most people think of when they think of malware. Computer **viruses** came from computer science studies of artificial life – known then as cellular automata – which gradually became more “life-like,” with the ability to propagate (make more of themselves), infect more hosts, become persistent, even hunt and kill each other. They resembled naturally occurring viruses in their behavior and thus the name stuck.

Viruses or **virii** are self-replicating pieces of software that, similar to biological viruses, attach themselves to another program, or, in the case of macro viruses, to another file. The virus is only run when the program is run or the file is opened. That's what makes viruses different from worms. If the program or file isn't accessed in any way, then the virus won't run and won't copy itself further.

Variants of viruses can use different trigger mechanisms like a certain time and date or a keystroke combination. These are usually designed for certain events such as remembrance of uprisings, crimes, acts of war or when the virus author's girlfriend puts out a restraining order against them.

Some malware consists of standalone programs that can appear to be software updates or pictures from someone's visit to the beach. Adobe PDF files have been a frequent launch point for many virus outbreaks, as well as Java. There are lots of reports of pirated software designed to look legitimate but that actually contains malware. That is why you need to look at the software **checksum** before downloading. Yes, **MD5 hash** values can be forged too, but we just want you be to as careful as possible when you download your legal copy of whatever you're getting.

A well-designed virus will evade detection, execute its payload and spread to other machines without the victim ever knowing what has happened until it's too late or maybe never.

Noted authority Dr. Fred Cohen lists some additional nasty ways malware can foul up your system and data:

- randomizing protection settings
- readable files become unreadable
- unreadable files become readable
- writable files become unwritable
- unwritable files become writable
- executables are not
- non-executables are
- setUID (trust level) privileges are set for untrusted programs



- when introduced into a competitor's manufacturing line it can lower the quality of products when the system is controlling a production operation (yikes!)

These days, you'll find that most malware is used as a payload delivery tool. A virus could be used to locate sensitive data in a network, open and keep open a connection for an attack, establish a DDoS, sniff financial information, or disrupt services such as manufacturing and infrastructure. Sophisticated malware will usually have several defense mechanisms, hold multiple exploits, and be written to survive and spread for as long as possible.

### The Polymorphic Virus

Once we figured out what a virus was (after 1988), they were easy enough to detect. They had a certain signature to identify them, either within themselves as a method to prevent re-infection, or simply they had a specific structure which it was possible to detect such as a payload. Then along came the **polymorphic** virus, "poly" meaning multiple and "morphic" meaning shape. This new breed of viruses changed themselves each time they replicated, rearranging their code, changing encryption and making themselves look totally different. This morphing created a huge problem for detection, as instantly there were no longer good signatures to detect viruses.

One of the easiest ways to make a virus change itself is by simple encryption. All the virus author has to do is use a random key generator to change the virus and make it unrecognizable every time the virus was copied. The idea made it difficult for **antivirus (AV)** vendors to locate a common code string for their signature-based AV software. The source string code was different every time because of the encryption.

The AV vendors decided to look at which parts of a polymorphic virus could not change, which would be the encryption/decryption portions of the malware. As you might expect, the virus writers thought of methods to alter their decryption functions and make those as random as the rest of the program. Rogue programmers added alternating dates, random clocks, different algorithms, operations and all sorts of techniques to make the entire polymorphic virus as undetectable as possible.

Virus creators turned to other methods of hiding malware like breaking the code into several segments. The first virus segment would be a harmless PDF but inside the PDF was a scripting call to perform a download of some more of the virus. The second portion of the virus is encrypted so malware detectors don't notice the install.

The creators think of ways to make a virus look anything but like a virus. Since antivirus programs look for files, events, behavior, or suspicious activities that might be a virus, the polymorphic authors decided to mimic functions of the operating system, peripherals and users.

In some cases, the virus replaces the real system files with their own variations. Sweet: every time you, say, open Notepad, the virus replicates.

### The Macro Virus

The **macro virus** makes use of the built-in ability of a number of programs to execute code. Programs such as Word and Excel have limited, but very powerful, versions of the Visual Basic programming language. This allows for the automation of repetitive tasks and the automatic configuration of specific settings. These macro languages can be exploited to attach viral code to documents, which will automatically copy itself on to other documents and propagate.

Since Word and Excel were built to operate as part of a suite of programs (Microsoft Office), a macro virus could take advantage of those special operating system privileges to spread its payload across entire corporate networks with ease. Office programs are allowed to use special (undocumented) developer calls and scripts within



the operating system for increased productivity, which also gives macro viruses access to protected areas of the operating system and network.

In most email clients, you can preview an attachment without opening the email. This is where a macro virus will attack, since a mini program is opening the attachment. That preview will activate the attachment even if the file is labeled "cutepuppy.jpg." File labels can be spoofed. Go check out **Lesson Nine: Hacking Email** for more information.

You can expect to find macro-type malware wherever there are client-side executable scripts, code, forms or sub-routines. This often occurs with HTML5, Java, Javascript and other add-ons that are part of browsers. You can check out more on this in **Lesson 10: Web Security**.

### Exercises

Research these questions:

- 6.1 What was the first virus? Don't trust the first answer you find. Check multiple sources. Five extra points for each classmate you can prove wrong.
- 6.2 Now: what was the first virus to be *released into the wild*? How did it propagate?
- 6.3 The **Klez** virus is well known for **spoofing**. What is spoofing, and how does Klez use it? Assume your computer is infected with Klez. How do you remove it? How do you find out?
- 6.4 Can a virus be useful or serve a useful function besides being evil? Think of the actual purpose of a virus before you make your decision.
- 6.5 What was the purpose of the Stuxnet virus? Based on what you read, did the virus achieve its goal(s)?
- 6.6 You just received an email with the following Subject: "Warning about your email account." The body of the message explains that your inappropriate use of email will result in your losing Internet privileges and that you should see the attachment for details. But you haven't done anything weird with email as far as you know. Are you suspicious? You should be. Research this information and determine what virus is attached to this message. (HINT: When you start thinking of breakfast – you're correct.)



## Game On: The Malware Teacher

The technology classroom stank like old fish and maybe rat fur but was at least laid out in orderly rows. Each desk had a sleeping computer monitor resting on it. Fly-encrusted florescent lights flickered against the sunlight from the row of windows. One student in front yawned and the rest of the class caught it, spreading towards the back rows. The teacher, Mr. Tri, hunched over the screen on his desk, scowling.

If the room hadn't smelled so awful, people would have started eating their lunch, chewing gum or chatting while their teacher struggled with basic computer competence. But in the stench they pinched their noses or breathed through a sleeve. Talking, eating and chewing were the last thing anyone in the room wanted to do. Several already had eyes glued to the clock: fifty-two minutes to go.

Eight minutes late, Jace snuck in the room's backdoor as quietly as she could. The smell made her gag, a loud gag. Mr. Tri detected the sudden noise and the change in air pressure from the door opening. He spun around quick enough to see Jace before she could duck underneath a desk. "Jace. You decided to join us today. What a wonderful surprise."

The teen looked around the room and saw eyes telling her to run while she had the chance, make a break for freedom. Run, Jace, run. Save yourself!

She made eye contact with Mr. Tri and replied, "I apologize for my tardiness. I was sick in the bathroom." It was a lame excuse but the teacher was preoccupied with something even lamer. The hacker lowered her backpack and headed for her assigned desk. Pinching her nose, she asked one of her classmates what was going on. He replied, "You'll find out soon enough."

"Jace, since you missed this morning's introduction, I'll bring you up to date. Someone, one of you students, installed a flu or a cold on these computers," He said accusing everyone in the room. "Now, until one of you fess up to the crime, you all will sit here enjoying this wonderful odor I brought in," Mr. Tri said. Jace looked around and saw the offending fur coat laced with sardines in the pockets, lying in the middle of the room.

"Ms. Jace do you know anything about this computer illness," he said as he pointed at the screen. Jace stood up and moved towards Mr. Tri as if she were approaching a hungry skunk. When she saw the text on the monitor and moved quickly to the keyboard and typed a few commands to see how the machine would respond.


"Hmm," she said to the screen. In a single motion, Jace reached into her backpack, pulled out her eyeglass case, opened it, and selected a USB thumb drive, thinking *gunslinger*. Her right hand was already typing while the left plugged in the drive. "Has anyone been updating the software on these computers?" The long silence had her imagining Mr. Tri's expression.

"What do you mean *update*?" he asked.

"Never mind."

First off, the browser was two versions old, and the Soda HTML extensions were famously exploitable by zero day vulnerabilities. Next, she saw another HTML5 product called Teepee that was two years old, at least. Clicking through folders of tools on the USB drive, Jace found and started Nmap to see what ports the computer was listening on. Nmap came back with a long list of open ports. Jace frowned.

Wireshark showed tons of TCP/IP traffic coming and going from five ports on the



machine. To fix that, she simply unplugged the network connection. The five ports continued to send SYN packets, even without a network connection. She turned her attention to the operating system's start-up files.

Jace wasn't aware that she said "hmmm" all the time when she was examining a computer, but the rest of the class noticed it. They began to gather around her in a semi-circle of curiosity. "Hmmm," Jace said, spotting several unusual programs in the boot loading process and system start-up.

She scanned through each file directory looking for unusual folders and their files' last-access dates. Again, this brought up a list of highly suspicious programs, directories and files.

Every bit of the bad file stuff was under one user name account name. Jace slapped a hand over her mouth just in time, then slowly turned to her right, lowered her hand and said softly to him, "Mr. Tri, it looks like the one who downloaded all this malware is a user named Super Tri." It wasn't until the laughter started that she looked over her shoulder and saw the whole class gathered behind her.

Oops.

### **Game Over**

## **Worms**

---

Worms are similar to viruses in that they propagate, but they use network services to move around. But they don't rely on someone running or accessing a file to trigger the self-replicating code; it executes on its own as soon as it can find a vulnerable host.

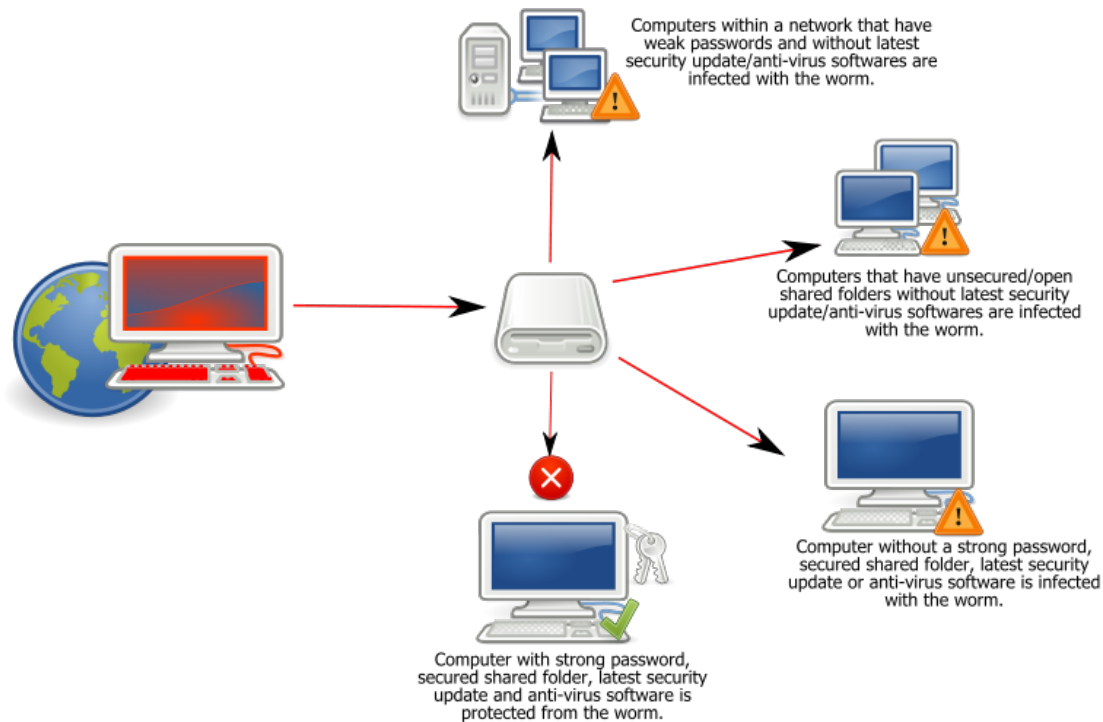
So a worm is a standalone program that, after it has been started, replicates without any need for human intervention. It will move from host to host, taking advantage of an unprotected network or service. Worms have overloaded servers and entire networks because of they're all about multiplying. Depending on how the worm was designed, a worm may not have a specific end point or target.

Worms have been used to map networks, to dig into hidden areas and to report their findings at predetermined connection points. This type of malware can be autonomous or work within a command and control structure.

There are several instances of worms in major systems in which nobody has been able to remove them, determine their purpose or keep tabs on their location. Worms are excellent for reconnaissance because they don't normally have a payload and use covert channels for communication if they communicate at all. If the worm never communicates, it's impossible to tell where it is and what it is supposed to do.

The good news for worms is that they usually only infect a system once. The bad news is about trying to locate that infection to remove the worm: Good luck.

## Worm: Win32 Conficker



**Figure 1.** How the **Conficker** Worm Spread. Photo courtesy of Wikipedia under Creative Commons. <http://en.wikipedia.org/wiki/File:Conficker.svg>


### Exercises

- 6.7 Which OSs were vulnerable to the first worm to spread over the Internet? Find the source code. Yes, really. No, it will not set your computer on fire. You can open the files in your browser and see how much fun the author was having. Curse him or envy him, depending.
- 6.8 Search for videos that show you how to use a worm hack tool. (Hint: use exactly that last phrase.) DO NOT CLICK ON THE LINKS. Considering that worms propagate through infected media like videos, why wouldn't you necessarily trust those videos? Or even those links?
- 6.9 How can you make a good decision to trust or not to trust these videos? For a start, do an Internet search and find the article, "Ten Tricks to Make Anyone Trust You (Temporarily)." Are any of these tricks being used on you?

### Trojans and Spyware

The bread and butter of malware falls into the category of spam. This is **trojan** and **spyware** with a touch of **adware** for an extra kick. The original **Trojan Horse** was created by the Greeks several thousand years ago. (Think about the film "Troy" if you've seen it). The basic concept is that you offer something that appears useful or benign to sneak something nasty into an otherwise secure computer. Examples include: game trailers; e-mail promising naked pictures of your favorite celebrity; a program, tool or utility; a file, such as a PDF; or pirated videos. You will often find them loaded into so-called **freeware** games. The concept of freeware is not to fill a free product with advertising or junk, but somehow that idea got mixed up.

Trojans are pieces of malware which masquerade as something either useful or tasty in order to get you to run them. There are at least two types of trojans. The first breed of



trojan malware pretends to be a useful program, picture, music, movie, or is an attachment within a program. The second type is a fake program that replaces the legitimate one on your system. Once they are inside a system they may do something unpleasant to your computer such as install a backdoor or rootkit, or – even worse – turn you machine into a zombie. Cue scary music in the background.

Your first clue that a trojan has been installed on your computer might be a massive slowdown and loss of resources. Your computer that is, not you. If you have a massive slowdown of your body and/or loss of resources then you have the flu. Go get a shot from your doc. Your computer is not so easy to fix. You are going to need your strength.

You might notice some applications won't load, or programs load that shouldn't be running at all. Don't expect your antivirus software to help because it didn't stop that trojan from installing; why should it work now?

Nope, this is all your mess and if you have one trojan on your computer, expect to find more. Since trojans are just vessels for dumping junk on your computer, you will need to figure out where the trojan came from. What was the last thing you downloaded, opened up or viewed from a friend?

To be fair to your friend, even very large organizations have given their employees, customers, and each other trojans before. Sony did it. Valve did it, twice. Microsoft may have done it but they keep calling them “undocumented features.”

We are going to talk about destroying malware later.

## Rootkits and Backdoors

---

Often when a computer has been compromised by an attacker, they'll want to get back into the machine. There are many variations on this, some of which have become quite famous – have a look on the Internet for “**Back Orifice**.”

Rootkits and backdoors are pieces of malware that create methods to keep access to a machine or network. They could range from the simple (a program listening on a port) to the very complex (programs which will hide processes in memory, modify log files, and listen to a port). Both the Sobig and MyDoom viruses install backdoors as part of their payload.

Both hardware and software manufactures have been accused of installing backdoors in products. Some of this is state-sponsored hacking, while some is just nosy companies. Sony installed spyware on users devices to enforce **Digital Rights Management (DRM)**. China has been charged with installing secret code in routers, hubs, and other products built in their country. These tactics have destroyed consumer trust in brands and products made in certain countries.

When you are dealing with rootkits, expect to lose your **master boot record (MBR)**, the software that boots your OS. Rootkits need to load themselves into memory before the operating system. They do this by hiding portions of themselves in the MBR. This means that successful removal of the rootkit will also damage your MBR.

To fix the MBR you will need to get to the command prompt through system recovery. At the command prompt, enter the following command and then hit enter:

```
bootrec.exe /FixMbr
```

If successful, you should be greeted with the message “**The operation completed successfully**.” The Master Boot Record has been repaired.”

While the above command does fix the MBR, there still might be an error with the system partition's **boot sector** and **Boot Configuration Data (BCD)**, which is to say there might be a *physical* issue to fix. This can happen if you install another operating system



alongside Windows 7, such as Windows XP. To write a new boot sector, try the following command:

```
bootrec.exe /FixBoot
```

### Exercises

- 6.10 Research Back Orifice. What exactly does it do? Who created it?
- 6.11 Research Windows Remote Desktop. What exactly does it do? Compare it to Back Orifice: how are they different?
- 6.12 Let's say you have a computer you want to make your computer dual-bootable, able to start two different versions of Windows. There's a trick to it, though (as usual). In what order must you install the Windows versions for this to work?

## Logic Bombs and Time Bombs

---

Logic bombs and time bombs are malware programs that sit quietly until some condition is met – perhaps a particular bit of data or a certain date. They do not normally propagate. For example: a program could be created that, should the administrator fail to log in for more than three weeks, would start to delete random bits of data from disks.

This occurred in a well-known case involving a programmer at a company called General Dynamics in 1992. He created a logic bomb that would delete critical data, set to be activated after he was gone. He expected that the company would then pay him significant amounts to come back and fix the problem. However, another programmer found the logic bomb before it went off, and the malicious programmer was convicted of the crime and fined \$5,000 US dollars. The judge was merciful – the charges the man faced in court carried fines of up to \$500,000 US dollars, plus jail time.

In 2009, a terminated employee at the mortgage loan giant Fannie May placed a logic bomb that was set to wipe out their 4,000 servers. Luckily, the malware was spotted before it activated. It wasn't such a lucky break for the ex-employee, though.

A logic bomb/time bomb is usually an insider attack committed by a disgruntled employee, contractor, or fired employee with access to the network. Prevention is the best method to stop this kind of threat. Enforce separation of duty so that no person has too much power over a system. Make sure every employee takes a vacation each year so that an evil doer can't keep covering their tracks.

And best of all, if your company fires somebody, take away their network access immediately. Don't let the employee finish up the day or check a few emails. Get them out of the office building, right after you take away their building access keys (codes). Place their network account in a special folder but remove all privileges for user access, especially remote access. That ought to help out some (as long as they don't know other employee's passwords).

### Exercises

- 6.13 What reasonable (and legal) uses might there be for time bomb and logic bomb coding?
- 6.14 How can you detect such a program on your system?



## Malware Today

---

Malware usually gives the attacker access to files or data on your computer, network, tablet or smartphone. Yes, your mobile phone can get malware too. **No computer system is immune from malware – including all personal electronics.**

Your mobile phone, or smartphone, is just a physically small computer. If you are surfing the web, using Facebook or opening email attachments, then you are vulnerable to malware on your phone. Malware may even come preinstalled. The issues are the same as with your computer; for example, you risk having your passwords hacked. More likely, the malware will wait for you to do some online banking and either clean out your bank account or steal your banking credentials and send them to the attacker.

Internet TV is also here. Now you can watch television and surf the Internet all at the same time. You can connect things up and have a “smart” home. Again, you'll have the same issues as you have on your computer. Researchers have hacked into Internet enabled TVs, onboard computers on cars and even refrigerators. Pretty much anything with an onboard computer can be attacked. Be aware that criminals can infiltrate your home, a private space where you feel secure, through your online interactions.

You may think you have nothing of value on your computer or smartphone, but your identity can be exploited. That is, an attacker could take information about you from your computer or phone together with publicly available information about you, say your Facebook photo, and there may be enough information to build a detailed profile. The attacker could try and open credit cards, or take out bank loans, in your name. This is known as **identity theft**. Creditors will then expect you to pay them back for the things the *attacker* bought. It can take years to prove you didn't spend the money and to clear your good name. It could delay you getting a loan to buy that “fast and furious” car you've been dreaming of.

We are digitally connected almost 24 hours a day and we expect our devices to remain part of the Internet even when we are not using them. Malware creators like that. Our phones are synched to our tablets which are synched to our computers which are synched to our cloud accounts. All of this information is at our finger tips and we want access to our music, files, movies, and personal data everywhere we go. Malware creators like that, too.

Currently, a lot of malware targets mobile devices. These devices have the least amount of security yet have the same accessibility to your data as a computer. On your computer, you probably have a firewall, antivirus software, and anti-spyware software installed. Your mobile devices probably don't have any of these protective measures. That needs to change.

Malware creators may be changing tactics away from ransom and denial of service attacks towards complete destruction of an organization's network data. Sony was attacked in October 2014 in a multipronged effort to release incriminating evidence, while destroying vital data in the background. This cyber assault used sophisticated malware against Sony to disrupt daily operations and render critical data useless.





## Feed Your Head: Malware Flavors

*According to Kaspersky in 2013 the top 20 malicious programs were:*


1. Malicious URL	93.01%
2. Trojan.Script.Generic	3.37%
3. AdWare.Win32.MegaSearch.am	0.91%
4. Trojan.Script.Iframer	0.88%
5. Exploit.Script.Blocker	0.49%
6. Trojan.Win32.Generic	0.28%
7. Trojan-Downloader.Script.Generic	0.22%
8. Trojan-Downloader.Win32.Generic	0.10%
9. Hoax.SWF.FakeAntivirus.i	0.09%
10. Exploit.Java.Generic	0.08%
11. Exploit.Script.Blocker.u	0.08%
12. Exploit.Script.Generic	0.07%
13. Trojan.JS.Iframe.aeq	0.06%
14. Packed.Multi.MultiPacked.gen	0.05%
15. AdWare.Win32.Agent.aece	0.04%
16. WebToolbar.Win32.MyWebSearch.rh	0.04%
17. AdWare.Win32.Agent.aeph	0.03%
18. Hoax.HTML.FraudLoad.i	0.02%
19. AdWare.Win32.Ibryte.heur	0.02%
20. Trojan-Downloader.HTML.Iframe.ahs	0.02%

### Exercises

- 6.15 Know the latest malware threats. That is, what new malware threats have emerged today? Go to the website of an antivirus software company and search for their threat monitor. Do an Internet search on "threat research and response."
- 6.16 Are there any threats that affect social networking sites? Look at different antivirus websites. Do they agree on what the latest threat is? How often do malware threats change (how many new ones are released each day)? How often should you update your anti-malware?
- 6.17 What are the issues relating to malware that could arise when you bring your own device (BYOD), say a laptop or smartphone, and connect it to a network at a friend's house, or at your work? What about coffee houses or restaurants?

Attackers have various motivations, but for the modern malware writer it is usually financial gain: to rob people of their money. They don't have to break into your house any more. They may empty your bank account, or spend big in your name. The other way that malware makes money is using your computer to distribute spam or phishing emails. Attackers can make a lot of money this way. That is, until your ISP blocks you.

In an even more ironic turn, crackers have come out in the open, offering malware as a service. An easy search will find you a botnet for rent or a cracker for hire to write



custom malware. Could you trust a malware writer? Can they leave themselves some sort of backdoor into your computer?

## Mobile Malware

Attackers used to focus on the network, but now they can easily circumvent the network defenses by targeting business mobile devices instead. In these next exercises, we will look at building malware for Android tablets and phones. Since almost all of these devices connect to a network in one way or another, there is a pretty good chance that they will log onto the company network at some point. Of course, many of the networks offer remote access but only for isolated segments of that network. This isn't true for web based services though and has become a weak spot for many companies' security.

Android runs as a virtual machine (VM) with a rebranded Linux flavor, designed for small devices but built for speed. The VM is called "Dalvik" in case you were wondering, which is a Java VM with much less overhead (demand for resources). The OS is built in C++ as are all the libraries included in the Android Software Development Kit (SDK). This means that there is a Linux kernel underneath all that GUI fluff. What this also means is that Android can run Java applications inside browsers and as standalone programs.

Third party programs can run native APIs to access built-in functions of Android like the Resource Manager, Telephone Manager and other main controls. This is a major vulnerability since there aren't many reasons for a game to have access to your location, photos, text messages or other private data. The third party applications are often written in Java while system applications are written in C++ (compiled for the processor in use).

### Exercises

- 6.18 Explore your own device's Android APKs (applications). Head over to <http://developer.sonymobile.com/knowledge-base/tools/> and look for APKAnalyser. This free tool will show you how that APK works and which APIs are called. It will also show you a very nice graphic of how that app works as a flowchart.
- 6.19 What are some of the ways we can determine where the device owner is?
- 6.20 Head over to <http://www.xray.io/#vulnerabilities> and take a look at known vulnerabilities for Android operating on an Arm processor. If you were writing malware, which of the listed issues would you try first? Remember that they are listed in alphabetical order, not by popularity. Most phones run on an Arm core.

## An Apple a Day

Now it's Apple's turn to be inspected. Apple has always marketed itself as being safe from malware because of the closed operating system and advanced security features. In truth, security in iOS for all Apple mobile devices depends on users only obtaining software from the official Apple Store. For jailbroken devices, this security feature can be bypassed, which means the Apple Store isn't an effective method to protect those devices. If the company relied solely on the protection offered by users purchasing software through official channels, than it isn't much of a practical plan at all.

Users like to share photos, attachments, messages, links and all kinds of other data. The shared data becomes an entry point for malware infections, just as with any operating system. Part of the reason we haven't seen much malware for iOS is because it is a relatively new popular platform. As iPhones and iPads become more attractive, they also become a larger target for malicious hackers. Now Apple products are a large



player in the mobile community so they are getting much more attention from malware writers.

One of the first prime-time malicious programs for the iPhone is called **Wirelurker**. This application spreads using the enterprise provisioning system, which is a function that allows a company to install custom applications without having to go through Apple Store approval. Luckily the malware doesn't do much beyond loading up a comic book unless the phone is jailbroken. Those phones will have payment information slurped up and sent back to a command and control server. Other proof of concept programs have been demonstrated in the past and shrugged off by Apple as "impossible." Yet the whole idea behind a proof of concept is to show that it is possible.

Anything that connects to the Internet is susceptible to getting hacked by way of malicious links, click-jacking, redirects, Java exploits plus a ton of other vulnerabilities. Apple products are no different.

### Exercises

- 6.21 Wirelurker is thought to compromise up to 800 million Apple users by using desktop to USB infection of iPhone and iPads. Why do you think such a powerful program uses a simple payload of installing a comic book application when it could install more dangerous software?
- 6.22 Look up exploit CVE-2014-4377 to see which operating systems and/or devices may be impacted. How would this exploit work if the user doesn't have Internet access? Safari will open up a rogue PDF even without Internet connectivity. Because this is an issue with the way Safari tags PDFs as images, multiple PDFs could be loaded up without the users knowledge, thus causing a buffer overflow that could be exploited.
- 6.23 The web page <http://www.exploit-db.com/platform/?p=ios> lists a collection of known vulnerabilities in iOS, which affect iPhone and iPad devices. Many of the items listed in the database involve multiple vulnerabilities ranging from Wi-Fi access to camera control. The database for iOS vulnerabilities only goes back to 2010. Which year has the largest collection of documented exploits and what do you think is the reason?

### Botnets

A **botnet** is usually several hundred up to millions of computers that have been attacked, compromised, and have a **rootkit** and **backdoor** installed without the owners' knowledge. They are unwitting hosts to malware, or **zombies**. The attacker (**bot master** or **bot herder**) can remotely command those machines to do anything he wants, from sending spam, to DDoS attacks, to stealing financial information.

If your computer is infected with a bot, it could be used in an attack. An infected computer could be responsible for an attack on the police servers. Legally, you are responsible for the behavior of your computer, just like you are for your cat or dog. What if your computer was involved in an attack on critical infrastructure in your country, like power or water supply plants?

Those kinds of attacks are called **cyberwar**, though that's a dangerous word, because what the police do is very different from what armies do.

Who is behind botnets? Sometimes individuals, but usually organized crime gangs. You certainly don't want to mess with that lot! It is said that the next war (on Earth, not in the galaxy) will be fought in cyberspace.

Would you like to be a botnet hunter? There are a few dedicated individuals who do this. The problem is that, in order to hunt down and bring down a botnet, it's possible



that you'll break some laws in the process. We'd best leave this to the professionals. They need to conduct their investigations within the constraints of the law.

Botnets are also used for **denial of service** attacks (**DoS**). Some of the recent DoS attacks have demanded money in exchange for calling off the attack. In the past, most DoS attacks relied on overwhelming servers with data requests in order to shut the servers down or force a reboot of the system. Some bot-nets have owned tens of thousands of machines which direct a single attack against another network to disrupt communication and thus business.

The bot-nets are individual computers located throughout the world but are controlled by one or more **command and control (C&C)** servers. Each machine is controlled by the C&C servers and told what and where to attack. The C&C servers are themselves controlled by another server called the mothership. By having layers of separate communications between the hackers and the attacking machines, locating the criminals behind the attacks is difficult.

### Free Stuff

People want their favorite music, TV shows, movies and more for free. Think again! How do you think an attacker might spread their malware? An effective way to spread malware (and build a botnet) would be to attach it to something that everyone wants for free. If you use an unsafe OS, don't turn off your antivirus software to speed things up.

What is antivirus software and what other countermeasures could help avoid getting malware? Each year vendors advertise their products as better than anyone else when it comes to virus detection. The largest antivirus manufacturers, Norton, McAfee, AVG, and Kaspersky hire research organizations (magazines, news outlets, social media) to endorse their products. In reality, some of the best software is open source and therefore free. The key is to do your own research to locate tools that meet your needs, not someone else's.

### Delivery Techniques

Very few people would purposely put malicious software on their own system so malware creators need a way to install their products without a user's knowledge. There are several techniques that have proven themselves effective ways to install programs without the owners knowing it. Some of the best methods are repackaging, updates, and attachments (SMS, email, web links, and other malicious URLs).

Repackaging involves the use of real programs that are offered as legitimate software through distribution systems. Malware builders would take that software and add their malware to it or recompile the code to include their payload. Google Play has had a difficult time controlling legitimate programs for Android devices. Since most users don't pay attention to the original size of the correct program, it is fairly simple to replace a good copy with a malicious copy.

Software updates are another area where malware creators can fool users into installing their programs. The malicious programmer is able to persuade a user into downloading an update for some software on the user's computer. The update advisory looks legitimate and may even point the initial URL to an actual software patch. The URL or update link is actually loading up malware while telling the user that their program is being updated. This technique has been done with Adobe, Microsoft, Java, and several other well known vendors.

We've already discussed attachments as a method for distributing malware. Yet, web links are still the easiest way for malicious software to appear of a system. Browser plug-ins for running Javascript, Ajax, PDF openers, PHP, Flash and other programs allow



malware to sneak in from malicious web pages. This means that you have to be on the look out for every possible access point when you are behind your keyboard.

One of the more interesting methods of installing malware is the use of multi-stage insertion. The malicious code is moved onto a user's system in sections to avoid detection. For example, a user might come across a link on a web site or a corrupt execution call inside the web browser from that web address. That single event allows a small program to run in the background of the machine. The program would make a minor adjustment to the system that opens a port or bypasses some security feature. Once that step is complete, another program is loaded that might just be the payload or could be a second-stage attack.

This process of multiple stage program loading can go on for as long as it is needed in order to install the malware and carry out an attack. Usually multi-stage malware is sophisticated enough that it is gathering information from large data sets, such as financial institutions or credit card database information. The massive attack on the American merchandise firm Target in late 2013 used a multi-stage attack that was updated at least five different times during the course of the breach.

Besides your typical file delivery mechanisms, some malware programmers are using communication channels built into computers to propagate. One particular program called **Flame** could use the system's own Bluetooth radio to transmit the malware to other machines nearby. Wi-Fi is also being used to transmit malicious code across airwaves. There has been rumors about some malware using high frequency audio tones to send bits to other devices. This technique has been proven to work in laboratory settings with super quiet environments. Don't expect this to work very well around your house though. You talk too loud and snore.

## Countermeasures

---

There are a number of ways that you can detect, remove and prevent malware. Some of these are common sense, others are technical alternatives. This section provides a brief explanation and examples.

### Antivirus Software

Antivirus software is available in many commercial and open source versions. These all use similar methods. They each have a database of known viruses, and they match the signatures of these against the files on the system to see if there are any infections (this is often called the **blacklist** approach). Often though, with modern viruses, these signatures are very small, and there may be false positives, things that appear to be viruses but are not.

Some virus scanners employ a technique known as **heuristics**, which means that they have a concept of how a virus behaves and try to determine if an unknown application matches these criteria. More recently, antivirus software has also crossed the boundary into **Host-based Intrusion Detection**, by keeping a list of files and checksums in order to increase the speed of scanning.

And yes, Apple Macs get malware too. Current estimates show 5,000 different types of malware for Apple products. There are exploit kits specifically designed to attack Macs. There are now numerous antivirus software programs for Macs. Search online for antivirus for Mac.

Do you use an antivirus on your iPhone or iPad? On your Android phone or tablet? On your Internet-connected TV or disc player? On your Linux box? Why don't you? Is antivirus software mandatory?



There's a list of free software and processes to fix malware problems located at <https://www.soldierx.com/tutorials/Malware-Removal-Guide>.

## Removing Unwanted Guests

Some malware is easier to remove than others. If you come in contact with some nasty virus, trojans or ransomware most antivirus software can remove the threat in a few seconds. There are other types of malware that don't go away very easy and require some research and work to remove, like rootkits. Some types of malware are almost impossible to remove without doing some damage to the data on that system.

One of the first steps to remove unwanted software to figure out what it is: you need to identify the malicious software. Most antivirus software scans will provide you with a name and it's up to you to research that type of software. The key is to have several different antivirus scanners on hand because one is never enough. Once you know the name of the malware, head over to <http://www.malwareremovalguides.info> and see what they recommend for removing that threat.

Each type of malware may need to be treated differently so do your research before you start deleting files off of a machine.

More times than not, if you have one virus you will also have several more hiding elsewhere. It's not uncommon to locate several dozen different strains of trojans sitting alongside adware or rootkits. Deal with each, working on the worst infections first.

## Malware Analysis

Imagine you work for an antivirus company and you discover new malware code that has never been detected before. You will need to assess the damage it could cause and what its intentions are, document and catalog the new malware, and most importantly, name it after yourself. Imagine that!

It would be a really bad idea to run malware on your own computer, or a shared computer connected to a network, for obvious reasons. If malware analysis seriously interests you, there is a lot more you need to know and you are going to need a test system exclusively for this purpose. It is pretty easy to write your own simple malware code or search online for virus code. Please be careful when you're getting into the "dark side" of the Internet. Malware writers are actual people, often with malicious and criminal intent; you don't want to hang out with them or invite them into your home.

With static analysis, it is possible to study a program without actually executing it. Tools of the trade are **disassemblers**, **decompilers**, and **source code analyzers**. Program disassembly involves converting a program file into a listing of machine language instructions; program decompilation is converting machine language instructions into the equivalent higher-level language source code; and static analysis is examining a program without actually executing it.

What if the malware is encrypted? If the code is encrypted, your job just got a bit harder but not impossible. Malicious code that is encrypted is usually a sign that the program is a multi-stage application. The stage of code that decrypts the program could already be somewhere else on the computer. You will need to look around for any script or application that downloaded around the same time that the malware was delivered.

For regular malware the usual procedure is to install and run it in a virtual machine. Depending on the type of malware, whether it is a stand alone executable, a Java app, a script or something else, you will need to decompile the program inside a



sandbox on the virtual machine. This is not for the faint of heart. Most malware has already been decompiled and cataloged by other researchers. You can save yourself time and effort by looking up this information and following it like a roadmap.

Two sites for loading up malware are <http://virusscan.jotti.org/en> or <https://www.virustotal.com/>. These sites run the code past well-known antivirus software and tell you the results. There are several items that you will want to pay close attention to. They are:

- **Propagation:** How the malware spreads
- **Infection:** How it installs, and remains installed despite disinfection attempts
- **Self-Defense:** How it conceals its presence and resists analysis
- **Capabilities:** Software functionality available to malware owner

Hint: Never trust or click on a pop up screen offering you free antivirus software if it says your computer is infected. This is almost always malware!

Keep in mind that the file extension JAR is a compressed Java file. If you ever want to inspect a Java element, take a look at <http://en.wikipedia.org/wiki/Decompiler> or the JAD project at <http://varanekas.com/jad/>.

### Exercises

- 6.24 Go online and find **Sandboxie**. (It never hurts to try a search along the lines of "like Sandboxie" too.) What OS is this used on? How does it work? What applications would you use it with?
- 6.25 Do you use a free antivirus software? No problem, but let's check it out. Go to the vendor's website and search for a comparison of the free software versus the commercial version (it's highly likely there is one). What's the difference between the two? What would you get for your money if you bought the full version?
- 6.26 Search in your favorite search engine for "compare antivirus software." Choose a current review (from this year). Which antivirus product is rated number one? What makes them different?
- 6.27 Now test your antivirus software to see if it has detected all the threats on your computer. First, go to Bitdefender's free online malware detector available at <http://quicksan.bitdefender.com>. Run the online scan. This may take some time so use your time wisely while you wait. Did Bitdefender detect any malware your antivirus software missed? If so, why didn't your antivirus detect it?
- 6.28 Test your AV software using a fake virus file. Go to [http://www.eicar.org/anti\\_virus\\_test\\_file.htm](http://www.eicar.org/anti_virus_test_file.htm) and read the "Antivirus or Anti-malware test file" information carefully. The file you will be testing is not an actual virus but rather designed to look like a virus to your antivirus software. Download the file. Wait to see what happens. What does your antivirus software do? Close your antivirus message, if you get one, and complete the download procedure.
- 6.29 Now click eicar\_com.zip. This compressed zip file contains a fake virus. What happened when you attempted to open the file? Is your antivirus program detecting this file as malicious?
- 6.30 Using the Internet, find an example of a trojan and of spyware. See eicar.com again.
- 6.31 Search the Internet for examples of rootkits and backdoors.
- 6.32 Now consider: could you sandbox your web browser so that anything it downloads is also trapped in the sandbox? Is this an effective alternative to antivirus?



## NIDS/NIPS

**Network intrusion detection systems (NIDS)** is similar to antivirus software. It looks for a particular signature or behavior from a worm or virus. It can then either alert the user (as an **IDS**, or **Intrusion Detection System**), or automatically stop the network traffic carrying the malware (as an **IPS**, or **Intrusion Prevention System**).

## HIDS/HIPS

**Host-based Intrusion Detection systems (HIDS)**, such as **Tripwire**, are capable of detecting changes made to files. It is reasonable to expect that an application, once it is installed, shouldn't change unless it's updated, so watching file information, such as its size, last modification date and checksum, makes it instantly obvious when something is wrong.

## Firewalls

Worms propagate across the network by exploiting vulnerabilities on each host. Apart from ensuring that vulnerable services aren't running, the next best thing is to ensure that your firewall doesn't allow connections. Many modern firewalls will provide some form of packet filtering similar to a HIPS, and will drop packets matching a certain signature.

## Sandboxes

The concept of a sandbox is simple. Your application has its own little world to play in and can't do anything to the rest of your computer. This is implemented as standard in the Java programming language, and can also be implemented through other utilities such as **chroot** in Linux. This restricts the damage that any malware can do to the host operating system by simply denying it the access required. In many OSs this kind of restriction is always built in. In at least one it is not. (Go on, guess which.)

Another option is to run a full machine inside a machine using a virtual machine product like XEN or VirtualBox. This isolates the virtual machine from the host operating system, only allowing access as defined by the user.

## Patch, Patch, Patch, Back-up

That's what most vendors will tell you: apply every patch, apply all patches, let us install all patches, automatically, all the time! That'll make you safe!

Except that:

1. Lots of the patches vendors push out don't apply to your particular system.
2. Every patch you install is yet more buggy, malware-prone code on your computer.
3. Patches break things just about as often as they fix them.
4. Patches can crash or destroy other stable software on your desktop machine – or your server.

Which is to say, the patch game is not the cure-all that the preachers say it is. The appropriate patches are usually beneficial, though they can cause problems. But allowing automatic updates (many, many server admins have learned the hard way) is actually very dangerous. (Microsoft has done everyone a big favor by teaching us this, over and over again.)

The key is to keep your software updated on a regular basis but test the patch before you install it on critical machines. If you can't test the patch, make sure you can reverse the patch installation. Cloud storage is getting cheaper each day and you can set up



automatic back-ups between your computer and your online account. Don't forget about local incremental back-ups. Your data is valuable, keep it safe.

## Scramble

Full disk encryption is another good idea to protect your data and your system from malware. Free software is capable of providing excellent encryption while still making your computer user friendly. One of the cool tricks of using disk encryption is that it replaces your boot sector with its own bootstrap. This keeps your risk of rootkit and boot sector malware infections lower.

Malicious code cannot attack something that it cannot see. Encrypted files keep sensitive information under your control so it would not be sent to the malware operator. This limits the malware's ability to capture useable information from you such as passwords, account details, those photos of you shooting milk out of your nose and your latest report card.

Don't just encrypt your hard drive. Encrypt all media and your phone.

## Exercises

- 6.33 Do a search on the term "automatic update causes." How many things did automatic updates cause? Or at least, how many results do you get?
- 6.34 Research antivirus software for mobile phones. Also search for anti-malware for tablets (e.g. iPad and Android). Are these tools effective? Who uses them?
- 6.35 Research Stuxnet, Duqu and Flame. For each:
- What systems did it affect?
  - What was its payload?
  - What made it different from any other malware?
  - How do you remove them from a system?
- 6.36 Matching Game: Research each of the following and match it to the type of countermeasure that it is:
- |   |           |
|---|-----------|
| <a href="http://www.virtualbox.org">http://www.virtualbox.org</a> | NIDS/NIPS |
| <a href="http://www.tripwire.org">http://www.tripwire.org</a>     | Antivirus |
| <a href="http://www.snort.org">http://www.snort.org</a>           | Firewalls |
| <a href="http://www.checkpoint.com">http://www.checkpoint.com</a> | Sandboxes |
| <a href="http://www.clamav.net">http://www.clamav.net</a>         | HIDS/HIPS |
- 6.37 Research how NIDS/NIPS and HIDS/HIPS work.
- 6.38 Research Firewall solutions on the net. You can analyze your firewall logs and submit your logs to the SANS Internet Storm Center, DShield at <http://isc.sans.edu/howto.html>.
- 6.39 Look up **chroot** on the internet. Read about this type of "jail" or "sandbox."
- 6.40 Draw an **Attack Tree**. Do an advanced search on "site:www.schneier.com."
- 6.41 Malware is never good for anyone, other than those trying to profit from it and risking being caught. Everyone wants to avoid being infected. Take a look at this flowchart to see exactly what you're avoiding when you protect yourself from malware: Inside the business of malware <http://www.computerschool.org/computers/malware/>.
- 6.42 Computerschool.org has an infographic on the Business of Malware. Find it.



## Conclusion

---

Any good immunity from malware comes with a strong understanding of malware. While we can't cover every possible type of malware (because by the time you read this there will be new ones), we've exposed you to some critical points. For instance, you can barely trust the shortcuts on your computer's desktop, and you can't trust at all any file that comes to you unrequested. The key issue is trust, which involves a keen awareness of how vulnerable you become when you give trust.

We don't want you to become deeply mistrustful of everything; that's an attitude that will lock you out of lots of opportunities. Instead, be very clear in your mind when you give anyone access to you, that you're giving them trust. The same principles that make networks secure can make you secure too. Network segmentation that allows only tightly controlled visibility, for instance, is a good practice in both networking and real life.

We also aren't encouraging you to build malware and unleash it on the world or on your acquaintances. Now, probably more than ever in human history, actions have consequences. Don't kid yourself that we couldn't find you if we tried, and we're not nearly as scary as some governments and law-enforcement agencies.

Instead, we want you to see how malware works and become sensitive to the scams in uses. That makes you not just safer from malware, but safer in your whole world.

Use this power only for good, young padawan.

Today's teens are in a world with major communication and productivity channels open to them and they don't have the knowledge to defend themselves against the fraud, identity theft, privacy leaks and other attacks made against them just for using the Internet. This is the reason for Hacker Highschool.

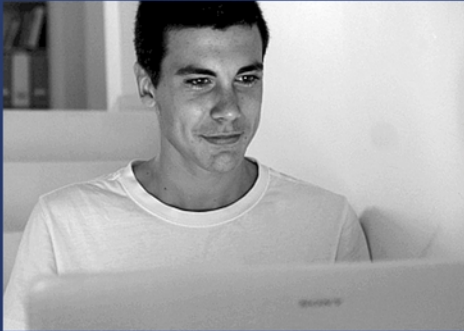
**The Hacker Highschool project is the development of security and privacy awareness learning materials for junior high and high school students.**

Hacker Highschool is a set of lessons and a practical means of making hackers. Beyond just providing cybersecurity awareness and critical Internet skills, we need to teach the young people of today how to be resourceful, creative, and logical, traits synonymous with hackers. The program contains free security and privacy awareness teaching materials and back-end support for teachers of accredited junior high, high schools, and home schooling. There are multiple workbooks available in multiple languages. These are lessons that challenge teens to be as resourceful as hackers, including safe Internet use, web privacy, researching on the internet, avoiding viruses and Trojans, legalities and ethics, and more.

**The HHS program is developed by ISECOM, a non-profit, open-source research group focused on security awareness and professional security development and accreditation.**

# Hacker HighSchool

## SECURITY AWARENESS FOR TEENS



## LESSON 7 ATTACK ANALYSIS



HACKING IS LEARNING  
[www.hackerhighschool.org](http://www.hackerhighschool.org)

ISECOM

Creative Commons 3.3 Attribution-Non-Commercial-NoDerivs ISECOM

[WWW.ISECOM.ORG](http://WWW.ISECOM.ORG) - [WWW.OSSTMM.ORG](http://WWW.OSSTMM.ORG) - [WWW.HACKERHIGHSCHOOL.ORG](http://WWW.HACKERHIGHSCHOOL.ORG) - [WWW.BADPEOPLEPROJECT.ORG](http://WWW.BADPEOPLEPROJECT.ORG) - [WWW.OSSTMMTRAINING.ORG](http://WWW.OSSTMMTRAINING.ORG)



## WARNING

The Hacker Highschool Project is a learning tool and as with any learning tool there are dangers. Some lessons, if abused, may result in physical injury. Some additional dangers may also exist where there is not enough research on possible effects of emanations from particular technologies. Students using these lessons should be supervised yet encouraged to learn, try and do. However ISECOM cannot accept responsibility for how any information herein is abused.

The following lessons and workbooks are open and publicly available under the following terms and conditions of ISECOM:

All works in the Hacker Highschool Project are provided for non-commercial use with elementary school students, junior high school students and high school students whether in a public institution, private institution, or a part of home-schooling. These materials may not be reproduced for sale in any form. The provision of any class, course, training, or camp with these materials for which a fee is charged is expressly forbidden without a license, including college classes, university classes, trade-school classes, summer or computer camps and similar. To purchase a license, visit the LICENSE section of the HHS web page at <http://www.hackerhighschool.org/licensing.html>.

The Hacker Highschool Project is an open community effort and if you find value in this project, we ask that you support us through the purchase of a license, a donation, or sponsorship.



## Table of Contents

Introduction.....	5
Continued Reading.....	5
Reasons to Attack.....	7
Just Because (Attacking for Fun).....	7
Cyber-Crime (For Profit).....	8
State Sponsored/Cyber Warfare (Bits instead of bullets).....	9
Feed Your Head: Stuxnet and Worse.....	11
Hacktivism (Is it contagious or do we need a vaccine?).....	12
Espionage (What's in your lunch box?).....	13
Feed Your Head: An Analyst Tip.....	14
Angry Employees (I got fired for playing video games).....	14
Types of Attacks.....	15
Spoofing (Who's at the door?).....	16
Game On: Try, Tri Again.....	18
Application-Layer Attacks.....	21
Remote Access Toolkits (RATs).....	23
DOS and DDOS.....	24
Slowdowns.....	25
Unicorns.....	25
Pay Per Service.....	26
Getting to Post.....	26
DDoS By the Numbers.....	26
Malware (Nobody liked me as a kid).....	27
Teaching a man to phish (Hacking the Wetware).....	28
Hacking the Technology that Surrounds Us.....	28
Attack Signatures: Detecting Different Types of Attacks.....	28
The Spoof.....	29
Sniffles.....	30
Packet Sniffing.....	31
Enter the Shark: Wireshark.....	31
Wireshark Fundamentals.....	32
Decoding the Packets.....	35
Summary.....	36
Protocol Hierarchy.....	37
Conversations.....	38
Hubs, Routers and Switches.....	41
Intrusion Detection Systems.....	42
Honeypots and Honeynets.....	43
Types of Honeypots.....	43
Building a Honeypot.....	44
Conclusion.....	46



## Contributors

---

Pete Herzog, ISECOM  
Marta Barceló, ISECOM  
Chuck Truett, ISECOM  
Kim Truett, ISECOM  
Bob Monroe, ISECOM  
Greg Playle, ISECOM  
Marco Ivaldi, ISECOM  
Rob Dodson

# ISECOM



## Introduction

---

You wake up feeling fresh after a great night's sleep and you look outside. The sun is shining and birds are singing, just like a Hollywood movie right before a monster comes out to attack the town. It's starting out to be a wonderful day, as long as the monster doesn't show up. You turn on your computer to check your email and messages. But wait! The computer, your loyal companion, isn't working the way it is supposed to. It sputters. It makes awful noises.

Files don't open up and applications are sluggish. Your network connection is frozen. Modem lights blaze in multiple colors even though you're not doing anything.

You check all of your cables, reboot the computer, scratch your head, kick the table but nothing seems to make your faithful digital device operate normally. Out of the corner of your eye you see the sunny sky fill with ugly dark clouds of despair. The hard drive sounds like someone threw a bunch of marbles onto the platter.

Yet, the computer sort of works. It kind of operates. It isn't completely dead but it isn't exactly the beautiful beast you know oh so well. Your security software won't run and your anti-malware programs refuse to turn on. Off in the distance you heard the fictitious roar of that Hollywood monster. You have been attacked!!!

Should you run and hide or stand your ground and face the beast with hopes of destroying the monster attacking your system? Running away isn't such a bad idea but Hacker Highschool doesn't recommend it. Let's take a deep breath, crack our knuckles, and think about our problem. You can fix this or at least gain control of the situation if you continue reading.

## Continued Reading

There are two predominant types of attacks: one is an attack against a computer and the other is an attack against a network. Other attack types that come to mind: application attack, human attack, physical attack... Oh wait! These are the OSSTMM **channels**.

A computer attack is a systematic attempt to gain access, disable things, delete content, or take over a computer or system of computers. Network attacks are a lot like computer attacks, but they add the additional element of probing the parts and pieces that make up that network: hubs, routers, switches and firewalls (use your imagination here).

Throughout this lesson, we will be discussing aspects of the Open Source Security Testing Methodology Manual (OSSTMM). Yes, it is a document for professional security people, but it works nicely for illustrating points of interaction in computers and networks. It's also brilliant. You see, these interactive points are potential weaknesses for computers and networks, so we need to be aware of and control those interactive points. If we don't have controls, or even worse don't know about access points, we will have entry locations for attacks, sort of like holes in a shoe. You don't want holes in your shoes because your toes will fall out.

Do you see how that works? Interactive points are dangerous and need to be controlled. No control means you have holes in your security plan and you have provided an attacker with wonderful entry locations to your networks or computers. Plus, nobody wants to lose their toes.

Attack analysis is not a forensic examination, nor a postmortem report that would be done after an attack. Attack analysis is an active process that needs to be part of your





proactive defense measures. You don't want to wait until your network is under siege before you start exercising your options. Let other folks draw up the graphs, log the events and scream into their cell phones during an attack. You are the one who has to keep calm and be a leader during aggressive network attacks. After all, you've engaged in professional study of the field. Right now.

Our goal is to show you the "whys," the "whats," the "hows," more "whats" and then a few more "hows" about attacks. ("Who" is always a very tricky question.) We plan on showing you why you might be attacked, who might be attacking you, what types of attacks are out there, what an attack looks like, how attacks are pulled off, what you should do when you are attacked and what you need to do after you've kicked the attackers butt.

Does this sound good to you?

Then read on.



## Reasons to Attack

First off, let's all agree on what constitutes an **attack**. To attack something means to deny, disrupt, destroy or limit a target's capabilities. You can puff your chest out if you want. Go ahead, we're not watching. That sounds so cool; deny, disrupt, destroy!

Network monitoring and remote port scanning aren't attacks. This means that intercepting data is no more of an attack than reading your neighbor's mail. You could argue that data interception or man-in-the-middle **exploits** do degrade a target's resources but those don't really hinder the adversary at all. The target can still conduct business; you just get to see what sort of business they are doing.

What they are doing is reconnaissance. If you are going to attack something you need to know what it is, how it works, and what kind of defenses it has. That is the purpose of port scanning, network monitoring, and so forth by the bad guy. It can also be an indicator....but we will come back to that later.

So, let's go with "deny, disrupt, destroy and limit" as a starting point for how we will talk about attacks. OSSTMM looks at an attack as a threat applied to a known vulnerability, within a system or something like that. As discussed before, vulnerabilities are weaknesses, or limitations in OSSTMM-speak, that leave you open to an attack. Exploits utilize these limitations to make successful attacks against a target. Does any of this make sense?

Okay, moving forward

The game of chess is about attacking and defending. Water polo is about attacking, not drowning and defending. Football is about attacking, taking your shirt off when you score a goal, and defending. There aren't many competitive activities that don't involve some form of offensive play. For those events that are passive, you know that they don't sell many stadium tickets. Humans are aggressive in nature. We love a challenge.

The Internet is its own game with its own set of challenges but very few rules. Along with all the incredible tools and knowledge at your fingertips, there are some incredibly bad people who take advantage of the connectivity provided by the Internet. Law enforcement has slowly come around to enforcing some of those rules but they are hampered by technology and jurisdiction.

### Just Because (Attacking for Fun)

Any article, media report or blog on cyber-attack statistics is incomplete (and most will tell you that) because many attacks go undetected, unreported or unnoticed. Imagine being a sophisticated hacker mastermind who creates this amazing attack against someone or something, only to have that attack ignored. Isn't that just plain rude! You go through all that effort of identifying a target, figuring out a proper attack vector, setting up the ploy and then conducting an attack, only to have all that hard criminal work go unnoticed. Some people just don't appreciate a good attack when they don't see one.



Hacking for fun isn't as popular as it once was. Maybe it never will be again. These days very few people are willing to risk long prison vacations just to thumb their nose at a major organization. Jail time takes the fun out of attacking networks. Yet there are still some hard core hackers out there that are willing to spend a few decades behind bars. These are the people you read about in the news mainly because they are newsworthy. Who in their right mind is determined enough to go after a major network, knowing that they'll be found some day?

As for those who still do attacks for fun, they usually build their own tools. This means that they need to locate vulnerabilities within systems and code a program to exploit a particular vulnerability. That is not an easy task unless they stick with social engineering or highly insecure networks. A small few will pay for an exploit service that is being monitored by international law enforcement. Some of the hackers live in countries that do not care if they attack certain targets in other countries. We'll get to that in just a moment. These days, cyber-attacks for fun occur to benefit the attacker, whether the benefit is bragging rights or something to pad their resume with.

An example of a hack for fun that turned into media attention bragging rights was performed by **Darwinare** in November 2012. Hactivist Darwinare gained access to the Australian Defense Force Academy and earned himself online chat interviews with two reporters. When he was asked about his attack on the military academy he was shy enough to say, "Oh, that old thing: I was bored. So simple, took like three minutes." After that project, he had to drop out of sight for seven months. So much for fame and fortune.

Cyber-attacks committed for reasons beyond fun fall into the rest of the categories below.

### Cyber-Crime (For Profit)

Face it: crime is profitable. If it weren't, nobody would be doing it. Jails are full of criminals who wanted cash but didn't expect to get caught. Yet, there they sit. If you remember, we mentioned two problems that face law enforcement when it comes to cyber-crime: technology and jurisdiction. Technology isn't getting any easier to understand and there isn't a slowdown in the amount of new technology being developed.

Criminals have taken advantage of technology since the invention of the wheel. There are cave drawings showing a cave man as he is being wheel-jacked by a cave woman. There is also a cave drawing of a cave man being attacked by a large dinosaur while he's making a cave drawing, in one of the first known instances of censorship. The first cave person attack was for profit while the second attack was for fun.

Cyber-crime makes up roughly 50% of all reported attacks, according to one source. As of this writing the world's number one spot for records lost to a data breach is 152,000,000 records. Yes, one hundred fifty two million records!

Ouch.

### Exercises

7.1 Find out what company lost those 152,000,000 records.



- 7.2 There is an open-source organization that maintains a daily view of data breach events. Find it.
- 7.3 Find the source of the 50% statistic above. What is the perspective or agenda of that source? Should you trust them completely?

Individuals commit a lot of today's cybercrime, but lots of others join cyber gangs. Conducting a cyber-attack by yourself means that you get to keep all the profits (if any), don't have to worry about being identified by your partners when they get caught and you can control the entire operation. If an attacker is part of a group, then she has to remember that the group is only as smart as the dumbest person in it.

The think tank Ponemon.org publishes an annual Cybercrime Study that focuses on the US, UK, Germany, Australia and Japan. Taken at the "big picture" level, the study shows the increase in successful attacks up 30%. Add one historic theft of \$45 million from credit cards and you have a scary idea of how much job potential you theoretically have as a criminal.

Here is something to keep in mind whenever you hear about a massive cyber-attack that stole millions of dollars: it is incredibly difficult to put a dollar figure on any type of crime, more so with cyber-crimes. When a new article comes out and says that a company lost three trillion dollars to a hacker, well, they are stretching the truth. Like really stretching the truth.

Cyber-crime attacks come in several flavors, ranging from identity theft to credit card skimming. The most popular attacks right now are denial of service attacks targeted against online retailers. See the **DOS and DDOS** section below for more information.

After denial of service attacks, the next most popular crime is simple theft. Theft is theft, plain and simple. Digital thieves steal people's identities, credit card information, bank account access, tax refunds, medical records, corporate confidential data and research. If something is stored electronically, it can usually be taken (or copied) by someone else. Each of these areas brings in billions of dollars every year for criminals and costs consumers trillions of dollars to recover and protect against future losses. (Nah, we would never inflate those figures.)

Many of these types of interactive point attacks range from stupidly simple to incredibly sophisticated. In the case of Darwinare, he claims his attack took three minutes. The Darwinare breach must have been a simple uncontrolled access point such as an easy-to-guess password or an unpatched vulnerability. Other attacks can take years to pull off or are completed in phases that span many years.

Since cyber-criminals act like real ones, they often commit the same type of crimes. Ransomware is a prime example of kidnapping or hijacking your computer system. Up pops a box saying you have a virus, trojan or some other type of malware. Since you have been visiting those naughty sites, you figure it might be true. The next thing you know you someone is making demands for money. Pay or you won't get your system back. Do you trust the message? Do you pay the ransom?

### State Sponsored/Cyber Warfare (Bits instead of bullets)

For typical military doctrine, there are several warfare components besides "Shoot at enemy." There are communication components, logistics, transportation, weapons, operations and stuff like that. For the operations segment, intelligence and information operations are critical parts. Within the **information operations (IO)** spectrum there is a tiny slice of ops called cyber warfare. That slice is further split into offensive and defensive operations. Cyber warfare isn't only about attacking an enemy's network, it's also about protecting your own network against attacks.



It's well documented that nations train, prepare and practice cyber warfare on a daily basis. It is also well documented that cyber warfare is considered an **Act of War** by those same entities. "Act of War" means that if one nation did this particular act, like drop bombs on another country, the bombed country has an internationally recognized reason to fight back. Without the backing of the international community, that nation is just committing an unprovoked attack on another nation. Unprovoked attacks on other nations are a bad idea. Not that they don't happen a lot.

Because of the international disgust for Acts of War, cyber warfare has morphed into clandestine operations or focused on intelligence gathering. Those nations that continue to commit cyber warfare claim the actions are beyond the control of that government or are committed by separatist groups. Overwhelming evidence suggests otherwise and is beyond the scope of this lesson. However, we still want to discuss this military action and what it means to you.

Cyber warfare is funded in the same way all military assets are and these functions are operated as an extension of the military arsenal. Tanks and jets are expensive to build, buy, operate and maintain. The same holds true for any cyber warfare unit. Enormous amounts of money are invested into these areas by most nations. In most cases, these units are manned by the best and brightest hackers in that country.

The premise of the units is to build an arsenal of digital weapons that can disrupt or destroy another country's ability to conduct warfare. The most effective weapons consist of zero-day exploits, which can target software, operating systems and control mechanisms. Some weapons are shock and destroy worms that move through a variety of systems to delete data. These programs do not rely on a particular operating system. They are the ultimate in cross-platform malware, are built to avoid detection yet are extremely efficient. Many of these weapons are only a few kilobytes in size.

Cyber weapons consist of three main components. These are the **delivery mechanism**, the **navigation system** and the **payload**. They are the same components used in missile technology but cost a fraction of the price. Missiles require a launch pad and are easy to spot on surveillance satellites. Cyber weapons barely need any kind of launch facility and can be activated from almost any location.

State sponsored hackers are privy to the source code of every piece of software imaginable. This enables the cyber soldiers to look deep into each program. Based on known information, almost every program has a bug every 5-10 lines of code. Being able to see the code allows these professionals to identify zero-day exploits, buffer overflows and system weaknesses in everything.

Military objectives range from aircraft avionics to artillery control computers, radar-jamming systems and infrastructure support controls. Remember that **all is fair in love and war**.

## Exercises

7.4 Research **EMP**. What is it?

7.5 You are a security consultant. Your client is nervous about the potential for EMP disruption of his giant cookie factory. Find the Executive Report from the federal commission charged with studying the threat of EMP. Give it a quick scan. Now prepare your short report to your client: is his facility vulnerable? Is an attack possible, or likely?

7.6 You are a hacker. The giant cookie factory next door is driving you crazy. How can you use EMP to knock out that factory?



## Feed Your Head: Stuxnet and Worse

If you would like to get a better understanding of state sponsored cyber weapons, take a look at **Stuxnet**. This is a good example of a weak weapon built by amateurs compared to newer systems. A properly designed national defense weapon would not have been recognized and would never have been allowed to be seen in public. One contributor's opinion on Stuxnet was based on the errors that were involved with its release:

"It should have never been detected. It should have never left the computers it was assigned to target. Stuxnet used some stolen certificates, and a few neat tricks in DLLs but it was discovered. That is careless.

"Let's look at the NSA's woes as a comparison.

"The only reason the NSA was caught was due to a rogue sysadmin. Otherwise, everything they have done is invisible. All of the sources seemed to show the FBI and CIA as buyers of Facebook data. There was no clue that the NSA was tapping at the source. Why buy the cow when you can own the farm?

"Stuxnet used several exploits that were considered zero-day because nobody had thought about those attack methods before. The primary reason for this neglect was because the malware was designed to attack **SCADA** systems, not networks. The worm exploited **PLCs [programmable logic controllers]**, not routers. A good portion of the industry has been screaming to add more protection to critical infrastructure. That is exactly what Stuxnet did, it attacked a tiny portion of infrastructure.

"The street traffic lights of Israel were attacked in 2013. That apparently didn't make the news. Stuxnet made the news because it was something different, something sexy. Causing traffic jams is state-sponsored cyber attacking and is a great example of small hits that slow your target down. Little nibbles like that cause eight-hour congestion in a main city.

"Aurora sort of made the news and all the attacks by the Chinese make the news, but not the small hit-and-runs. The attacks that keep me awake at night are the ones we can't see and will never see. Those exploits won't be in the news either because we created those attacks ourselves. We provided the ammo thanks to the pictures we posted on Facebook, the emails we sent talking about our vacation, the text messages we get about traffic jams, the cookies we gather when we shop online, the medication we refill online, the life insurance information we update via the web and all the million bits of data that are picked up all around us.

"Everyone has a camera on their cell phone. Everyone. A friend of mine is a cop and I asked him if video cameras are helping or hindering his job. He said that the judge only sees a snapshot in time so all those video clips are hurting their ability to be effective. Now, lets magnify that a million times over the next ten years of your life. Everything is digital and everything about you is traceable. That is much more dangerous than a state sponsored Stuxnet worm."

Once upon a time, SCADA systems were considered safe from attack, working in isolation with little outside contact like true introverts. But they require



administration and maintenance like all other systems. This leads to the predictable human vulnerabilities.

So, thinks the administrator, *if I don't let the techs bring in USB sticks they'll complain*, and voila! Someone plants a backdoor.

### Exercise

7.7 In your web browser, go to a search site. Search on the terms "SCADA hacked" followed by the current year. Scan a few of the results; there will be plenty.

Now add the term "cheat sheet." How's your luck with this? We'll bet it's pretty good.

As a side note, SCADA infrastructure security is usually not concerned with **Confidentiality** (because there's no valuable information to steal from SCADA networks, except maybe access credentials). However, it is very concerned with **Integrity** and **Availability**.

## Hactivism (Is it contagious or do we need a vaccine?)

Our official position is that activism in any positive manner that furthers a just cause is okay (the negative is cyber-bullying or worse). Law and justice are two very different things. However, not everyone can mobilize people, pay for full-page ads in the NY Times Op-Ed or drum up thousands of signatures. So, you need to use what you know and do what you can with what you have. If you're a hacker then what you know is **hactivism**.

Protest is the basic human right to express our opinion. When we add our voice to an issue, we are exercising our freedom of speech. If we add the connectivity of the Internet and hacking tools to a cause that anyone thinks is worth fight for, then we have hactivism. This activity may be viewed as heroic to some people but considered anti-social disobedience by others.

The main difference between hactivism and criminal hacking is the valor of it. When you see sit-ins and rallies and such protests where people defy an authority they feel is wrong, they risk arrest for a cause. They are willing to be arrested to stand up to what's right. But if you use acts of vandalism or theft under the cover of anonymity then you're just being a criminal.

In this sense, hactivism is something a hacker uses where ingenuity can be greater than deep-pocket resources of the offending group. Where one can't afford to have an organization call thousands of people with a message, a hacker does the same with a script, free telephone services and an audio file. The idea is to be within the legal confines of what's allowed yet making your point.

Back in the early days of Internet when people carried pagers instead of phones and still used modems to get online, there once was a hacker who made a point. This hacker was mad at a national insurance company that refused to reimburse payment on medical care they supposedly covered. Phone calls went nowhere. Letters went



nowhere. The media didn't care or were paid not to report on this big corporation. What's a hacker to do?

This hacker ran a program called Tone Loc to determine the range of pager phone numbers in the local exchange and then dialed them all with the phone number to the local corporate boss. Local calls were free after all. Then he did it again sending out the number to their claims desk. And again and again and again. He called thousands of pagers every day creating a **Smurf Attack**, where nearly everybody who got that number on their pager called it back to ask why they paged them. After a few days, so many people were upset with this that it made the news. Once it was in the news already, reporters were more than happy to print and report on other negative stories about that corporation.

And as the hacker got his story out to the news about unpaid claims, many other people followed with similar stories. This led to a local investigation which found lies and tricks used by the corporation to avoid paying out legitimate claims. This led to a national investigation and criminal charges and huge fines against the corporation. Back then there was no word for hacktivism but that's what it was. It was genius! And possibly quite illegal.

### Espionage (What's in your lunch box?)

A company that's trying to purchase another company is playing something like a game of poker. If the other player knows what cards you have, you'll have a tough time winning the game.

In 2009, the FBI contacted the CEO of a popular soft drink company to tell them that they were victims of a massive attack. The attack took several months but it also happened when the soft drink company was conducting a major acquisition deal with an overseas drink manufacturer. The deal fell through for unknown reasons but it might be reasonable to suggest that the vast amount of internal data taken during the attack had something to do with the failure. The other overseas company knew which cards were in play.

We call that **espionage**.

Espionage is lying, cheating, stealing, hurting, maiming, killing and everything in between that involves gaining information. There are three reasons for espionage: military, political and industrial. Military and political were covered in State Sponsored/Cyber Warfare section above. So, we turn your attention to industrial espionage. Isn't that cool of us?

Just nod your head in agreement.

Industrial espionage is just another fancy name for an attack that has a business purpose. The purpose is to gather intelligence, disrupt business or slow down another competing company. Research and product development are expensive and difficult to keep secret. An organization can save themselves lots of cash by stealing the work of another company.

The same principle applies when your classmate looks over your shoulder during a test. He doesn't know the answer but you do. If you are caught, you both get in trouble even if you had nothing to do with the cheating. You could call that academic espionage.

In a polite society, there are legal, moral and ethical issues that keep companies from spying on each other. So, they hire other companies to do that work for them. Business intelligence is a massive sector. This work would be considered illegal if the true customer were ever located. So, they have **Non-Disclosure Agreements (NDAs)**. These





written contracts forbid one party from saying anything about the other party if they are ever caught.

Lots of fun, eh?

### **Feed Your Head: An Analyst Tip**

One of our contributors, a professional security analyst, gave us this valuable piece of insight:

“Since there are a lot of similarities between espionage and state-sponsored hacking you might wonder how we would catch them. Like most spies, they are in their greatest danger when they are either trying to get away or pass their information. So it is with these two efforts. The information, to be of use, must be sent somewhere. Too often we fail to monitor outbound traffic, we are trying to keep the bad guys out. But just as spies get in, so do the bad guys. So it is better to watch for our secrets to be passed out of the network.”

Traditionally, security pros are looking hard at inbound requests, probes and attempts at intrusion. That's not stupid, but if you're not monitoring *outbound* traffic, you may be missing the most critical information you can get: what the crooks are stealing.

### **Exercises**

- 7.8 Now you are selling your client on SET (the Social-Engineer Toolkit). What the heck does it do? Are you selling him a product, or services?
- 7.9 You want to find out if that annoying cookie factory next door has any web cams you can access, or anything else for that matter. You've heard (just now) that there's a place online with a name like "Shodan" where you can look for these gizmos. Find that site.
- 7.10 What additional kinds of information is available from that site?  
 How can it be used for to help with analysis after an attack?  
 How could you use it before an attack to make yourself look like a genius?

### **Angry Employees (I got fired for playing video games)**

Getting fired from a job is a part of life. It happens.

Once someone is fired, they usually box up their cubicle pictures, are escorted out the front door by some nice men with big sticks and then sit in their car cussing for a while. Once they are done throwing a tantrum, they build a resume and start looking for a new job. Depending on the employee, the cycle may repeat over and over again.

Sometimes an employee (ex-employee) feels as though they were unjustly fired from their job. These people like to get revenge. Those people who work in IT love using their skills to sabotage the companies' network, plant logic bombs or destroy every account in the system. Yes, those ex-employees get their revenge but they also get a knock on their front door by the local law enforcement a few days later.



These scenarios happen all the time and they never have a happy ending for anyone. The attacks are very successful mainly because the employee knows the inner working of the network. Sometimes they are the only people who have access to certain parts of the network or they are the only ones who know how to do a vital function on the network. Unfortunately, all these characteristics make the perpetrator very easy to identify.

Sometimes an employee feels angry because they were passed over for a promotion or given a crappy parking space in the company lot. (You know they're trying to tell you something when they make you park next to the dumpster.) In those circumstances, the employee has time to plan the attack, place Trojans and logic bombs, set up command and control remote servers and generally plot terrible revenge.

One recent plot included an ex-employee conducting attacks from a company domain in other countries. When the attacks were investigated, the company was found liable for the massive attacks. It took months before the reasons for the attacks could be uncovered, but they were inevitably traced to the fired employee. In the meantime, several countries were very upset with the innocent company and banned them from conducting international business. Imagine dollar signs flying out the window.

## Types of Attacks

---

Now that we've looked at the reasons for attacks, we're going to explore some of the popular forms of attacks. Remember that this lesson will focus on attacks that deny, disrupt, destroy or limit computer or network capabilities. The question of what is an attack and what isn't is tricky. Malware is a perfect example of an attack, like Stuxnet. That tool seemed to have been designed to cripple the centrifuges used for making nuclear fuel. It was an attack.

Sniffing emails and reading company data are not attacks because no real tangible damage is done (yeah, reputations vanish and lawsuits fly but there's no direct impact on the utility of the system itself). Man-in-the-middle exploits may be considered attacks only if the attacker inserts erroneous data into the packet stream that may cause some (tangible) damage along the way to routers, servers or data.

Likewise, cross-site scripting, buffer overflows and SQL injections aren't attacks. They are **exploits**, a means to gain access to a network to launch an attack. Brute force is not an attack; it is a method to get passwords to obtain access through elevated privileges. What someone does next might or might not be an attack. This is like a boxer sparring. He may swing at you, but he hasn't hit you with that haymaker and knocked you out. Yet.

It would be impossible to name all the different kinds of attacks that are available, known or being created at this very moment. Cyber-attacks take several forms, use alternate methods of executing their mission, rely on a variety of tools to make that attack successful and can morph themselves over the lifetime of the attack. To make things a bit easier to understand, we're going to cover generalities of attack structures.

Buildings like houses and skyscrapers have unique types of structures and so do attacks. This is the best analogy we could come up with so help us out here. Each attack has strengths and weakness depending on how they are used or where they are employed.



## Spoofing (Who's at the door?)

When you were a kid, you probably enjoyed pretending to be someone or something you weren't. It's fun to play that game when you are young but when you get older, it serves other purposes. **Spoofing** is pretending to be someone or something you aren't. You can spoof an email, an account, a person, a network connection or a car. Ok, maybe pretending to be a car is asking too much but that would be kinda cool. Look, I'm a VW camper.

In the digital world, we spoof digital things. If we are setting up for an attack, we spoof to obtain information to get into a network, and to try to hide our origin. You'd think this is a no-brainer but not every hacker knows to do this. If you don't spoof then you might as well hand out a business card telling everyone what your name is and where you live. Spoofing help to cover your tracks and obtain access.

The Common Vulnerabilities and Exposures database from Mitre (**cve.mitre.org**) lists thousands of spoofing exploits in their collection. And that list is just a shadow of what the Open Source Vulnerability Database had before it closed April 2016. The Mitre list of spoofs includes cellphone SMS backups, spoofing in Apache servers, DNS spoofing and ways to make a lonely spoofing salad for a light lunch or snack. You might think of spoofing as a multipurpose tool that is reinvented as new technology emerges. There is even a spoof attack on an ordering application for a major fast-food chain, using Android, for the hungry hackers out there.

There are multiple types of spoofing and as many reasons to spoof for attack purposes. One common use for spoofing is using a proxy or five to mask the location of the attacker. By routing attack commands through several servers and proxies, the attacker can evade detection and avoid capture (if they do everything perfectly). Now think of this in light of zombies, the victims of **command and control (C&C)** attack vectors and the unwilling slaves of botnets. The execution modules they deliver are already inside the victim's network. The controller or **mothership** maintains a link between itself and the attack modules inside the victim's machines.

In these sophisticated attack structures, there will be several C&C sub-servers located throughout the world. These C&C minions communicate with each attack module to ensure data is flowing or the attack is progressing as planned. If an attack module is discovered on a computer, the best a victim can expect is to locate one of the minion C&C servers, not the main mothership. All connections are spoofed to look legitimate, all IP traffic locations are spoofed to bypass IDS and everything else is spoofed to avoid locating the main attacking servers.

### Exercise

7.11 A popular open source tool used to conduct spoofing attacks is **Eftercap**. You can find your own copy at <http://ettercap.sourceforge.net/downloads.html> or get the Fedora Security Spin at [http://fedoraproject.org/wiki/Security\\_Lab](http://fedoraproject.org/wiki/Security_Lab). Point your browser to <http://www.thegeekstuff.com/2012/05/ettercap-tutorial/> to see an example of DNS Spoofing.

A major challenge to spoofing comes from network authentication and application integrity methods. We know that there are many ways to fake our way into a restricted building but many of the primary access points have angry guards waiting on the other side. In a digital sense, those guards are control processes who may conduct a full body cavity search on anything trying to pass through that interactive point. Trust us, you don't want that type of search done if you are trying to spoof your way in.

Another weak point in spoofing techniques is deep packet inspection. Data packets at critical (or all) network connections are screened for contents, sending location,



possible modifications and potential threats. The software is fast and powerful. Deep packet inspection techniques will usually identify any type of spoofed data and either block the data or sound the alarms. Either way, those spoofed data packets will be logged and audited. Remember, spoofing is lying about your identity. It's not the power of invisibility.



## Game On: Try, Tri Again

The classroom stank and Mr. Tri's shirt was buttoned wrong as usual. His once-white dress shirt skipped a button between the second and third hole down the front. Normally, the pudgy man had some fashion mistake like his shirt being untucked, a back pocket flipped inside out, mismatched socks, a watch on backwards, some hideous mismatch of color and patterns between his pants and shirt. Six months into the school year, most of the high school students were used to the unmarried teacher attire. There was a rumor he lived with a blind mother.

Mr. Tri did get quite a laugh any time he attempted to grow a mustache or beard, though. His facial hair grew in different colors, lengths and various stages of patchiness. Depending on the angle of sunlight or the amount of time he had spent growing his fuzz, he could look either hideous or hilarious. On this particular day Mr. Tri was growing either muttonchops, a biker beard or a ponytail beard. It was too early to tell but ugly either way.

He stood in front of the horrified students of Technology 101 and began his unrehearsed lecture.

"Children, today we are going to talk about computer attacks and what they mean to us as keyboard users. There are some idiots who believe that computer attacks are different from network attacks. This is very incorrectly. An attack is an attack no matter what as long as digits are used. Digits are dangerous in the wrong hands. Hackers attack computers and steal digits which are traded for money and drugs. Digits are like drugs to some hackers, they must have more and more digits to feed their hacker cravings. Isn't that right Ms. Jace," Mr. Tri announced as he pointed to Jace near the back of the class.

Jace had tuned out the teacher even before she sat down so she was caught by surprise when he called her name, "Huh, I'm sorry. What was that?" Shanya sitting next to Jace repeated the teacher's comments in a whisper.

Mr. Tri clearly thought he had the advantage over Jace. He sniffed though his nose, which sounded like a car backfiring, and said, "Ms. Jace did you have too many digits last night, perhaps while hacking?"

Jace shot back, "I'm sorry Mr. Tri, from way back here it sounded like you said that you had too many donuts last night. I wouldn't know why you had too many donuts last night."

"Digits, I said digits, not donuts," he yelled, his face expanding to twice its normal size. Its angry red glow lit the first two rows of desks. The students in those desks felt the temperature rise several degrees from



the teacher's supernova head.

Jace let the slightest smirk creep out of the left corner of her mouth as she asked, "What about digits? Digits are just characters, numbers, or symbols. Did you mean bits, or eight-bit bytes? Or four-bit numbers used in hexadecimal notation? Since we use the bits in bytes like on/off switches, there are 256 possible combinations..." she was saying when she was abruptly cut off.

"I'm not talking about any of that gibberish. I am talking about computer attacks. Now, listen up." Mr. Tri realized that he'd made a massive mistake in telling the school's foremost hacker to listen to his unresearched, unrehearsed, uneducated banter on a topic he could barely spell.

The small smirk on her face grew large as she replied, "Oh, I apologize. I didn't realize you were going to cover one of my favorite subjects. Please continue." Several of the students looked like they didn't know whether they should laugh or run from the room. Jace sat down and pulled out a pencil and paper for note taking. Mr. Tri felt his knees trembling as he saw her ready to take notes on his ill-prepared topic.

"Students aren't supposed to take notes. They are just supposed to recite whatever we tell them to," Mr. Tri mumbled to himself. "If they start taking notes then they'll figure out we don't have anything to teach them. They might even go out and learn on their own and then I'd be out of a job. I can't have that, I need my job." He sweated down to the deepest levels of his tiny soul.

Out of the thick, locker-room air, an idea fell onto Mr. Tri's thin brain. *Brilliant*, he thought.

"Oh, Ms. Jace. I didn't know that this was a topic of interesting for you," he said. The class was used to the fact that this adult couldn't teach, couldn't dress, didn't bathe and couldn't speak very well either.

"Why don't you give us a quick class on your knowledge information about them computer attacks," the runt said as he offered the floor to Jace. That would get him out of a big jam and make sure Jace didn't start taking notes in his class.

Jace nodded, stood up and went to the front of the room as Mr. Tri slithered off to one side.

The teen began, "cyber attacks can create different types of destruction. Cybercriminals can do more damage over a wider area using a computer than if they were using most modern weapons."

Several of the younger guys in the group snorted with immature remarks about tanks against a mouse pad and USB drives versus a cruise missile. Jace kept talking like she couldn't hear them. "None of those military weapons could take down an entire city or country, but



several cyber-attacks have crippled targets that size! In March 2011, the country of Georgia was taken over by a series of cyber attacks against their banks, news stations, power grid and their government. In April 2007, the country of Estonia was almost shut down due to coordinated attacks against their government, banks, TV stations and all digital communication. Can a tank or a cruise missile do that?" she directly asked the boys. Somehow they now had nothing to say.

Jace had made her point and her peers were partly scared and partly impressed out of their boredom. It made sense: everything these days needed some type of electronics. Elevators, hospitals, traffic lights, phone networks, all needed programmable circuits to operate them. Now, even basic services like water and electricity could be attacked, disrupted or destroyed. Jace continued the remainder of her talk without interruptions. Until she noticed that Mr. Tri wasn't in the classroom anymore.

### **Game Over**



## Application-Layer Attacks

Nothing in life is perfect. Nowhere is this statement truer than in digital technology. Software and hardware have bugs, backdoors, vulnerabilities and errors in them even before they reach the intended consumer. Application layer attacks target application services (server-side and client-side). These types of attacks include **buffer overflows**, **cross-site scripting (XSS)**, **Injection** (such as command injection and SQL injection), **directory traversals** and exploits against every other interactive point you could possibly imagine. As we saw from the OSVDB, there are entire databases dedicated to documenting vulnerabilities, daily. Duh.

### Exercises

- 7.12 Look at the OSVDB website. Who maintains this database? Why? And why should you trust them?
- 7.13 Look at <http://exploit-db.com>. Who maintains this list? Why? And you trust them why?
- 7.14 Look at the NVD website. Who maintains this one? Why? And why are they trustworthy?
- 7.15 Look at the CVE website, and answer the same questions.

Don't forget to check for the hardware vulnerabilities too. We did mention that all that hardware is running applications, didn't we? If you follow the news you know that certain governments have been inserting backdoors into hardware being sent to countries they are completely friendly with, at least in theory.

If your organization were under attack, understanding application layer attacks might be your first step to stopping the attack. There are just so many types to choose from. Applications are in everything digital and these applications interact with open connections you may never even know about. These connections include using ports that you might not expect software to send packets through. Know your ports and especially know which applications access multiple ports to communicate.

A recently reported vulnerability is a good example of this, and a good opportunity to get familiar with the dataloss discussion boards:

<http://lists.osvdb.org/pipermail/dataloss-discuss/2012-March/003930.html>

### Exercise

- 7.16 Who exactly is "security curmudgeon?" Track him or her down. Does this person ever reveal their real identity?

In reality, you should have already conducted an analysis of all access points, as recommended by the OSSTMM. The manual will take you through an intensive





examination of every possible application interface that could yield a possible exploit. This testing should be performed before an attack, not after. To put it a better way, use the OSSTMM on everything under your control every chance you get.

You'll be the life of every party, trust us. Ladies dig OSSTMM guys and guys love hearing about OSSTMM from ladies.

In the OSI model, these are Layer 7 attacks. Since everyone including your grandmother has a web page or uses the Internet, a large number of network attacks are aimed at web applications. Organizations may not use secure coding practices for in-house programs and many lack the resources to perform proper security auditing of their public web application software. This common industry practice leaves more exploits open with every new web widget and web application. Mobile applications are easy targets because more people have smartphones than computers. More people put sensitive information on their smartphones, too. They also take their smartphones to work with them. It's a win-win situation for every attacker.

Low-level application vulnerabilities can be chained together to run a series of commands with the privileges of the "root" user on the device. An attacker can obtain unauthorized access to the device and plant backdoors or access configuration files containing credentials for other systems (like Active Directory/LDAP credentials) that can be used in further attacks.

Then there are the apps practically everyone uses, like Adobe Reader and Flash. Apple refuses to offer Adobe Flash in iOS because they feel Adobe has too many unsolved security issues. And that's just a video plug-in.

Let's take a look at how many applications run on a small device. Even before you turn the device on, there is an internal clock. You turn your device on and the circus starts. As power is applied, it is monitored by an on-board application that checks to ensure correct voltage. If it has enough juice, then the built-in circuits check to see what sort of thing they're in. It might be a toaster, it might be a Titan super computer, it just needs an application to see what its initial purpose is.

Before we have even the slightest evidence of life on the screen, we have already run three to four applications. The device's read-only memory lives in built-in chips that use a hard coded application to tell the OS about its size, file storage capacity, if it's bootable, did it pass the self-test, things like that. We are running five applications and the device isn't even ready to work yet. Yet, each one of the internal applications communicates with each other and the CPU before you see the start screen. Once that device is operational, you could easily have thirty programs running just on your smartphone. Let's multiply that a few hundred times for a desktop computer and multiply that a thousand times more for networks.

Application level attack potential changes every time a new device or program is added, updated, removed or reconfigured.

Updates and patches are the traditional solution to application vulnerabilities. Oops, this form is vulnerable to XSS; better fix it. Dang, that input allows a buffer overflow; better fix that too. Fix that buggy code by piling on thousands of more lines of buggy code! Or fail completely, as some patches do, and crash ALL your users' systems (it's happened more than once).

Some attacks use file replacement to keep their activity hidden. Malware and other attack techniques will name their programs "calc.exe" or "notepad" to hide them in plain sight within the victim's network. As the victim updates their programs, that malicious code can be overwritten with the correct application. To combat this, an attacker will usually place a second copy of their code somewhere else in the system.



This second copy will routinely check to make sure the attack package is where it was meant to be. If the malware is overwritten, the second copy just writes it there again.

### Exercises

7.17 Mobile devices aren't exempt from malware. List the application marketplaces for the top three mobile operating systems.

For each marketplace, do research to determine if they have ever distributed malware.

If it has, how was it delivered?

How did it get into the market?

And what type of malware was it?

7.18 What is Project un1c0rn?

Go to their website. What are you looking at?

How can you use this information?

### Remote Access Toolkits (RATs)

This type of attack can be used by the very beginner script kiddie but it is still an effective method to obtain access to networks and data. You don't have to know anything about scripting to launch preconfigured programs like Poison Ivy. They provide remote access that's almost identical to Windows Remote Desktop. Have you used it? It's useful for troubleshooting, training and breaking into a computer from a distance.

Let's say that you forgot a file on your home computer but you are at the coffee shop. Remote Desktop into your home computer and transmit that file to your new location or even work on that file as if you were sitting at the home computer. It's quite handy. It's also quite dangerous if not configured correctly.

That is the key to security for all digital life forms: configure them correctly.

Right out of the box, most products and applications are designed to be used by the widest possible population using the most open configuration settings. This means things are supposed to be easy for the least computer savvy person you know, like a grandparent. It's up to the user to configure, tweak, lock down and most importantly, read the manual. You may have heard the phrase "RTM." (Sometimes people add another initial.) Yup, that stands for "read the manual." Most people don't.

Weak passwords are easy ways to gain access to remote connections. You could stand outside any public hotspot, sniff the packets for a few minutes and you will probably obtain several passwords for company remote servers. We at Hacker Highschool do not recommend that you do this, but this is the kind of testing a smart security person will do, along with warning network users to stay away from public access WiFi unless proper protection measures are taken (hint: a secure VPN).

### Exercises

7.19 Who are the primary users of RATs, and for what purpose? This may be a tricky question to answer until you do some research on **Advanced Persistent Threats (APTs)**. (We discussed these in Lesson 6, Malware.) APTs frequently use RATs.

7.20 If you scanned your own computer for open ports, which port number would make you suspect it was infected with a RAT?



## DOS and DDOS

**Denial of Service (DoS)** attacks and **Distributed Denial of Service (DDoS)** attacks are commonly associated with web sites and ecommerce. However, both of these attacks can be used against any device that communicates: an email server, a proxy, a switch, an IDS and so forth. We are just used to hearing of these being used against web servers.

We are experiencing massive demand on our support capacity, we are going to get to everyone it will just take time.

### Code Spaces : Is Down!

Dear Customers,

On Tuesday the 17th of June 2014 we received a well orchestrated DDOS against our servers, this happens quite often and we normally overcome them in a way that is transparent to the Code Spaces community. On this occasion however the DDOS was just the start.

An **unauthorised** person who at this point who is still unknown (All we can say is that we have no reason to think its anyone who is or was employed with Code Spaces) had gained access to our Amazon EC2 control panel and had left a number of messages for us to contact them using a hotmail address

Reaching out to the address started a chain of events that revolved around the person trying to extort a large fee in order to resolve the DDOS.

Upon realisation that somebody had access to our control panel we started to investigate how access had been gained and what access that person had to the data in our systems, it became clear that so far **no** machine access had been achieved due to the intruder not having our Private Keys.

At this point we took action to take control back of our panel by changing passwords, however the intruder had prepared for this and had already created a number of backup logins to the panel and upon seeing us make the attempted recovery of the account he proceeded to randomly delete artifacts from the panel. We finally managed to get our panel access back but not before he had removed all EBS snapshots, S3 buckets, all AMI's, some EBS instances and several machine instances.

**Figure 7.1** The Codespaces.com DDOS

Web attacks occur so often that they don't make headlines anymore. One of the major problems with web attacks is the loss of business that happens when a company can't conduct transactions over the web. Amazon, Google, Facebook, the New York Times and every major web content provider has been the target of DoS or DDoS attacks. The basic idea behind these attacks is to keep a web server network too busy to handle normal IP traffic. These attacks can be as simple as sending partial header requests to a server or as complicated as having tens of thousands of zombie computers overload a network with bogus requests.

Due to the limitations of a single computer, it is difficult for one machine to disrupt the service of a communication server. This isn't to say that there are no DoS attacks. There are and we'll show you one in particular. But you're more likely to see large networks of computers working to distribute an attack across multiple fronts to disable networks. This is a DDoS attack. Those are much more common and it's harder to track the true attackers.

DDoS requires a massive network of machines that are infected with command and control software that propagates across thousands of unsuspecting computers. Most of the time, the computer owner has no idea that they are part of a DDoS. These machines are controlled by higher-level control servers located throughout the area. Above the controlled servers is the mothership server that passes commands down to the control servers, which they relay to the individual bots/zombies.



Locating the control servers is difficult at best and finding the mothership is rare. If the control servers are located or compromised, the mothership servers unplug and disappear. In the meantime, the individually controlled computers that unwittingly participated in the DDoS cannot be legally prosecuted since they didn't know they were part of a crime. Right? (Wrong.)

Criminal hackers have figured out many new twists on the DDoS concept but those ideas are beyond the scope of this lesson. We'll be covering a range of DoS and DDoS attacks and how they work.

### Exercise

7.21 Read up on Rustock.

Is it a trojan?

Is it a root kit?

Is it a proxy?

Is it a back door?

Find out how to remove it. Particularly note the Registry keys and the files you have to remove.

And once you have, is that really the end of your problems?

### Slowdowns

Let's start with the simple slow Denial of Service attack. A program like **Slowloris** sends HTTP header packets to the victim. The trick is that the packets Slowloris sends are never complete requests or they don't contain all the information the web server needs to respond to the HTTP request. Think of it as the old joke, "How do you keep an idiot in suspense? I'll tell you tomorrow."

The attack tries to make as many connections as possible. This is a slow attack, like you getting out of bed on a cold morning. Slowloris mainly works against older Apache servers, where the server will wait for the full header information before processing that request. The attack will send additional HTTP information but never enough to complete the request; it just tries to keep the connection open as long as possible.

This attack can be mitigated by limiting the number of connections a single IP address can open and restricting slow connections to a minimum. Newer Apache server software comes with a module to reduce the effectiveness of this attack called `mod_reqtimeout`.

### Unicorns

UDP Unicorn attacks User Datagram Protocol, which is the primeval portion of the Internet protocols. Remember way back when we talked about protocols? Yeah, we told you about the connectionless UDP that doesn't use any handshakes, unlike TCP (all that SYN – SYN ACK stuff). It just sends data and forgets about it. This works great for streaming video, when data is being sent in large masses and missing one or five packets isn't going to be noticed by the user.

The Unicorn attack exploits **Windows sockets (Winsock)** to make your dreams come true. It does this by flooding a target with multithreaded UDP packets. Similar UDP-LAG attacks just try to slow down a server, thus the name "Lag." It takes a pretty fat connection to overload another server but this is an old school method of attack that is still out there, lagging.



## Pay Per Service

Imagine this: you can buy criminal **Software-as-a-Service (SaaS)**. Usually SaaS is something like email services, but there have been, are and will be services like Blackhole where you could pay by the thousand computers for sophisticated attacks against the victim of your choice. Of course, this service business earned its creator, Paunch, many exciting adventures with the Russian legal and prison systems. Another pay-to-hack too is TwBooter, a web service that calls itself an "Administrative Network Stresser Tool." Whatever you want to call it, it does things similar to Blackhole. You give it a target, pay your fee and clap your hands in glory as you watch some web site become the victim of a DoS. No intelligence required.

## Getting to Post

GET and POST attacks overwhelm a victim's server by filling up their memory buffers with requests. Some of the attacks require the server to decrypt its own data in a circular process. It's kind of like that annoying game where you repeat everything the other person says. In this attack, though, the server has no idea it is repeating itself thousands of times a minute.

**HTTP GET Flooding** is exactly what its name says: it floods the victim with GET requests to overwhelm the network. The GET and POST attacks work very well in SSL sessions under HTTPS. These attacks are more difficult to identify because the data requests are encrypted.

**RUDY**, or the **R-U-Dead-Yet** attack, is a form of POST attack, but it works by sending a never-ending content length request for a POST query. The server keeps waiting for the rest of the POST content length but it never comes. It's like winning the lottery: It never happens, to you anyway.

## DDoS By the Numbers

Distributed Denial of Service attacks require lots of data and plenty of bandwidth to overwhelm the victim's servers. To achieve this feat, most attackers will leverage other resources like botnets, using other servers (that don't belong to them), or being really creative with protocols. DDoS attacks utilize almost every layer of the OSI model (remember the OSI model from earlier lessons?) and the protocols associated with them.

Layer 3 and 4 attacks have used the Network Time Protocol (NTP) servers to flood targets with amplified data requests. The protocol was designed back when the Internet was young and security wasn't an issue. Many of the original internet protocols are still used today and still lack basic security measures. With the NTP attack, an attacker spoofs a request for time from one of the NTP servers located throughout the world to synchronize time across networks. A NTP request is a small unauthenticated request from one computer for a time update. The NTP returns to the requestor a longer string of data that includes the time.

In this attack, the data requested is amplified by the fact the NTP server returns more data than is sent to it in the first place. The NTP servers don't require verification from the requesting user which allows an attacker to spoof the return IP address. An attacker sets up crafted packets that have the target as the destination address. These packets can be launched from a single server that allows IP spoofing.

One attack on 10 February 2014 generated a peak of 400 Gbps against a cloud server. Now that is some serious amplification of data directed at a target. The requests generate 206 times more data than are sent. So if the attacker sends 1,000 8 bit NTP



requests (MONLIST) at 1,000 NTP servers, the results will be 16,480,000 bits sent back to a target. It's slightly more technical than this but you get the idea.

Switches in protocols and commands are being leveraged to create massive floods of data against targets. The Open DNS attack works the same way but doesn't return as much data as the NTP attack. A similar attack using SNMP servers could yield a return of data at 650 times the rate.

### Exercises

7.22 You want to launch an attack against the computer of someone in your class, and you're interested in Low Orbit Ion Cannon (LOIC).

Will it do what you want?

Can you find it online? Do so.

How do you use it? Explain.

7.23 Find information on attack trees. Create an attack tree that diagrams the steps necessary for you to launch a LOIC attack against your classmate's computer.

### Malware (Nobody liked me as a kid)

If you haven't read the lesson yet, don't forget that Lesson 6 deals with malware.

In the early days of viruses, many would simply delete the victim's data. Others would play a silly tune while wiping the file allocation table or posting a message announcing to the user that the computer is infected. These were very destructive programs and only seemed to come in a few flavors.

Those flavors were:

1. Delete data on the computer.
2. Overload networks by propagation and resource hogging.
3. Just plain messing with users' heads by deleting text, changing page order, altering typed characters and so on. The kind of stuff your kid brother does to you, or you do to your big sister.

These early forms of malware evolved into polymorphic (able to change their own structure to avoid signature detection), macro level scripts (which depended on a particular application like MS Word) and somewhat more sophisticated programs that began extorting money from users. Virus makers turned to profitable **ransomware** programs that encrypt user data and demand money from the computer owner to decrypt the data. As you might expect, those that paid the ransom did not always get their data back. (Surprise: Pirate.)

Malware began its life as a form of attacking computers and that fact hasn't changed one byte.



## Teaching a man to phish (Hacking the Wetware)

Delivering these malware packages to someone's system has become much easier. Phishing is the primary method, although did you check out that USB drive you found lying in the driveway? Great new program on it wasn't there? We spent the whole night reworking it so that when you put it into your computer it loaded our software along with the game. When you are done playing with the game and your computer, we will launch our attacks, using your system, your IP and your persona. If you want to see how much you know about phishing, the OpenDNS quiz is a good place to start (<http://www.opendns.com/phishing-quiz/>).

### Exercises

- 7.24 Put your security consultant hat on again. Your client wants to know if security training really is effective in making employees safer. There are big names on both sides of the debate.
- List two Godzilla-class security professionals who say it isn't effective, and very briefly, why.
- List two who say it is, and why.
- 7.25 Now you're selling your client on SET (the Social-Engineer Toolkit). What the heck does it do? Are you selling him a product, or services?

## Hacking the Technology that Surrounds Us

One of the hottest topics in the technology world is the **Internet of Things (IoT)**, the networked devices that include everything from your home's electric meter to the **fire pressure management system (TPMS)** on the cars around you. Most of these communicate using familiar protocols, which means most of them can be manipulated using familiar methods: spoofing, DDoS, physical mischief, etc.

### Exercises

- 7.26 What is the wireless protocol used by TPMSs?
- How does your car know which sensors are its own, when every car around it has them too?
- Is this identification method susceptible to spoofing? DDoS? What else? In other words, are there vulnerabilities in this system?
- 7.27 Does the car you're in most often have a built-in GPS unit?
- What are the vulnerabilities in THIS system?

## Attack Signatures: Detecting Different Types of Attacks

Some attacks are like mosquito bites: you don't notice them until after you've been bitten (and had all your blood sucked out). Other attacks can take place over months and years, siphoning off your proprietary data the entire time.

A spoofing attack can remain hidden inside a network while a DDoS will wake up the entire IT staff as the phones begin ringing off the hook. Everyone will be asking, why did you take the network down? You didn't. You were eating a sandwich. Somebody else is taking your network down.

Attack detection techniques rely on **signature recognition** and **anomaly detection**. Signature detection works great if the attacker is using known vulnerabilities, exploits, or



typical tools (like script kiddies do). The problem with looking for attack signatures is that the programs need to know what those are ahead of time. Signature recognition programs don't work against zero-day exploits because there isn't any signature to detect until after the attack.

Anomalies within a network are an everyday occurrence. If the intrusion detection system sends an alarm every time a data burst occurs, you'll be spending your entire work day resetting the system. A few bad log-on attempts and there goes your weekend. From a practical standpoint: what is an anomaly anyway? There is no easy method to distinguish normal data flow from an attack, other than a DDoS or DoS. And deep packet inspections require additional resources and possible delays in data transmission.

Network attacks that are carried out by people unfamiliar with your company will gather information ahead of time. Scanning by outside IP addresses is a normal part of any network so you will have to look for certain patterns like:

1. Scans that repeat the same time each day or night (weekends and holidays are great times to recon networks)
2. Scans that come from within the domain (because internal scans are considered "passive" traffic, the attacker may not bother with a disguise)
3. Scans that seem to use the same technique/tool
4. Scans that seem to come from an internal IP address
5. Scans that focus on known or new vulnerabilities (CVEs)
6. Scans against hardware such as routers, IDS, printers and other network connected devices. They all have IP addresses so they make great access points.
7. WiFi network scans, Bluetooth scans and remote access log-in attempts from portable devices in the local area. Look at that coffee shop across the street.

### Exercise

7.28 Here's where you begin your education with the open-source intrusion-detection application, Snort. First, find the website where Snort is distributed and supported.

Like many malware detectors, Snort relies heavily on signatures. Find a Snort signature for the CryptoLocker malware.

### The Spoof

Spoofing may be detectable by tracking redirected URLs in user web browsers (or by disallowing redirects altogether). Spotting spoofed sites in email links is fairly tough because of the social engineering factor: People are curious and trusting. Training and educating company staff is a good starting point for preventing users from clicking on malware-linked sites. The problem is hackers are excellent at enticing a user to open up or click on a things. You know what you do when you get an email from your mother that tells you to look at some video of your relative doing something funny. One click and the payload is already loaded. Too bad, no video of Uncle Mika slipping in the bathroom.

Spoofing may be used as part of an overall complex attack, such as reconnaissance or information gathering. Creating a spoofed web site might be a simple method to get network users to upload a small segment of a larger attack tool. It would be like getting one foot in a network's door. Once that small script or program is inside a user's browsers, the malware phones home to retrieve the rest of the program. These actions





can be detected if you are looking for outbound traffic on unexpected ports to unusual URLs. You should not see a local user uploading data to an external source; this is almost always a bad thing. Too few network security professionals look at outbound activities, though, for better or worse, depending on which side you're on.

IP packet spoof detection requires more work since only a few of the current network protocols confirm the authenticity of inbound data packet addresses. Forged certificates, man-in-the-middle intercepts and hijacked sessions can all be made to look like trusted data sources. Add to the fact that spoofing can happen at multiple network levels such as network layer spoofing, transport layer spoofing, session and application layer spoofing (discussed earlier) and data link layer (MAC address) spoofing.

Proper identification of suspected spoofed data packets needs to work in conjunction with IDS, routers and firewalls within a network. An intruder may not even use the reply data that your network provided, they may just be looking for a connection. If she asks for DNS resolution from inside your network, she may not care if she gets a correct DNS entry back (although this could be handy); she's just probing for hosts. Usually. This is where **Time to Live (TTL)** becomes useful for not only detecting spoofed packets but stopping spoofed data. Basically, the TTL setting of a packet tells the network how long to keep kicking the packet around. Packets shouldn't be hanging around forever, and if they are trying to, they deserve suspicion.

Inside intranets, data packets traveling along similar routes should take roughly the same path and arrive at the same time, every time. If there are packets that do not seem to follow this basic principle, or appear to bounce through different paths, those packets may be spoofed. Routers automatically tune TTLs (keep them as short as possible) to minimize and flush out wandering (spoofed) packets. This is a basic first line of defense.

However, different protocols use different TTLs. This is one reason why you will need to depend on correctly configured firewalls, routers and user training. Spoofing is a constant challenge to battle.

### Exercises

- 7.29 Time for research: find one common command-line tool that lets you find the path to a target, using a switch that specifies the maximum number of hops (TTL), as a way to detect spoofing. (Yes, you have used this tool before, in earlier lessons.)
- 7.30 And more research: find one easily-available command-line tool that lets you create spoofed packets (or heck, any kind of packets you can imagine).

### Sniffles

Sniffing packets is not as simple as plugging your computer into the network and capturing traffic. It's often more difficult to decide where to place the sniffer than it is to analyze the traffic. The main devices that handle network traffic do so differently, so you have to be aware of the network's physical setup. So, how do you collect traffic from the network?

First, if you're going to have to collect everyone's traffic, on a wired Ethernet network you'll need a **mirror port** or **trunk port** on a switch. Otherwise, on a switched network, the only traffic you'll see is broadcast traffic and your own. But be very clear: WiFi is not switched networking. WiFi functions like a hub: you can see everyone's packets.

If you're attached to a mirror port or have put your WiFi card into **promiscuous mode**, a packet sniffer application can monitor network traffic on all computers on the network.



## Packet Sniffing

---

A packet sniffing program is designed to capture the traffic packets that move along the network. You get to check out the packet content and make some determinations about the validity of the packet. In Linux/Mac/Unix, the native **tcpdump** command can capture traffic, save it to a file, look for search strings and a lot more. When you're dealing with automated processes (come on, you're a hacker, you want to automate everything), using tcpdump at the command line is the way to go.

### Enter the Shark: Wireshark

Full-on GUI tools like **Wireshark** are often called **network protocol analyzers**. They let you capture and interactively browse the traffic running on a computer network. Wireshark is the de facto (and often **de jure** [by law]) standard across many industries and educational institutions. For Windows users, you must also install the **WinPcap** driver, which you'll be reminded of during installation. WinPcap is also available from [www.winpcap.polito.it](http://www.winpcap.polito.it), if you find yourself needing it separately.

### Windows Installation

Download and install Wireshark (<http://www.wireshark.org>). Then follow these steps:

1. Double-click the installer file to begin installation and then click **Next** in the introductory window.
2. Accept defaults all the way through.
3. **When the dialog asks if you want to install WinPcap, make sure the Install WinPcap box is checked (indicating "yes").**
4. Click **Install** and the process will begin.

### Linux Install

The first step to installing Wireshark on Linux is to download the correct installation package. Not all versions are supported. Usually you're going to need root privileges.

#### RPM-based Systems

For RPM-based distributions (Red Hat, Fedora and SUSE), you can download the appropriate package from the Wireshark page. Open a terminal window and a command like this (use the filename of the actual installation package you download):

```
rpm -ivh wireshark-0.99.3.i386.rpm
```

But you can usually install it without downloading it with this command:

```
yum install wireshark
```

This command goes out and gets a slick pre-configured package from the **system repositories** and installs it for you. Nice, huh?

#### DEB-based Systems

On a DEB-based system (Debian, Ubuntu and many more) you can install Wireshark from system repositories, so you don't need to download anything unless you really want to here either. Open a terminal window and type the following:

```
apt-get install wireshark
```

### Mac OS X Install

Different versions of Mac OS X require different procedures to install Wireshark. Check the online documentation, but generally the steps are:

1. Download the DMG package from the Wireshark site, and the Xquartz package from <http://xquartz.macosforge.org>.
2. Open the Wireshark.dmg and copy Wireshark.app to the Applications folder.
3. Open the Xquartz.dmg and copy Xquartz to the Applications/Utilities folder.
4. When you start Wireshark you'll be prompted to *Choose Application for X11* since it doesn't find it up in the Applications folder. You need to manually locate it by browsing down to it in Applications/Utilities/XQuartz.

## Wireshark Fundamentals

To find anomalies on your network when you might be under attack, you'll have to know what daily normal network activity looks like. With your network operating smoothly, you can baseline your activities. Deviations from this baseline mean something is amiss.

### Exercise

7.31 Packet capture with Wireshark: follow these steps.

Open Wireshark, and from the main drop-down menu, select **Capture** and then **Interface**. A list of interfaces with their IP address should be visible.

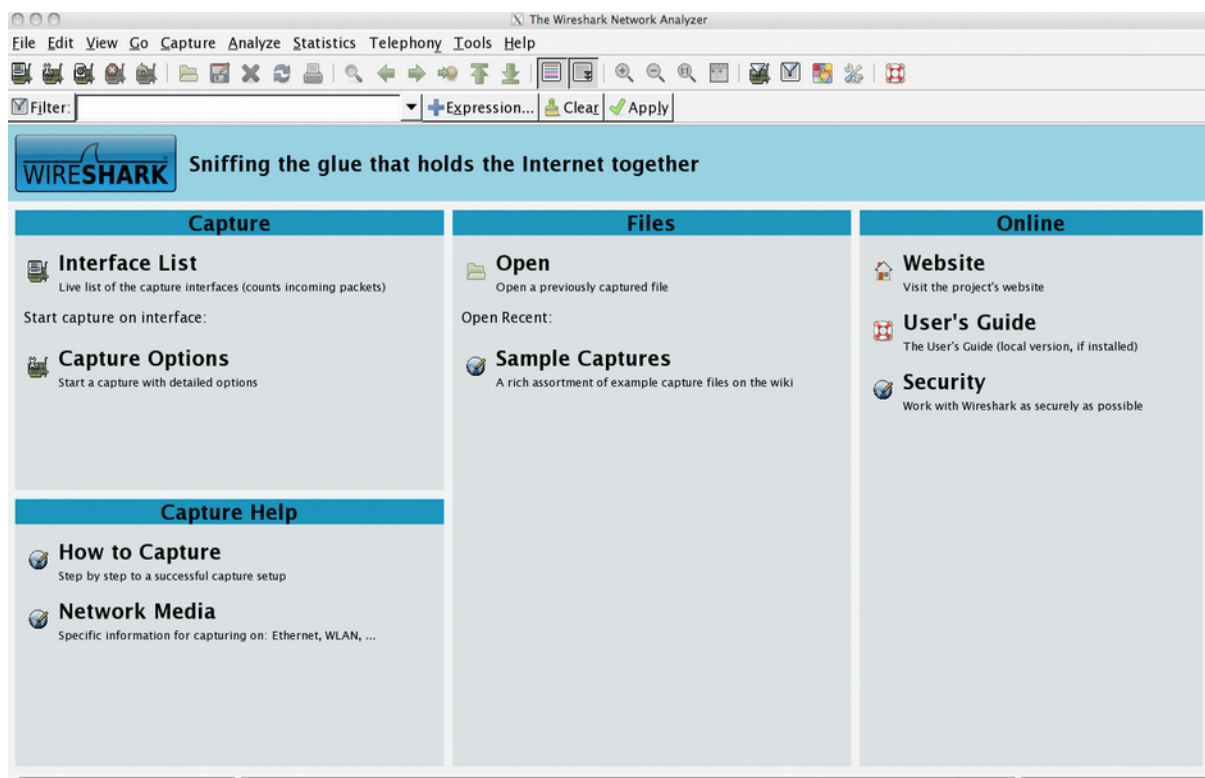


Figure 7.1 Wireshark

Choose the interface you want to use and click **Start** or simply click the interface under the Interface List section. Data should start filling the window.

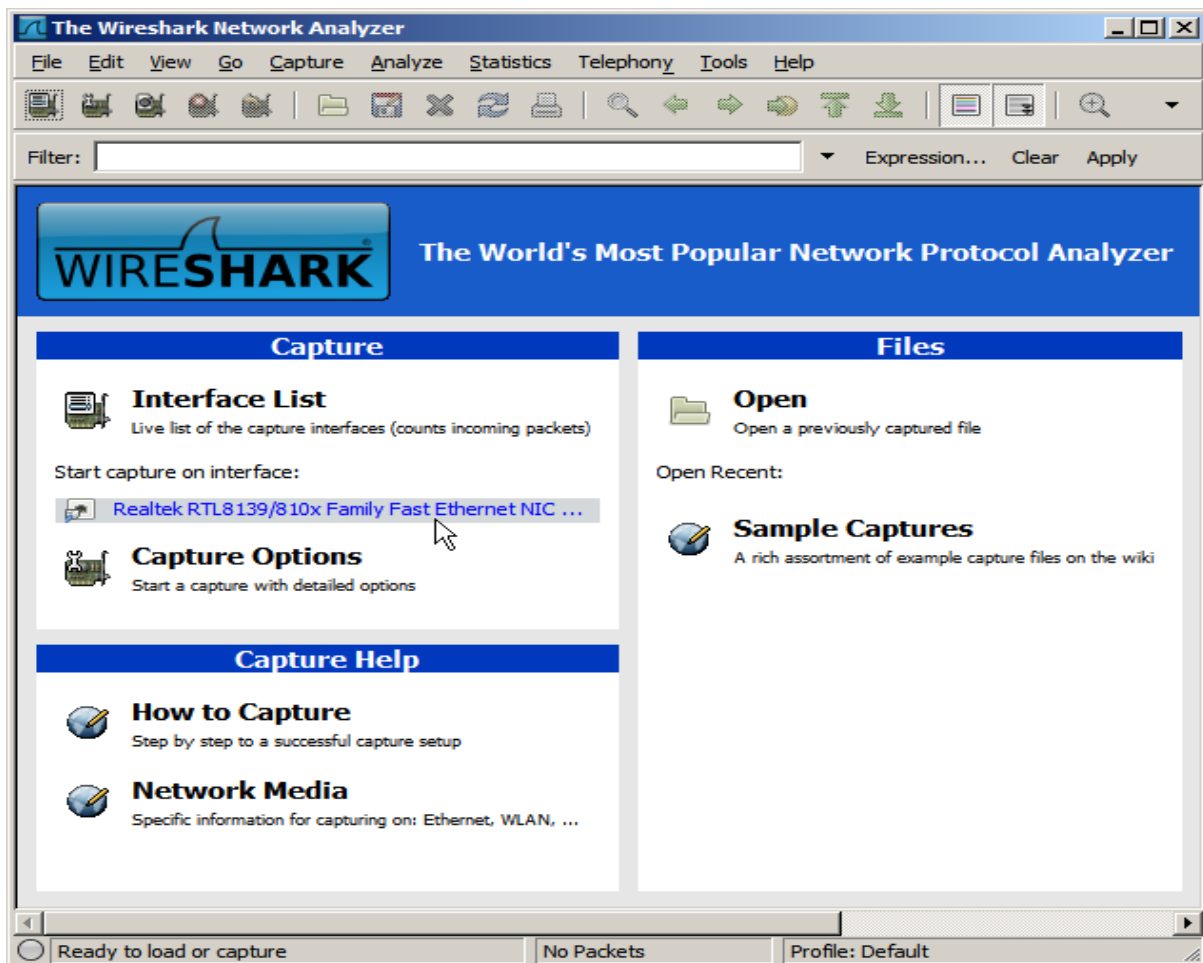
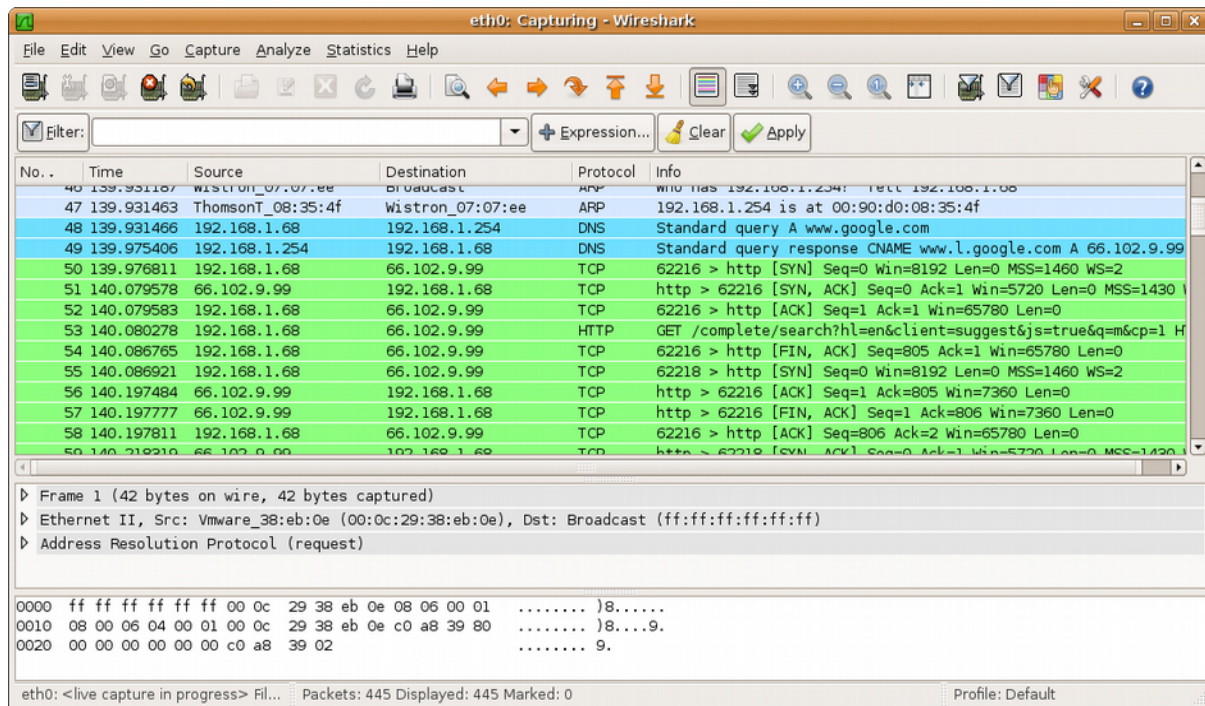


Figure 7.2 Wireshark Interface Selection

This will open another window that shows the activity that Wireshark sees on your network.

Open each of the following screens in your local copy of Wireshark.



**Figure 7.3** Capture

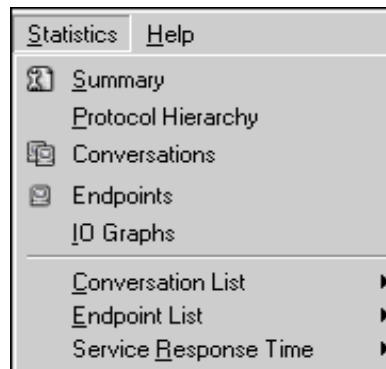
In the packet capture window, the top pane displays a table containing all the packets in the current capture file. This includes the packet number, the relative time of the packet capture, the source and destination of the packet, the packet's protocol and some general information found in the packet.

The middle pane contains a hierarchical display of the information about a single packet.

The lower pane displays the packet in its raw, unprocessed form. It shows how the packet looked as it crossed the wire.

## Decoding the Packets

Now that you can see network traffic, you have to figure out what it all means. Wireshark provides a number of charts that are valuable in establishing what normal network traffic looks like. There are a lot of different statistics to consult: click on the *Statistics* field in the menu bar at the top of the screen.



**Figure 7.4** Statistics Menu

These statistics are compilations of data Wireshark observed. Conversations and endpoints identify sources of significant amounts of traffic. This tells you what the traffic flow of your network should look like. Some items you might consider looking at include ARP or ICMP packets. Large numbers of such packets might suggest a problem.

## Summary

Basic global statistics are available in the summary window such as:

- Capture file properties
- Capture time
- Capture filter information
- Display filter information

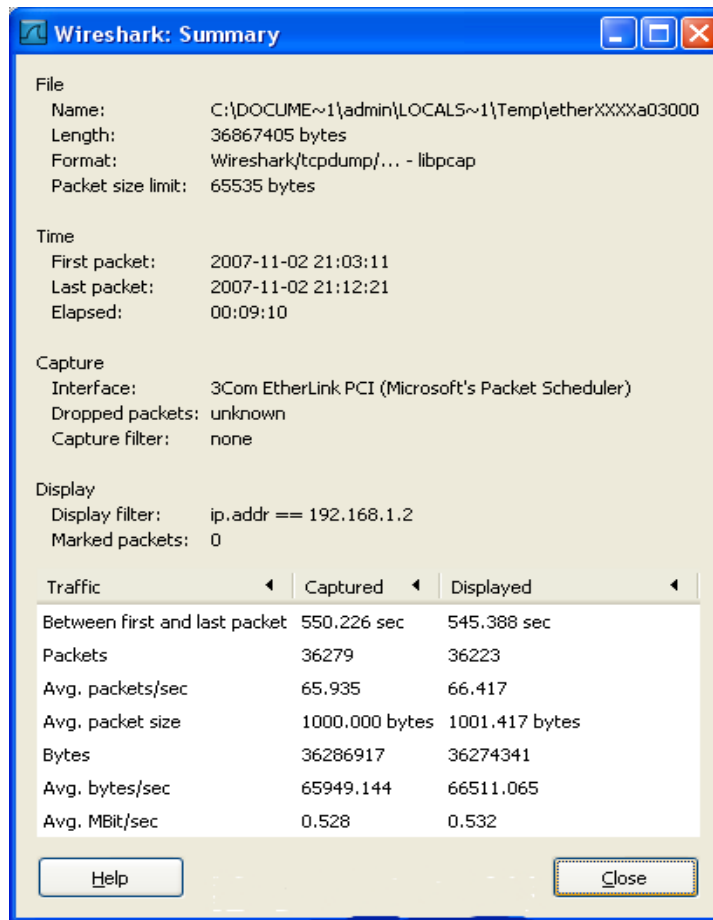


Figure 7.5 Summary

### Protocol Hierarchy

The protocol hierarchy shows a dissection by OSI layer of the displayed data.

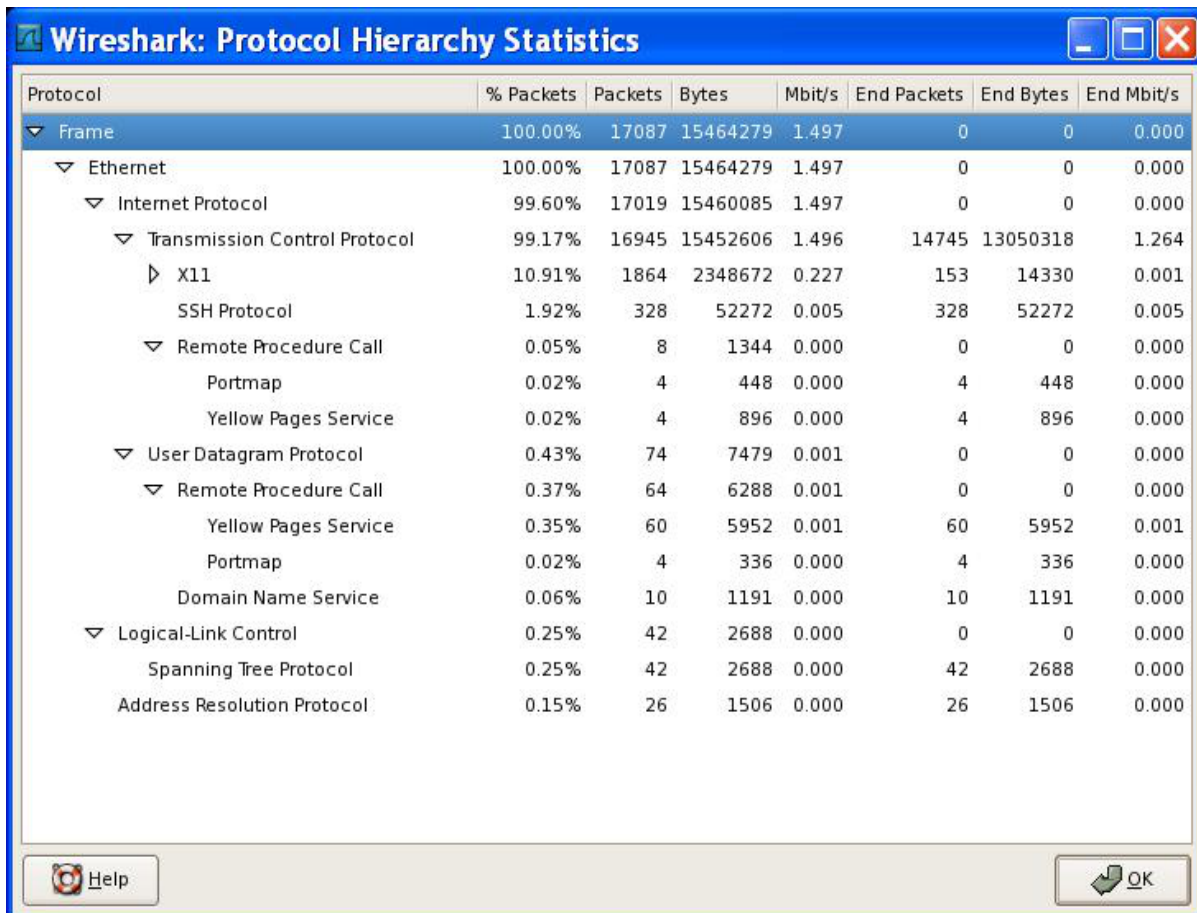


Figure 7.6 Protocol Hierarchy

### Conversations

If you use a TCP/IP application or protocol, you should find four active tabs for Ethernet, IP, TCP and UDP conversations. A "conversation" represents the traffic between two hosts. The number in the tab after the protocol indicates the number of conversations, for example "Ethernet:6".

### Ethernet Conversations

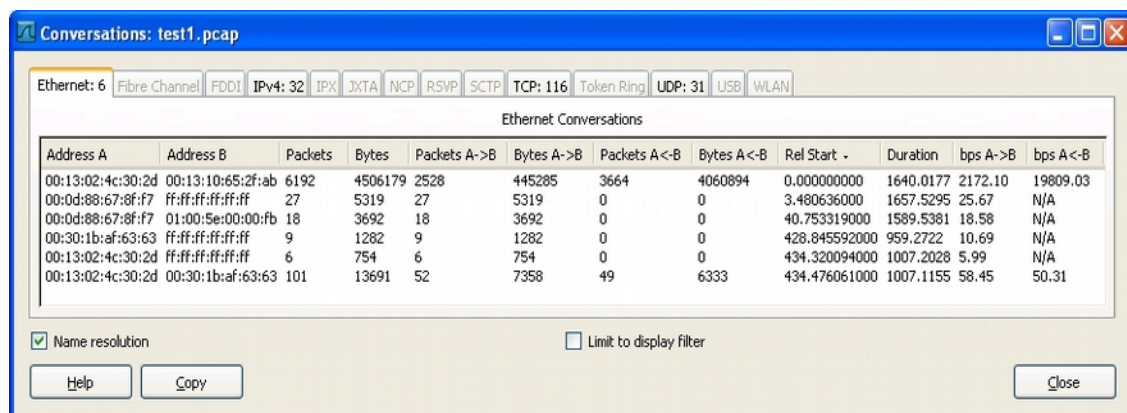


Figure 7.7 Ethernet Conversations





### IP Conversations

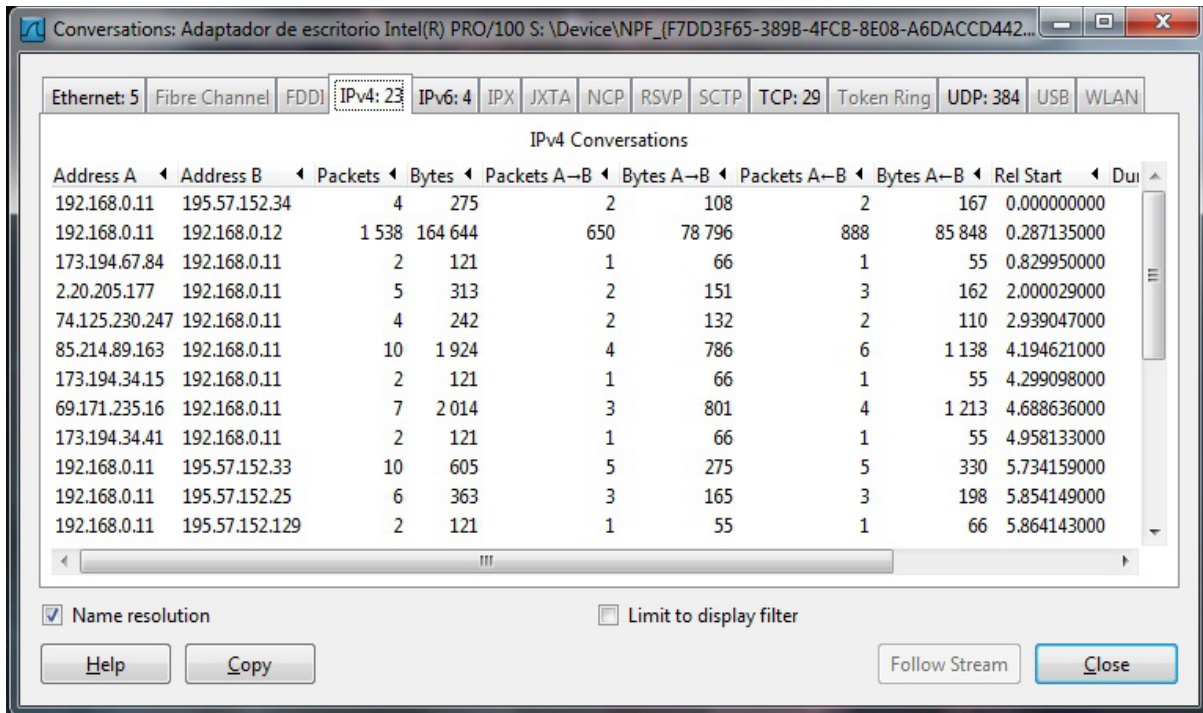


Figure 7.8 IP Conversations

### TCP conversations

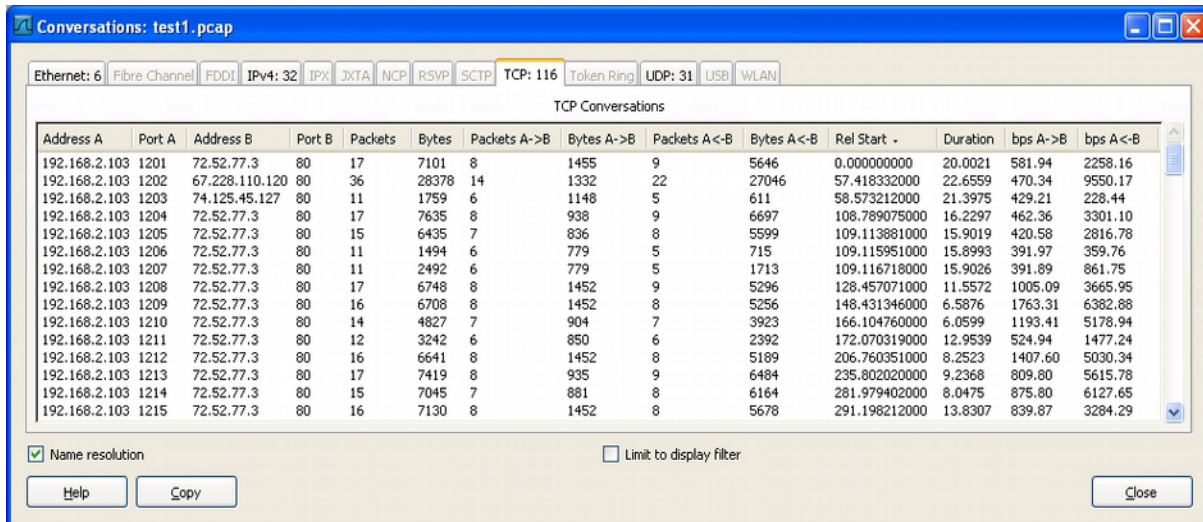


Figure 7.9 TCP Conversations

As you review this information from your computer, which programs might be involved in these conversations, in light of information from the lesson on Ports and Protocols?

### Endpoints

The endpoints provide statistics about received and transmitted data on a *per machine basis*. The number after the protocol indicates the number of endpoints. For instance: "Ethernet:6".

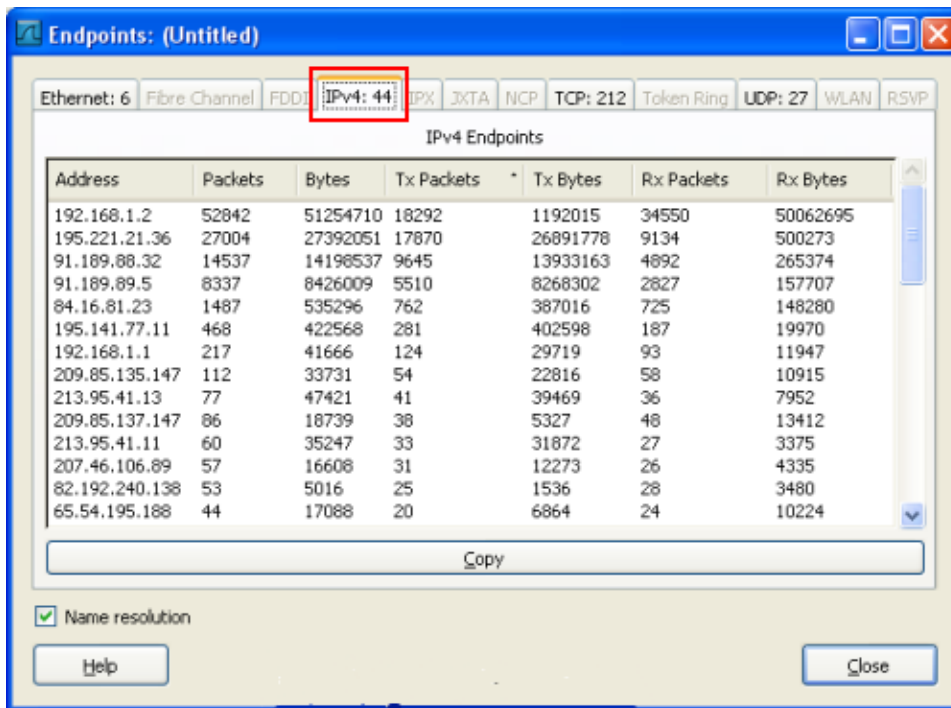


Figure 7.10 Endpoints

Which endpoints are consuming the most traffic? Why might that be?

Output

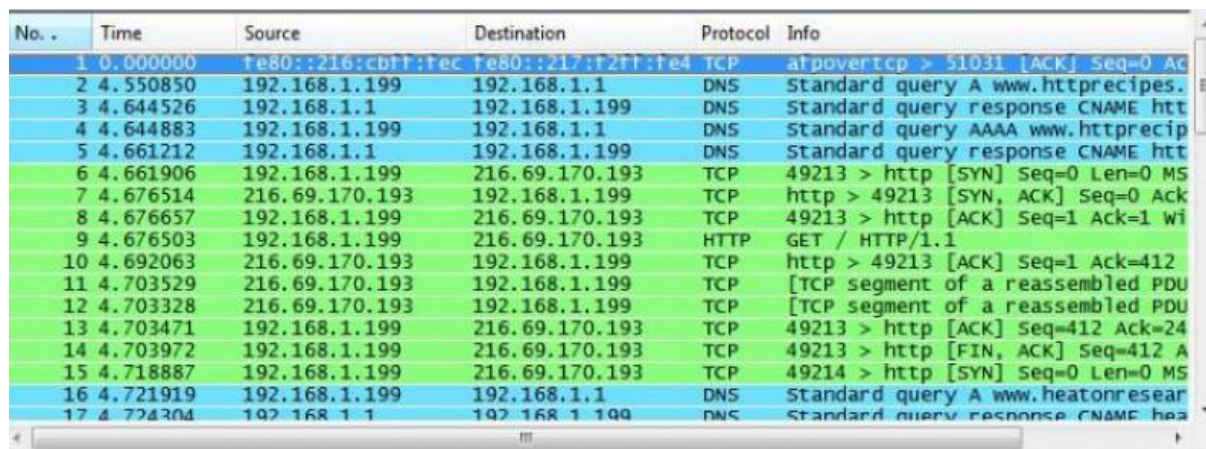


Figure 7.11 Output

In this example, the seventeen packets show activity collected by Wireshark. The easiest information to decode is the *Source* and *Destination* columns. The 192.168.1.x IPs are local network systems. The 216.69.170.193 is not local.

The next column to look at is the *Protocol* column. This column tells you what protocol was being used.

Packets two through five represent a DNS query to identify a specific website.

The last column, *Info*, provides more detailed information about the packets.

Packets six through eight identify the three way handshake of a TCP connection:



Packet 6 – SYN Packet

Packet 7 – SYN ACK Reply

Packet 8 – ACK Packet

Packet 9 represents a request to reuse a connection multiple times to download images, scripts, stylesheets *et cetera* after the page has been delivered.

Packet 10 represents an acknowledgment of the request.

Packets 11 and 12 provide information about packet segmentation. A "PDU" is a "Protocol Data Unit." One unit of information being transferred in accordance with a given protocol will be disassembled into many packets (smaller pieces) if it's too large to fit in one packet. When the receiving side gets the packets they are then reassembled before they are sent up the stack.

### Exercises

7.32 By now you should be familiar with Wireshark.  
Can you look for a particular string of text in the packets you capture? Find out how.

Now, start a capture.

Go to a search engine, and search for the word "password."

Check in Wireshark: does it see the word "password," or is your traffic encrypted and unreadable?

Try this with at least three search engines.

Which ones encrypt your traffic? Why do you suppose they do this?

Be clear that this is exactly how information is leaked: when it's outbound.

7.33 Are you getting tired of looking at individual packets? Now it's time to learn about a nice feature of Wireshark called "following TCP streams." The whole idea of TCP is taking traffic apart and putting it back together again, so why not get rid of the whole "packetizing" operation and look at the original data?

Find out how to do this, and demonstrate this skill to your instructor.

7.34 You are a double-top-secret agent, and you've managed to break into the Elbownian Embassy's VoIP system. You are familiar with Voice over IP, right? Basically it's telephone over the Internet. Use Wireshark to see how many VoIP calls are active.

7.35 If you can pinpoint the TCP stream for a VoIP call, and you can follow that stream, and you can save that stream, can you play back that call?

### Hubs, Routers and Switches

If you're not familiar with these devices, read **Lesson 3, Beneath the Internet**, for more details. The functions of a router, hub and a switch are all quite different even if at times they are all integrated into a single device. Let's start with the hub and the switch since these two devices have similar roles on the network. Each serves as a central connection for all of your network equipment and handles a data type known as **ethernet frames**. Frames carry your data. When a frame is received, it is transmitted on to the physical port the destination PC is plugged into. The big difference between these two devices is in the method for delivering frames.



In a **hub**, incoming frames are broadcasted to all ports. It doesn't matter that the frame is only destined for one machine. The hub has no way of distinguishing which port a frame should be sent to. Passing it along to every port ensures that it will reach its intended destination. This puts a lot of traffic on the network and can lead to poor network response times. Since a hub broadcasts every packet to every machine or node on the hub, a filter in each computer discards packets not addressed to it. A packet sniffer disables this filter to capture and analyze some or all packets traveling through the hub, depending on the sniffer's configuration.

A **switch**, on the other hand, keeps a record of the Media Access Control (MAC) or physical addresses of all the devices connected to it. With this information, a switch can identify which system is on which port. So when a frame is received, the switch knows exactly which port to send it to, without significantly increasing network response times. That's why a switch is considered to be a much better choice than a hub. Rather than a central hub that broadcasts all traffic on the network to all machines, the switch acts like a central switchboard. It receives packets directly from the originating computer and sends them directly to the machine to which they are addressed. This makes sniffing packets on a switch much more difficult. You can only see traffic that is intended for your machine - unless you use more advanced techniques such as ARP poisoning (see Ettercap above or Cain and Abel for a Windows tool) By the way, have you noticed that *\*all\** popular sniffing/MITM tools have been developed by Italians? Including the Winpcap port. What's up with that?

**Routers** are completely different devices. Where a hub or switch is concerned with transmitting ethernet frames at the local Layer 2, a router's job, as its name implies, is to route IP packets to other networks, which is a Layer 3 operation. A packet contains the source address it came from and the data, and the destination address of where it's going.

A router is designed to join two or more networks, commonly two Local Area Networks (LANs) or Wide Area Networks (WANs), or a LAN and its ISP's network. Routers are located at gateways, the places where two or more networks connect. Using headers and forwarding tables, routers determine the best path for forwarding the packets. Routers use protocols like ICMP to communicate with each other and configure the best route between any two hosts. The same packet sniffing issues apply to routers that apply to switches.

## Intrusion Detection Systems

---

You've probably realized that, to use a packet sniffer to detect unauthorized activity in real time, you'll have to sit at your computer, watching the output of the packet sniffer and desperately hoping to see some kind of pattern. An **intrusion detection system (IDS)** does this job for you. IDSs combine the ability to record network activity with sets of rules that allow them to flag unauthorized activity and generate real-time warnings.

### Exercises

- 7.36 Open Wireshark and start a live capture. Now open your web browser and look for a plain text document to download. Download and save the text file to your hard drive, then close the web browser and end the capture session in Wireshark. Look through the packets captured by Wireshark, paying close attention to the ASCII dump in the bottom pane. What do you see? If you have access to an email account, try checking your email while Wireshark is performing a capture. What do you see there?



- 7.37 On the *Capture Options* Screen, make sure that the box marked "Capture packets in promiscuous mode" is checked. This option may allow you to capture packets directed to or coming from other computers. Begin the capture and see what happens. Do you see any traffic that is intended for a computer other than yours?
- 7.38 What do you know about the hardware that connects your computer to the network? Does it connect to the other computers through a switch, a router or a hub? Go to a web search engine and try to find out which piece or pieces of hardware would make it most difficult to capture packets from other computers.
- 7.39 If you are sitting at a coffee shop, library or airport, using WiFi, and you wanted to capture traffic, could you? Could someone else be doing the same to you? What security controls could you use to prevent that?
- 7.40 Research intrusion detection systems. How are they different from firewalls? What do they have in common with packet sniffers? What kinds of unauthorized activity can they detect? What kinds of activity might they be unable to detect?

## Honeypots and Honeynets

People who like to watch monkeys go to the zoo, because there might be monkeys there. People who like to watch birds put out bird feeders and the birds come to them. People who like to watch fish build aquariums and bring the fish to themselves. But what do you do if you want to watch hackers? You put out a **honeypot**. Think about it this way – you're a bear. You may not know much (being a bear) but you do know that honey is tasty and there is nothing better on a warm summer day than a big handful of honey. So you see a big pot full of honey sitting out in the center of a clearing and you're thinking, "Yum!" But once you stick your paw in the honey pot, you risk getting stuck. If nothing else, you're going to leave big, sticky paw prints everywhere and everyone is going to know that someone has been in the honey and there's a good chance that anyone who follows the big, sticky paw prints is going to discover that it's you. More than one bear has been trapped because he liked tasty honey.

A honeypot is a computer system or virtual machine that serves no other purpose than to lure in hackers. A **honeynet** is a network of honeypots. In a honeypot, there are no authorized users – no real data is stored in the system, no real work is performed on it – so, every access, every attempt to use it, can be identified as unauthorized. Instead of sifting through logs to identify intrusions, the system administrator knows that every access is an intrusion, so a large part of the work is already done.

### Types of Honeypots

There are two types of honeypots: production and research.

**Production honeypots** are used primarily as warning systems. A production honeypot identifies an intrusion and generates an alarm. They can show you that an intruder has identified the system or network as an object of interest, but not much else. For example, if you wanted to know if bears lived near your clearing, you might set out ten tiny pots of honey. If you checked them in the morning and found one or more of them empty, then you would know that bears had been in the vicinity, but you wouldn't know anything else about the bears.

**Research honeypots** are used to collect information about hacker's activities. A research honeypot lures in hackers and then keeps them occupied while it quietly records their actions. For example, if – instead of simply documenting their presence – you wanted to study the bears then you might set out one big, tasty, sticky pot of



honey in the middle of your clearing, but then you would surround that pot with movie cameras, still cameras, tape recorders and research assistants with clipboards and pith helmets.

The two types of honeypots differ primarily in their complexity. You can more easily set up and maintain a production honeypot because of its simplicity and the limited amount of information that you hope to collect. In a production honeypot, you just want to know that you've been hit; you don't care so much whether the hackers stay around. However, in a research honeypot, you want the hackers to stay, so that you can see what they are doing. This makes setting up and maintaining a research honeypot more difficult. You must make the system look like a real, working system that offers files or services that the hackers find interesting. A bear who knows what a honeypot looks like might spend a minute looking at an empty pot, but only a full pot full of tasty honey is going to keep the bear hanging around long enough for you to study it.

Honeynets are harder yet; they have to have what appears to be real, live traffic on them.

### Building a Honeypot

In the most basic sense, a honeypot is nothing more than a computer system that is set up with the expectation that it will be compromised by intruders. Essentially, this means that if you connect a computer with an insecure operating system to the Internet, then let it sit there, waiting to be compromised, you have created a honeypot! But this isn't a very useful honeypot. It's more like leaving your honey out in the clearing, then going home to the city. When you come back, the honey will be gone, but you won't know anything about who, how, when or why. You don't learn anything from your honeypot, unless you have some way of gathering information regarding it. To be useful, even the most basic honeypot must have some type of intrusion detection system.

The intrusion detection system could be as simple as a firewall. Normally a firewall is used to prevent unauthorized users from accessing a computer system, but they also log everything that passes through or is stopped. Reviewing the logs produced by the firewall can provide basic information about attempts to access the honeypot.

More complex honeypots might add hardware, such as switches, routers or hubs to further monitor or control network access. They may also use packet sniffers to gather additional information about network traffic.

Research honeypots may also run programs that simulate normal use, making it appear that the honeypot is actually being accessed by authorized users and teasing potential intruders with falsified emails, passwords and data. These types of programs can also be used to disguise operating systems, making it appear, for example, that a Linux based computer is running Windows.

But the thing about honey – it's sticky and there's always a chance that your honeypot is going to turn into a bees' nest. And when the bees come home, you don't want to be the one with your hand stuck in the honey. An improperly configured honeypot can easily be turned into a launching pad for additional attacks. If a hacker compromises your honeypot, then promptly launches an assault on a large corporation or uses your honeypot to distribute a flood of spam, there's a good chance that you will be identified as the one responsible.

Correctly configured honeypots control network traffic going into and out of the computer. A simple production honeypot might allow incoming traffic through the firewall, but stop all outgoing traffic. This is a simple, effective solution, but intruders will



quickly realize that it is not a real, working computer system. A slightly more complex honeypot might allow some outgoing traffic, but not all.

Research honeypots – which want to keep the intruders interested as long as possible – sometimes use **manglers**, which audit outgoing traffic and disarm potentially dangerous data by modifying it so that it is ineffective.

[www.sicherheitstacho.eu](http://www.sicherheitstacho.eu) has set up live feeds of cyber attacks as they happen. The data is based off 180 sensors (honeypots) located around the world. The site shows who is attacking who, the amount of data in the attack (DDoS), and is updated every few seconds.

### Exercises

- 7.41 Honeypots can be useful tools for research and for spotting intruders, but using them to capture and prosecute these intruders is another question. Different jurisdictions have different definitions and standards and judges and juries often have varying views, so there are many questions that need to be considered. Do honeypots represent an attempt at entrapment in your country?
- 7.42 Is recording a hacker's activities a form of wiretapping in your country?
- 7.43 And on the specific question of honeypots – can it be illegal to compromise a system that was designed to be compromised? These questions have yet to be thoroughly tested. Discuss this for a bit – what are your thoughts and why?



## Conclusion

---

The news is filled with stories on cyber attacks. Some of the attacks seem sophisticated while others seem to happen by chance. The largest and smallest organizations are being targeted on a regular basis by one form of digital crime or another. Most movie plots involving action have at least one hacker in them that uses Nmap to destroy the enemy. It's like the world has become one big series of digital wars. Expect to see some TV reality show where cyber criminals face off next. Like the next season of 24.

The reasons for entities to attack each other is as varied as the tools they use. These days most of the attacks are well funded and aimed at criminal behavior. In the old days, attacks were not. Digital crime pays, as does espionage and nation/state warfare. Criminals are using multiple layers of attack to confuse the target.

Financial sectors are being targeted for many types of cyber attacks since that is where the money is. The fastest growing sector for cyber crime is mobile platforms. Malware plays a huge part in the increase of these crimes across the globe. It seems as though attackers are going after anything these days.

To combat and protect yourself, you need to secure your computer/network by thinking about all possible access points. One of the best ways to do this is to think like an attacker. Use the same tools they use against your own domain to see what needs to be strengthened.

Don't focus on the threats as much as your own system. Educate yourself and stay up on news about different types of attacks. The best defense is a good offense. Hacker Highschool encourages you to explore the world around you but do no harm. If you have an issue or a cause, we understand, but caution you to remember the implications of your actions.



Today's teens are in a world with major communication and productivity channels open to them and they don't have the knowledge to defend themselves against the fraud, identity theft, privacy leaks and other attacks made against them just for using the Internet. This is the reason for Hacker Highschool.

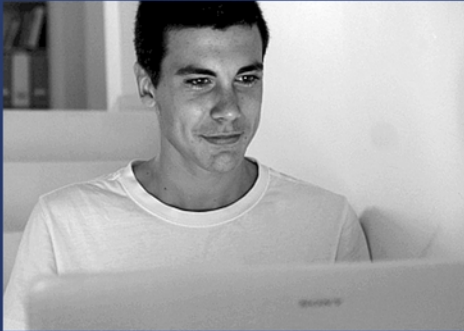
**The Hacker Highschool project is the development of security and privacy awareness learning materials for junior high and high school students.**

Hacker Highschool is a set of lessons and a practical means of making hackers. Beyond just providing cybersecurity awareness and critical Internet skills, we need to teach the young people of today how to be resourceful, creative, and logical, traits synonymous with hackers. The program contains free security and privacy awareness teaching materials and back-end support for teachers of accredited junior high, high schools, and home schooling. There are multiple workbooks available in multiple languages. These are lessons that challenge teens to be as resourceful as hackers, including safe Internet use, web privacy, researching on the internet, avoiding viruses and Trojans, legalities and ethics, and more.

**The HHS program is developed by ISECOM, a non-profit, open-source research group focused on security awareness and professional security development and accreditation.**

# Hacker Highschool

## SECURITY AWARENESS FOR TEENS



## LESSON 8 DIGITAL FORENSICS AND COUNTER FORENSICS



HACKING IS LEARNING  
[www.hackerhighschool.org](http://www.hackerhighschool.org)

ISECOM

Creative Commons 3.3 Attribution-Non-Commercial-NoDerivs ISECOM

[WWW.ISECOM.ORG](http://WWW.ISECOM.ORG) - [WWW.OSSTMM.ORG](http://WWW.OSSTMM.ORG) - [WWW.HACKERHIGHSCHOOL.ORG](http://WWW.HACKERHIGHSCHOOL.ORG) - [WWW.BADPEOPLEPROJECT.ORG](http://WWW.BADPEOPLEPROJECT.ORG) - [WWW.OSSTMMTRAINING.ORG](http://WWW.OSSTMMTRAINING.ORG)



## WARNING

---

The Hacker Highschool Project is a learning tool and as with any learning tool there are dangers. Some lessons, if abused, may result in physical injury. Some additional dangers may also exist where there is not enough research on possible effects of emanations from particular technologies. Students using these lessons should be supervised yet encouraged to learn, try, and do. However ISECOM cannot accept responsibility for how any information herein is abused.

The following lessons and workbooks are open and publicly available under the following terms and conditions of ISECOM:

All works in the Hacker Highschool Project are provided for non-commercial use with elementary school students, junior high school students, and high school students whether in a public institution, private institution, or a part of home-schooling. These materials may not be reproduced for sale in any form. The provision of any class, course, training, or camp with these materials for which a fee is charged is expressly forbidden without a license, including college classes, university classes, trade-school classes, summer or computer camps, and similar. To purchase a license, visit the LICENSE section of the HHS web page at <http://www.hackerhighschool.org/licensing.html>.

The Hacker Highschool Project is an open community effort and if you find value in this project, we ask that you support us through the purchase of a license, a donation, or sponsorship.



## Table of Contents

WARNING.....	2
Contributors.....	5
Introduction.....	6
The Magically Disappearing Data Trick (Where and How to Hide Data).....	7
First Things First – Large Data Sets.....	7
You Can't Get There From Here.....	8
Software Tools.....	8
Digging the Tunnel.....	9
Passing the Buck.....	9
Working From Home.....	10
Next Things Next – Small Bits of Bytes.....	10
Swamped by Swaps.....	10
Give 'em Some Slack.....	11
File Makeovers.....	11
The Disappearing Data Magic Trick (Making Data Irrecoverable).....	12
Wash, Rinse, Repeat.....	12
More Software Tools.....	13
Boot and Nuke.....	13
Eraser.....	13
Sderase.....	13
Hammer, Drill, Bigger Hammer.....	13
Planting a Garden.....	14
Seeding the Garden.....	15
Feed Your Head: Digital Forensics Principles.....	16
Digital Forensic Methodology.....	16
Digital Forensics Process.....	16
Evidence gets lost all the time!.....	17
This is an Exercise for Law Enforcement Personnel Only.....	17
Home Away from Home, or When Business Gets Too Personal.....	18
Software Tools and Collections.....	19
Data Media Analysis.....	20
Time for a Date.....	20
Getting in Time With Offset.....	20
EXIF Data.....	21
Imaging Tools.....	21
Give Them the Boot(ing).....	21
Deleted Data.....	21
Formatting Media.....	22
Precautions While Collecting Evidence from a Data Storage Device.....	22
Steganography: A look at security controversy.....	23
Steganography: It's Real, It's Easy and It Works.....	24
Steganography Is a Scam.....	26
Windows Forensics.....	26
Laptops Are Treasure Troves.....	27
Volatile Information.....	27
Tools for Collecting Volatile Information On Windows.....	27
Non-volatile Information.....	28
Ready? Roll Cameras, Action.....	29
Windows Server 2008 Event Log editing and location.....	29
Linux Forensics.....	29
Linux Slack.....	30
Silly String.....	30



Grep.....	30
More Command-line Tools.....	30
Finding a Haystack in a Needle.....	31
Encryption, Decryption and File Formats.....	31
Feed Your Head: Real Case Studies.....	33
Mobile Forensics.....	33
Connect the Blue Wire to the Red Square.....	34
Some Disassembly Required.....	34
So Many Devices, So Little Time.....	35
iPhone Forensics Example.....	35
Phone Software Tools.....	36
Now What?.....	36
Network Forensics.....	37
Firewall Logs.....	37
Packet Sniffers.....	37
Intrusion Detection Systems (IDS).....	38
Router and Network Management Logs.....	38
Tools of the Network Trade.....	38
E-Mail Headers.....	39
Game On: Getting Down and Dirty.....	39
Let the Fun Begin.....	42
Reconnaissance.....	42
Software and hardware vulnerabilities.....	42
OpenVAS.....	42
Weapons to Hack Networks.....	43
Counter Forensics.....	44
Just Who Has the Advantage.....	45
You Gotta Be Social.....	45
Head in the Clouds.....	46
Issues with Cloud Forensics.....	46
Conclusion.....	47



## Contributors

---

Pete Herzog, ISECOM  
Marta Barceló, ISECOM  
Chuck Truett, ISECOM  
Kim Truett, ISECOM  
Marco Ivaldi, ISECOM  
Bob Monroe, ISECOM  
Simone Onofri, ISECOM  
Greg Playle, ISECOM  
Willy Nassar  
Ken Withey

**ISECOM**



## Introduction

---

If you are attempting to go through all the effort of learning to hack and actually conducting some hacking, you will need to learn how to cover your tracks. It is safe to assume that you will be the focus of an investigation if you pull off a really great hack. Never mind the “whys” and “hows” of the hack; investigators are going to look for evidence to connect you (the suspect) to the crime. The investigators you are interested in are given the lovely name of “Digital Forensic Examiners.” That name sounds a bit scary. Don’t worry, this lesson will tell you all about those investigators.

Each of the lessons in Hacker Highschool is like a sip of water from a vast ocean of information. You are getting a small taste of the massive topics to whet your appetite for hacking. In this particular lesson, you are being armed with sophisticated knowledge to keep you safe. Knowing how to use this knowledge is entirely up to you. This lesson is going to show you safe locations to put data and how to hide your treasures from prying eyes. What good is hacking a system and obtaining vital information unless you can store your prize in a safe place?

After being a hacker for a while, you will find yourself overloaded with all sorts of media that you need to get rid of. Maybe you don’t want that 256 megabyte USB thumb drive anymore. Perhaps that 1 gig SD card is too small for any useful purpose other than a book place marker. Whatever the case might be, it is not a good idea to toss that media in the trash. That old hard drive, yeah, the one you used when you were saving images from lingerie department store web pages. Yes, that drive definitely can’t go in the trash in its current condition. We’ll show you ways to blow useless data into bits (pun). You’ll learn how to ensure nobody ever reads that media again. Evidence needs to be erased.

Once we delete your unwanted collection, you will probably go right back to exploring domains. Hacking a system means that you will leave little clues of your entry, your exploits, and your exit. If these digital tracks are left in the system, you will be getting some attention from the local authorities. You don’t want that, do you? Much like your dirty room, you need to know how to clean up after yourself. Everything from concealing your location, altering your entrance methods, changing the system logs time, moving the data without being noticed, and setting up a backdoor needs to be planned for and handled as you go. We’ll discuss the best techniques to use.

If you happened to get that dreaded knock on your front door by a team of gun-toting law agents and find yourself on the wrong end of that barrel, we will talk about simple but effective ways to side step or slow down your investigation. Maybe you need a lawyer, maybe you don’t. There are lots of ways to stay one step ahead of law enforcement. There are even more ways to have fun with forensic examiners.

Counter forensics is exactly what the name says it is. Think of digital forensics as a game of hide and go seek; you make all of your moves before the other person even starts. How you apply counter forensic tactics depends on what you are trying to accomplish. Are trying to delete evidence, slow an investigator down, tamper with the evidence to make it appear unreliable, or just have some fun with the gate guards. This area will provide an overview of all the topics we have discussed and possibly make sense of it all.

Only a few hackers ever work alone. These days, hacking is a business. Hacking organizations have offices; they have a management structure, and payroll systems. One could only wonder what health and retirement plans they offer employees. The organized hacking business has a fairly good communication system, partly thanks to encryption. You will also need a communication method that protects both yourself



and the receiver from unwanted listeners. Whomever you will be working with, you will be introduced to methods for better protection. If your cellphone becomes a piece of evidence, this lesson will show you how to disable tracking mechanisms and SIM card intercept. We will discuss SIM card modifications and using the Advanced Encryption Standard (AES) to safely send and receive VOIP on a cellular line.

You will be introduced to the weapons used on the battlefield. Why show up for a gunfight with a knife? This section will cover the latest and greatest commercial and open source forensic software used. Along the way, we will touch on methods to bypass commercial firewalls, IDSs, behavior management tools, and other bumps in the road. You do not want to leave traces of your activities, or more importantly, show them how you bypassed their expensive equipment. It will be useful knowledge to know the weaknesses of forensic tools and ways to exploit those issues.

To conclude our lesson, we will walk you through the most effective steps you might need to infiltrate systems. Once inside, you will be guided through methods to enter systems undetected, conduct your mission, leave a backdoor, clean up the logs and activity trackers, and then exit without being noticed.

## **The Magically Disappearing Data Trick (Where and How to Hide Data)**

Suppose for a minute that you just stumbled on (hacked into) a site with very special information. Let your imagination play with the idea of "special information." Whatever that information is, you happen to grab a copy of it. Nice job.

So, here you are with several megabytes of information on your computer. Where are you planning to store it all? Keeping it on your computer is not recommended. With your imagination already engaged, suppose there are eighty-five armed law enforcement agents heading towards your house. These agents have mean attack dogs, a helicopter with guns, and none of them has had their morning cup of coffee yet. Not even the dogs. You need to make that special information magically disappear, and quick.

### **First Things First – Large Data Sets**

You can have all the (borrowed) data you want on your computer (not a smart move) as long as you use strong encryption so it isn't in plain text anymore. A better idea would be to put that incriminating information behind a secret revolving bookcase. You do have a secret revolving bookcase, don't you? Alright, we'll move that encrypted data somewhere else since you are the only person who doesn't have a secret revolving bookcase. The secret revolving bookcases are on sale, by the way. Buy two if you can.

An old trick to store incriminating data is to put it on someone's computer, presumably without their knowledge. Many other hackers use this same technique, since it allows them to place the burden of proof on the shoulders of law enforcement. It is difficult to find a person guilty of a computer crime if that person doesn't have any evidence that links them to the crime.

Let's get back to those cops, angry dogs and helicopters with guns heading your way. Before you found that one site with the special data, you may have located a few other servers that didn't interest you. For example, those three servers for "Diapers International" had low security, plenty of spare room, and small spurts of activity. (There's a joke in there somewhere, trust us).





Go back to “Diapers International” and take a peek at their server package. If anything smells suspicious, get out fast. Otherwise, look for a directory that is either used frequently or hardly used at all. Both types of directory activity have their advantages and disadvantages.

In an active directory, you can create multiple subdirectories and store your data without the transfer of data being noticed as much. The size of the data load should not set off any alarms because that main directory is in constant use. The bad news is that directory could be watched more closely than others due to its value to the organization. That active location is most likely being backed up on a regular basis. You don’t want additional copies of your valuable data (evidence) floating out there.

Inactive or dead directories are popular spots to hide data. These locations may have served a purpose to the organization at some point.

This is where you will want to create a maze of subdirectories or set up a hidden directory. If you go with the maze, create a mental map of how you are going to navigate this maze to store your data. The idea is to build a pattern of subdirectories that you will store your encrypted data in. That pattern of storage needs to confuse anyone who locates your stash but you will know exactly what that pattern is. For example, if you build a directory under a few other directories, start branching off additional sublevels. For each sublevel or tree branch, those will split off into more sublevels or branches. Your storage pattern might be something as simple as, left sublevel, right sublevel, right, right, left.

## **You Can’t Get There From Here**

---

A simple question you might be asking yourself is “how do I keep my socks from smelling so bad?” Sorry, we can’t help with that one, El Stinko, but, we can show you how to move large amounts of data from your computer to the “Diapers International” without being noticed. The Internet Control Message Protocol (ICMP) is long forgotten protocol that has magical covert powers if hacked a bit.

When you scan a port, you are really sending a TCP SYN request (layer 4) to see if that port responds. A proper ping uses ICMP, which doesn’t use ports. Although ICMP rests on top of the Internet Protocol (IP), it is not a layer four protocol. This comes in very handy when we get into the firewall and network traffic logging.

Firewalls operate at several levels of the OSI model, restricting or allowing data flow based on the criteria that is given. The higher the stack layer, the deeper the firewall can inspect the contents of each packet request. At the lower layers, the firewall can still intercept and control data movement but it doesn’t know as much about the data as it does at the higher layers. This is where ICMP packets become fun ways to deliver content.

The technique is known as “ICMP Tunneling.” Before we can do much with this covert communication, we need some software tools.

### **Software Tools**

- Wireshark - [www.wireshark.org/](http://www.wireshark.org/)
- Hping - <http://www.hping.org/>
- Kali Linux - <http://www.kali.org/>



ICMP packets have plenty of room after the header to store data (roughly 41k per packet). The idea here is to handcraft ICMP packets loaded with your data and send them through a covert ICMP tunnel to the location you want. You can generate ICMP packets using hping or nping (from the nmap people) and insert your payload at the same time. With these ping tools, you can customize the Ethernet header, the IP header and the payloads.

## Digging the Tunnel

---

Why should you have to do all the hard work? Why not have the server on the other end do some of the work for you? You will need to set up an ICMP tunnel between your computer and the storage server, you know that much. Now it's time to think like a hacker.

### Exercises

- 8.1 The first thing you need to do is figure out how this technique works. Here's where you explore the value of one of the most valuable resources for hackers on the Internet: Youtube. Given the pace of change on the Web, there may be a new video site of the day by the time you read this; but given the strength of certain search-based businesses, it's likely Youtube may still be your primary video resource.

To see a demonstration of how the ICMP tunnel works, head to <http://www.youtube.com/watch?v=ADHtjwwkErl>

- 8.2 There are two sides to a tunnel: the server side and the client side. You'll see that the creator of the video promises to release the source code, but never did on Youtube (despite visitor pleas).

Now you need to find the source code, which has indeed been posted to the Internet. Find the code. One further hint: notice the author's name.

- 8.3 You would need to get the server-side code onto your target, and run it, for this to work. How could you get that code onto a target?

Once both of these daemons are running host to host, the server will begin sniffing for ICMP packets. You will be sending commands through the tunnel to the server using ping and the server will respond in turn with ping packets. The server daemon will begin collecting your packets and placing the data where you have instructed. If the data flow is large, the server will establish additional multiple pings. The client side daemon will receive transmission updates through the same type of sniffer used on the server.

## Passing the Buck

With portable drives having ever-higher capacity and smaller sizes, physically hiding large amounts of data is straightforward; put the media in a safe place away from your house and your computer. When those agents and mean dogs break down your front door, expect them to search every place imaginable; even your underwear drawer. Consider the fact that these people search houses every day for a living. They know all the hiding spots. Don't hand the media to a friend for safe keeping either, that's just not cool.

Before you even think about places to hide your treasure, encrypt the media, the data, or both first. Try TrueCrypt at [www.truecrypt.org/](http://www.truecrypt.org/).



Place the media in a sealed plastic bag, something that is weatherproof. Use a straw to suck the air out of the bag to reduce moisture content as well as size. Don't dig a hole in the ground near your house to bury the media because fresh dirt will look suspicious to the dogs and the agents. Instead, look for hiding spots that are high off the ground. People rarely look up for some odd reason. Just make sure you can get to that spot when you need to. Expect any tape you plan on using to secure the bag to fail. Use twist ties, zip ties, twine, shoestring, or other objects that will ensure your bag doesn't come loose and fly away.

A favorite hiding spot is anything near a police station or in a police station. There are very few good hiding spots inside a station but plenty near the outside. Use your imagination but also think logically about placing, recovering, and leaving the area without drawing attention to yourself. Your activities may be better suited for daytime, since the dark tends to alarm people more. Planting a bag in broad daylight, usually later in the afternoon, wouldn't draw nearly as much attention as it would in the evening.

## Working From Home

Many organizations offer free cloud storage with nothing more than a valid email account. Some places like [www.Adrive.com](http://www.Adrive.com) will give you 50 GB of free online storage. Google, Apple, Microsoft and many others provide various amounts of free storage. These cloud services, using a one-time email account, can be useful locations to store data. All you have to do is clear out your browser cache each time you visit and/or go incognito with Chrome so there are no traces of your visits on your computer. Some of these cloud services will allow you to synchronize your computer files with the online account. Disable this function and remove all entries that point to the accounts. It is easier and safer to view your data through a web browser instead of using the cloud's interface.

## Next Things Next – Small Bits of Bytes

If you have small amounts of data, like passwords, private keys, or a secret recipe for soup, you can slip that data into places that will not be noticed. Don't go so far as trying to hide data in your DNA, we've already tried that and it gave us the lousy sense of humor you are reading. Plus, we twitch a lot. There are better ways.

Malware creators have long known that there is storage space on Windows systems in the Master Boot Record (MBR). It's not much space but enough to hide a private key or a DLL. Your lunch bag will not fit in the MBR, so don't try it, we already did.

## Swamped by Swaps

**Swap files** are places on a media drive that is temporary RAM. The swap file space allows the computer to run faster even if it runs out of RAM to execute programs. UNIX and Linux set aside a permanent block of media for swap storage. Even if the computer is turned off, this hard drive swap file space can still contain data from previous events.

Windows swap files (**page files**) can get quite large and hold pieces of recent files. This could be even more dangerous if you were connected to a windows based server. Windows servers store a significant amount of user data that can be handy to the forensic examiner. Take a look at "temp" directories for the swap files.



## Give 'em Some Slack

Files are stored in **clusters**. Depending on the operating system, the clusters can vary in size. If you created a file on your computer, that file might only need 50% of the cluster's space. This leaves a cluster with open space left. This open space within a cluster is called **file slack** or just **slack** for short. If you delete a file that was in that partial cluster space, that space is still available even if the file was deleted.

The 50% cluster space that was previously occupied with a file, will keep that data intact. These data remnants remain in the cluster until it is filled with other data. Windows automatically creates slack space as soon as any file is created, viewed, modified, or saved.

## File Makeovers

Some of the best places to hide data is to hide it in plain sight. File modification is just a fancy way of changing the name of a file, altering the extension of files, or changing the file attributes. By now, you should already know how to change the name of a file. You made a file "Evil Plans" earlier, now let's get creative. Would you put all of your passwords in a file and name that file "Passwords?" No, of course not. Nor should you put all of your work in files that can be easily identified.

When looking to modified files, look at file extensions. File compression is an easy way to cover tracks and save space, however, those files will be the first one the agents will be checking. So, you will need to alter the file extension. This can be accomplished by editing the last three characters of the file name.

Changing a .doc file to a .gif is as simple as changing an .odt file to a .avi. Creating the altered files can become tricky and time consuming. Look at file sizes, created dates, and modified dates to give you ideas of how to customize each file. An .odt file should not be a gigabyte in size, as well as an .avi file should not be a few kilobytes either. An .avi file should be several gigabytes in size.

Look at the file dates too. The files that were created or accessed within a week of the criminal event and after the event should ring a bell in your head. Alter those dates to any day at least a year before your hack. If you really want to have some fun, change the dates to impossible dates, such as 30 February or 21 March 2112. Don't forget Pi Day, using that date and time will really show who knows their math and who doesn't.

## Exercises

- 8.4 The date is 21 December 2012. During a forensic examination you locate several files. Along with files you can see the size of the file and the file type. Digging deeper, you also notice the date those files were created and the last time each file was opened or modified. Look at each of the following files and see if any file looks suspicious.

Name of file	Type of file	Size of file	File creation date	Last time file was accessed or modified
Passwords.exe	Executable	13KB	May 2008	12/19/12
Fall 2012 vacation.jpg	Picture	12948KB	June 2009	12/19/12
Planstokillwife.doc.	Word document	2KB	December 2012	12/20/12
Love songs.mp3	Music file	7985340KB	Unknown	Unknown



Which of these files look suspicious?

Which files would you analyze first?

Are any of these files fakes?

Why did you chose your response for each file?

There will be many times in your life were you may think using a hammer will help solve computer hardware issues. A well-known tool store once sold a tool set called the "Ultimate Tool Kit," inside of which was a box of ten different types of hammers. In the field of forensics, hammers will not help you solve any cases. Using a hammer may create other challenges, possibly making you a suspect.

## The Disappearing Data Magic Trick (Making Data Irrecoverable)

Behind these two doors we have option 1, which is digital sanitation and option 2, physical destruction of media. Each option has merit, but it will be up to you as to which method you want to use. The folks here at Hacker Highschool love the sound of an electric drill boring holes into old hard drives. If you put that sound to music, you would have an outstanding remix. However, if you can't bear to see or hear holes plunged violently into hardware, we have media sanitation to remove, unwanted data. A kinder, gentler way of blasting the heck out of unwanted data bits.

After you have been using a hard drive or any other type of digital storage device, you will probably get to the point where it doesn't serve a useful purpose. It is a really awful idea to throw that media in the trash or give it to someone who might use it again unless your data is removed first. Do you really want someone finding that jpeg of you in your Batman costume last year? What about those old receipts from your last part-time job? Would you want those homemade videos of you dancing in your underwear to end up on that video site? Well, let's get rid of those worrisome digital memories on that old media before you pass it along to someone else.

### **Wash, Rinse, Repeat**

Sanitizing media is inexpensive (not very much fun) and provides secure destruction of sensitive data. You can eradicate data, wiping those digits off the face of the earth using open source software. One of the simplest methods is to encrypt your media using True Crypt. Once the entire physical chunk of storage is encrypted, it is now somewhat safe to toss that hardware away. The logic is that the entire data image cannot be decrypted unless you provide the passphrase. Pretty simple, right? If those eighty-five agents get their hands on your old media, it is useless to them since you are the only person who can unlock the data. If another person obtains your old media, they will have to reformat and repartition it before it can be used.

There are roughly two standards for proper media destruction. The first one is US DOD 5220.22-M and the other is the Gutmann algorithm. DOD 5220.22 is a US National Industrial Security Program Operating Manual that provides instruction on destruction of data. The U.S. Department of Defense like to destroy things too, so they only authorize complete destruction as a means to remove data.

The Gutmann algorithm, named after Dr. Peter Gutmann and Colin Plumb, gives a little more latitude on physical annihilation of hardware. The algorithm requires the media to



be overwritten thirty-five times in a manufacture specific pattern. Different drives require different overwrite patterns. Although this method is an outstanding researched backed technique, it has been outdated due to the size of newer drives and built-in controller settings.

## More Software Tools

---

Within the open source community, there are some great software tools will make your data impossible to recover. The software will not damage your media but will make the data on it unrepairable. When running the software you press the "start" button, don't expect to ever see that data again. Not even in the afterlife.

### Boot and Nuke

<http://www.Dban.org>

Boot and Nuke comes as an ISO image that you burn to a CD and boot your system off of it. Once the software is up and running, you just select which drive you want sanitized (Nuked). Dban is an industry standard for bulk data destruction and emergency uses. Once Dban has been used on a drive, there is no forensic recovery possible. That data is gone, bye-bye.

### Eraser

<http://sourceforge.net/projects/eraser/files/latest/download>

This program is strictly made for Windows. Even though Window Vista on up has ability to format and write ones over the media, Eraser formats the drive and writes random data over the drive many times. The program does this function several times, format, write, format write and so on until a pattern is completed.

### Sderase

<http://sourceforge.net/projects/sderase/?source=directory>

SD is a newcomer to disk wiping, just released August 28<sup>th</sup> 2012. The programs creator made an interesting comment on the web site. SDerase proclaims that it meets US DOD 5220.22-M data sanitation requirements. US DOD 5220.22-M mandates that the only acceptable method for media and data removal is physical destruction of the media. We have yet to see any software that can perform physical damage.

### Hammer, Drill, Bigger Hammer

The one method that has stood the test of time for media eradication; the one method that all experts agree on its success, is physical destruction. Smash it, pound it, pulverize it, or tear it apart. Use your imagination to find ways of ruining media. A magnet will only work on magnetic material, so flash drives will just laugh at you if you put a speaker magnet next to it. The laughing media will certainly pause to reflect its pending doom when you show up with a hammer, though.

A typical carpenter's hammer will apply approximately a lot of damage to any solid object it impacts. Larger hammers apply larger amounts of damage (and fun). Keep your thumbs clear. A rock can perform the same function as a hammer, on your thumb and on the media you want to destroy. From a Return on Investment (ROI) perspective,



a rock is more economical but can require continuous replacement with long-term use.

Likewise, an electric drill using a large bit can produce excellent demolition results. The best method for drilling destruction is to drill several holes in various places on the media. There are additional hazards involved with using a drill that must be considered:

- Wear safety glasses or similar eye protection
- Do not try and hold the media in your lap while you drill
- Do not try and hold the media in your hand while you drill
- Do not ask a friend or family member to hold the media in their lap or hands while you drill
- To reduce damage to your drill and drill bit, place cardboard or wood under the media before drilling

Once you are done turning your old media into tiny pieces, your next step will be to spread the parts over several garbage cans. Grab a slingshot and practice hitting cans with the remaining parts. Make an art project out of the ruins. There are all sorts of ways to separate the small parts over a large area. You might as well have fun scattering them.

## **Planting a Garden**

---

To survive in this line of work, you need to be a bit paranoid. Okay, a lot paranoid. Being alert and planning ahead is a good idea and not something to be taken lightly (this same advice can be applied towards early retirement planning). In the physical world, we leave hair strands, fibers from our clothes, fingerprints, shoe prints, and other evidence of our presence. Unlike the physical world, it is possible to enter a digital area, spend some time playing around, and exit that area without leaving a single piece of evidence that you were ever there. Consider how this can work for better and for worse – and exactly how bad it could be, to be on the wrong side of this process.

We will cover the whole process later on, but here we are going to focus on hiding your tracks. In networking there are two types of devices. The first device is basically a “dumb” device, which means that the device doesn’t keep a log of activities. These devices are common switches, hubs, bridges, and so forth. They just do whatever it is they were designed to do.

On the other side, we have “intelligent” devices that do keep logs of certain activities and can invoke decisions based on the filters and configurations that are installed. These devices fall into the category of firewalls, routers, range extenders, servers, and other network hardware that keeps track of data flow. These are the devices that you will need to pay attention to because they are the ones that will monitor, record, and possibly disrupt your hack. These network roadblocks are covered in depth at other HHS lessons.

You need to know how to deal with these devices to cover your tracks and if needed, lead those eight-five agents somewhere else. In your planning, it might help to work backwards on a timeline. This allows you to set the amount of time you will be in that network and minimize the chances of you being caught by controlling your exposure time.



## Seeding the Garden

You will need to consider multiple ways to properly cover your tracks, before you exit the target network. If you just depend on a single method, such as erasing all log files, you are leaving yourself open to other tracking methods. Erasing log files may sound like a super idea but what happens if there are hidden redundant logs? Oops. We need to choose several courses of action that complement each other but do not interfere with your overall plans. Consider these points from the perspective of the investigator – and that of the perpetrator.

Planting logic bombs have been used in the past by outsourced vendors who haven't been paid, angry admins, and ransom-minded folks. Each of those examples place logic bombs where maximum damage to data will occur. Complete network data destruction is not a great idea if you want to keep a low profile after a network breach. A logic bomb that will simply delete or corrupt log files if triggered by an audit within so many days (five) or hours after your exit, would work well to cover your tracks and not alarm too many people.

CCleaner (<http://www.ccleaner.com/>) is a free Windows-based program that has consistently performed well for home and commercial users. (It was originally called Crap Cleaner but when they suddenly went big time they realized they'd have to have a more respectable name.) With this 332 KB utility, you can select which log files you want to delete or edit on any machine you have admin access to. You can even clear your browser history, erasing you own tracks once your job is complete. CCleaner will try to make a system restore point before it alters anything. Your two choices are to not allow a restore point or look for a file in the root directory labeled "cc\_20110928\_203957" or something like that. Remove and delete that file before leaving, even if that file is on your own drive.

Root kits hide activity and is valuable for Linux-based servers that do not have many security holes to use.





## Feed Your Head: Digital Forensics Principles

Like any science, digital forensics relies on well-defined methods, which are basically concerned with keeping evidence intact. Consider this logic: if you've lost control of your evidence, even for a minute, it's not evidence any more – it may have been altered.

Data files can be altered without seeming so. It is easy to change the date, check sum, and last access dates after someone has edited a file or a log. One of the forensic investigator's duties is to prove that the evidence collected has not been altered in any way. Basically, you will need to prove that nothing has happened to that data while it was collected, analyzed, in custody and at all times.

This is why it is important to use proven procedures to collect, inspect, and handle all data. One simple mistake could render all your hard work worthless. This is where a good technique comes into play. By conducting each investigation in the same manner and using the same process to document your actions, you can prove that nothing has tampered with your evidence.

### Digital Forensic Methodology

Digital forensics is a branch of criminology, so it's all about a procedural collection of legally useful evidence. This means that you are only looking for evidence that matches the crime, not poking around just because you can. The following figure shows the steps you'd take in helping a criminal investigation.

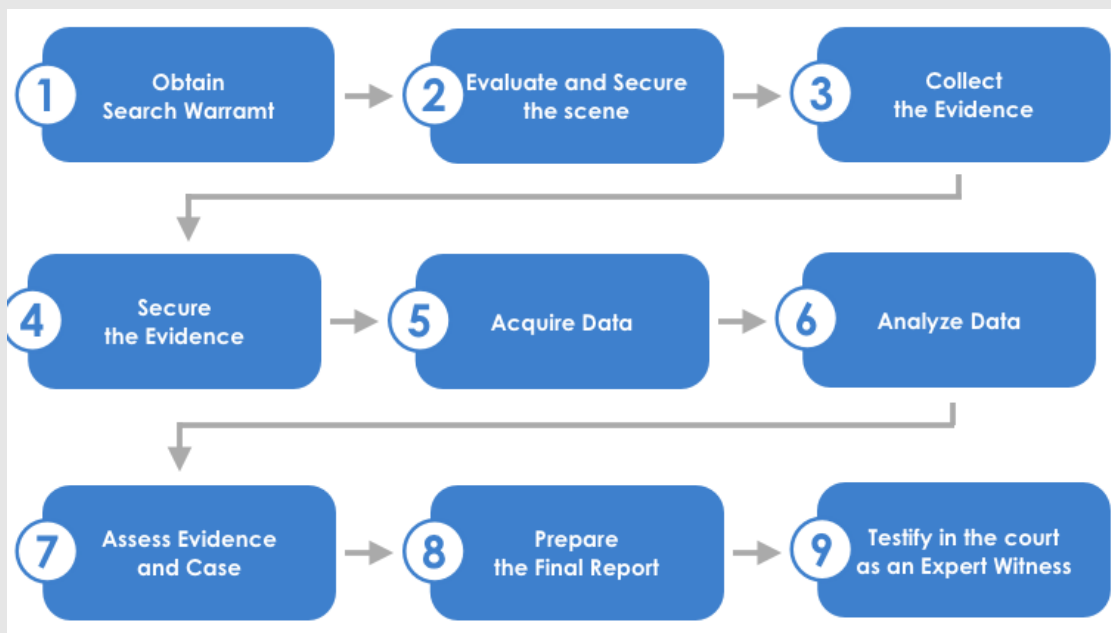


Figure 8.1: Digital Forensics Methodology

### Digital Forensics Process

When you're investigating a computer crime, you need to base your work on a process, such as a policy, procedures, and checklists. Your forensic process must be repeatable and hold up to scrutiny by other forensic experts. You should

develop an investigative process that requires constant documentation of everything action that evidence goes through. If it isn't documented, it didn't happen.

An example of this process is:

1. Identification of evidence (must be documented)
2. Collection of evidence (must be documented in Chain of Custody)
3. Preservation of evidence (must be documented in Chain of Custody)
4. Analysis or interpretation (of course, must be documented)
5. Communication of your findings and documentation

Probably the most critical single document will be the **Chain of Custody** documents, which should state exactly what has been taken as evidence, by whom, and who has subsequent custody. Lose custody of your evidence, even for a few minutes, and it's probably worthless now.

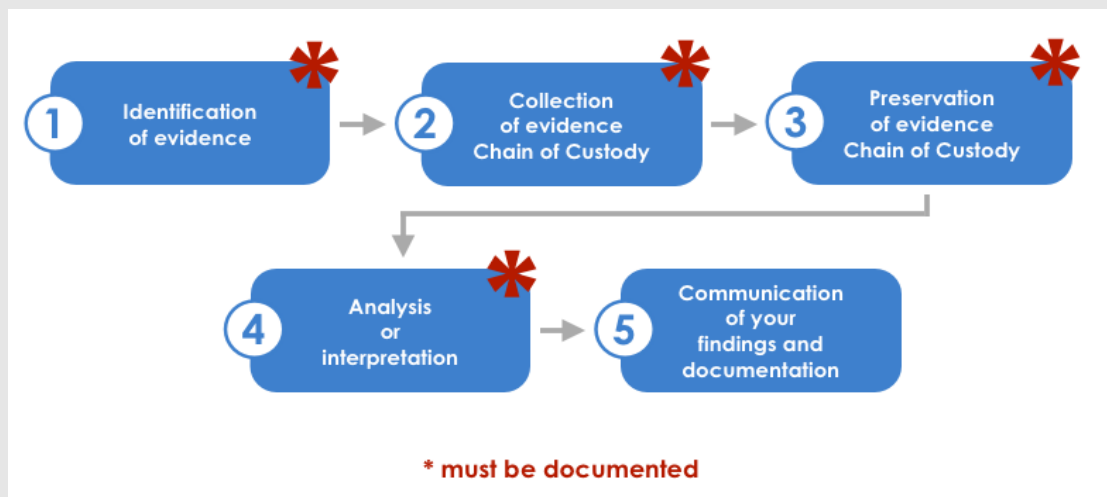


Figure 8.2: Digital Forensic Process

### Evidence gets lost all the time!

Don't let all your hard work become wasted because a piece of vital information wasn't labeled or never made it through your Chain of Custody procedures. Unlike your teachers, a judge will not accept "my dog ate my forensic evidence" as an excuse for something missing.

On the other hand, lost evidence may be a wonderful thing, depending on your perspective.

## This is an Exercise for Law Enforcement Personnel Only

Police criminal records are stored on local servers, usually housed in the main headquarters with a backup file system stored at a satellite police station. Within the primary criminal records database, there are several subsets of information. These subsets are where daily investigative data is kept, individual background history (arrest warrants, recent prior offenses), results of forensic test requests,



As with the FBI's criminal mainframe, there isn't any public accessible link via the Internet. These workstations can access the Internet themselves though. Communication links are available through workstations in the police buildings, with privileges granted based on job function. Along with the workstations, police vehicles are equipped with remote communication encrypted laptops. These vehicle computers are highly capable with access to most of the typical police databases and the Internet.

Currently, these portable computers remain on even when the vehicle is turned off for short amounts of time. Communications are handled through wireless data networking operating on the "Part 90 private and public Land Mobile Radio (LMR) two-way radio system licensees operating legacy wideband (25 kHz) voice dispatch or data/supervisory control and data acquisition radio systems in the 150-174 MHz (VHF) and 421-512 MHz (UHF). The FCC has mandated these frequency bands must make the transition to the narrowband technology (12.5 kHz or less) by January 2013.

### Exercise

- 8.5 This narrowbanding is causing a major cost crunch for many law enforcement agencies since a great majority of them invested in frequency hopping radios. Frequency hopping allows a radio to "hop" across a spectrum of radio frequencies, making it difficult to jam or intercept. Each radio is set to "hop" based on a master radio and the time, plus or minus 3 seconds. Once the slave radio is synched to the master radio, all transmissions will sound perfectly normal, even as the radios bounce through 70 frequencies a minute. Narrowbanding shuts down this hopping capability, since the radios are restricted to only a few channels. Can you find out what frequencies are used in your area?

Local law enforcement agencies have established Public Utility Contracts (PUCs) with major wireless carriers to provide data usually through the entire range of their jurisdiction. The wireless frequencies are the same as your typical data cellular device, EDGE, 2G, 3G, and 4G LTE. The only difference in data transmission is the SSL encryption used between the servers and the mobile computers. Programs like Snort and Wireshark work well to intercept data packets, however, the computer needs to be stationary or the police have to be chasing after you.

Earlier we talked about hiding things near police stations. There is another reason to hang around a police station, especially the motor pool where the cruisers are parked. This is perfect location for packet capturing of login and password credentials. When the police vehicle is first unlocked and prepared for the next patrol shift, the onboard computer must be authenticated and synchronized to the data servers. This is performed at the beginning of every shift change, since one police officer is replacing another police officer.

Another recent implementation to the mobile computers, VOIP has been added. The primary purpose of this addition was to stop interception of police radio transmissions by bad guys and nosey reporters. VOIP is whole other system wrought with vulnerabilities.

## Home Away from Home, or When Business Gets Too Personal

One major flaw in the law enforcement community is their use of email, both on and off duty. The use of cell phones for personal calls, emails forwarded to home accounts, Facebook and other communications has blurred the lines of "official business." FBI



agents take work home with them all the time, just as police officers do. Getting that work data in and out of the office is as simple as a mouse click, for anyone.

Consider this, most email user names start with some combination of first name, last name, separated by a dot and followed by the agencies address. This would look like First.Last@police.state. Gov. Let's say the email address you want is for police officer Dean Martin with the New York State police department. That email address would be D.martin@troopers.ny.gov. Each police department publicly displays their URL on their web site, usually under "Contact Us" or "Complaints."

In many cases, the first portion of their email address will be the officer's user name. Where this comes in handy is the open sharing of information between other law enforcement agencies. Once you have gained access to one email account, it is easier to intercept those communications and move towards the active investigation database.

One problem will be with accessing the databases. This access is controlled by a need to know and access is only granted to those areas the officer is working on. If you log in as one agent and attempt to access a part of a database that agent should not have access to, red flags will go off. This all goes back to the reconnaissance portion of your hack. Know as much as you can about who is doing what with your case.

There several flavors of security/forensics distributions of Linux on the Internet. Consider going to [www.securitydistro.com](http://www.securitydistro.com) and trying several. Before you go trying some of these software kits on your parents computer, read the documentation. Each software package contains powerful elements that can easily ruin your day if not properly executed. Granted, the whole idea behind hacking is to learn by trying. Just be careful, be educated and never forget that your actions have effects on others. Someday YOU will be the "other."

## Software Tools and Collections

The most common digital forensics/security testing open source or free software is in collections of tools such as:

- **Kali Linux** ([www.kali.org/](http://www.kali.org/))
- **Sleuthkit** ([www.sleuthkit.org](http://www.sleuthkit.org))
- **Katana** ([sourceforge.net/projects/katana-usb/](http://sourceforge.net/projects/katana-usb/))
- **CAINE** ([www.caine-live.net/](http://www.caine-live.net/))
- **Wireshark** ([www.wireshark.org/](http://www.wireshark.org/))
- **DEFT** ([www.deftlinux.net/](http://www.deftlinux.net/))
- **HELIX** ([https://www.e-fense.com/store/index.php?\\_a=viewProd&productId=11](https://www.e-fense.com/store/index.php?_a=viewProd&productId=11))
- **UNetBootin** (<http://unetbootin.sourceforge.net/>)

## Exercises

8.6 Head over to any one of the forensic software providers listed above. Follow the directions to create your own live CD. The word "live" means that the media can boot up your computer and load its own operating system without needing to use the one already installed on that machine.

8.7 Now use UnetBootin to make a bootable USB drive loaded with the same forensic tools. Remember that these tools are running on Linux (various flavors)



so don't worry about compatibility with the operating system you are already using.

8.8 Play around with the software tools and read the documentation. While you are at it, mount your own hard drive and attempt to recover any files that you might have deleted recently. Once you recover a deleted file, rename that file to "Evil Plans" using the same file extension it already had. You'll be using that "Evil Plans" file later on.

8.9 Many of the software packages have graphical user interfaces (GUIs) and some run using the command line. Take a close look at those programs that run from the command line. Notice how the switches (/s) at the end of each command can create powerful tools within themselves.

## Data Media Analysis

Computer forensics investigators use various software tools for analyzing and recovering data on various forms of media. There are two basic reasons to conduct a forensic analysis: to reconstruct an attack after it occurred, and to examine a device that may have been used to carry out a crime.

The first step before proceeding with any type of data analyze is to make an exact image of the evidence and to only work with that image. The software tools mentioned earlier allow investigators to perform the following tasks and more:

- Search for text on media devices in file space, slack space, and unallocated space
- Find and recover data from files that have been deleted or hidden
- Find data in encrypted files
- Repair FAT (FAT16, FAT32, eFAT) partition tables and boot records
- Recover data from damaged NTFS partitions (often Linux can do this when Windows can't)
- Joined and split files
- Analyze and compare files
- Clone devices that hold data
- Make data images and backups
- Erase confidential file securely
- Edit files using a hex editor
- Crack certain encrypted folders and files
- Alter file attributes or remove restrictive permissions (read or write only)

## Time for a Date

### Getting in Time With Offset

Event time is usually crucial, so the **offset** between the time of the system from which evidence has been taken and atomic time should be recorded (don't forget the timezone!). Typically, this is done AFTER evidence has been secured since it involves starting the system.

Knowing when an event happened or didn't happen is a crucial fact that must be established for each piece of evidence. If a suspect admits that they "never sent a threatening email" to the victim, your job will be to locate that email and confirm when it was sent and by whom. Throughout this lesson we will convey this same



message until we think you've had enough. Then, we will bring it up one more time, just to be sure you are sick of hearing it.

### EXIF Data

Digital photos are encoded with **metadata** known as EXIF or Exchangeable File Image File Format. The original idea of using EXIF was to offer photographers precise data on each photo, such as shutter speed, color balance, and time and date of the photo. The amazing array of information includes even more additional data if the camera has a GPS activated, including location services.

Most of the cameras that input this tracking data are cell phones. Cell phone cameras include in the EXIF personal data about the users name and if the phones GPS is operating, the EXIF will provide the location where the photo was taken.

Granted, all this information can be spoofed however, few people know about this metadata in the first place. One picture posted in a social media sight can be enough to locate your suspect.

### Imaging Tools

Just as with hard disks, any data storage media that could be evidence should be imaged and then stored, so your analysis work is only done on the image. You never want to work directly with the original evidence because doing so could alter the information on that media. Each of the forensic software collections mentioned above can create an exact image of most forms of media. If your forensic lab computer can read the media, those software tools can image it.

Use hashing techniques to ensure that the binary image is an exact bit-for-bit copy of the original. Take a hash of the original. Create the image, and then take a hash of the image. If the two hashes are the same, you have an identical copy. This should be performed by the same software we discussed earlier. It's no use to work on an image that isn't the exact same as the original evidence.

### Give Them the Boot(ing)

Bootting is the process by which a small program actually initializes the operating system installed on a computer or on the booting device. Part of this process involves looking into the boot sector to find out where the operating system is. USB drives can become a boot device as can a CD/DVD, ZIP drives, flash media cards, and network interface card (using PXE).

A "live" CD/DVD/USB or other media means the device can boot up the computer. As long as the computer BIOS allows for booting from other media, this bootable media can load all sorts of operating systems including virtual machines and dual booting.

The ability to boot from several types of media can allow a suspect to boot a computer with their own operating system and store all their evidence on that same device. This form of boot-up would not leave any trace of activity on the suspect's computer and would make your job all that much more difficult.

### Deleted Data

A killer usually wants to get rid of the dead body and the weapon they used as quickly as possible after the crime. The killer wants to destroy any evidence that would link



them to the murder. A computer crime suspect will want to do the exact same thing. Digital evidence can be removed easier and quicker if the suspect knows what they are doing. (Don't take this as an invitation to commit "the perfect crime." We guarantee there is no such thing. Honest. We would know.)

To delete traces of old files, Linux uses the command **dd**

```
dd if=/dev/zero of=/home/filename
synch
rm /home/filename
sych
```

To delete files and remove traces of those files in Windows:

1. Using Explorer, select the files or folders and hit the "delete" key.
2. Clear all files in Temp directory or use software like CCleaner.
3. Once the files are deleted, select the Recycle Bin.
4. Right click on the Recycle Bin and select "Empty Recycle Bin."
5. Create a new Restore Point under "Systems" and delete the older Restore Points.
6. Reboot.

CCleaner lets you select which log files you want to delete or edit on any machine you have admin access to. A suspect can even clear their browser history, erasing their own tracks once their job is complete. CCleaner will try to make a system restore point before it alters anything. Your two choices are to not allow a restore point or look for a file in the root directory labeled "cc\_20110928\_203957" or something like that. A suspect will remove and delete that file before leaving, even if that file is on a portable drive.

## Formatting Media

Most media needs to be formatted before it can be used for a particular operating system. As a rule of thumb, formatting destroys all data that was previously on that media. If you come across a hard drive or other media that was recently formatted. It may contain evidence that the suspect wants to remove. With the software tools listed earlier, you have the capabilities to recover files and folder from that media.

There are programs out there that will format the media, write random information on the new formatted drive, reformat and continue this process as many times as you wish. Under these extreme conditions, recovering the original files and folders will be quite difficult. The key to recovering anything is to identify this event and media as quickly as possible.

## Precautions While Collecting Evidence from a Data Storage Device

These are the rules for when you're on the opposite side: when it comes to collecting media for forensic examination. You will not need a hammer or a drill. In this situation, you will need to be careful and non-destructive.

- Hold the media only by outer edges and avoid scratches or dropping it.
- Use water-based markers for writing on evidence.



- Place digital evidence devices in a waterproof and labeled bag.
- Take special precautions with storage media that are cracked or damaged.
- Do not rinse media with water to remove surface dirt, possible drug contamination, grease, an/or oils.
- Do not use any type of cleaner based on organic or petroleum solvents near the evidence.
- Create an image of the data on the media and work with the image to prevent damage to the original data.

### Exercise

8.10 While analyzing a suspect's 4 gig xd card you notice that the partition shows a 2.5 gig logical partition and nothing else. On the xd card you locate family pictures, routine documents and other mundane data. You do notice one encrypted file that is using 192 bit ASE block cipher inside a folder named "kids pix."

Why is the xd card only 2.5 gigs when it should be 4 gigs?

Does it concern you that there is an encrypted file located in a strange folder?

What do you know about AES and what does a 192 bit block cipher mean to you during your forensic investigation?

Can you crack this file?

### Steganography: A look at security controversy

The topic of steganography gives you a chance to look at how differently security experts can think. It is a totally workable means to secretly transfer data; it's just never been found in the wild. Is anybody using this stuff?





## Steganography: It's Real, It's Easy and It Works

When you're performing digital forensic investigations, it is not enough to simply recover photos, documents, videos, audio and VoIP packet data contained on the suspect media without also testing that evidence for potential hidden evidence such as steganography. While it may appear to be a benign picture, that picture may contain a plethora of hidden information.

Steganography, often referred to as **stego**, is the ability to hide information within transmissions without anyone being able to notice any change or modification to the original host without the use of special software tools. For example, a picture containing a hidden stego message looks identical to the casual viewer and gives no obvious indications that any modifications have been made to the original. While similar to encryption in that stego is used to hide objects and data, making it unnoticeable and unreadable, stego but should not be confused with cryptography. Steganography embeds the information in such things as documents or images while cryptography encrypts the information using a cypher or encryption key that is used to scramble and later unscramble the message.

In a recent case, stego was used, and detected by the FBI. Ten stego criminals were then released to Russia as part of a modern day spy swap. You can read more about that here: <http://www.reuters.com/article/2010/07/08/us-russia-usa-spy-idUSTRE66618Y20100708>

Steganography uses many different techniques from data insertion to algorithmic but to make the concept easier to understand let's just say that steganography inserts data into a host file in a manner that does not readily change the host file that can then be distributed to other persons who can then reconstruct the hidden message(s) contained in that host file. While graphics, bitmap images in particular are the most commonly used steganography hosts, the host can be audio files, videos, or documents as well.

There are more than 600 known stego creation and detection tools available on the Internet. But even with all the tools, a person who is trained to use a hex editor can readily detect steganography "infected" hosts if they have access to a library of clean original images, documents, videos and audio files with which to compare the suspect hosts against. Steganography detection is also aided with the usage of Steganography signature libraries similar to anti-virus definition detection as well as the comparison of steganography based hash values. Steganography hash values are available at sites such as <http://www.hashkeeper.org> or <http://www.stegoarchive.com>

A few examples of common steganography creation tools include **S-Toolsv4**, **JP Hide-and-Seek**, **JStegShell**, **ImageHide**, **ES Stego** and **Dounds Stegonagrophy**. Whereas **StegDetect** and **Stegbreak** are tools used to help detect steganography infected hosts. For more information about Steganography you can visit <http://Stegano.net>

### Exercises

Steganography Retrieval Techniques

8.11 Obtain a copy of Dound's Steganography.

[http://download.cnet.com/Dound-s-Steganography/3640-2092\\_4-8880146.html](http://download.cnet.com/Dound-s-Steganography/3640-2092_4-8880146.html)

8.12 Create and encode a message.



1. Locate a .bmp image, and save the image to the desktop.
  2. Launch Dound's Steganography. To have 32-bit color settings, refer to the "how to use" file that comes with the program. The settings must be in place for the program to work properly.
  3. Click the File tab, select Open, navigate to the saved .bmp image, and click Open. The image appears in the image field below the Message field.
  4. Type the text message of your choice to be hidden in the Message field.
  5. Click the Function tab, and select Encode Message, which will encode—hide—the data behind the photo. After the encoding is complete, the message Encoding Complete appears. Click Ok.
  6. Click the File tab, and select Save As. Give the file a unique name, and select the location to save the file.
  7. Close the program and then reopen it.
  8. Click the File tab, and select Open. Navigate to the file with the hidden data, and select Open. The .bmp image will appear in the image field.
  9. Click the Function tab, and select Decode Message. The hidden text will be decoded and displayed in the Message field.
- 8.13 Demonstrate how to hide data behind an image file.
1. Locate a .bmp image.
  2. Use Dound's Steganography to open the image.
  3. Enter data in the Dound's Steganography message box.
  4. Encode the .bmp image you located in Step 1 with hidden data.
  5. Save the file.
  6. Email the file to another student.
- 8.14 Open image Emailed to you from other student(s), and decode their hidden text.
1. Were you able to hide your text message using Dound's Steganography and encode the image?
  2. Was the other student able to open, decode, and view your hidden text?
  3. Were you able to discover or decode their text message?

Or, for a completely alternate point of view, read on.



## Steganography Is a Scam

A reviewer who is now paying us not to reveal his name had this to say (this is why we keep reminding you to think twice about what you post/email/text):

I want to state my protest of having Steganography as a portion of Lesson 8. After reading countless papers, articles and crap on the subject, I think there are so many easier ways to hide data. In 2009, the U.S. Department of Justice funded an eight month investigation into locating terrorist messages in pornography. The investigation was funded to the University of Texas. At the end of eight months, it was reported with great satisfaction that over 130,000 pornographic images were analyzed for terrorist messages but none actually had any hidden text. The entire investigation relied on 18 postgraduate students to pore over every Internet porn site they could locate. The investigators were all male.

What I understood from all this money was that we ended up with a bunch of horny college students and no useful data. In case you were wondering, the Department of Homeland Security and the U.S. Air Force replicated the same type of study (independent of each other and totally unaware that the same study had already been performed) looking for hidden messages in dirty pictures. Each study found nothing except one message was found in a small batch. Those pictures were deemed to be a hoax by someone trying to see if hiding messages in dirty picture was a worthy prospect. I swear!

Steganography is a bunch of horse pucky unless you guys know something that I don't. The topic is used to fill up spaces in otherwise empty security books. I don't want to fall prey to the same stupid material. I've even interviewed one of the world's leading scientists on the topic and he didn't convince me.

Personally we are delighted with this kind of controversy. First, of course, it simply gets people thinking. And then come the questions: Were these all-male testing crews just capitalizing on the opportunity to look at porn all day? For pay? The fact that three different organizations did these "studies" sounds like it might support this notion. But even further, was porn even the right kind of pictures to test?

### Exercise

- 8.15            What would be a better type of picture or media for sending stego messages? Where would be the ideal place to share it? Make it so.

## Windows Forensics

---

Windows can be its own worst enemy when it comes to maintaining data. The operating system is a resource hog, fills up a hard drive, and never seems to sit idle. How are you expected to examine an engine spark-plug in a car moving at a high rate of speed with no chance of the car slowing down? Oh, and Windows is constantly moving files around and modifying them, even the ones you are looking for.

We'll start this messy affair by covering the different types of volatile and non volatile information an investigator can collect from a Windows system. This section goes into more details about grabbing and analyzing data in memory, the registry, events, and files.



### Laptops Are Treasure Troves

Some forensic cases will involve a data breach. The latest historical data shows that laptop incidents caused the highest loss of corporate data than any other form. Hacking was far behind laptop theft with hacking breaches only accounting for 16% of all reported breaches. What does this mean to you?

Laptops are often issued to employees without much accountability or restrictions. This would be like handing out the keys to company cars and not caring who drove which car where. The results of such negligence is a whole bunch of company laptops being misplaced, forgotten, or taken without any oversight. Those same laptops are usually set up for remote access into the enterprise server. It could be your job to determine how a bad guy was able to access the organizations network. Missing laptops might be your answer.

Keep this in mind as well, when you lose YOUR laptop. What could people find out from your laptop? How happy would you be about it?

### Volatile Information

**Volatile information** is information that is lost when a system is powered down or otherwise loses power. Volatile information exists in physical memory or RAM, and consists of information about processes, network connections, open files, clipboard contents etc. This information describes the state of the system at a particular point in time.

When performing a live analysis of a computer, one of the first things investigators should collect is the content of RAM. By collecting the contents of RAM first, investigators minimize the impact of their data collection activity on the contents of RAM.

These are some of the specific types of volatile information that investigators should collect:

- system time
- logged-on user (s)
- open files
- network connections
- process information
- process-to-port mapping
- process memory
- network status
- clipboard contents
- service/driver information
- command history
- mapped drives, shares

### Tools for Collecting Volatile Information On Windows

To collect volatile information from a Windows system, you could use the following free software tools, which belong to the **Sysinternals** suite provided by Microsoft. You can



download it for free from Microsoft's website: <http://technet.microsoft.com/en-us/sysinternals/bb842062>. After downloading it, you should install it on the root (C:\) of your forensic workstation hard disk. You'll use these commands (unsurprisingly) in the command line interface, like this:

```
psloggedon
```

This Sysinternals program enables you to see who is logged on the system locally as well as those users who are logged on remotely.

```
time /t command
```

Use this command to see the system actual time. Windows shows file times in UTC which is also GMT (Universal time). The file time is shown down to the 100th nanosecond in a hexadecimal 8 bit format. Windows system time is shown in 32 bit, displaying month, day, year, weekday, hour, minute, second, and millisecond.

```
net session
```

This command shows not only the names of the users accessing the system via a remote logon session but also the IP address and the types of clients from which they are accessing the system.

```
openfiles
```

This command lists users logged in to a system remotely; investigators should also see what files they have open, if any. This command is used to list or disconnect all files and folders that are open on a system.

```
psfile
```

This program also belongs to the Sysinternals suite discussed above. It's a command line program that shows a list of files on a system that are open remotely. It allows a user to close open files either by name or by file identifier.

```
net file
```

This command display the names of all open shared files on a system and the number of files locks, and closes individual shared files and removes file locks.

The Microsoft tech web site listed above for Sysinternals explains each tool within the suite and ways the tools can be modified with switches. Overall, this package is a powerful set of utilities for forensic specialists and network technicians.

One last spot you might want to look for deleted files is in the thumbs preview database for windows. Look for a file listed as thumbs.db\_. This will show you all the thumbnail images of files viewed in explorer as thumbnails.

## Non-volatile Information

**Non-volatile information** is kept on secondary storage devices and persists after a system is powered down. It's not perishable and can be collected after the volatile information is collected. The following are some of the specific types of non-volatile information that investigators should collect:

- hidden files
- slack space
- swap files
- index.dat files
- meta data



- hidden ADS (Alternate Data Streams)
- Windows Search index
- unallocated clusters
- unused partitions
- registry settings
- connected devices
- event logs

### Ready? Roll Cameras, Action

Anytime an object (a file) is acted upon by another object (an intruder), there will be residual effects. The effects might not be easy to locate or detect, but those actions (of deleting or modifying) will cause some other results elsewhere. To reduce detectable actions, a professional hacker will use the tools that are already built into the system. They won't introduce new software, instead they will use the system tools in a manner that seems normal.

### Windows Server 2008 Event Log editing and location

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\EventLog\Application]
    %WinDir%\System32\Winevt\Logs
```

You can also use Windows Powershell to view all of the security logs in one command:

```
get-eventlog security
```

If you want to look at a specific security event, try

```
$events = get-eventlog security -newest 20
```

### Exercises

- 8.16 Even if you aren't using Windows Sever, locate the following logs in windows: Set up, Application, Forwarded Events, and Security. What "number" of events are in each log for your computer? What "type" are each event listed as?
- 8.17 While we are looking at event logs, let's create a "custom view" so you can see critical events from selected logs. Wow, look at that. Can you import a custom view? Which filters would you select for large events such as "Application" logs?
- 8.18 Grab a copy of Sysinternals from the web page listed. You will see that this program is a group of smaller programs, very powerful programs. Take a close look at the use of switches for some of the mini programs. Can any of those programs be used together (combined) to create a whole new program?

### Linux Forensics

---

Linux is often used in computer forensics because it:

- Treats every device as file



- Does not need a separate write blocker (forensics requires a hardware write blocker to keep the integrity of the data intact)
- Is highly flexible to work across many operating systems and file types
- Can be booted from removable media
- Is often bundled as a forensic tool kit with multiple tools

Linux, as with Unix, does not have **alternative datastreams** that are connected to files. The datastreams associated with Linux are not destroyed if you are using most file wiping utilities. To wipe a file securely, the file should not be recoverable since it should be deleted from the media. A proper wiping means that the file could only be recovered at extreme expense or not at all.

A file removed using the command

```
/bin/rm file_name
```

still remains on the media and can be recovered without much effort.

## Linux Slack

Linux file systems do contain slack space, just as Windows does. The slack space is much smaller, roughly 4K per block. This means that a suspect can hide about 4 KB of data in a small file block. The same techniques we discussed in Windows slack space can be applied to Linux slack space. This space is undetectable by filesystem and disk usage tools. When data is removed or deleted, the slack space will remain with the contents of any hidden data.

## Silly String

Text strings in Linux are fairly easy to search for and locate using the command

```
/dev/hdaX | grep 'text you want to look for'
```

Depending on the size of the media, this search can take quite a while because it will look for that text everywhere in that partition. You will not want to use a hex editor, since this will take even longer to perform. A hex editor can be useful to determine to contents of that media, though.

## Grep

Grep is an immensely powerful Linux tool. It is used to find certain lines within a file. This allows you to quickly find files that contain certain things within a directory or file system. It also allows for searching on regular expressions. There are search patterns that allow you to specify criteria that the search must match. For example: finding all strings in the dictionary that start with "s" and finish with "t" to help with doing a crossword.

```
grep ^s.*t$ /usr/share/dict/words
```

## More Command-line Tools

The "Live" forensic tools we discussed earlier are complete Linux forensic toolkits. Linux itself has a number of simple utilities for imaging and basic disk analysis, including the following:



Tool	Description
dd	The dd command can copy data from from any disk that Linux can mount and access.  This command can make a bit-stream disk-to-disk file, disk-to-image file, block-to-block copy/block-to-file copy.
sfdisk and fdisk	Displays the disk structure.
grep	Searches files for instances of an expression or pattern.
md5sum and shasum	Creates and stores an MD5 or SHA-1 hash of a file or list of files (including devices).
file	Reads file header information to tell its type, regardless of name or extension.
xxd	A command-line hex dump tool
ghex and khexedit	Gnome and KDE (X windows interface) hex editors

## Finding a Haystack in a Needle

Open Source forensic software includes powerful search tools that let you search for many combinations and permutations of factors for deep data searching. There is no need to buy expensive commercial tools, which is the wonderful part of using Open Source software. Linux provides you with plenty of scope to construct similar tools using standard utilities. The following text details the use of find, grep and strings, and then describes the use of the pipe to combine them.

## Encryption, Decryption and File Formats

Many of the files that you will come across will not be immediately readable. Most programs have their own proprietary file formats, while others use standard formats – for example the standard picture formats - gif, jpg, png, etc. Linux provides an excellent utility to help you to determine what a given file is. Remember the **file** command from above?

Command Line Switch	Effect
-k	Don't stop at the first match, keep going
-L	Follow symbolic links
-z	Attempt to look inside compressed files





These switches let you try to read a file. There are a number of file conversion utilities available to you under Linux, and even more available on the Internet, as well as a number of file viewers for various formats. Sometimes it may require more than one step to get to a place where you can really work with the data – try to think laterally!

Occasionally, you will come across files which have been encrypted or password protected. The complication that this presents varies, from encryption that is easily broken to stuff that would even give the best decryption professionals a headache. It pays to examine the area surrounding the computer that you are dealing with. People aren't very good at remembering passwords; they may well be written down somewhere nearby. Common choices for passwords also involve: pets, relatives, dates (marriage, date of birth), telephone numbers, car registrations, and other simple combinations (123456, abcdef, qwerty etc.). People are also reluctant to use more than one or two passwords for everything, so if you can reverse engineer a password on one file or application, try it on the others. It is highly likely to be the same. Take a look at Lesson 11 Passwords for more information on cracking passwords.

## Exercises

- 8.19 Boot up with Linux and create a file named "evil plans" on a USB drive or any other portable rewritable media.
- 8.20 Delete that file using whatever technique you wish.
- 8.21 Hand that media to your lab partner and tell them you lost a file. Ask them to recover the lost file but don't tell your partner the name of the file you "lost."
- 8.22 Try this recovery process with other types of media and operating systems, changing out with your lab partner.
- 8.23 How many times does it take to format a drive or removable media to ensure that all previous data or a single file is erased?
- 8.24 If a partition is removed and reallocated, is the previous data lost forever or can it be recovered? What tools would you use to attempt such a task?
- 8.25 Hide a secret file in the slack space of another file. Delete the main file. Can the hidden data be recovered and how would you do it if it is possible at all?
- 8.26 If the encryption method is too strong to be broken, it may be necessary to perform a dictionary attack (also called a brute force attack). Find out what a dictionary attack is.
- 8.27 Find out what Truecrypt is and how it works. Learn about hidden containers. Do you think you would be able to access such an archive? How? (One answer: <http://xkcd.com/538/>)



## Feed Your Head: Real Case Studies

Here are some examples to show digital forensics at work.

Who	What
Morgan Stanley	In a Florida court case, Morgan Stanley (MS) repeatedly failed to turn over data related to a fraud suit against them. 1423 backup tapes were concealed by MS that contained emails detailing the fraud. A fired MS technician revealed to the court that those tapes existed and many other tapes were deliberately mislabeled. Forensic examination confirmed this intentional fraud. The judge fined MS \$1.6 billion dollars for acting with "malice or evil" intent.
David Kernell	In September 2008, the defendant hacked into Sarah Palin's Yahoo email account. Before the FBI arrived to investigate this crime, Kernell uninstalled his web browser and fragmented his hard drive. The government was able to provide sufficient forensic and testimonial evidence of his crime to convict him on a number of counts.
TJX A.K.A. Albert Gonzalez	One of the longest convictions ever handed down for a computer crime was given to TJX. Gonzalez was convicted of stealing 90 million credit card and debit card numbers. The defendant ran a gang of cyber-thieves over several years and bought a yacht for himself using the stolen money. Teams of digital forensic examiners were called in to crack the case and provide evidence. TJX was issued a 20 year prison sentence and ordered to pay \$25,000.

## Mobile Forensics

Using mobile communications as a tool in your planning and/or execution of the hack can provide you with a completely new set of options. Cell phones use several forms of signaling, one is the radio that links your phone to the closest antenna receiver, the next is the Bluetooth link that works for short-range connections, the GPS signal locator can be used for other functions, and lastly the phone has digital connection capabilities. We want to focus on the digital portion of a cellphone.

Inside a cell phone is a Subscriber Identification Module (SIM) card that identifies your phone to you and your service provider. This SIM card is also the same card that stores some of your phone numbers and other text data. This card has an onboard microprocessor.

SIM cards contain a special set of numbers known as International Mobile Subscriber Identity (IMSI). The IMSI is the phone number for that device and can be thought of as a Machine Access Code (MAC) address for a cellular phone. The first set of numbers of a MDN are assigned to the manufacture. SIM card editors like the ones available at Dekart [http://www.dekart.com/products/card\\_management/sim\\_manager/](http://www.dekart.com/products/card_management/sim_manager/) will assist you in viewing this number set.



If you were going to conduct special business on a cell phone that might be traced, it is quite possible to have several SIM cards on hand. Changing out the cards after each call makes it nearly impossible to trace a cellular call. International SIM cards with preloaded calling credits are available in Europe, Korea, Japan, and other countries that do not have a cellular monopoly as in the United States.

One point to consider is that cellular devices are tracked from cellular tower to tower, even if the device is just on. This is part of the normal communication hand-off to ensure the cellular caller can make a connection quickly, at any time. In the near future, towers will track and maintain logs of each cellular device that pass through their zones while communicating. This may sound contradictory to the paragraph above, however, the SIM card contains the hand-set identifier. Changing out SIM chips is almost like changing out cellular devices.

Short Message Service (SMS) are stored by each cellular phone carrier for several days or none at all. This shows how quickly evidence can disappear and timely response is critical. The messages are saved on the user's phone, usually on the SIM card or on the external memory card.

So how do you feel about the track-ability of YOUR phone?

### Connect the Blue Wire to the Red Square

Another aspect of cellular digital communications is the ability to use Voice Over Internet Protocol (VOIP). This communication tool uses VOIP software to create data voice communications between you and another VOIP user, bypassing cell phone usage charges. Wonderful, right? How could this possibly help you?

Well, since VOIP is digital and a piece of software, we can encrypt the packets if you are using the Android OS. The Advanced Encryption Standard (AED) is a block cipher and can provide many levels of security. You will want to use the lowest encryption level, since VOIP is already going to be slow over a data phone.

### Some Disassembly Required

Before you attempt to recover any data from a cellphone, turn off the cellular signal, as in put the phone in "Airplane" mode. Cellular providers can disable or delete all data from a device if that device is reported as lost or stolen. Don't be that one person who forgets to disable the signal to the mother ship.

Older devices used proprietary cabled for charging and transferring data. These cables changed from device to device and never seemed to be interchangeable to anything. These days, most devices connect using a mini USB cable on one end and a standard USB on the other. Apple products are the exception to this standard for "security" reasons.

As secure as this sounds, the Apple to computer interface cable ends as a USB connection at one end. If that doesn't seem to work well enough, you can purchase the Ipad Camera adapter kit. The kit includes a straight USB link to the pad plus another adapter that connects a SD card directly to the Pad. So this gives you the option of plugging a SD card or a USB drive directly into the Pad. Cool, huh?

Cellular devices can store data in any one of three local areas. These areas are: The phone's built-in memory, the SIM card, and the external memory card. The good stuff (real evidence) is often located on the phone's internal memory and on the SIM card. SMS enabled devices often include software for "predictive text." Predictive text files can include portions or entire text messages that may not be located elsewhere.



## So Many Devices, So Little Time

Way back when, phones just called out and received voice transmissions. These things were connected to a wall, a deck, or payphone booth. These days phones are no longer just phones, they are portable networked computers. Cellular communications comes in all different sizes and models. An iPad isn't a phone but it can communicate in many of the same ways a cellphone can. A tablet, Android OS, Pocket PCs, all have many of the same features as a phone but cannot be called a phone at all. "Dude, let me borrow your tablet so I can call my friend," is something you're not going to hear just yet, but you will. Then you'll have to worry they're going to find your love letters, or your porn, or put love letters or porn on your phone.

Products that run on the Android OS are fairly straightforward to exam due to Google's open operating system. Android is based on the Linux kernel, which was covered earlier. Google provides its Android source code and a developer's tool kit free of charge. Other device OSs include Black Berry, Windows, Windows CE, Nokia, Symbian, and Linux.

Each OS will need to store files in some order and there are not too many different ways to name "SMS" or "Video" files. A little snooping around on each different device using the software listed below should give you the evidence you are looking for (which is, by the way, why you shouldn't fee invulnerable, or even "protected," on your device).

Besides the fact that cellular devices have Bluetooth, data transmission, and WiFi communication capabilities, many are GPS enabled as well. All of these signals store information on the phone, the SIM card or on the external memory card. Forensic software allows an examination of each type of history, including the GPS. If the suspect enabled their GPS, all the waypoints and location history can be recovered to provide even more evidence. <http://www.gpsvisualizer.com/> allows you to upload GPS data and will create maps to show you where that where that data leads.

Don't forget about the suspect's vehicle GPS as well as the on-board computer. Any vehicle built over the past decade (since 1985 in the U.S.) has a diagnostic computer that tracks speed, fuel consumption, ignition sequence, plus more information that may help you solve the case. Expect shoes to start keeping track of your location. You might even be able to make phone calls with them too.

**iDevices:** There are some folks have dedicated time and effort into open source projects such as IPBackup Analyzer. The purpose of this program is to look at data that is backed up on an iPhone and make it readable. You can find this open source software at <http://ipbackupanalyzer.com/>. One of the unique issues with Apple mobile products is the requirement for a back-up passcode. The passcode can be bypassed using software tools, which will allow for examination of text messages, phone contacts, pictures, video, emails and all evidence you might need to examine.

### iPhone Forensics Example

Check out this article on an iPhone forensic investigation:  
<http://www.nxtbook.com/nxtbooks/evidencetechnology/20120910/#/30>

Warning -This article deals with a sensitive topic that you may find offensive.



## Phone Software Tools

Most of the major phone forensic software builders have found a niche market that allows them to charge a premium for their tools. There are a few open source and free products out there you might want to look into. Like anything else, each tool has pro's and con's but, you will need to have a working knowledge of several tools to be successful.

**Oxygen:** This software and hardware manufacture offers several types of cellular forensic products. This software is free for limited time use, roughly six months. If you can't get the data you want out of a device in six months, you might want to try the "Hammer" method. You can download the free version of Oxygen Forensic Suite 2012 at <http://www.oxygen-forensic.com/en/freeware/>. This software is capable of reading iPhone back ups, even if the data is protected by iTunes passwords. Nice.

**Bit Pim:** An open source project, Bit Pim has been used for a number of years. This free software has one tiny drawback, the lack of support for newer smart phones. To be honest, Bit Pim doesn't work on quite a few newer smart phones. Luckily for you, the authors will try and create a package for you if you ask them nicely. Actually, you have to follow their rules listed in the Web site under "FYI", if you want any response from them. Failure to adhere to their request process will result in nothing. You will get nothing from them, period. Read their documentation at <http://www.bitpim.org/>.

**Sleuth Kit:** We talked at Sleuth Kit earlier in this lesson. Another open source program with tons of features, including cellular forensics, Sleuth Kit gives you the same capabilities as many commercial products. You can find more than enough information about the product including an entire Wiki at [www.sleuthkit.org](http://www.sleuthkit.org).

<https://viaforensics.com/products/tools/> offers several free links to Android OS forensic tools. This site offers a book on Android Forensics plus several scripts for gathering your own data. These folks are also the one who have a section dedicated to iPhone forensics at <https://viaforensics.com/iphone-forensics/howto-iphone-forensics-free-andor-open-source-tools-91411.html>. Remember, an apple a day, keeps the iTunes back-up away.

## Now What?

If a digital device is evidence in a case, do not turn off the device if at all possible. Find a battery charger, get a charger, or build a charger if you have to but keep that device running if it is already turned on. This is critical if the phone is a pay-as-you-go since there isn't a signed contract with a mobile carrier. These phones are difficult to trace because they are disposable.

Of course you can't examine nor copy the SIM data without removing the battery. This is another reason to keep the cellular device powered by another way without relying on the battery. With our luck, the device battery will always be just about dead anyways. Bad guys never remember to charge of their devices.

The forensic examination should be done using direct cables from the device to your awesome lab computer. This means that all other communication means need to be shut off. Bluetooth, WiFi, GPS, and whatever else has to be turned off before an examination can begin. Failure to do so could render the evidence useless in a court of law.



## Exercises

8.28 Grab a Mini USB cable and connect a cellular device to your computer. The device should ask you one question with three possible answers. What are those three answers? Which one should you choose if you want to evaluate the data on your device?

Once you have a link between your computer and the target device, look to see what information you can obtain on your own. How far could you get without any special software? Could you read any data on the device itself or just what is on the external memory card?

Disconnect the link and download any cellular forensic software that you like onto your computer. Install the software. Turn your target device off, then on again. Now reestablish that cable connection you had from the first question. Okay, now you can run the new forensic software you downloaded. Can you access all of the SIM data or just parts of it?

Do not touch your PUK or Pin!! Most phones will lock-up if the PUK or Pin is guessed too many times. What is the PUK or Pin and why it is critical to you as an examiner?

8.29 Steal someone's cell, ha, ha, just kidding. Borrow someone's cellphone and connect it to your high-speed Cray Supercomputer. Can you access their SMSs, photos, contacts, caller log? What is the serial number of that phone; the one that is hard coded into the SIM, not the one on the inside of the case?

## Network Forensics

---

Network forensics are used to find out where a computer is located and to prove whether a particular file was sent from a particular computer over a network. While network forensics can be very complicated, we will cover some of the basics that can be applied to everyday life, and how you can find things out – or be found out.

### Firewall Logs

Who's connecting to you? The firewall is a utility that can control connections between two points in a network. There are many types of firewalls. Regardless of the firewall type and job of the firewall, it is the firewall logs, which give you the details. By using the logs you can find patterns of attacks and abuse to your firewall.

As with any log file, the integrity of those files are essential. Think of log files as a **smoking gun**. Each file is stamped with time/date and certain property rights. Firewall logs are considered "smart" logs because they are generated from a device that has perimeters and is not a simple hub or switch box. Each packet is not recorded but each request and connection is recorded. You are looking for connections between specific IP addresses or the transmission of files between two connections.

### Packet Sniffers

**Packets** of data flow through the veins of every networked device. Since there are literally millions of packets moving between servers and other devices, looking at individual packets had always been thought to be impossible. With the increased power of computers and better software technology, we now have the capabilities to search through millions of transmitted packets to locate those that meet our requirements. We call this technique "packet sniffing."



Imagine that you are on a bus loaded with people. Everyone is talking but you want to hear just one conversation from two seats away. Your brain has the ability to tune out other noise and focus on that one conversation. Packet sniffing does the same thing; it filters out all the other noise and concentrates on the packets you are interested in.

Packet sniffers come in all kinds of shapes and sizes but every type must be placed between the data flow transmissions. You can't hear a conversation if you are not with the people who are talking. Packet sniffers can be active (looking) or passive (listening) yet, either type will gather packets that match your requests. The trick for an intruder is how to gather, store, and transmit those packets in your network without getting caught.

## Intrusion Detection Systems (IDS)

This tempting name is a generic term for anything that can detect, alert, or shut down abnormal network activities. Snort is a perfect example of a program that can look for abnormal behavior in network traffic. An example of odd behavior would be if you were on vacation and your email account became active. Your account started to send and receive all types of attachments and redirected emails, as though someone were using your email account. An IDS would pick up on this weird behavior and either act on its own to shut off the account or notify someone that some strange stuff is going on while you are away.

IDS were designed to be the watchdog of network traffic. Each type of IDS looks for protocols, signatures, ports and other locations where odd behavior might happen. Some systems deny all and only allow authenticated users through, while other IDS's bait and wait. The IDS's logs are full of wonderful details on odd behavior.

## Router and Network Management Logs

As mentioned in the firewall logs, routers and network management logs are very detailed in their collection of typical activities. Occasionally something will pop up in the logs that will trigger a forensic expert's interest on a case. Besides being evidence to prove when certain events occurred, log files are difficult to tamper with. The software tools we have mentioned before have programs to automate log filtering. Using automation tools will save you both time and sanity in the long run.

## Tools of the Network Trade

There are a variety of open source software tools that should be part of any network evidence collectors kit, starting with the tried and true **Wireshark**. Since network traffic is data packets, or chunks of information, Wireshark captures and analyzes packets. Instead of making you going line by line through each packet to identify headers, routing information, sender, and the contents of each packet, Wireshark does all the heavy lifting for you. Plus, Wireshark is a cross platform utility.

**Netcat** at <http://netcat.sourceforge.net/> is another powerful open source program that analyzes all network traffic including TCP and UDP, inbound and outbound, Ethernet and IP, including any service or port you'd like to look at. Like Wireshark, Netcat is a cross platform application. Both are actively updated by a team of volunteers.

Netcat has a **hexdump** utility built into the software and can capture/analyze packets.



## E-Mail Headers

E-mails come with information of every computer they pass through to get to you. This is added to the **header** portion of the email information. Sometimes the most important information is in the headers. To view the headers, however, is not always so simple. Various mail clients will all have different ways to view this. The real trick to reading headers, though, is to know they are read backwards. The top of the list is the receiver. Each route the email travels goes with each line until the very last line is the computer or network that the mail was sent from.

This is only true if the email sender used their real email address to send it. Emails can be spoofed, IP addressed can be faked, and all sorts of other tricks might be used to disguise the real sender. The header can provide some clues but don't expect to solve any cases based just on email header information.

Within the email header, there is segment called "Message-ID." This set of characters is provided by the first email server when the message was sent. Since each ID is unique, proper logging can help you identify the location of the original sender. Look for the link listed right after the series of numbers and letter in the ID.

The sender "From" information in the header is configured by the email client and should not be considered reliable. Time stamps can also be misleading because email clients can be configured to send emails hours or days after the email was written. This is a technique known as "Delayed Send."

## Exercises

- 8.30 Grab any Spam email you have in your email box. Using the information provided, dissect the email header in an attempt to locate the source of that spam. How did that spammer get your email address?
- 8.31 Determine how to look at your e-mail headers in the e-mails you receive. Are there any particular fields in those headers that seem foreign to you? You probably have several email accounts. Forge yourself an email, try your best to hide your actual location.
- 8.32 Send that spoofed/forged email to your lab partner telling them that they need to pick up something, like donuts, for the next class. Make sure the spoofed sender is the instructor, otherwise you might not get your donuts.
- 8.33 If you have a social media email address, send yourself an email from that social media site to your regular email account. Take a look at the social media email header to see if you can tell how that email was routed.
- 8.34 Now, try that same exercise again but send the social media email as an anonymous to you regular account. Check the anonymous email header to see if you can tell how well your identification is hidden by the social media service.

### Game On: Getting Down and Dirty

"Please tell me what you are doing in the school dumpster," Mooka asked, scratching his windblown hair. Two legs dangled against the lip of the large green trash bin while the other half of Jace grappled with bags of garbage in the container.

"Just hold the lid open for me," yelled the trash intruder. The side of the metal bin





released a loud “clunk” as something large struck the inside. “I got it! Now pull me up,” Jace yelled in Moko’s general direction. The echo of her voice in the trash bin sounded as if she were talking through a soup can with a rubber balloon stuffed inside. The thought of soup made her nauseas as she clung onto the prize junk she dug out of the dumpster.

Moko grabbed the two legs, trying hard to avoid being kicked in the face and pivoted the smelly hacker up and over the trash dumpster’s opening. To both of their surprise, Jace popped up out of the trash bin holding a used mobile tablet in her arms and she didn’t even drop it, yet. With her shoes planted softly back on the parking lot asphalt, she held the battered slim case as a trophy for her dirty work.

“Check it out, this is the old teacher’s lounge time card device,” Jace beamed with glory. As any lady would do, she brushed back her tattered hair so she would look her best in this moment of triumph. In that brief second, the slippery case slid out of the single hand and landed nicely on top of her foot.

Moko had never heard Jace cuss that much before and he certainly had never seen her howl in pain that loud. He did his best to avoid enjoying his best friend in pain, mainly because he was usually the one who suffered the injuries when they were together.

Moko put his hands on his hips and lectured to the injured girl, “I haven’t heard that much cussing since the last time I went to church!”

Jace forgot all about her wounded toes as she looked with strange eyes at her friend, “What are you talking about? You’re weird dude.”

He dropped his arms down and explained, “At church they’re always talking about hell and us being “damned” and stuff like that.” Moko was trying to lighten up the mood, to cheer up Jace even though his humor was awful.

“Moko, that joke was more painful than my foot right now,” jabbed Jace. “But check out the other cool gadget I found in the trash,” she said as she pulled out a smashed cellphone from her hip pocket.

With a tone of pure boredom Moko responded with, “Wow, a wrecked cellphone to go along with a destroyed tablet. What are you going to do with all this wonderful treasure?”

Jace twisted her head slightly as her mouth formed a grin reserved for evil geniuses. “Just wait,” she said.

Back at the lab, well, more like back at Jace’s small bedroom in the apartment she shared with her grandmother, Jace had the back covers pulled off the devices. Even after she had wiped down the devices the stench was still lingering in the poorly ventilated room.

“So, what are we looking at,” Moko asked as he peered over Jace’s slim shoulder at the pair of broken gadgets. He backed off a step as soon as he realized that she smelled just as bad as the trash dumpster.

“First of all, I need to find out what operating system these things use,” she replied without looking at Moko.

“But you can figure that out just by looking at the case. That one runs IMO because it was built by Anvil and has the Anvil stamp right on it and that smaller one runs Robot. It says right on the back of the thing.”



“Dude, just because it was built and packaged by a brand doesn’t mean it runs that operating system. You can root the devices and replace whatever OS you want, plus you can add modifications to the internal chips using EPROMs to dual boot. And don’t get me started when it comes to fake phones made in Hinad. You never know what is on those things.”

Mokoa took another step back and said, “Okay, I’ll step back and shut up now.”

“No you won’t. You can’t be quiet any more than I can. I’ll walk you through the steps I take to gather data. Grab a seat,” Jace said knowing that the only chair around was in the kitchen.

“Oh, and while you’re getting a chair grab me some cookies and a tall glass of ice water.”

Mokoa knew the drill since Jace was an expert at getting him to do chores for her.

It took him three trips to the kitchen and back to get the cookies, water, chair, and a snack for himself. He finally sat down behind Jace as she began her tutorial.

“80% of these mobile devices run different flavors of two OS’s, Robot and Anvil. Robot has a million versions of it, with each version slightly different based on who manufactured it. IMO was written by one single company so there isn’t much variation in those mobile devices. Robot is much easier to image then IMO, since IMO is a closed OS and is very tight on security access. If this device is running IMO but it was jailbroken, I’ll have an easier time obtaining the encrypted pin.”

Mokoa understand most of what Jace was saying but knew not to disturb her while she worked. He could ask questions when she stopped to drink water or bite into a cookie. Otherwise, he kept quiet and watched her work.

Jace continued, “Each OS stores data differently. Luckily, Robot was written based on the Linux kernel and Software Developer Kits (SDK) are easy to download from the creators of Robot. This is fairly open-source software. IMO isn’t. If the mobile phone or tablet has IMO and the user employed a PIN to lock the machine, our work is much tougher. It’s not impossible to recover the data; it just means more work for me.”

A laptop was pulled out from inside Jace’s knapsack; again, Mokoa was given the task of fetching the computer for Jace. She flipped open the portable computer and had the programs running while she hunted for USB cables in her desk drawer. While she rummaged for cables, Mokoa reached over and pressed buttons on each of the trashed machines. None of the buttons seemed to work.

“Hey genius,” Jace shot out, “I already tried that. I wouldn’t be digging for connection cables if the things powered up.”

Mokoa felt a bit stupid as usual. Jace never missed small details like trying to power up a machine first.

“Yes! I found the two I need. I hope they work,” Jace said as she untangled a mess of wires.

“Um, Jace, had you thought about taking a shower before we keep going? You really smell bad, like Mr. Tri bad,” Mokoa couldn’t bear the stench any longer.

“BAD! You think I as smell as bad as Mr. Tri! I’ll show you bad,” She steamed and back handed Mokoa faster than he ever expected since he was sitting behind her.

“What was that for? You stink and I can’t stand sitting next to you. I’m leaving until



you shower and apologize for that smack," Mokoia said as he was walking out of Jace's room. The apartment door slammed shut.

Upset by the whole situation, she smelled her hair and really felt bad for what she did to her friend. Jace cursed to herself.

### Game Over

## Let the Fun Begin

---

A critical part of a hack is thinking through the entire process before touching the keyboard.

- How are you going to get inside your target?
- What controls do you need to disable or monitor during your network visit?
- What do you want and where is the location of your target?
- How are you going to transfer the data you want and where are you going to store it?
- What logs and audits need to be restarted or edited as you exit to cover your tracks?
- Where are you planning to keep the new data for your safety and use?

Social engineering is an excellent tool for gaining access to physical locations and networks. Recognizing it is a great way to be immune to (some or most of) it.

## Reconnaissance

**Recon** is learning the networks vulnerabilities, the types of servers you will be dealing with. What security measures are being used and what are their vulnerabilities. Can you turn those security devices to your advantage? Where are the network logs and audit logs kept? Are you going to install a backdoor for return work? What attack vectors are you comfortable using and will work across each network?

## Software and hardware vulnerabilities

You can locate all known exploits and vulnerabilities on all types of products by going to <http://www.cvedetails.com/> or [www.cve.mitre.org/](http://www.cve.mitre.org/). Both of these web sites should be part of your attack methodology as soon as you learn anything about the networks you will be dealing with.

## OpenVAS

OpenVAS at <http://www.openvas.org/> is an open source vulnerability scanner and manager. The organizations own **Network Vulnerability Test (NVT)** database is used to update the scanner on a daily basis. This "One-stop-shopping" for vulnerabilities can be compared with CVE, without all the extra technical jargon. The software is a collection of tools that you can shape to fit your needs, even if you just want to know which vulnerabilities apply to an Apache web server.

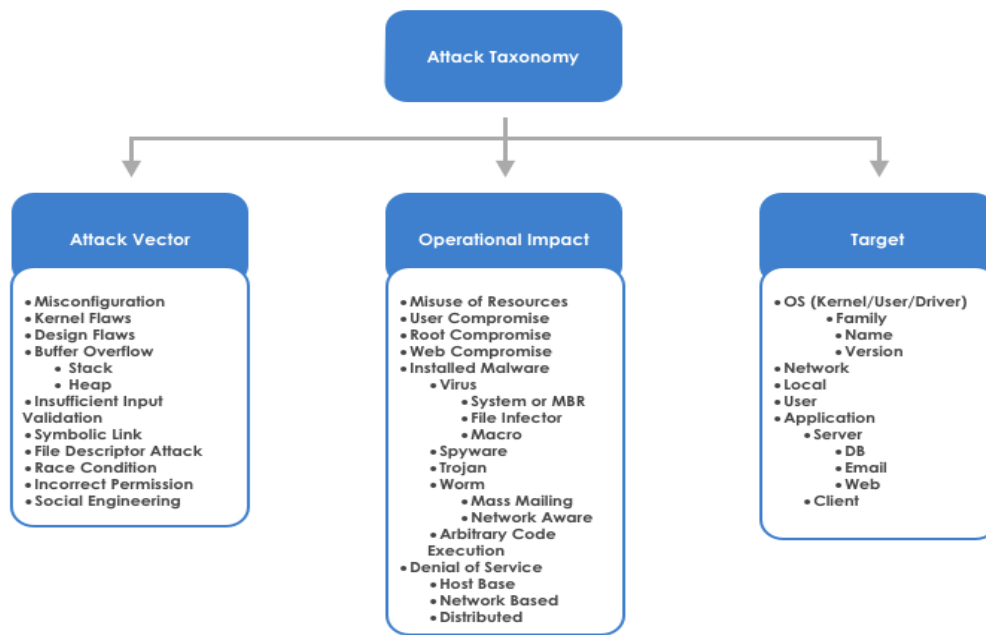


Figure 8.3: Attack Taxonomy

**Attack Vectors:** These are methods to enter networks, using a variety of tools or known vulnerabilities. You will often see this term used alongside “malware,” since attack vectors are mainly viewed as malicious network entry points by the security professionals. In our use of the term, we are merely showing you the types of ways to enter networks, in a specific category. Repeat after me, “I will not use Attack Vectors to plant malware.” Hold your hand in an official looking manner when you say that, too, so it sounds like a pledge or something. Because this stuff can land you in jail, and we don’t want to be the people who catch you. We’ll leave that for your friends.

## Weapons to Hack Networks

**Blackhole** is a package of “mouse click” exploits aimed at giving any hacker with any degree of skill, several ways to gain administrator access on a network. Unlike most other exploit software; Blackhole 1.0 earned a name for itself in the security industry because the program introduced several zero-day vulnerabilities. The creators of version 2.0 are promising “dynamic URL’s” provided by another AV company, which would effectively build custom exploits just for you.

<http://malware.dontneedcoffee.com/2012/09/blackhole2.0.html> Cost is \$50 for one day of use. Such a deal!

**THC-Hydra** 7.3 was updated in May 2012 and Hydra works to crack network logon passwords. The program has an excellent use for switches in the command prompt for Linux. This program can be run through a proxy (to hide your location), through FTP, IRC, HTTP, and several other protocols.

<http://www.thc.org/thc-hydra/>

**Metasploit** has been around as a penetration testing software in the open source community. There are now several commercial flavors of Metasploit, with the one you want being free. Like many community based tools, Metasploit has a large library of add-ons, plug-in, and configurations. Many security professionals have this software as



part of their “have to have” toolbox. You will want it because there are additional exploits added all the time to the Metasploit engine.

<http://www.metasploit.com/>

The **Fedora Security Spin** is another community project aimed at education and the safe testing of security tools. Fedora Linux is really customizable: you can create your own **spin** with exactly the apps you want, and nothing you don't want. The Security Spin is all about security testing and auditing, which sounds kind of formal but actually is really fun. Like many security tools, the package is distributed in an ISO format that can be put on a USB thumb drive or CD. And like Fedora itself, the vast majority of professionally built security software is open source.

<http://spins.fedoraproject.org/security/#home>

**Cain and Able** is the ultimate in Script Kiddie software. Cain was originally designed as a standalone program to recover passwords from SAM dumps (Windows password files). The program is still excellent at performing that same task, just finding anyone running an old version of Windows is difficult. Able was added to increase the usefulness of the tool and build a penetration-testing package. Cain and Able together offer really easy access to really insecure networks.

<http://www.oxid.it/cain.html>

**Fyodor** boasts 125 Top Security Network Security Tools. This list has been available for a number of years and is updated (sort of). Not only does the web site provide you a brief description of each tool but it also provides you a like to the software. Some of the tools are quite old but useful when working with FORTRAN or an abacus. Check the site regularly, at least every few years, for new tools that you have already heard of.

<http://sectools.org/>

## Exercises

8.35 Head over to Metasploit.com and download the most recent copy. Burn the ISO onto either a bootable USB device or a “live” DVD.”

The kind folks at Metasploit provide a vulnerable server that allows you to work with the tools in Metasploit without getting into legal trouble. The server is called “Metasploitable.” Create an account, get Metasploitable and try out your new pentesting software.

<http://updates.metasploit.com/data/Metasploitable.zip.torrent>

## Counter Forensics

---

Counter forensic software tries to perform one or both functions of deleting all log files and/or erasing all data that could have been altered during a network visit. Both methods could ring very loud alarms if not used correctly, bringing in the eighty-five agents that have not had their morning coffee yet. Counter forensic tools are mainly used on a single computer to remove, hide, cover-up, and generally make a forensic examiners job difficult or impossible.

There are a few issues that need to be considered if you plan on using counter forensic software. The first issue is the examiners determination that counter forensic software was actually used on your machine. This alone would raise suspicion as to why anyone would use this software if they didn't have anything to hide.

Second, locating and deleting every bit of data remnants from swap files, temp directories, pagefiles, and every speck of data that could link you to the hack.



Third, forensic examiners are paid either by salary or by the hour. They are primarily paid to produce enough evidence to convict someone. If you have created an evidence recovery challenge that would take too long for any examiner to make a case against you, they will likely stop looking at some point. Time is money. Take your time to add more work for an examiner ahead of time, this time and every time, unless you want to spend time doing hard time for a long time.

### Just Who Has the Advantage

There are several opportunities for a criminal to use counter-forensic methods on a device. These include:

- Many forensic examiners do not know how to deal with advanced users who can manipulate the operating system or hide data. The push for new digital forensic personnel has created a "shake n bake" process where the person attends a few classes and is handed a piece of software to use. This leaves so much experience needed to the will of the software manufactures.
- Forensic software doesn't have a set standard for the scientific process of collecting, analyzing, and reporting a repeatable method. Different software will show different results. Thus, results cannot be replicated. This is bad, very bad.
- There is no common body of knowledge amongst digital forensic experts. This means there are several ways to slice an apple, none are right or wrong. There is no established method to conduct a digital forensic examination or even publish the results.
- Digital forensic examiners and software/hardware hasn't been designed for field environments. This stuff was created to be used in a nice clean lab, with perfect conditions, and all the tools you would ever need. In the real world, this is never the case.
- A simple alteration to the evidence, such as a delayed file update or system time change would render the entire forensic collection useless. The data would not be accepted by a court of law because of a simple change.

### You Gotta Be Social

Facebook, Twitter, Google, Tumblr and all those social media sites are part of cloud storage. Each of these services offer easy access to users to communicate with friends, meet others, share ideas, post pictures, post their calendars, and socialize in a digital environment. Many of these cloud providers seem to give away their web products without any regard for their own profit. It all appears to be "free."

The concept is simple: provide an online place where people can interact and give them ways to express themselves in an environment that the user thinks is private. As more users join that cloud playground, collect information on each user to create precise marketing material for that user. The cloud service can then sell that targeted marketing information to advertisers or product manufactures directly. You could say that it is a "win-win" because users get a nice place to socialize and cloud services can earn enough to stay in business.

Of course, that isn't all the ways these social media providers earn money. Facebook recently announced it had 1 billion users. With that many people across the globe accessing Facebook, that site has become the world's largest personal photo and identification database ever created. Everything that is posted on any of these social sites becomes property of that service. All that personal information is worth a tremendous amount of money.



Think of social media this way; if you are not being sold anything, than you are being sold to someone else. You are the product.

## Head in the Clouds

Current forensic laws, tools, and techniques do not work in a cloud setting. Because of the design in cloud computing, any forensic analysis will involve shared resources. What this means is when a forensic examiner attempts to retrieve suspected evidence, they are going to also grab data that belongs to other people, as well. This doesn't mean that an attack against one cloud service is going to go unreported or not investigated. The cloud provider will conduct their own investigation and look at legal issues. The crimes that involve one account, one suspect, one victim, or one event are going to be tricky because the cloud service may not be willing to help you out. As always, it depends on which side of the conflict you're on.

## Issues with Cloud Forensics

1. No jurisdiction over data. Most cloud providers have redundant data centers located in several places throughout the world.
2. Massive increase of cellular devices accessing/loading/creating/altering and moving data in the cloud. This means that data could be in several places at the same time.
3. No central role for management to help filter out suspects. You do not own the data storage, you are just renting it.
4. No access control to keep data segregated for a forensic investigation. Customers can get to that data at any moment, at any time.
5. Lack of physical infrastructure to create a time-line or determine timestamps or log events.
6. Terms and conditions between organization and Cloud provider may not allow a forensic investigation that will meet your requirements.
7. Retrieval of evidence without modifying it is extremely difficult.
8. Each cloud service handles their data storage and service conditions differently.

If a crime was committed against the cloud provider, the provider has the jurisdiction over that criminal activity. The cloud customer may have limited or no access to cloud data, more so, if the social media site owns that data. This is the case with services such as Facebook, where the content is user driven but owned by the cloud service. (Are you surprised to find that your user content is owned by the social media site, not you?)

## Exercises

- 8.36 There are a few techniques that work if you are trying to identify the sender of data unless that sender uses a proxy. Chrome and certain add-ons to FireFox allow you to view the content of HTTP source. How can you use the information this reveals to block the sender? Do so on your browser.



## Conclusion

---

Digital forensics is not an easy task, nor it is an easy profession. You must be detail oriented, able to document everything you do to the evidence you find, think like a criminal and have an enormous amount of patience to locate all the evidence. Besides that, you need to be willing and able to be an expert witness, if called to testify in a case.

On the other hand, some education and experience with forensics techniques and tools can help you maintain the privacy and confidentiality you've been losing fast in our digital world.

If you were brave enough to complete this lesson, you know we discussed where media comes into play as source to hide data, boot up a computer and hide evidence within data or within the operating system. You were introduced to some very tricky places that data can be hidden and how to thwart forensic experts.

Digital forensics is filled with areas that require expert knowledge or at least a fairly good understanding of that area. This lesson was designed to provide you a taste of what you can expect if you want to work in this amazing field – or just be an informed computer user.



Today's teens are in a world with major communication and productivity channels open to them and they don't have the knowledge to defend themselves against the fraud, identity theft, privacy leaks and other attacks made against them just for using the Internet. This is the reason for Hacker Highschool.

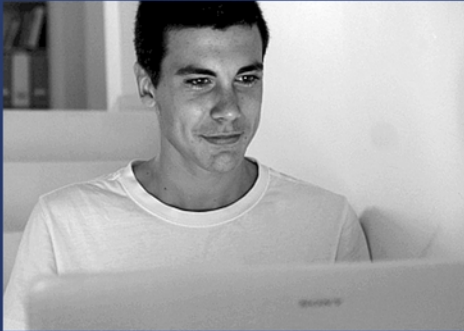
**The Hacker Highschool project is the development of security and privacy awareness learning materials for junior high and high school students.**

Hacker Highschool is a set of lessons and a practical means of making hackers. Beyond just providing cybersecurity awareness and critical Internet skills, we need to teach the young people of today how to be resourceful, creative, and logical, traits synonymous with hackers. The program contains free security and privacy awareness teaching materials and back-end support for teachers of accredited junior high, high schools, and home schooling. There are multiple workbooks available in multiple languages. These are lessons that challenge teens to be as resourceful as hackers, including safe Internet use, web privacy, researching on the internet, avoiding viruses and Trojans, legalities and ethics, and more.

**The HHS program is developed by ISECOM, a non-profit, open-source research group focused on security awareness and professional security development and accreditation.**

# Hacker HighSchool

## SECURITY AWARENESS FOR TEENS



## LESSON 9: HACKING EMAIL



HACKING IS LEARNING  
[www.hackerhighschool.org](http://www.hackerhighschool.org)

ISECOM

Creative Commons 3.3 Attribution-Non-Commercial-NoDerivs ISECOM

[WWW.ISECOM.ORG](http://WWW.ISECOM.ORG) - [WWW.OSSTMM.ORG](http://WWW.OSSTMM.ORG) - [WWW.HACKERHIGHSCHOOL.ORG](http://WWW.HACKERHIGHSCHOOL.ORG) - [WWW.BADPEOPLEPROJECT.ORG](http://WWW.BADPEOPLEPROJECT.ORG) - [WWW.OSSTMMTRAINING.ORG](http://WWW.OSSTMMTRAINING.ORG)



## WARNING

---

The Hacker Highschool Project is a learning tool and as with any learning tool there are dangers. Some lessons if abused may result in physical injury. Some additional dangers may also exist where there is not enough research on possible effects of emanations from particular technologies. Students using these lessons should be supervised yet encouraged to learn, try, and do. However ISECOM cannot accept responsibility for how any information herein is abused.

The following lessons and workbooks are open and publicly available under the following terms and conditions of ISECOM:

All works in the Hacker Highschool Project are provided for non-commercial use with elementary school students, junior high school students, and high school students whether in a public institution, private institution, or a part of home-schooling. These materials may not be reproduced for sale in any form. The provision of any class, course, training, or camp with these materials for which a fee is charged is expressly forbidden without a license including college classes, university classes, trade-school classes, summer or computer camps, and similar. To purchase a license, visit the LICENSE section of the HHS web page at <http://www.hackerhighschool.org/licensing.html>.

The HHS Project is an open community effort and if you find value in this project we ask that you support us through the purchase of a license, a donation, or sponsorship.



## Table of Contents

WARNING.....	2
Contributors.....	4
Introduction.....	5
Overall: How Email Works.....	6
Feed Your Head: Email Headers.....	9
Dig Me.....	12
Game On: The Bug Trap.....	14
The Risky Business of Email Composition.....	17
Receiving Email.....	18
Responding to Email.....	19
Cryptography Protecting Contents From Disclosure.....	20
PGP and GPG.....	21
MIME Your Manners.....	21
Key Trust.....	22
Sending An Encrypted Email Using GPG.....	22
Receiving An Encrypted Email Using GPG.....	22
GPG Implications.....	22
Email Server-Side Vulnerabilities and Threats.....	24
Bandwidth Eating.....	24
Email Server Vulnerabilities.....	25
Email Server Threats.....	25
Email for Fun and Profit.....	25
The Key to Success.....	26
Email Client-Side Vulnerabilities and Threats.....	27
Turn On The Lights.....	27
Malware, Trojans, And Rootkits, Oh My.....	28
This Email Looks Legitimate, Let's Open It Up.....	28
Exciting Tricks With Email Systems (Hacking the Postman).....	29
SEAK And Ye Shall Find.....	29
Spoofing Versus Malware.....	30
Stupid Email Tricks.....	31
Outsmarting The Email Bots (Email Obfuscation) .....	31
Conclusion.....	33
The Ultimate Disclaimer.....	34



## Contributors

---

Pete Herzog, ISECOM

Bob Monroe, ISECOM

Greg Playle, ISECOM

Marco Ivaldi, ISECOM

Simone Onofri, ISECOM

Peter Houppermans

Andrea Zwirner

# ISECOM



## Introduction

---

Email has been around for a long time; like longer than those socks stuffed under your bed. It predates the Internet (not your dirty socks), and is one of the first forms of electronic information exchange. Before email, we had smoke signals, half-naked guys running as messengers, bricks with notes attached, Morse code, large rocks slung over castle walls with curse words written on them, and a variety of other analog communication methods like the telephone and paper "snail mail" (not really delivered by snails). Many of these original message transmission required special tools, training, or lots of rocks. Luckily, enterprising authors created text that could be written on stone tablets or bound in books and thrown at people or read by them. One of the first books was *Smoke Signals for Dummies*.

Email is based on simple store and forward principles. It can be relatively easy to use (unless you are in a huge hurry), very robust and so cheap that it is often abused for commercial and criminal purposes. Its asynchronous design allows communication to take place without the need for sender and receiver to both be online at the same time. Kind of like when your mother is talking to you and you're not paying attention until she asks you a question. You are not there for the transmission but you better be a quick deceiver. Um, receiver. A quick receiver.

In this lesson, we will focus on modern Internet email and hacking or security issues you can use for fun and profit.

## Overall: How Email Works

**First, we are going to pretend that you are an email.** You will follow the transmission and receipt of yourself as an email, and we will identify the various components that move you along.

1. Email (you) is (are) created either using an email **client** such as Outlook, Mail, Eudora, Pegasus or Thunderbird, or on a web service like Yahoo Mail, using a web interface. It's almost funny how much email mimics "snail mail," because your message is enclosed in an envelope, like in Figure 9.1.



**Figure 9.1:** Email message, headers and envelope

2. You are sent to a mail server called a **Mail Transmission Agent (MTA)**, which queues you for transmission. Modern mail systems do this typically via encrypted **SMTP (Simple Mail Transport Protocol)** since they require authentication to prevent abuse, and encryption protects credentials from disclosure, along with the email contents. MTAs accepting email (you) without some sort of authentication are called "open relays" and tend to be abused by senders of junk mail, also known as UCE (Unsolicited Commercial Email) or **spam**.
3. For each address ("recipient") in the message, the MTA first checks if a recipient is local (right on the same computer). If not, the MTA uses a so-called MX record (explained



below) to find the server for the relevant domain. If there is no valid receiving host found, a failure message for that specific address is sent back to the sender.

4. The MTA attempts to deliver you to each address. If this fails, the MTA re-queues the message to try again later until timeout occurs and a delivery failure message is returned, usually in 48 hours. So you have to hang around for about two days. This delivery may initially be deliberately delayed by the receiving MTA as an anti-spam technique: spam software is typically less intelligent and will not queue and retry delivery (the technique is called **greylisting**). By default, this delivery takes place via **unencrypted** SMTP. Encrypted connections are the exception rather than the rule.
5. Optionally, a mail relay picks you up and routes you to your final destination. This typically happens in environments with spam and virus filtering and where security dictates a layered model, such as enterprise or government networks.

Did you catch that reference to a **layered security model**? Those heavy-duty government security guys can't create M&Ms, hard on the outside but soft in the middle. They put in lots of layers of armor: router controls and firewalls, intrusion detection systems (IDS), anti-virus, anti-malware, spam control and a whole lot more.

Which sounds pretty tough to hack. But never forget: every program you install adds more code, with more vulnerabilities, and the same goes for hardware. That cool VPN device, for instance, might give you "secure" VPN – or it might offer backdoors of its own. It depends a lot on whether you're Red Team or Blue Team how much you like this.

6. The receiving MTA expands the address if it is an alias or a mailing list. These do not need to be in the same domain: an alias can expand into a whole new email address on another server. After expansion, you are re-queued for further delivery.
7. When an email address refers to a local mailbox, you are now moved into that mailbox (unless the mailbox has exceeded its storage quota). You might be too big. You gotta stop eating so much junk food.
8. You are then picked up via the POP3 or IMAP protocol by webmail or a mail client. Here too, the connection is generally encrypted (with SSL) to prevent leaking login credentials; the protocols are POP3S and SSL IMAP. POP3 is a "pick up" process: it downloads messages, then deletes them from the server (this can be date driven). IMAP is a synchronization process that seeks to keep clients' mailboxes identical to what is on the server account (for mobile devices this is typically within a date range to preserve device storage), which makes IMAP perfect to maintain email on multiple devices at the same time.
9. Finally, most email clients now have junk mail detection built in, usually based on Bayesian, pattern scoring principles. Try sending your friend an email with "Viagra" in the Subject to see how this works.

#### The Three Stages of Spam Filtering

- a. Receiving servers first check on origin: an SMTP connection is refused from blacklisted servers (various companies exist to provide these lists).



- b. When a connection is accepted, email is then scanned for content. Some organizations are concerned they may have a message falsely marked as junk mail; they may require suspect email to be marked as junk, but still delivered.
- c. Finally, most email clients now have junk mail detection built in, usually based on Bayesian, pattern-scoring principles.

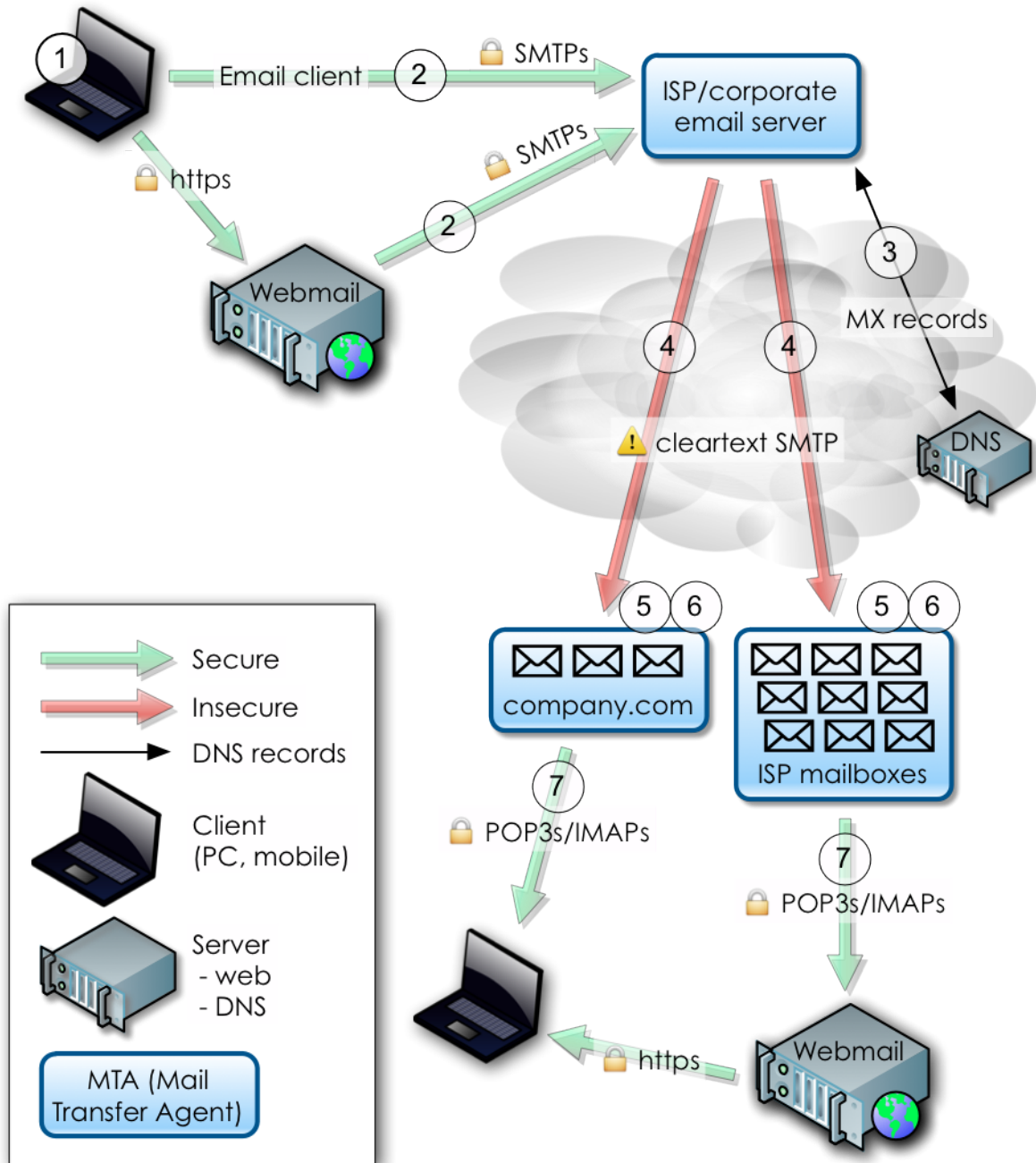


Figure 9.2: Email process flow

So, there ya go. That was easy, wasn't it? You start at one place and may or may not end up at another place, depending on whether:

- You have the correct address



- You are spam
- You are too big
- The receiving mail box is too small
- or you are too old.

With all this great information you now know how emails move through the digital world. Everything you need to know about life can come from email traffic.

- Know where you are going.
- Don't eat spam.
- Get big mail boxes (communicate a lot).
- Don't get big (Eat right and exercise to maintain a healthy lifestyle).
- Lastly, don't get old.

See how easy that is?

### Feed Your Head: Email Headers

A **message**, from the SMTP point of view, consists of **headers** and a **body**. Headers are machine-parseable statements containing information of all kinds, the most basic ones being headers like 'To:' for the mail recipient or 'Subject:'. The sender address might be quickly dismissed as a basic piece of information easy to describe but we'll see that it's a more complex concept.

The body of the message contains everything else (everything other than headers) and it's not normally supposed to be parsed by MTAs (although, as we'll see, it might happen for filtering purposes). Usually the body of the message contains simple text but it can also be HTML (which often annoys really technical people), and in multi-part messages (i.e. messages with attachments) MIME is used. MIME stands for Multipurpose Internet Mail Extensions and it's a standard that is used for sending character encodings other than plain ASCII and binary content. MIME is automatically used by the email client when needed.

Some headers can be removed, some can be modified and some will be added by different components in the mail flow process. Every MTA should always add a "Received" header for tracking its role the email path during transmission. In theory, by looking at the headers you should always be able to track the original sender. We'll soon see why this is not always the case.

There's a set of headers that every email should have in order to be parseable by the SMTP standard, some headers that most SMTP implementations consider standard but that really are not, and some custom headers (X-\*) that are customizable and can contain any sort of message. Think of it as a way to shift user-definable content from the body to the headers. Some of the most widely used examples are filtering applications information (X-Spam) and MUA (X-Mailer). (It's not uncommon to spot very interesting customized headers in the wild; email from security consultants may have weird ones!)

Consider this example.

[Sample message]

```

From root@isecom.org Sat Sep 30 13:50:39 2006
Return-Path: <root@isecom.org>
Received: from iseecom.org (localhost.localdomain [127.0.0.1])
    by iseecom.org (8.13.8/8.13.7) with ESMTP id k8UBodHB001194
    for <test@isecom.org>; Sat, 30 Sep 2006 13:50:39 +0200
Received: (from root@localhost)
    by iseecom.org (8.13.8/8.13.5/Submit) id k8UBoNcZ001193
    for root; Sat, 30 Sep 2006 13:50:23 +0200
Date: Sat, 30 Sep 2006 13:50:23 +0200
Message-Id: <200609301150.k8UBoNcZ001193@isecom.org>
From: root@isecom.org
To: test@isecom.org
Subject: foobar

```

test

If you look at your raw mailbox, you can sometimes see an additional "From" followed by a space and then a sender address, without the colon seen in the usual "From:" header. That's an internal separator for messages defined by the mbox storage format and it's not really an SMTP header.

The **Mail Delivery Agent (MDA)**, which is the component responsible for storing the message in final delivery, also has the task of protecting any existing line that begins with "From" in the body of the message, a process that's prone to misinterpretation.

The sample message shown above was transmitted with the following SMTP transaction:

```

CONNECT [127.0.0.1]
220 iseecom.org ESMTP Sendmail 8.13.8/8.13.7; Sat, 30 Sep 2006
14:08:38 +0200
EHLO iseecom.org
250-iseecom.org Hello localhost.localdomain [127.0.0.1], pleased to
meet you
250-ENHANCEDSTATUSCODES
250-PIPELINING
250-8BITMIME
250-SIZE 5000000
250-DSN

```

```

250-ETRN
250-DELIVERBY
250 HELP
MAIL From:<root@isecom.org> SIZE=57
250 2.1.0 <root@isecom.org>... Sender ok
RCPT To:<test@isecom.org>
DATA
250 2.1.5 <test@isecom.org>... Recipient ok
Received: (from root@localhost)
    by isecom.org (8.13.8/8.13.5/Submit) id k8UC8EMj001346
    for root; Sat, 30 Sep 2006 14:08:14 +0200
Date: Sat, 30 Sep 2006 14:08:14 +0200
Message-Id: <200609301208.k8UC8EMj001346@isecom.org>
From: root@isecom.org
To: test@isecom.org
Subject: foobar

test
.
250 2.0.0 k8UC8c3M001347 Message accepted for delivery
QUIT
221 2.0.0 isecom.org closing connection

```

The path of an email message is traced with the "Received" headers:

```

Delivered-To: <spoofer@isecom.org>
Return-Path: test@isecom.org
Received: from smtp.isecom.org (smtp.isecom.org [140.211.166.183])
    by azzurra.isecom.org (8.13.6/8.13.6) with ESMTTP id
    k4KL5UOq014773
    (version=TLSv1/SSLv3 cipher=DHE-RSA-AES256-SHA bits=256
    verify=NO)
    for <spoofer@isecom.org>; Sat, 20 May 2006 21:05:30 GMT
Received: by smtp.isecom.org (Postfix)
    id D138A64413; Sat, 20 May 2006 21:05:29 +0000 (UTC)
Delivered-To: spoofer@isecom.org
Received: from localhost (localhost [127.0.0.1])

```

```

    by smtp.isecom.org (Postfix) with ESMTP id B87EF64409
    for <spoofer@isecom.org>; Sat, 20 May 2006 21:05:29 +0000
(UTC)
Received: from smtp.isecom.org ([127.0.0.1])
    by localhost (smtp.isecom.org [127.0.0.1]) (amavisd-new, port 10024)
    with ESMTP id 24780-13 for <spoofer@isecom.org>;
    Sat, 20 May 2006 21:05:23 +0000 (UTC)
Received: from mail2.isecom.org (bsiC.pl [83.18.69.210])
    (using TLSv1 with cipher DHE-RSA-AES256-SHA (256/256 bits))
    (No client certificate requested)
    by smtp.isecom.org (Postfix) with ESMTP id 6B37E64405
    for <spoofer@isecom.org>; Sat, 20 May 2006 21:05:23 +0000
(UTC)
Received: from localhost (localhost.isecom.org [127.0.0.1])
    by mail2.isecom.org (Postfix) with ESMTP id BDF11B02DE
    for <spoofer@isecom.org>; Sat, 20 May 2006 23:12:55 +0200
(CEST)
Received: from mail2.isecom.org ([127.0.0.1])
    by localhost ([127.0.0.1]) (amavisd-new, port 10024) with ESMTP
    id 11508-04 for <spoofer@isecom.org>; Sat, 20 May 2006 23:12:42
+0200 (CEST)
Received: from localhost (unknown [192.168.0.5])
    by mail2.isecom.org (Postfix) with ESMTP id 54666B02DC
    for <spoofer@isecom.org>; Sat, 20 May 2006 23:12:41 +0200
(CEST)
Date: Sat, 20 May 2006 23:05:04 +0200
From: John Doe <test@isecom.org>
To: spoofer@isecom.org

```

## Dig Me

When you're working in Linux and UNIX in general, **dig** is your best friend for testing DNS settings. MX records are very important to email delivery, so let's have a look at them briefly. MX records are for email, and have no relationship to websites for the same domain. The web server of "domain.com" may be a completely different system from the mail server, which is why those DNS records are identified differently.

The way to get MX records is by using the **dig** command from a UNIX, Linux, or OSX command line. **dig** is a DNS information tool, and as any UNIX program it has a gazillion options. We will just use one format. Using

```
dig <domain name> MX
```

tells dig to extract only mail exchange records from the relevant domain. Another easy example is

```
dig <servername> <type>
```

For example the public DNS server 213.133.105.2 ns.second-ns.de can be used for testing. See which server the client receives the answer.

```
dig sleepyowl.net
sleepyowl.net.          600      IN       A        78.31.70.238
;; SERVER: 192.168.51.254#53 (192.168.51.254)
```

The local router 192.168.51.254 answered and the response is the A entry. Any entry can be queried and the DNS server can be selected with @:

```
dig MX google.com           # Get the mail MX entry
dig @127.0.0.1 NS sun.com   # To test the local server
dig @204.97.212.10 NS MX heise.de # Query an external server
dig AXFR @ns1.xname.org cb.vu # Get the full zone (zone transfer)
```

The command host is also powerful.

```
host -t MX cb.vu           # Get the mail MX entry
host -t NS -T sun.com     # Get the NS record
host -a sleepyowl.net     # Get everything
```

As a larger example, here are the MX records for Google's *gmail.com* domain:

```
;; ANSWER SECTION:
gmail.com.          893      IN       MX       10 alt1.gmail-smtp-in.1.google.com.
gmail.com.          893      IN       MX       40 alt4.gmail-smtp-in.1.google.com.
gmail.com.          893      IN       MX       30 alt3.gmail-smtp-in.1.google.com.
gmail.com.          893      IN       MX       20 alt2.gmail-smtp-in.1.google.com.
gmail.com.          893      IN       MX       5  gmail-smtp-in-v4v6.1.google.com.
```

There are three values in each line that are of interest. The "893" is a **time to live** value (how many seconds, or how many routers to hop, depending) you will find in every DNS record – it indicates how long a DNS is allowed to cache the record before the information is considered stale and has to be retrieved again.

The "10" in the top line, and "40", "30", "20" and "5" in subsequent lines are "preference" values, followed by a **Fully Qualified Domain Name (FQDN)** of a system prepared to handle email. The preference values are used by the MTA to decide which of the machines in the MX records list to try first, and which to contact next, should the first one fail or refuse email. If no server is found to accept the email, a failure message is sent back



to the email originator (using the “reply-to” or “from” information). Lower values indicate preferred MTAs. Thus, the last entry in the list above will be tried first, with the rest as fallback if the first system fails or is overloaded.

A service can also offer records with identical preferences; here is the response from yahoo.com where the preferences value is set to “1” on all records:

```
;; ANSWER SECTION:
```

```
yahoo.com.          48      IN      MX      1 mta6.am0.yahoodns.net.
yahoo.com.          48      IN      MX      1 mta5.am0.yahoodns.net.
yahoo.com.          48      IN      MX      1 mta7.am0.yahoodns.net.
```

Doing this means the email load will be distributed over the 3 systems equally. The very low TTL value of “48” suggests this DNS entry is dynamically controlled, a sign of an active load balancer. Load balancers do pretty much what their name says they do; they make sure traffic (inbound, outbound, high priority, low priority) gets the level of attention it deserves.

Last but not least, you can also identify whether the receiving domain uses mail filtering. The famous domain no10.gsi.gov.uk (the domain of Britain's Prime Minister) shows that a company called MessageLabs is presently responsible for mail filtering:

```
;; ANSWER SECTION:
```

```
no10.gsi.gov.uk.   3600   IN      MX      20 cluster.gsi2.message-labs.com.
no10.gsi.gov.uk.   3600   IN      MX      10 cluster.gsi.message-labs.com.
```

You do not need to fear black helicopters when you look this up: this information is public, as email would otherwise not work. Besides, the UK military only has green helicopters!

## Exercises

- 9.1 Does your email platform support a “Delivery Receipt” or any kind of delivery flag that lets you know (at least) that your mail reached some destination? If it does exchange messages with a friend and examine the headers from that traffic.
- 9.2 Choose a domain name. Find out which system handles email for that domain by looking up MX records.

## Game On: The Bug Trap

The cafeteria floor was slightly damp, almost like fly paper, with a stickiness pulling at the bottom of her rubber soled shoes. Jace glanced at the reflective sheen of the gunk floor wondering how a place that serves food could smell so bad but keep a mirror shine. The odor reminded her of when her Grandpa used to place the cockroach traps behind the apartment couch. As Grandpa pulled out the old trap, Jace could see the encrusted remains. It seemed like the entire inside of the roach trap was filled with dead bugs.

She was never shy about asking questions. “Why don't the roaches just leave the box? Can't they see all their friends in there dead?” she asked Grandpa more than once just to be sure she got the right answer each time. Jace loved to watch



Grandpa work, often getting in the way by putting her head right over his shoulder. He never complained. He always loved having his granddaughter right by his side as much as possible.

"Jace, the roaches are attracted to this trap by the smell inside the box. As soon as they go inside the trap, the floor is really sticky and they get stuck in the box. It's like they are glued to the floor. They don't seem to notice all the other dead insects inside and they die too," her grandfather would explain in roughly the same version each time he was asked.

"Roaches aren't very smart," little Jace would reply with a smug smile.

"Yes, my dear, you are much smarter than a cockroach."

"Thank you, I guess."

Back at the school cafeteria, coffee brown hair fell into her face as she continued to look at the mirror floor; she needed to get back her classwork.

Just out the corner of her right eye, Jace caught a sudden commotion at the cafeteria double doors. They swung open as several people burst into the large open room. Instantly going into clandestine mode, she leaned forward and let her thin shoulder-length hair cover her face. She heard, "There, she's over there trying to hide herself! Grab Jace. Don't let her move!" Several older voices shrilled with the excitement of a witch hunt.

The teen hacker held her position at the table, clenching her knapsack and pretending to be unaware of the coming attack. Knives, pitchforks, torches, angry mobs and all those monster movie images compressed themselves into her calculating mind. Curiosity got the best of her and she looked up to see the school principal, his secretary, Mr. Tri, three freshmen table tennis players and several other oddballs approaching. The roar of their voices was tremendous as it bounced against the polished floor towards her.

"Hold on! I said hold on," a familiar voice commanded somewhere behind the angry freaks. The mob lost momentum. The freshmen parted the crowd so the Chief of Police could step through the cluster of confusion. "Alright folks, thank you for your overzealous help locating Ms. Jace for me. Now, I would like to have a moment alone with her," the chief said in a calming voice. He'd used that same voice to talk down a jumper off a 14 story building years ago. It worked then, it sort of worked now. The crowd became a loose net of individuals trying to look busy, tying their shoelaces, too obviously stretching to listen to the chief's private conversation.

"Hi Jace," Chief couldn't think of anything else to open with.

"Yes, Police Chief. How may I help you at MY school, while I am enjoying MY lunch. With MY peers all staring at ME," she almost broke her jaw clenching her teeth.

"I apologize. I'm sorry to interrupt your massive gathering of friends here but I need your help now," the chief said, trying to keep his cool, but also letting Jace know that this wasn't the time to be difficult. Jace unclenched her hands from her pack and looked the Chief in the face. He tilted his head towards the double cafeteria doors, motioning her to follow him.

Jace looked down at her unfinished sandwich, reluctant. The Chief didn't take his eyes off of her. He raised his right hand and slapped his fingers in the air. Jace flinched. The entire lunchroom flinched. Principal Mantral realized what the signal





meant and came rushing forward with a clear plastic bag.

"Cookies, eh," Jace asked.

"Chocolate chip pecan, made by Officer Hank's wife," Chief replied. "Deal?"

"Deal," she replied with one cookie already in her mouth.

As the two walked out of the school, the chief asked if Jace had ever ridden in a SWAT van before. "That was the only vehicle I could get at the last minute. Sorry," he said. The two of them left the school looking like rock stars in the SWAT van. Jace laughed looking in the rearview mirror at the stunned students and school staff.

"Here's what is going on. Someone is looking at my email. I don't know how or who or why but I do know that my emails are being hacked. I need you to help me stop this. It's causing major problems with our law enforcement capabilities," the Chief didn't give Jace a chance to interrupt. "When you set up our network last summer, you did a bunch of extra security stuff. It hasn't been enough. I can tell you that one email three weeks ago concerned a technicality on a particular suspect we had. Only me and the District Attorney knew about this problem."

The police chief reached across the van console to grab a cookie from the open bag. Jace jokingly slapped his hand away. He reached down towards his baton that he didn't wear since he was a wasn't a street cop anymore. Jace relented and handed him a large pecan piece to keep him talking, which he did, cookie crumbs littering his uniform.

"Two hours after the DA and I emailed each other, I get a call from the front desk telling me that the suspect just posted bail. The suspect's lawyer found out about the technicality and got a judge to sign off on release for the suspect. There were only two people who knew about this technicality and that was through my email," the Chief said.

He continued, "Last week I got a call about the possibility of some missing evidence at a crime scene. That was just an anonymous phone call. There wasn't any specific item mentioned in that call. I wrote a quick email to our evidence clerk asking for the inventory log for the entire weeks' worth of cases, especially the log from that day. The clerk emails me the log and I compare it to the police report from the crime. Being the thorough investigator I am, I delete all the information in the log file that isn't relevant and forwarded the log file to our Internal Investigation team."

Jace is trying to understand what he is saying in-between all his police jargon. "So?" she blurted out, feeling much better after her sandwich and five cookies.

The chief looked a bit annoyed but answered anyway, "So! So there isn't any missing evidence that we can tell. Later that day, I get another call from the DA asking me where the murder weapon is from that same case. It didn't occur to me that the gun was missing from the evidence locker. Again, two hours later another suspect is out on bail because the police and forensic team didn't document or turn in the pistol used in the crime."

Jace, wishing she had a large cold glass of milk, chimed in, "So the mysterious caller was checking to see if the gun has in police custody. The email you sent to your Internal Thugs, verified that the weapon was never entered in as police evidence."

The police chief had an amazing smile of satisfaction when she finished her conclusion. "Ya know, Jace, you'd make a terrific police detective when you get



older.”

Jace shot back, “Yeah, well, I have too much self-respect to be a cop. I’d rather be a lawyer or a politician or some other lower form of life.” Luckily she started to giggle as she said the last part because the chief was getting angry with her insult. “Just joking, Chief.”

**Game continues...**

## The Risky Business of Email Composition

- **Disclosure.** Think about whom you are emailing, why and how. Not only is email transmission by default insecure when it leaves the local MTA, you are also releasing information. The use of encryption such as PGP, GPG and S/MIME requires both sides to be similarly equipped, and is generally perceived as very complicated to use (translated: users avoid it with enthusiasm). An alternative way to protect the transmission would be the use of the *same* email provider: that way, the message never needs to travel across the Internet in an unencrypted form. This is where the *how* question appears: are you sure your provider or that of the recipient (or one or the other of your governments) is not listening in? Take that into consideration when you handle something confidential.
- **Rerouting.** An email address does not need to remain in the domain it is sent to, but could be redirected elsewhere. As an example, the US company *pobox.com* does not sell mailbox services, only aliases. The main risk is that your email may thus travel over various, different legal jurisdictions before it arrives at its destination. In our example, a *pobox.com* alias will always go via MTAs in the US first, and is thus at risk of interception under the ongoing abuse of the US PATRIOT Act.
- **Privacy violations.** A recipient using services such as Facebook or Google exposes his email to automated scanning of content, even though the *sender* never gave that permission!
- **Distribution lists.** If you use an email distribution list, use the BCC (blind carbon copy) field for it. Email addresses in the TO: and CC: field are visible to every recipient of the email, and could end up giving away the contents of your email list to an uncontrolled third party, and expose your recipients to spam and other junk mail.
- **Conflict.** An email is like a letter, but is written and sent much more quickly, which leaves you less time to consider its contents. Writing email is like driving: it’s best not done in anger. In case of emotional involvement, write a draft and leave it for an hour, then reconsider if you really should send it. It could save a friendship or a career.
- **Misaddressing.** One of the main causes of email going astray is misaddressing. This is a consequence of mail clients trying to autofill an address from the characters typed by the user. Always check if the recipient is indeed the intended one.
- **Multiple recipients.** When you send email to more than one person, make sure the content is appropriate for all recipients. Also, it is good practice and ethically correct to visibly copy someone in if you use their information or talk about them.
- **Legal issues.** Disclaimers under your email may look impressive, but have no legal value other than a copyright notice. You sent the email, so you cannot disavow its content (to a degree, of course you could always claim its sender was spoofed) and



you cannot prescribe what an incorrect recipient should do with an email because you probably don't have a contractual relation with them. (See the Ultimate Disclaimer at the end of this lesson.)

- **Top posting.** When you reply to an email, does your email program automatically put your reply on top of the original message? This seems to be the default these days, but unfortunately it's ... rude. Recipients who have to start from your reply and work their way down to some context aren't likely to love you. On the other hand, weren't they in on the original message to begin with? Consider, at least, whether you want to engage in "top posting."
- **Autoresponders.** "You sent me an email so I'm sending you this automatic email response to let you know I won't get your email until I return, so heaven help us both if you've got an autoresponder too, because this is going to go in circles until the end of the universe." This stuff can drive you crazy, but it's also an awfully convenient notice to evildoers that you're probably not at home. What was your street address again?
- **Signatures.** Do you use a signature, an automatic "Yours Truly, Stardonk Cluck, Program Manager for Automated Actions" that sticks itself to the bottom of every message you send? They're not necessarily bad – until they get really long. And ten of them stack up at the bottom of a long back-and-forth conversation. And they are all in HTML, not polite plain text, so that your graphic of a gorilla climbing a skyscraper appears over and over and over. Be kind about using a signature, and don't subject your recipient to the dangers of HTML email at all if you can help it.

### Exercises

- 9.3 Head over to <http://www.gaijin.at/en/olsmailheader.php> and insert an email header you've taken from any email. This program is an analyzer that will provide you information about that email header. From the information given, what can you do with these results?

## Receiving Email

Mail clients contact the servers on which mailboxes are stored, and check if the top message count has not changed. Some clients do this periodically (for instance, every 30 minutes), some do it manually (usually to preserve bandwidth) and some maintain a simple permanent connection with the mail server so that they receive an update as soon as new email arrives (called **push notification**).

When an email is inbound, the mail client or webmail environment will pick this up via POP3 or IMAP. Mobile clients usually download only the header and a small portion of the message to save bandwidth, leaving it up to the user to decide if the whole message should be picked up, left for later or deleted.

The early days of email communication took place over unreliable, slow connections, and the handling of attachment such as documents, spreadsheets or images still shows this. An attachment must always be downloaded in full before it can be displayed. Users using webmail on a third party computer such as in cyber cafes must be careful: **viewing an attachment means leaving a copy behind on the system's hard disk**. By default, those are **not** erased after use.

Webmail on an untrusted computer naturally carries another risk: unless you use one-time passwords, you may leave behind your email access credentials because there is no guarantee the third party machine is not infected or monitored.



Systems with active email should have up-to-date antivirus protection, but you should understand that antivirus only protects against **known** malware. Especially with targeted attacks, it can take several days before malware is added to virus scanners; some malware is never added at all.

Incoming email contains a travel history in the headers. Every system mail passes through adds a line in the hidden part of the header, with the latest one on top. However, be aware this is also easy to forge: keep in mind that not all entries may be real.

## Exercise

9.4 Consider the “temporary” files people leave behind when they use email. You can see a lot by looking into temp directories (there are usually more than one). Windows, for instance, makes it easy to see what’s in temp directories even if you don’t know where they are: the `%temp%` variable knows all of them.

Open a command-line interface in Windows and type

```
dir %temp%
```

What do you see?

For an even handier view, use Windows Explorer by typing this command:

```
explorer %temp%
```

9.5 Open up the email header on any email. See if you can locate any additional receivers besides you. They might be located in the carbon copy (cc) section of the email header.

- Select several emails. Check the mail path and origin via mail headers. Check what other information is available in the headers (hint: email client and antivirus software versions; encryption algorithms; etc.).
- Compare the sender and reply-to addresses
- Have a look at some junk mail. What do you see in those headers compared to normal emails? Check where all the links go (just as text, not by going there). Do the links URLs go where the text says they are going?

## Responding to Email

Responding to email needs to be done with some care. How many times have you said something or done something that you didn’t mean or wish you could take back?

First of all, NEVER react to what is clearly junk mail, even if it is to unsubscribe. All you do is confirm that (a) the email account is live and (b) someone at that email address actually reads junk mail. The result of the cancellation attempt is thus, ironically, *more* junk mail.

Check for address disclosure. Do all recipients need to be visible? If you use a mailing list, are all recipients still valid? Does every recipient really need to see your answer?

Be hygienic. Does the whole email need to be repeated or can you just use the parts that are relevant? If you re-use parts of a previous email you can show that by “quoting” - a way to make it visible you are repeating part of an email, and then respond to that specific part.

Be careful with quoting: is all that you repeat actually meant for the recipients you are selecting now, or are you including (parts of) a discussion that was confidential and not meant for the new recipient(s)? Avoid quoting the whole original message including signature and (usually extensive) disclaimer. Be aware that whatever you send can be forwarded on to anyone without your permission or knowledge too. It is a good and polite

habit to copy a person in when you talk about them or refer to something they have done, so they know what has been said about them, and it prompts you to avoid stating something you may regret later. Delivery flags are helpful in keeping an eye on email as it travels to its destination, and by looking up MX records you can work out where the email will go. You could use geolocation to give you a physical location as well. Delivery flags also chew up bandwidth. Because a delivery flag was set, your email server must send a response. Not everyone appreciates emails flagged as "urgent" or "important." Flags such as these are usually an indicator that the message is spam, if they weren't sent by a co-worker.

## Exercises

9.6 Forward an email to another account and compare headers.

- How can headers be used against you, and how you could prevent this from happening?
- Can you forward an email that was sent to you as a blind copy (BCC)?

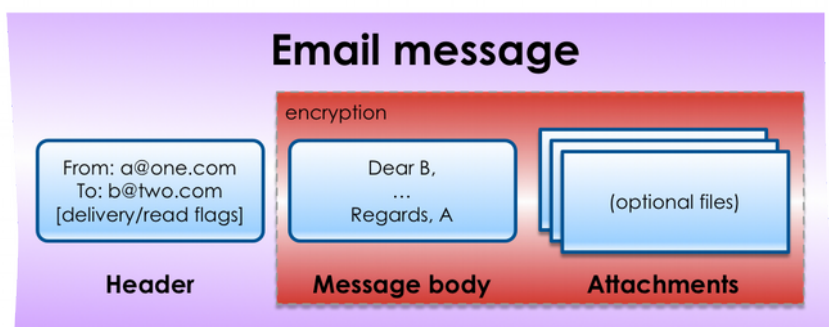
9.7 Write yourself an email and send it to yourself. During delivery, quickly retract that email (unsubscribe). If the email was successfully pulled back, take a look at the header of that draft email. Copy that header to a text editor and see if you can locate which email server stopped that email from going through. Cool huh?

## Cryptography Protecting Contents From Disclosure

The simplicity of email makes it also vulnerable. The sender cannot be sure that an email is not altered on its way to the recipient, there is no way to make sure that only the receiver can read it and a receiver cannot be sure it is actually sent by the person listed in the email as the sender.

One way to ensure confidentiality is to encrypt a document before attaching it to an email. For example, it is possible to encrypt text documents and spreadsheets like those produced by OpenOffice, and PDF files support encryption too. However, applying cryptography to email itself is easier, and also allows for the email contents to be secured.

Email headers still need to be left in cleartext so that the mail servers can process and deliver the email.



**Figure 9.3:** Email encryption

Email security can be provided in two different ways: using PGP (or GPG) and S/MIME. Both use encryption to assure:

- **Confidentiality:** can only the intended recipient(s) read this email?
- **Integrity:** is the email content unchanged?



- **Authenticity:** did the email really come from a particular sender?

(An easy way to remember this list is the acronym they form: **CIA**.)

In general, authenticity and integrity are combined in electronically **signing** an email: the email gets checksummed, and the result is encrypted and embedded in an electronic signature that could only have been made by the person who holds the right private key (see “PGP and GPG” below).

Confidentiality is assured by using someone's public key to encrypt the message body, so that only the holder of the correct private key can decrypt and read it (see “PGP and GPG” below). For extra assurance, such a message can be signed too.

You should keep in mind that encrypted email is rather uncommon, especially in an era where people voluntarily let their email be scanned by companies such as Google and Facebook. In some countries, you must make sure that you have the means to access your email when authorities demand this of you, for instance in the US by the TSA when you cross the border, and in the UK when served with a warrant under the Regulation of Investigative Powers Act.

## PGP and GPG

PGP stands for Pretty Good Privacy and was developed by Phil Zimmermann. The history of PGP is interesting and certainly worth reading, but for the purposes of this chapter we will only focus on its use.

You are more likely to come across the Open Source version called GPG (GNU Privacy Guard). GPG is available for free for many platforms, and only uses open, publicly evaluated algorithms.

GPG works on the principle of **public/private key management**, which means that keys have a PUBLIC part you can give to anyone who wants to send you encrypted email, and a PRIVATE part you have to keep secret, which is the only way to decipher the message you have received. The combination of private and public key is called a **key pair**, and it is generally the first thing you generate when you install GPG on a machine. The key pair is protected by a password so that it cannot be altered by anyone but the owner. Alterations may be necessary because you want to change the email addresses the key supports, or want to make use of other functions.

Because you need someone's public key to encrypt a message to them, servers such as [pgp.mit.edu](http://pgp.mit.edu) exist where you can download the key or keys associated with a specific email address and upload your own. It is possible that keys have expired or passwords have been lost, so always use the latest key or even better, ask your recipient to send theirs and confirm the key fingerprint (a short version of the checksum).

## MIME Your Manners

**MIME (Multi-Purpose Internet Mail Extensions)** is an email extension of the Simple Mail Transfer Protocol (SMTP). MIME gives you the ability to transfer different types of media and data like audio, video, images, compressed files, and applications as attachments to email. The MIME header is inserted at the beginning of the email and the receiving email client uses this information to determine which program is associated with the attached file. MIME in itself does not provide any security to emails or attachments.

**S/MIME (Secure/Multipurpose Internet Mail Extensions)** is a protocol that adds digital signatures and encryption to Internet MIME message attachments. Using digital signatures, S/MIME allows for authentication, message integrity and **non-repudiation** of origin (“non-



repudiation" means you can't deny you sent it). S/MIME provides privacy and data security (using encryption) to emails that use this protocol.

S/MIME is both a security tool and a security issue since users can send sensitive data or secrets as attachments to outbound emails in order to avoid detection. Therefore, the use of S/MIME in a corporate setting should be carefully monitored on the email servers.

## Key Trust

How do you ensure that a key for an email recipient is really theirs, and not uploaded by someone else? The solution to this is that keys can be signed by others. Imagine you already have the key of someone else who you trust, and who knows the person you want to email with. That other person can **sign** the public key, which means you invest a bit more trust in the key, provided you know this other person. This is known as **inherited trust**. You can also find another way to get in touch with the person and either receive their full public key, or receive the key "fingerprint" - a checksum of the key which is quick to verify. On a key server, a key can also have an ID - yet another checksum serving the same goal.

## Sending An Encrypted Email Using GPG

Most email clients support plugins that make the handling of keys and encryption easier. The best thing to do is to check beforehand if your recipient has a public key and get it from a key server, or from the recipient themselves.

Then, compose your email as normal (once again we strongly recommend plain text email over HTML), add any attachments and tell your email client to encrypt and send the email. If you decided to sign the email, the email client will use your private key to sign the message first, then use the public key of your recipient to encrypt the email and any attachments. If you protect your key pair with a passphrase (you'd better!), your email client will ask for that passphrase.

## Receiving An Encrypted Email Using GPG

Email encrypted with GPG contains either an attachment flagged as GPG, or has a block of text with a header that tells a GPG-capable email client it has just received an encrypted message. The email client will now access your private key (possibly via a password) and decrypt the message and any attachments. If the message was not encrypted with your public key this decryption will simply fail. If the message was signed by the sender, the GPG plugin will use the corresponding public key to verify that signature too.

GPG plugins will alert you to problems with signatures or attachments, but in general you will find that once installed, the use of GPG is quite easy.

## GPG Implications

Be aware that the majority of email is not encrypted, and that probably includes your own. Some people think that using encryption is suspicious and draws attention all by itself. It's your right to have privacy when communicating, so don't worry too much about the opinions of others.

GPG is not easy to use in a webmail environment (aside from the obvious question whether you can trust a third party to encrypt properly and not mount a man in the middle attack on your secrecy) and also doesn't work well on mobile clients. Be wary of

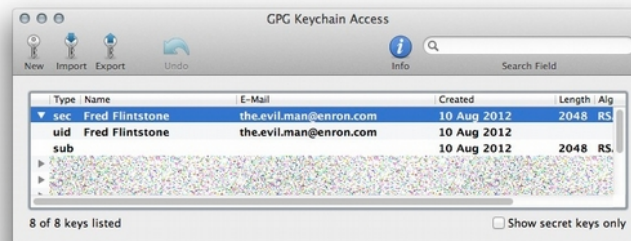
mobile apps that claim to solve this issue: some have been found to send your data elsewhere for processing!

There are online mail services that sell encrypted, “enhanced security” email accounts. But be careful to read the fine print on your user agreement. One provider’s reads like this:

“I understand that this service is not suitable for illegal activity and that the providers of this service will cooperate fully with authorities pursuing evidence via valid legal channels.”

Of course, “legal channels” include programs like Echelon, Carnivore, PRISM, the Patriot Act and XKeyscore. Look ‘em up and ask yourself just how “enhanced” you think this paid “security” really is.

Some countries mandate that you must be able to decrypt any information when ordered to do so by court. For example, in the UK you can be served under the Regulation of Power Act 2000, and non-compliance is termed as contempt of court, automatically resulting in jail time. This has unpleasant implications: if you have been experimenting with encryption and have forgotten the keys or passwords, you will effectively face jail for being forgetful (yes, you will be guilty until you can prove your innocence). It is thus good practice to erase any encrypted material and email that you can no longer access. In a corporate setting you must manage and document key and passphrase changes and disposal of encrypted information very carefully.



**Figure 9.4:** The GPG Keychain

Last but not least, it is interesting to note that the use of email addresses to identify a key is a **convention**, not a mandated standard. It is entirely possible to generate and use keys for email addresses that do not exist. Such keys are still accepted at public servers. This is known as “hiding in plain sight,” and it means that there is no relation between the email addresses and the key used to encrypt/decrypt traffic. In the above image, for instance, both person and email address are fictitious.

The disadvantage of doing this is that it breaks an established way of working, and plugins such as **Enigmail** may need some convincing before they support this more creative approach. A further area for experimentation is expired keys: expiration doesn't stop the keys from working.





## Exercises

- 9.8 Download the GPG support for your email client program and install it.
- 9.9 Find out how to generate your own key. Do so. Keep it local; don't accept any offers to upload your public key to a public key management server.
- 9.10 Add other email addresses to your key, then change the passphrase.
- 9.11 Now publish your public key to a key management server.
- 9.12 Compose an email to someone using GPG. How would you go about getting their key? Do it.
- 9.13 What can you do with messages when you only have your own key?
- 9.14 Create a new key for a fake email address. How easy is it to do on your machine?

## Email Server-Side Vulnerabilities and Threats

Both small and large organizations use email servers to send and receive these electronic messages, unless they outsource the task or use a cloud service. Email serves multiple purposes to its users: some are good purposes and some are evil (hear madman laughing in the distance). Email servers are the first line of attack/defense on a network perimeter.

Emails have been used to send family vacation pictures, piano recital songs, birthday cards, have a bad day cards, homework assignments, excuses for not turning homework, company communications, marketing, newsletters and an assortment of other media. Besides the "Have a Bad Day" email, all the emails mentioned above have a friendly use. Email serves a valuable purpose for daily communications.

On the other hand, emails have been exploited to send out flame letters, porn, pirated MP3s, classified information, corporate research secrets, taunts, cyber bullying, malware, phishing, and spam. In 2012, email attachments moved to second place behind rogue web sites as the primary delivery tool for malware. A vital part of our communal society has become perverted for criminal use.

## Bandwidth Eating

Email servers should be configured to block the bad stuff and allow the good stuff to pass to you. This sounds easy enough and it will be easy for us to tell you about it. You'll have all the hard work of making it happen (again, mad scientist laughing in the distance). All email traffic coming and going through a network eats up vital bandwidth. You will never hear someone complain, "My connection is too fast." The sooner you can detect and inspect email traffic (outbound and especially inbound) on your email server, the less bandwidth is wasted. Besides preserving bandwidth, the ability to filter bad emails early on will save work on CPU server processing.

Some studies estimate that 80% of all inbound email is spam. Do you really want to wait until that junk gets into your email inbox before this stuff is detected and deleted? The sooner spam is intercepted by your email servers, the better. One technique used is when spam is detected, the server will eliminate it after a certain amount of time. This prevents deletion of email traffic that a user might be expecting. Your organization's marketing department might really want that one email with the subject line "How to improve your performance." Turn off automatic email confirmations and receipts to save bandwidth on the email server, too. Your users won't mind, trust us.

Since your email servers are access points exposed to attacks coming from the Internet, you should take extra precautions for anyone with admin rights. Those who have admin rights should never send or receive email while they are logged in with admin privileges. In



fact, those with admin rights should only use those rights for internal network maintenance. Over the years, many networks have been compromised when an admin logged in and surfed the Internet or sent emails while working with escalated privileges.

### Email Server Vulnerabilities

As the name might suggest, an email server is just like any other server. The server will have vulnerabilities that can be exploited. The Common Vulnerabilities and Enumeration database at <http://cve.mitre.org> listed a total of 1043 email server vulnerabilities in 2012. Many of these issues can be resolved through proper server configuration and user privileges. Other issues can only be solved by the software manufacture or by being vigilant when shopping for server software.

For a complete list of all known email server vulnerabilities go to <http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=email+server>.

### Email Server Threats

Large web email clients like Gmail, Yahoo, and Microsoft migrated to a new cryptographic email signature program called **DomainKeys Identified Mail (DKIM)**. DKIM wraps a cryptographic signature around an email that verifies the domain name that the message was sent through. DKIM helps filter out spoofed messages from legitimate ones. The specifications for DKIM can be found at <http://www.dkim.org/>.

The problem involves DKIM test messages. According to **US-CERT (United States Computer Emergency Readiness Team)**, an evil hacker can send a flag that it is testing DKIM in messages. Some recipients will "accept DKIM messages in testing mode when the messages should be treated as if they were not DKIM signed."

This isn't the first DKIM problem that has attracted CERN's attention. The signature key length used for encryption was vulnerable to cracking if the key size was too small. DKIM standards set the minimum key size to 1024, with any email using a smaller key being rejected by the program. But DKIM operations didn't reject smaller key sized emails. Instead, the emails were sent along their merry way, fully vulnerable to factor cracking. Once the key was cracked, a hacker could spoof emails or send out malware using that user's email key and address.

DKIM is designed to act as a "trust" verification tool for email. The system uses public-key cryptography, just like PGP does. With proper use, an email can be traced back to its original sender through a domain verification process. Basically, you are identifying yourself as the sender by your domain origin. This should drastically reduce spoofed emails, filter spam, and prove that you sent that message. Security folks call this non-repudiation.

In non-repudiation, the information provider data cannot be changed. The information is not refutable. If you said, "I want to wear a dress," that statement cannot be contested. You said it, that is a fact and you will not be able to retract that statement. This is important when dealing with contracts, legal matter and excuses to your father for not taking out the trash.

### Email for Fun and Profit

Thanks to the profitable market for corporate espionage, email is a simple method to find client contact lists, customer information, meeting notes, new product developments,



answers to the next math test and all kinds of valuable data. We are not even going to get into government espionage, simply because we all know this has been taking place since the dawn of man. There are several primitive cave paintings depicting one caveman spying on another caveman's woolly mammoth with envy. One can only imagine the spy caveman returning to his tribe to describe the newest version of Mammoth 2.0.

A simple but often missed email security method is scanning of all email attachments. Scanning needs to be performed on all data packets, compressed files, unknown file types, split files, files that can do the splits, files that spit, meta data, files with URLs, and pretty much everything that can be done to a file. This scanning should be focused on inbound traffic but don't forget to be suspicious when large attachments are leaving your network. Sensitive company information needs to be encrypted, especially if sent by email, to anyone inside or outside the network. Oh, by the way, sensitive information really should never leave the network. If a user is sending information outside the network, you might want to keep an eye on their activities.

Large organizations like the Veterans Affairs Hospitals in the US use **data loss prevention (DLP)** software to do all this stuff. Never attempt to playfully email your ex's medical records from the VA Hospital, for instance, since the people who come play with you will make you wish you had just cut your throat and gone *straight* to ... well, you know where.

## The Key to Success

Keyword filtering is a type of application layer filtering (layer 7) that lets you block all messages containing particular keywords or phrases (text strings) that commonly appear in spam (for instance, "Viagra" or "hot sexy babes"). Other forms of email filtering include:

- **Address blocking:** a filtering method that blocks mail from particular IP addresses, email addresses or domains of known spammers.
- **Bayesian filtering:** "intelligent" software that can analyze spam messages and learn to recognize other messages as spam using **heuristics** (patterns of behavior).
- **Blacklisting:** lists of known spammers' addresses can be shared, so each user doesn't have to develop a list from scratch. These lists are available from several providers, and are highly valuable for address blocking.
- **Whitelisting:** a filtering method that, instead of specifying which senders should be blocked, specifies which senders should be allowed. Again, these lists are used as part of address blocking.
- **Greylisting:** this temporarily blocks email from unknown sources. Legitimate email will be re-transmitted, but spam usually won't.
- **Challenge/Response filtering:** replies to email from senders not on a "trusted senders" list with a challenge, usually involving solving a task that is easy for humans but difficult for automated bots or scripts.

There are many open-source and for-pay applications that can do these kinds of filtering, some better and some worse. If you've ever had to deal with these PITAs (we'll let you figure out on your own what a PITA is) you'll see them as the challenges to clever script writers that they are.



## Email Client-Side Vulnerabilities and Threats

Incoming email may contain malware, usually in the form of an attachment or a web link. When you see these clues think "Scam!"

- An unexpected origin: who sent you this mail, and would they have sent it? A favorite trick of spammers is to use other people's valid email address as the origin so spam passes filters and users are more likely to open the email.
- A "too good to be true" event such as a lottery win, inheritance or bank "mistakes" in your favor. Look up the "Nigerian scam." Do any of the sample messages you'll find look familiar? Practically everyone has gotten one.
- A domain mismatch between "from" and "reply-to" addresses (compare them).
- Weird, strangely incorrect or over-complex use of language.
- Unexplained or illogical urgency. (Why would this email be so urgent?)
- Embedded web links which go to different domains than the human readable text suggests (e.g. a link that appears to go to [www.bank.com](http://www.bank.com) in reality goes to [www.133thacker.org](http://www.133thacker.org) with a fake banking site). Most mail clients now show the real website address when a mouse cursor is hovered over the relevant text.
- Attachments with active content, such as .exe or .html. These are especially risky on platforms that auto-execute content.

### Exercise

- 9.15 Go to <http://www.419eater.com/>. What is scam baiting? Can you find instructions? Can you find *precautions*? This is dangerous stuff. Knowing about it doesn't mean you should do it. But you shouldn't be defenseless either.

## Turn On The Lights

Email content is a wonderful way to get users to click on malicious links. One common tool is the **Blackhole Exploit Kit**. Sounds scary, doesn't it! Can you say "Blackhole Exploit Kit" five times really quick without making any mistakes? Blackhole is a web application exploitation program that takes advantage of known vulnerabilities in Java and Adobe applications. It's used to send a phishing email to users, trying to get the user to click on a compromised web link.

**Phishing** is an attempt to gather important information from a victim by using **social engineering**, persuasive emails sent out to thousands of users. Typical phishing emails appear to come from a well-known and trusted organization. The attackers will use the exact same logo, similar reply email address, and as much professional wording as they can to fool as many people as they can. The email will ask the reader to "verify" or "update" credit card data, personal bank account information and other stuff that you would only give to a trusted source.

When the victim clicks on the official-looking link, the link sends them to a compromised page where the malware payload is installed on their machine. The user is unaware they're being **pwned**, and not all anti-virus programs detect the installation. Once the payload is on the user's computer, the phishing spam sends now control that computer as well as any content they want to collect from it.

Blackhole spam pretends to come from legitimate companies like Amazon, Visa, Twitter, UPS and many organizations that wouldn't raise a user's suspicion. This program is rented



by paying for server time on the Blackhole server. Recent fees ranged from \$50 for a day to \$150 a month.

### Malware, Trojans, And Rootkits, Oh My

The media loves to play up email cracking, because fear sells. (Remember that when anyone who's selling you something tries to scare you.) But the truth is, malware has been around a long, long time.

Dr. Fred Cohen wrote his PhD dissertation on the idea of a computer virus in 1984, published it in 1985, and it was yanked from public viewing a few weeks later. 1985 was a long time ago and the media still acts like malware is a massive new threat to the entire world.

We are not going to tell you every transmittable threat; you can look them up yourself in Lesson 6, Malware. We are going to show you how email security works from both the inside and the outside.

### This Email Looks Legitimate, Let's Open It Up

STOP!!!! Don't open that email just yet. In fact, don't even preview the email. There are several ways email has been and is still used as an attack tool. Social engineering is the preeminent technique to get people to open email, open attachments or click on malicious web links inside an email or message. Social engineering preys on several human emotions we all have including curiosity, wanting to be helpful, trust in our friends, greed and many types of financial or medical concerns. See Lesson 20 for much more about social engineering.

Our curiosity with new or unknown information can be used to persuade us to do stupid things. When you're sent an email that has "Subject: Re: Re: Thanks!" if you're a typical user, you'll want to know why you're being thanked. In this case, you are being thanked ahead of time for opening up an email with a malicious payload.

These types of emails ask you to call a phone number, click on a URL in the message or do anything that can give away your goodies.

### Exercises

Consider an email with this at the top:

```

from:      Mr Norman Chan <naveen.kumar@iitg.ac.in>

reply-to:  2259575299@qq.com

to:        (your email address)

date:      Mon, Nov 19, 2012 at 7:40 AM

mailed-by: iitg.ac.in
  
```

9.16 Would you respond to an email that has this in its subject line: "Hello,I'm Norman Chan,i have a bussiness worth 47.1M USD for you to handle with me?" The sender is "naveen.kumar@iitg.ac.in."

9.17 Investigate this email address to see if Norman Chan owns a business worth 47.1M. Also check the reply-to address, "2259575299@qq.com." DO NOT GO TO QQ.COM.

- 9.18 Max out your browser's security settings before attempting this. Do a little research into qq.com but do not go to this URL. DO NOT OPEN THIS URL. Based on your research of qq.com, would this site raise any alarms for you?

## Exciting Tricks With Email Systems (Hacking the Postman)

Email seems to always play some part when it comes to a security breach or a huge network attack. Every virus, every bit of malware, every phishing event seems to involve email as either a major transport mechanism or a way to enter a system to begin an attack. Email may not be as popular as other communication forms like SMS or IM, but it's what's used the most in the corporate and government world. Now we're going to look closely at the very idea of email and how it can be used as a weapon or a shield.

When mapping a network, we need to know several entry points. If we rely on just one entry point, what do we do when that vulnerability gets patched or fixed? Multiple entry points to a network allow us more freedom to move around that network and give us more avenues of approach. Avenues of approach are a good thing, trust us.

Knowing the user name scheme used by an organization for email or network access gives us a major advantage. Once we know a user name, we can then focus on obtaining that user's password. Most, but not all, organizations use "firstname.lastname@companyname.com. Some use variations of first initial and lastname@companyname.com. Others use the last name followed by the first initial @companyname.com. Pretty lame, right? How much more lame is it when the email address is also the user's login name? This mistake is far too common.

Organizations have a directory within their network that allows users to know who's who and where they work or what they do. That internal directory is a gold mine of information for an attacker. You can look up people in Facebook or other social media to learn more about each user. You can find out when they're going on vacation, what they do, what their hobbies are and other clues to the types of passwords they would use. This information is also valuable if you want to pursue social engineering against that person (for fun and profit).

## **SEAK And Ye Shall Find**

Let's look at gathering email addresses and using email as a hacking tool. Email hacking is closely linked with social engineering, just in case we haven't pointed that out enough already. The **Social Engineering Automation Kit (SEAK)** at <http://www.seak.com.ar/> is designed to use search engines to locate email addresses in a network or on a web site. SEAK is basically a set of Perl scripts that allow search engines to look deep into web pages and networks then report back all the email addresses it finds. SEAK can also be used to locate people in the same way.

SEAK has a brother program at <https://github.com/FreedomCoder/Esearchy-ng>, called **Esearchy**. We didn't name it so don't blame us for the strange name. Esearchy does the same things as SEAK but does it in a Windows environment and searches documents too. Esearchy looks for passwords hidden in metadata, along with any other useful information like email addresses that are available to the public.

Another tool, **Maltego**, is an open-source intelligence and forensics analyzer. It provides tools for discovering data from open sources, and showing that information as a graph, which is handy for link analysis and data mining. The whole point is analyzing real-world relationships between people, groups, websites, domains, networks and online services like your favorite social networks.



An easy tool that you may have heard of is Google search. If you want to see all the employee profile information for a company you can use this command:

```
site:www.google.com intitle:"Google Profile" "Companies I've worked for"
"at company_name"
```

If you want to search all email addresses in a domain or URL, you can use Esearchy. You'd type the following all on one line, substituting a real domain for "company."

```
esearchy -q"@company" -y
AwgiZ8rV34Ejo9hDAsmE925sNwU0iwXoFxBSEky8wulviJqXjwyPP7No9DYdCaUW28y0.i8p
yTh4 -b 220E2E31383CA320FF7E022ABBB8B9959F3C0CFE --enable-bing --enable-
google --enable-yahoo --enable-pgp -m 500
```

**Gpscan** is a Ruby application that can automate this search and produce even more results. When paired with the command above, Gpscan becomes a powerful script tool for reconnaissance and social engineering. You can find Gpscan at <http://www.digininja.org/projects/gpscan.php>.

While you're using any of these, take the time to learn how these tools work. Pay particular attention to the syntax available for each tool and what each one does. You can learn quite a lot about how search engines can be used to hunt for and return email addresses, and possibly a few passwords. Also know which engines are used for the job. Some of the best search engines on the Internet are difficult to find.

## Exercises

9.19 Now it's time for you to research a security tool on your own. Find **FOCA** (the metadata tool). What does it do? Would you want it as an arrow in your quiver?

## Spoofing Versus Malware

In 2007, the CEO of a Fortune 500 company received an email that appeared to be from one of his senior staff. The email's **From:** line showed that the email was sent by a close associate. The **Subject:** line said something like, "How to reduce energy costs." When the CEO opened the email he saw an attachment and a link that also appeared to be legitimate. The CEO opened the attachment and didn't see anything on his screen so he closed the email.

Several months later, the FBI met with that same CEO to tell him that several terabytes of data had been stolen from his company due to a malware infection from the CEO's machine. The FBI confirmed which email had the malware attachment and that "How to reduce energy costs" was the culprit. The message had been spoofed.

This situation happens every day. Your uncle will call you and ask, "Why are you sending me so many email ads?" At school, your buddy keeps getting spam from you for useless products. Why are you sending out all these spam messages?!

You're not.

Your email address has either been spoofed or your computer email program has been hacked. To find out whether an email address was spoofed, you will need to look at the email header of the sent email. We talked about this already. Now let's put your knowledge to work.

Ask anyone who received a spam email from you to FWD it to you, whole, not just by quoting the message. The header will tell you if your email address was spoofed or not. Look at the **Reply** and **Sent** portion of the email. As we already saw in previous exercises, the header will show that the email was either sent by you - or by someone else.



## Stupid Email Tricks

When it comes to privacy, web based email is anything but private. The web service can be asked to provide all of your emails, contacts, calendars and such, if presented with legal documents showing a cause to release that data. One old but still useful trick is to create a web email account using some name other than your own. Whoever you are sending secret emails to has the same access to the account as you. You create your email and save it as a draft. You never send the email, just create a draft of the contents. The email stays in your account but is not traceable since it was never sent. Your partner logs into the same account and read the draft email. Once read, the draft can be deleted or altered to create a new draft email for you to read. It's like Ping-Pong without a ball. The same can be done with a shared Google Doc, by the way.

**NOTE:** You don't get to be the director of the United States Central Intelligence Agency without knowing this little trick.

The email meta data is actually outlined in RFC 2822. Someone thought that it was a good idea to include meta data in email! What medication were they on when they thought this one up? The email meta data can include the following information:

- To
- From
- CC
- BCC
- Date
- Subject
- Sender
- Received
- Message-ID
- References
- Resent
- Return-Path
- Time/Date
- Encrypted
- In-Reply-To

Okay, don't worry too much about RFC 2822 on email meta data. The reason is, the RFC only covers electronic text traffic. Your SMS texts, all your Instant messages, and those photos you shared contain this wonderful invasion of your privacy, as well. However, this hidden data is covered under a different RFC. Listen to the sound of the evil scientist laughing once again in the background. "Wha ha, ha, ha, snort, ha, ha, ha, snort, ha, ha ha - Igor get me a tissue."

## Outsmarting The Email Bots (Email Obfuscation)

This is so simple that you will laugh at yourself for not thinking of this earlier.

When you need to send your email address to someone else or to some account or some other crazy reason; are you going to send it in plain text? If you do, you're opening





yourself up to spambots. These spambots are vicious creatures that have mother issues and they slime their way through the Internet looking for email addresses.

It's like connecting a brand new computer to the Internet without enabling any security features: An email address sent in plain text is just asking for trouble. You might as well connect a J-2 crossover to the Viper 115 ROJ input module inside a .002 latent relay link with a Blast Ion K-0a801 inside a quad reverberator! How messed up is that! Bad idea, very bad idea.

To outsmart these, you might want to try altering your email address when you send it. As with anything else, there is a trade-off between ease of use and security. There are several techniques, just use your imagination.

```
somebodyatsome.whereelse
```

```
Somebody@somedotwhereelse
```

```
somebody2some.whereelse
```

These have been used successfully to pass on an email address and bypass those weak bots. We'll see how long that lasts.

### Exercises

9.20 Visit this URL to take a look at Etherios EasyDescribe:

```
http://appexchange.salesforce.com/listingDetail?listingId=a0N300000018leZEAQ
```

This is a free metadata viewer/extractor. Grab some of those emails you have stashed away and run them through Etherios. What metadata is in those emails that is not in the header?

9.21 If some data is not in the header, where could that metadata be hidden in the email?

9.22 Does RFC 2822 require metadata to be embedded in electronic text or is it just a standardized method for Internet email traffic?

9.23 Using whatever means you think is best, try and locate the correct business email address of the three company CEOs listed below. Hint, first find out who they are.

Coca Cola

Kia

British Aerospace Engineering (BAE)



## Conclusion

---

Now that you are thoroughly educated (or confused) about email, you can quickly see that this simple communication tool is not very simple at all. The way email works through systems can become quite complex and requires you to meet certain criteria. Remember how we sent you as an email through the typical send, route and receive process. You did pretty well, too. You might want to consider a career as an email. Just a thought.

Email etiquette is important since writing and sending off an email written when you were angry or upset could cause you trouble later on. Stop sending everyone CC'd replies. If you are going to respond to a bunch of folks, use BCC to protect others' email address privacy.

Along the same lines of protecting privacy, we discussed using encryption tools like PGP or GPG to send and receive emails away from prying eyes. The cool part of that section was actually creating a key to use for real. It wasn't all that painful was it? If you answered "yes," sorry. We are certain that the local fast food restaurant is still hiring, since security just isn't your thing. We hope that you didn't answer "yes," because we need as many security experts as we can get.

After that section, we went into heavy security topics. Ok, maybe they weren't all that hard core but we thought you would enjoy some of it. You have to admit that email server and client-side vulnerabilities and threats were fun! Well, we had fun writing it. We got to look at spam, spam and more spam. We know that it eats up valuable bandwidth so you need to filter it as soon as possible in the routing chain. We also know that the state of Hawaii consumes more real Spam annually than any other state or country. They like their Spam.

Dig is an important email tool for Linux and Unix users for tracking down specific information. If you see large chunks of data leaving your network as email attachments, you know to take a closer look to ensure company secrets aren't part of those emails. One interesting point we discussed was the use of Blackhole server to exploit vulnerabilities within networks. This tool was been widely used for sending malware in email, knowing that someone in the domain is likely to open that email or click on the infected link within that electronic document. This kind of threat can be stopped by filtering email traffic and educating users on this issue. User education is a key element in security.

At this point in a conclusion you would get some advice that may or may not interest you. Not here. Just know that email security is a challenge to cyber security. How you view that fact depends on which side of the fence you're on.



## The Ultimate Disclaimer

Contributor Peter Houppermans, author of *The Evil Guide to Privacy*, writes:

If you ever feel the need to highlight the folly of email disclaimers, please find here one Peter Houppermans cobbled together from very old USENET posts together with some of his own. A critical part of being a successful hacker is having a functional sense of humor: not only does it make you unpredictable, it also provides armor against those times when IT gets the better of you. Laugh, dust yourself off and rejoin the fight. Computers cannot win; we'll unplug them.

*The opinions expressed herein are those of the author, do not necessarily represent those of his employer or someone else, are probably silly, and have nothing to do with the recent consumption of liquor to the value of the GDP of a small country.*

*The information in this email and any attachments is purely nonsensical and unlikely to be politically correct. It is intended solely for the attention of world + dog and its mother. I sincerely believe it's totally pointless to tell you what to do with this email if I managed to send it to the wrong place. This email does not contain nudity (yet), and no cuddly animals or whales were hurt during its production as there weren't any available. In true consulting style this email has been composed entirely from recycled keystrokes and cut-n-paste from many, many other leaving emails and other versions to other audiences (minus the embarrassing parts - I take PayPal, hint). May harm the digestive system if swallowed (especially when printed on cardboard), batteries not included. Author may sue, contents may settle. For more culture, add yoghurt. Unsuitable for people under 18 or those lacking an operable sense of humour. Do not hold upside down, open other end. If you received this email in error, well done.*

*Any attachments should not be trusted or relied upon, but may prove highly entertaining.*

*This disclaimer is meant for educational purposes only. Send no money now. Ask your doctor or pharmacist. To prevent electric shock, do not open back panel. No user serviceable parts inside. You may or may not have additional rights which may vary from country to country. Not recommended for children under twelve years of age. Batteries not included. Limit 1 per customer. Does not come with any other figures. Any resemblance to real persons, living or dead, is purely coincidental. Keep away from open flame or spark. Void where prohibited. Some assembly required. All rights reserved. List each check separately by bank number. Contents may settle during shipment. Use only as directed. Parental discretion advised. No other warranty expressed or implied. Unauthorized copying of this signature strictly prohibited. Do not read while operating a motor vehicle or heavy equipment. Postage will be paid by addressee. In case of eye contact, flush with water. Subject to approval. This is not an offer to sell securities. Apply only to affected area. May be too intense for some viewers. Do not fold, spindle, or mutilate. Use other side for additional listings. For recreational use only. Shipping and handling extra. No animals were harmed in the production of this signature. Do not disturb. All models over 18 years of age. If condition persists, consult your physician. Freshest if consumed before date on carton. Prices subject to change without notice. Times approximate. No postage necessary if mailed in Singapore. If swallowed, do not induce vomiting. Breaking seal constitutes acceptance of agreement. For off-road use only. As seen on TV. We reserve the right to limit quantities. One size fits all. Do not leave funds without*



collecting a receipt. Many suitcases look alike. Contains a substantial amount of non-active ingredients. Colours may, in time, fade. We have sent the forms which seem to be right for you. Slippery when wet. This product is only warranted to the original retail purchaser or gift recipient. For office use only. Net weight before cooking. Not affiliated with the Red Cross. Surfaces should be clean of paint, grease, dirt, etc. Drop in any mailbox. Edited for television. Keep cool; process promptly. \$2.98/min AE/V/MC. Post office will not deliver without postage. Simulated picture. List was current at time of printing. Penalty for private use. Return to sender, no forwarding order on file, unable to forward. Do not expose to direct sunlight. Not responsible for direct, indirect, incidental, or consequential damages resulting from any defect, error, or failure to perform. No Canadian coins. Do not puncture or incinerate empty container. See label for sequence. Prices subject to change without notice. Do not write below this line. Time lock safe, clerk cannot open. At participating locations only. Serial numbers must be visible. Align parts carefully, then bond. Falling rock zone. Keep out of reach of children. Lost ticket pays maximum rate. Your cancelled check is your receipt. Check paper path. Place stamp here. Avoid contact with skin. Sanitized for your protection. Be sure each item is properly endorsed. Penalty for early withdrawal. Sign here without admitting guilt. No solicitors. Slightly higher west of the Mississippi. Storage temperature: -30 C (-22 F) to 40 C (104 F). Employees and their families are not eligible. Beware of dog. Contestants have been briefed on some questions before the show. No purchase necessary. Limited time offer, call now to ensure prompt delivery. You must be present to win. No passes accepted for this engagement. Extinguish all pilot lights. Processed at location stamped in code at top of carton. Shading within a signature may occur. Use only in well-ventilated areas. Replace with same type. Accessories sold separately. Booths for two or more. Check here if tax deductible. Keep away from fire or flame. Some equipment shown is optional. Price does not include taxes. Hard hat area. Pre-recorded for this time zone. Reproduction strictly prohibited. Adults 18 and over only. Detach and keep for your reference. No alcohol, dogs, or horses. Demo package, not for resale. List at least two alternate dates. First pull up, then pull down. Call toll free before deciding. Driver does not carry cash. Some of the trademarks mentioned in this product appear for identification purposes only. Record additional transactions on back of previous stub. This supersedes all previous notices. Tag not to be removed under penalty of law.

*If you've read all the way to here you're either exceptionally curious. Or a lawyer.*

Today's teens are in a world with major communication and productivity channels open to them and they don't have the knowledge to defend themselves against the fraud, identity theft, privacy leaks and other attacks made against them just for using the Internet. This is the reason for Hacker Highschool.

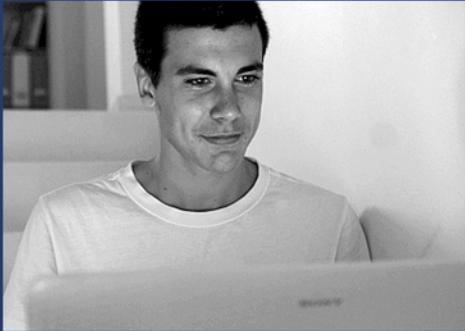
**The Hacker Highschool project is the development of security and privacy awareness learning materials for junior high and high school students.**

Hacker Highschool is a set of lessons and a practical means of making hackers. Beyond just providing cybersecurity awareness and critical Internet skills, we need to teach the young people of today how to be resourceful, creative, and logical, traits synonymous with hackers. The program contains free security and privacy awareness teaching materials and back-end support for teachers of accredited junior high, high schools, and home schooling. There are multiple workbooks available in multiple languages. These are lessons that challenge teens to be as resourceful as hackers, including safe Internet use, web privacy, researching on the internet, avoiding viruses and Trojans, legalities and ethics, and more.

**The HHS program is developed by ISECOM, a non-profit, open-source research group focused on security awareness and professional security development and accreditation.**

# Hacker HighSchool

## SECURITY AWARENESS FOR TEENS



## LESSON 10 WEB SECURITY AND PRIVACY



HACKING IS LEARNING  
[www.hackerhighschool.org](http://www.hackerhighschool.org)

ISECOM

Creative Commons 3.3 Attribution-Non-Commercial-NoDerivs ISECOM

[WWW.ISECOM.ORG](http://WWW.ISECOM.ORG) - [WWW.OSSTMM.ORG](http://WWW.OSSTMM.ORG) - [WWW.HACKERHIGHSCHOOL.ORG](http://WWW.HACKERHIGHSCHOOL.ORG) - [WWW.BADPEOPLEPROJECT.ORG](http://WWW.BADPEOPLEPROJECT.ORG) - [WWW.OSSTMMTRAINING.ORG](http://WWW.OSSTMMTRAINING.ORG)



## WARNING

The Hacker Highschool Project is a learning tool and as with any learning tool there are dangers. Some lessons, if abused, may result in physical injury. Some additional dangers may also exist where there is not enough research on possible effects of emanations from particular technologies. Students using these lessons should be supervised yet encouraged to learn, try, and do. However ISECOM cannot accept responsibility for how any information herein is abused.

The following lessons and workbooks are open and publicly available under the following terms and conditions of ISECOM:

All works in the Hacker Highschool Project are provided for non-commercial use with elementary school students, junior high school students, and high school students whether in a public institution, private institution, or a part of home-schooling. These materials may not be reproduced for sale in any form. The provision of any class, course, training, or camp with these materials for which a fee is charged is expressly forbidden without a license, including college classes, university classes, trade-school classes, summer or computer camps, and similar. To purchase a license, visit the LICENSE section of the HHS web page at <http://www.hackerhighschool.org/licensing.html>.

The Hacker Highschool Project Project is an open community effort and if you find value in this project, we ask that you support us through the purchase of a license, a donation, or sponsorship.



## Table of Contents

WARNING.....	2
Contributors.....	4
Introduction and Objectives.....	5
How the Web Really Works.....	7
Reality Check.....	7
Security - Here to Save the Day.....	8
Game On: Who Hacked The Website?.....	9
Getting More Details.....	11
Mightier and Mitre-er.....	13
Browsing Behind Bars: SSL.....	13
Applied SSL.....	14
The Properties of Trust.....	15
Employing a Middleman: Proxies.....	16
Using Someone Else's Server.....	16
Using a Local Proxy.....	16
The Onion Router.....	16
SQL Injection - The Lion.....	22
Buffer Overflows - The Tiger.....	22
Cross Site Scripting (XSS) - The Bear.....	22
Bear Traps and Tiger Rugs.....	23
Using a Local Proxy for Web Application Testing.....	24
Building Secure Web Applications.....	27
Protecting Your Server.....	28
Feed Your Head: Understanding HTTP.....	33
Going Deeper.....	34





## Contributors

---

Pete Herzog, ISECOM  
Marta Barceló, ISECOM  
Chuck Truett, ISECOM  
Kim Truett, ISECOM  
Marco Ivaldi, ISECOM  
Bob Monroe, ISECOM  
Greg Playle, ISECOM  
Cor Rosielle, ISECOM  
Mario Platt  
Monique Castillo

**ISECOM**



## Introduction and Objectives

---

What you do on the World Wide Web (the Web) is your own business, isn't it? You might think so, but it's just not true. What you do on the web is about as private and anonymous as where you go when you leave the house. You'd think your comings and goings are your own business, and the people of ISECOM would agree with you. But in most countries government and private investigators are free to follow you around, collecting data on where you go, whom you meet, and what you say. The focus of this lesson is learning how to protect yourself on the web and to do that, you will have to know where the dangers are.

Web security is a whole lot more than making sure your browser doesn't collect tracking cookies. In this lesson we hope to show you how deep the web actually goes. When most people think of the Internet they think of the web. Since you have already read some of the other lessons, you know that this isn't true. The world wide web is just a small fraction of the digital communication structure we know as the Internet. With web security we have to look at web servers, the software running on both the client and the server-side, as well as all the services we depend on. Our phones are the most connected devices ever made, yet few people understand the difference between Java and Javascript.

The code of most web applications is no longer just Hyper Text Mark Up Language (HTML), either. Entire operating systems are being used to enhance your online experience to sell you things or just collect personal information about you. Those pictures you post to your friends on social media are being used by other entities. You are unknowingly part of the largest privacy invasion ever created. And you are the one providing all the information.

Our focus here is not to scare you but rather educate you about how far web security really goes. We will cover the foundation of web applications, their uses and misuses, vulnerabilities, security practices plus some tricks of the hacking trade. As always, Hacker Highschool is not here to teach you criminal techniques. We endeavor to show you how to be security conscience. You must think on your own and learn on a continuous basis about the world around you.

Your phone, computer, tablet, vehicle and even home appliances are connecting to the internet whether you know it or not. Wouldn't you like to know what they are saying about you along with the data that is being collected? We hope to show you ways to protect your privacy and personal data. Perhaps you will take up the challenge and begin teaching others as the Institute for Security and Open Methodologies (ISECOM) does. Education is a powerful weapon.



## Fundamentals of Web Security

The focus of this lesson is learning how to protect your privacy on the web and how to keep your own web sites safe against intelligent attacks. To do that, you will have to know where the dangers are and how to bypass those dangers. Our topic here is going to expose you to the massive security challenges facing web site owners. You probably heard about attacks such as SQL injections, buffer overflows, cross site scripting and more. This lesson is going to demonstrate what these attacks really mean to you as well as methods to exploit/defend each web based vulnerability.

First off, you need to understand that the original idea behind the Internet was to link academic organizations with certain **U.S. Department of Defense (DoD)** organizations. This basically meant that the Internet (**ARPANET**) was built on the premise that security wasn't needed, since you already had to be either a research university or a government entity to connect to the network. The original Internet linked a bunch of big computers all over the U.S. to each other so security wasn't really an issue, the availability and durability of these connections were some of the real concerns.

This is where the **Transport Control Protocol (TCP)** was first introduced to deliver packets of data from one big (really big, like we are talking about a computer the size of your garage) computer to another big computer far away. Way back then, nobody thought the Internet would ever get as large as it has today. **Internet Protocol V4 (IP as in TCP/IP v4)** addressing used a 32 bit number sequence which limited addresses to a specific amount of addressable devices (4,294,967,296 addresses, to be exact). Remembering that back then there were only so many universities and defense organizations that would need an IP address. The idea was to have a connection of computers that would provide information to anyone connected to it. The logic back then was "why would anyone ever want to limit the freedom of information exchange?"

Along came private companies that connected to the Internet. Soon, others followed and more and more became all part of the Internet we see today. However, since security wasn't part of the original plans for the Internet, we have many security challenges that keep us security experts nicely employed. Keep reading and you will have the skills you need to earn a good paying job as well.

You may have heard the phrase "Keep It Simple, Stupid", or KISS for short when it comes to planning or performing a task. One fundamental law of engineering is that "complexity causes failures" so you want to try and keep your design planning as simple as possible. In security, we just shorten the whole phrase to "Complexity Breeds Insecurity".

The **World Wide Web** has evolved into an extremely complex system of interconnected computers, weird wires & cables, sloppy software, and a few really bad people who want to take things that do not belong to them. This whole mess we think of as the web sits poorly balanced on top of an ancient architecture built when security was not an issue nor was it designed into the original plans. Think of the web as a single car train with millions of other train cars stacked onto of each other's roof, each using different construction material, with the original train half leaning off the tracks.

Yes, security professionals have our work cut out for us. You can be sure that employment in the security field will continue to rise in the years to come. Why? Because the Internet is a bullet train moving at light speed and expanding even faster. Too many people depend on the Internet each and every day to allow engineers time to rebuild the Internet from scratch. We can't repair the train's engine while the whole thing is still moving, so we lubricate the wheels a bit or we dust off some gauges. The Internet train has to continue running until the neighbors build some other new Internet using spare parts from the one we have now. Until then, work on your resume.



## How the Web Really Works

---

The web seems pretty simple: you get onto the Internet, open a browser, type in a website URL, and the page appears. But the devil's in the details, and some of them can hurt you. So how does the web really work?

A quick trip to the fine folks who make standards for the web, the **World Wide Web Consortium (W3C, at <http://www.w3.org>)**, will teach you all you want to know about how the web works. Don't miss the history of the web at <http://www.w3.org/History.html>. The problem seems to be that definitions and standards might teach you how to be safe. But, probably not. The people who want to hurt you don't follow standards or laws.

### Reality Check

So what really happens when you go to a website? Assuming you are already connected to the Internet, here's how it works step by step:

1. You open your browser (Explorer, Firefox, Chrome, Opera, Safari, or any number of other browsers out there).
2. You type in the **Uniform Resource Locator (URL)** into the browser's address line (the website address, like ISECOM.org; the real nerds often call them URI for **Uniform Resource Indicators**).
3. The website's URL is saved in the browser's history on the hard disk. Yes, there's a record of everywhere you've been, right there on your hard drive.
4. Your computer asks your default **Domain Name Server (DNS)** to look up the IP address of the website. The DNS connects the name [www.ISECOM.ORG](http://www.ISECOM.ORG) to the IP address of 216.92.116.13. Use [www.whois.net](http://www.whois.net) to locate detailed information on a web page and try a Reverse IP lookup to find the actual numbered address instead of the site names.
5. Your computer connects to the website's server, at the IP address it was given by the DNS, to TCP port 80 for "**http://**" websites, or TCP port 443 if you go to an "**https://**" secure web site. If you use **https://** there are more steps like getting server certificates that we won't cover in this example.
6. Your computer requests the page you ask for (like History.html), or if you specify a folder the web server sends a default page, usually index.html. It's the server that decides this default file, not your browser.
7. Your IP address, the web page you are visiting and details about your browser are likely to be stored on the web server and/or proxy servers in between (see below).
8. The requested web page is stored in the browsers **cache**. Yes, there's a copy of every page you've visited, right there on your hard drive, unless the web server explicitly asks your browser not to store it (cache is often disabled for protected web pages served through HTTPS, or at least it should be).
9. Most web pages contain other elements, like pictures, ads, style sheets (instructions to your browser on how the page should be displayed), **Javascript** (little programs, to do checks or make the web page look fancy and smooth). All these elements are retrieved in a similar manner as typing a URL by yourself.
10. The browser nearly instantaneously shows you what it has stored in your browser's cache. There's a difference between "perceived speed" and "actual speed" during your web surfing. This is the difference between how fast something is downloaded (actual) and how fast your browser and computer can render the page and graphics to show them to you (perceived). Remember: Just because you didn't see web page elements doesn't mean it didn't end up in your browser's cache.



The World Wide Web (web) is a massive **client-server network**. Clients are typical users who run web browsers to display or capture Internet data. Servers are web servers, such as Internet Information Server (IIS) on Windows or Apache on Unix/Linux. So the browser asks for a page, and the web server returns content in the form of **Hyper Text Markup Language (HTML)** pages.

## Security - Here to Save the Day

---

Where does security come in? Well, if you're running a public web server, it's sort of like a retail store window. The front window is a great place to advertise and display your goods, however you don't want that store window left open so passers-by can take things for free. Ideally, you'd make sure that if someone throws a brick, your window doesn't shatter, either! Unfortunately, web servers operate complex software, and you can't have complex software without getting some bugs as well. Less scrupulous members of society will exploit these vulnerabilities to access data they shouldn't have access to (like credit card information, your medical records, and those pictures of you with the black eye that your little sister gave you). More on these vulnerabilities later.

Your browser *will* have vulnerabilities as well, and to make things worse, the creators of your web browser has its own agenda. Some bad Web sites can inject malicious code into your browser, or employ malicious links that can infect your computer. And let's not forget social media sites that watch your every move as well as collect every picture you post and every word you type. Last year, one specific social media site sold every bit of data they collected on its users to seven different companies. That social media site didn't lose FACE with their users though, since this massive sale of personal data wasn't listed in many BOOKS at that time.

### Exercises

- 10.1 Locate your browser's cache and temp files. Examine the directory and the files inside. What is in those directories? How could this data be used against you? Is there a way to prevent the browser from caching files, or to clear the temp and cache when you close the browser?
- 10.2 Where can you find your browser's history file? Look at that history. What does it tell you about yourself? Is there a way to prevent the browser from recording this history? How can you clear this history?
- 10.3 Most on-line businesses collect a lot of information about you, which may seem harmless yet it becomes a privacy concern over time. Install the Firefox add-on called Ghostery. What does it do? What does it tell you?
- 10.4 Locate your favorite web sites and try to learn as much as you can about them. Who owns those sites? Where are the web servers located (physical location)? How long have those IP addresses been registered and who is listed as the Registrant Name?
- 10.5 When does your favorite web site domain name expire and can you purchase that domain name when it expires?



## Game On: Who Hacked The Website?

"It's not like you were going to be the only person with access to the police web site. I've known you for years now and I don't even trust you with my electronics," Mokoia reasoned with Jace. He subconsciously braced himself next to the junky old metal police desk just in case Jace decided to smack him again. That previous hit almost cost them their relationship. Luckily Jace had learned the value of their friendship then, and she wasn't about to screw that up again, ever.

Without looking up from the monitor Jace replied, "Agreed, but I didn't give anyone here any password information about their web site. I just set it up and figured that they wanted me to maintain it. Hell, I built their entire network here for free. Nobody ever asked me for admin rights or passwords except for individual user accounts. I am root."

As the two teens crowded into the former police interview room recently converted into a server area, the puzzled hacker behind the keyboard was more interested in recovering the web site post-attack. Mokoia was concerned about who had accessed the police web console and created a back-up in the public directory. Jace was working in the digital world while Mokoia was thinking in the real world. They were a good team.

A blast of fresh air hit the small room as Officer Hank pushed into the room with the teens. "Hey Jace, how's it coming along and who is this handsome guy next to you? Do I need to frisk him for weapons, like chocolate or flowers?" Hank asked in a fatherly manner.

Mokoia stood up in reaction to the unexpected police officer standing over him. Actually, Officer Hank was towering over Mokoia. Hank held out his meaty hands to either shake Jace's friend's hand or place it manly on his shoulder. Mokoia started to introduce himself, "Hi I'm Mo."

"Mokoia, yeah. I've seen you a few times. Jace tells me all about you. I think she likes you," Hank gave a light jab at Jace's shoulder to soften his joke about her liking Mokoia.

"Shut up you two, I'm trying to work here," Jace growled back. Both guys stepped back from the cranky girl and shared a laugh at her reaction.

Jace confronted Officer Hank while looking at the screen and not at him, "You wanna tell me who in this building has admin access to this network besides me? Or more to the point, who screwed up the web site? I worked my butt off trying to make this network and web content perfect for you cops and all I get is this mess."

The burly police officer interrupted Jace, "Hold on their Miss High and Mighty. First off, you forgot to give anyone here the passwords to update the site. Secondly, we needed to add more information to our network. And C, you haven't been around lately to help make those changes happen. So you need to just chill out and remember that I've got a wooden baton and you don't."

*Jace, put in her place, Mokoia thought with a slim smile. By a cop, too. I like this guy.*

The flustered teen flopped back into the creaky old desk chair. She explained what she knew about how the web attack happened, what she had to restore and how the issue needed to be corrected. Hank squatted down next to her listening with interest. Every now and then, Mokoia added additional details.

Officer Hank explained his situation, "The chief wants dynamic content as part of the public web site. He wants things like traffic web cams, weather reports, most-wanted criminals, community volunteer patrol details and all kinds of stuff. He wants our web to give the users real-time information about events as they happen."

Jace sat leaning against the dilapidated police interrogation desk to absorb all this.



She asked, "Okay, we can do all that. Now I need to know how you guys want me to implement the dynamic content. We could use Ruby on Rails, PHP, jQuery and HTML5 with push or pull data. What's your flavor?"

Hank just started back at Jace, blank look on his face.

A long pause, too long for Mokoia, was broken when Jace tilted her slender head and continued the conversation in a slower voice, "There are plenty of security concerns with any dynamic content we use. First, we have the top three application level issues. They sound like a crappy movie script."

Mokoia looked away. Jace spun around to face the computer screen. "Our biggest three issues with dynamic content are SQL injection, cross-site scripting and leaving unnecessary services running inside web applications. SQL injection is passing SQL code into an application. Potential attack strings are built from fragments of SQL syntax. They get executed on the database server side if the Web application doesn't screen out potential code. For example, we can have problems if we don't protect ourselves against malicious input like a single-quote character, which could close the SQL string and give the attacker unintended system and application access."

Without taking a breath, Jace continued, "The easiest way to see how this works is to type this string for the e-mail address and password in a web form input:

```
' OR '1' = '1
[note that there must be a space at the end]
```

"The condition will be satisfied by the always-true condition '1' = '1', the SQL statement will show us everything in the current table and the login will succeed."

Jace tossed her hair out of her face and pressed forward: "In order to eliminate this vulnerability, we need to pass the name and password to the database in a way that special characters aren't interpreted as part of the SQL command. The easiest way to do this is to avoid constructing ad hoc SQL statements, and instead pass the e-mail address and password as parameters to a stored procedure. Basically we tell the SQL database what an email address should look like and only accept valid inputs."

Hank and Mokoia hated feeling stupid around Jace but that was something they were both getting used to. The police officer tried, "Great to hear that you know so much about these web security problems. But, can you fix all this stuff and give the chief the type of web site he wants?"

She squared up her shoulders like a soldier given a proper order. Jace answered back, "Hank, I'm on it. Don't you worry. I'll take care of everything, dude."

Hank smirked and thought, *That's what I'm worried about, dude.*

## Game Over

### Rattling the Locks

Assume you have put up your own web server to host your blog, personal website, photos and whatnot. How could all that information be exploited and used against you? Let's find out, shall we?

Standard HTML pages are transferred using **Hyper Text Transfer Protocol (HTTP)**. It's a simple text-based system for connecting to a server, making a request and getting an answer. This means that we also can connect easily to a server using command line tools like **telnet** and **netcat**, and get information about what software is running on a specific server. Look at what you get when you run this simple two-line command:

```
netcat isecom.org 80      [now press <Enter>]
HEAD / HTTP/1.0          [now press <Enter> twice]
```

```

HTTP/1.1 200 OK
Date: Wed, 29 Feb 2012 23:25:54 GMT
Server: Apache/2.2.22
Last-Modified: Tue, 07 Feb 2012 18:41:18 GMT
ETag: "3dad-4b8641fe22f80"
Accept-Ranges: bytes
Content-Length: 15789
Identity: The Institute for Security and Open Methodologies
P3P: Not supported at this time
Connection: close
Content-Type: text/html
  
```

Every web server and version will return different information at this request – an IIS server will return the following:

```

netcat www.microsoft.com 80
HEAD / HTTP/1.0

HTTP/1.1 200 OK
Connection: close
Date: Fri, 07 Jan 2005 11:00:45 GMT
Server: Microsoft-IIS/6.0
P3P: CP="ALL IND DSP COR ADM CONo CUR CUSo IVAo IVDo PSA PSD TAI TELo
OUR SAMo CNT COM INT NAV ONL PHY PRE PUR UNI"
X-Powered-By: ASP.NET
X-AspNet-Version: 1.1.4322
Cache-Control: public, max-age=9057
Expires: Fri, 07 Jan 2005 13:31:43 GMT
Last-Modified: Fri, 07 Jan 2005 10:45:03 GMT
Content-Type: text/html
Content-Length: 12934
  
```

## Getting More Details

You can take this further and obtain more information by using the “OPTIONS” modifier in the HTTP request:

```

netcat isecom.org 80      [now press <Enter>]
OPTIONS / HTTP/1.0      [now press <Enter> twice]

HTTP/1.1 200 OK
Date: Fri, 07 Jan 2005 10:32:38 GMT
Server: Apache/1.3.27 Ben-SSL/1.48 (Unix) PHP/4.2.3
Content-Length: 0
Allow: GET, HEAD, POST, PUT, DELETE, CONNECT, OPTIONS, PATCH, PROPFIND,
PROPPATCH, MKCOL, COPY, MOVE, LOCK, UNLOCK, TRACE
Connection: close
  
```

This gives you all of the HTTP commands (or “methods”) to which the server will respond in a specific directory (in this case “/”, the root directory of the web server or “document root”).

Notice that HTTP is completely “in the clear”; everyone can see what you’re browsing for. Consider how this might be changed; look up and learn about “secure searching”.

Doing all of this by hand is rather tedious, and matching it manually against a database of known signatures and vulnerabilities is more than anyone would want to do. Fortunately for us, some very enterprising people have come up with automated solutions like **nikto**.

Nikto is a **Perl** script that carries out various tests automatically. It runs a scan and provides a detailed report:

```

./nikto.pl -host www.hackerHigh School.org
- Nikto v2.1.5
  
```



```

-----
+ Target IP:          216.92.116.13
+ Target Hostname:    www.hackerHigh School.org
+ Target Port:        80
+ Start Time:         2012-03-20 13:33:49 (GMT-6)
-----
+ Server: Apache/2.2.22
+ ETag header found on server, fields: 0x2f42 0x4b8485316c580
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS
+ /cgi-sys/formmail.pl: Many versions of FormMail have remote
vulnerabilities, including file access, information disclosure and email
abuse. FormMail access should be restricted as much as possible or a more
secure solution found.
+ /cgi-sys/cgiwrap/~bin: cgiwrap can be used to enumerate user accounts.
Recompile cgiwrap with the '--with-quiet-errors' option to stop user
enumeration.
+ /cgi-sys/cgiwrap/~daemon: cgiwrap can be used to enumerate user accounts.
Recompile cgiwrap with the '--with-quiet-errors' option to stop user
enumeration.
+ /cgi-sys/cgiwrap/~ftp: cgiwrap can be used to enumerate user accounts.
Recompile cgiwrap with the '--with-quiet-errors' option to stop user
enumeration.
+ /cgi-sys/cgiwrap/~mysql: cgiwrap can be used to enumerate user accounts.
Recompile cgiwrap with the '--with-quiet-errors' option to stop user
enumeration.
+ /cgi-sys/cgiwrap/~operator: cgiwrap can be used to enumerate user
accounts. Recompile cgiwrap with the '--with-quiet-errors' option to stop
user enumeration.
+ /cgi-sys/cgiwrap/~root: cgiwrap can be used to enumerate user accounts.
Recompile cgiwrap with the '--with-quiet-errors' option to stop user
enumeration.
+ /cgi-sys/cgiwrap/~sshd: cgiwrap can be used to enumerate user accounts.
Recompile cgiwrap with the '--with-quiet-errors' option to stop user
enumeration.
+ /cgi-sys/cgiwrap/~uucp: cgiwrap can be used to enumerate user accounts.
Recompile cgiwrap with the '--with-quiet-errors' option to stop user
enumeration.
+ /cgi-sys/cgiwrap/~www: cgiwrap can be used to enumerate user accounts.
Recompile cgiwrap with the '--with-quiet-errors' option to stop user
enumeration.
+ /cgi-sys/cgiwrap/~OYG3G: Based on error message, cgiwrap can likely be
used to find valid user accounts. Recompile cgiwrap with the '--with-quiet-
errors' option to stop user enumeration.
+ /cgi-sys/cgiwrap/~root: cgiwrap can be used to enumerate user accounts.
Recompile cgiwrap with the '--with-quiet-errors' option to stop user
enumeration.
+ /cgi-sys/cgiwrap: Some versions of cgiwrap allow anyone to execute
commands remotely.
+ /cgi-sys/Count.cgi: This may allow attackers to execute arbitrary
commands on the server
+ OSVDB-3092: /readme.txt: This might be interesting...
+ OSVDB-3093: /cgi-sys/counter-ord: This might be interesting... has been
seen in web logs from an unknown scanner.
+ OSVDB-3093: /cgi-sys/counterbanner: This might be interesting... has been
seen in web logs from an unknown scanner.
+ OSVDB-3093: /cgi-sys/counterbanner-ord: This might be interesting... has
been seen in web logs from an unknown scanner.
+ OSVDB-3093: /cgi-sys/counterfiglet-ord: This might be interesting... has
been seen in web logs from an unknown scanner.
+ OSVDB-3093: /cgi-sys/counterfiglet/nc/: This might be interesting... has
been seen in web logs from an unknown scanner.
+ 6474 items checked: 1 error(s) and 22 item(s) reported on remote host
+ End Time:          2012-03-20 13:50:22 (GMT-6) (993 seconds)
-----
1 host(s) tested

```



**Note:** Almost every one of these lines represents a possible vulnerability or exploitable code. Using various options you can fine tune nikto to do exactly what you need, including stealth scans, mutation and cookie detection.

## Mightier and Mitre-er

Finding a vulnerability is all well and good but what you do with that information is a whole different story. Security professionals will take the scan results of their own web servers and patch, update, remove, repair or do whatever they need to in order to close each vulnerability. The nice folks over at **Mitre.org** operate several databases (<http://mitre.org/work/cybersecurity.html>) that collect and catalog every known vulnerability you could imagine.

These databases are given scary names like “**Common Weakness Enumeration (CWE)**” and “**Common Vulnerabilities and Exposures (CVE)**” yet they are fairly simple to operate. These systems are a collection of other tools and data with a search engine built into each database. Each data repository is focused on different aspects of hardware, software, services, system configurations, and compliance requirements.

Looking at the results nikto gave us earlier, you can see near the bottom of the log are the letters **OSVDB** followed by a bunch of numbers. OSVDB stands for the **Open Source Vulnerability Database** located at [OSVDB.org](http://OSVDB.org). In the log results from nikto the numbers after OSVDB identify a specific type of vulnerability.

## Exercises

10.6 Install **netcat** on a Linux virtual machine or use a live Linux CD/DVD to boot up your computer:

```
sudo apt-get install netcat [for Debian/Ubuntu family]
```

or

```
sudo yum install netcat [for Red Hat/Fedora family]
```

10.7 Install nikto on your Linux virtual machine:

```
sudo apt-get install nikto [ for Debian/Ubuntu family]
```

or

```
sudo yum install nikto [for Red Hat/Fedora family]
```

10.8 Repeat the experiments above, targeting [www.hackerhighschool.org](http://www.hackerhighschool.org).

**Note:** Do not try this on any other public or private web server, including your school's. Bad, bad idea.

## Browsing Behind Bars: SSL

When the web started to take off, it wasn't long before everyone realized that HTTP in plain text wasn't good for security. The next variation was to apply encryption to it. This came in the form of **Secure Sockets Layer/Transport Layer Security (SSL/TLS**, simply called **SSL**), a cryptographic suite encompassing secure ciphers implementing 40 to 128 bit (or more) symmetric key encryption methods. A 40 bit key is not as secure than a 128 bit key, and with specialized hardware, 40 bit is breakable within a reasonable period of time, like during a lunch break (okay, maybe a bit more than that). The 128 bit key will take much longer: cracking it with only brute force will require somewhere between a trillion years and the total age of the universe. A streaming cipher suite like RC4 only offers protection for about the square root of the key space, or half the length of the key; so a 128 bit key

offers protection of 64 bits, which can be cracked on a modern PC in relative short time – think days. One thing to remember is that the stronger the key algorithm you use, the longer it will take to encrypt and decrypt code. Use encryption sparingly or only apply as much strength as you need for web surfing.

Along with SSL, an open source version is available called “**OpenSSL**” and can be found at [openssl.org](http://openssl.org). OpenSSL works alongside Transport Layer Security to provide an entire library of cryptographic recipes. OpenSSL is a command line tool with many options to work with. Turn to <http://www.openssl.org/docs/apps/openssl.html> for all the latest information on the command interface and library updates.

For known HTTPS attacks there are more complex approaches using something called a **known cyphertext attack**. This involves calculating the encryption key by analyzing a large number of messages (over a million) to deduce the key. Along with **cyphertext**, you will find multiple attack methods that are discovered every day.

## Applied SSL

You shouldn't rush to try and crack 128 bit encryption. Since SSL just encrypts standard HTTP traffic, if we set up an SSL tunnel, we can query the server just as we did earlier. Creating an SSL tunnel is a snap, especially since there are utilities like **openssl** and **stunnel** made just for the job.

## Exercises

10.9 Macs and Linux machines have openssl already installed. Try this command:

```
openssl s_client -connect www.hackthissite.org:443
```

This connection will only stay open a few seconds, but you can run any HTTP command, like:

```
GET www.hackthissite.org/pages/index/index.php
```

10.10 Check to see if stunnel is installed on your Linux machine or on the live CD. If it's not installed, install it. Go ahead and give it a test drive.

10.11 Once it's installed or at least running on your machine, run the command:

```
stunnel
```

This will tell you the location of the configuration file you need to create.

10.12 At that location, create stunnel.conf and enter this text (replace *ssl.enabled.host* with the name of a real SSL server that you want to connect to):

```
client=yes
verify=0 [psuedo-https]
accept = 80
connect = ssl.enabled.host:443
TIMEOUTclose = 0
```

10.13 Once that's done you start stunnel with this command:

```
stunnel &
```

Stunnel will map your local computer port 80 (default HTTP port) to SSL/TLS port 443 (default SSL port) and uses plain text, so you can open another shell and connect to it using any of the methods listed above, for instance:

```
netcat 127.0.0.1 80      [hit <Enter>]
HEAD / HTTP/1.0        [hit <Enter> twice]
```

```
HTTP/1.1 200 OK
Server: Netscape-Enterprise/4.1
Date: Fri, 07 Jan 2005 10:32:38 GMT
Content-type: text/html
Last-modified: Fri, 07 Jan 2005 05:32:38 GMT
```

Content-length: 5437  
 Accept-ranges: bytes  
 Connection: close

## The Properties of Trust

The whole SSL-thing is designed around trust. The browsers contain certificates which are really just long numbers that serve as keys. When you use SSL, the S in HTTPS, the certificate of the browser is used to check the certificate of the server and see if it's a trustworthy web server. The theory is that if the domain name matches the SSL certificate of the server as the domain that we visited then it's trustworthy. Of course who says it's trustworthy is a whole other problem. It wouldn't be the first time that a criminal infiltrated the certificate authorities and got the keys to fake their own server certificates that your browser then tells you is trustworthy. In another case, a known skeezy organization that was responsible for making spyware paid its way into being a certificate authority and added to all the browsers. So trust is a big deal on the web. It's another way to attack and it's another way that humans badly understand how to protect against it going wrong.

ISECOM spent time researching trust and discovered 10 main properties. You can read more about these properties in the [OSSTMM](#). But basically, these are 10 things that need to be evaluated to have logical (not emotional, gut-feeling) trust:



## The 10 Trust Properties

1. **Size.** The number of subjects the trust extends to.
2. **Transparency.** The level of visibility of all operational parts and processes of the subject and its environment.
3. **Symmetry of trust.** The vector (direction) of the trust.
4. **Subjugation.** The amount of influence over the subject by the source.
5. **Consistency.** A historical evidence of compromise or corruption of the subject.
6. **Integrity.** The amount and timely notice of change within the target.
7. **Offsets.** These are offsets of sufficient assurance, the compensation paid to the source or punishment for the subject when the trust is broken. It is a value placed on the trust with the target.
8. **Value of reward.** The financial offset for risk is the amount of win or gain for the source where the potential gain for giving trust to the subject is sufficient to offset the risk of breach of trust.
9. **Components.** This is the number of elements which currently provide resources which the subject relies on either directly or indirectly.
10. **Porosity.** This is the amount of separation between the subject and the external environment.

Our parent organization ISECOM pioneered the field of trust analysis: the study of reasons we trust – and which reasons are actually good ones.

**Figure 10.1:** Trust Properties



## Exercises

10.14 Look at Property 8. Do you think **reward** is a valid reason to trust? Do you think it is a GOOD reason to trust? Would you trust someone because they gave you money? And now: Do you use Gmail? Do you trust the company that provides it because it's free?

## Employing a Middleman: Proxies

---

### Using Someone Else's Server

A **proxy server** (or just **proxy**) is a middleman in the HTTP transaction process:

- The client (your browser) sends its request to the proxy;
- The proxy stops and holds your request, then sends its own request to the web server;
- The web server (which doesn't even know who you really are) responds to the proxy; and
- The proxy relays the response back to the client, completing the transaction.

A proxy can be a server on your own network that lets you pass your connection through it. This is handy because it gives you some protection, since the proxy hides your identity and acts as a firewall between you and the rest of the web. But you can also use a proxy server that's out on the Internet, which hides you even better (on-line you can find many lists of publicly available proxies, such as <http://tools.rosinstrument.com/proxy/>). These **external proxy servers** provide critical access to the outside world for people in countries that censor or cut off their ISP's connections to the Internet.

But, as you might have figured by now, proxy servers are vulnerable to attacks themselves, and can become jumping-off points for launching attacks on other servers. Not to mention that you may be going through one and not even be aware of it – which means **IT'S RECORDING EVERYTHING YOU DO**. This is something you especially want to consider if you use a free public proxy server. There is absolutely no guarantee the owner of that proxy is honest and will not use your username/password or credit card details. Schools and businesses usually have a proxy to enforce their "appropriate use" policies.

### Exercises

10.15 Are you behind a proxy? How can you find out?

10.16 If you lived in a country that blocked access to some web sites, could you find external proxies that would let you access them? If so, wouldn't the government also look for them and block them? How might you overcome this?

### Using a Local Proxy

You can run a proxy server right on your local computer. It won't change your source IP address (because its address is the same as yours), but it can prevent caching and filter out undesirable content.

### Exercise

10.17 Find a piece of software called Privoxy. If you install it, what will it give you? Are you a candidate to use it?

### The Onion Router

The Onion Router, or **TOR**, was created to hide your IP address – many times over. When you use the TOR network, your traffic gets encrypted and passed along through a tangle

of routers, and eventually emerges ... somewhere. But in theory your traffic can't be traced back to you. In theory. In reality, some things – like using Flash on TOR – has given unsuspecting users bad surprises.

Technically, you could set up TOR yourself, which involves some interesting configuration. We recommend it for the learning experience. However, most of us mere mortals will appreciate the **TOR Browser**, which has all the defaults set for safety, and lets you do all that interesting research in a separate browser.

## Exercises

10.18 Who created TOR? Why?

10.19 Find out where you get it. Get it. Open it up and make it work.

## HTML Programming: A Brief Introduction

HTML is a set of instructions that explains how information sent from a web server (Apache, IIS) is to be displayed in a browser (Firefox, Opera). It is the heart of the web. The **World Wide Web Consortium (W3C)** is the standards organization governing the workings of HTML.

HTML can do much more than just display data on a web page. It can also provide data entry forms, for server-side processing by a higher level language (Perl, PHP, etc). In a business setting this is where HTML is most useful, but in a hacker setting this is where HTML is most vulnerable.

**Note:** HTML is supposed to be a standard format across all platforms (browsers and operating systems), yet this isn't always the case. Some browsers read the HTML slightly different than other browsers just as other OS's won't be compatible with other operating systems. Be sure to cross check your HTML code against other types of browsers to ensure it is interpreted correctly and doesn't show up as a pile of gibberish.

## Reading HTML

HTML works by using **tags** or **markup**.-Most opening tags, **<h1>**, for instance, must have a closing tag, **</h1>**. Just a few tags have no closing tag, like **<br>** and **<img>**.-The markup **<h1>** tells the browser where to start and stop displaying a large, bold heading, for example. **Well-formed HTML** has proper opening and closing tags (and follows other rules as well). One factor to consider in coding (writing) HTML is the need to be read by several different platforms.

Take, for example, the code:

```
<html>
<head>
  <title>My Hello World Page</title>
</head>
<body>
  <h1>Hello World!</h1>
  <p>I'm a new page.</p>
</body>
</html>
```

We are telling the browser this is an HTML document with the tag **<html>**, and inside the **<head>** we give it the title "My Hello World Page" with the **<title>** tag. The My Hello World Page tag tells our browser "here's the information to show to the viewer." Finally, the **<h1>** tag tells the browser to display the information in "Heading 1" style, and the **<p>** tag tells

the browser “here’s a regular paragraph.” The tags that have a “/” are the closing tags, which tell the browser to stop displaying the contents as described by the opening tag.

### Exercise

10.20 Search the W3C.org website for the HTML standard. Are the HTML5 standards in the same place? Why wouldn’t they be?

10.21 Copy the code above and paste it into a text file called **hello.htm**. Open that file in your browser of choice and you should see something similar to this:

# Hello World!

I’m a new page.

### Viewing HTML Source Code

All browsers contain a way to view the underlying HTML code that generated the web page you are looking at. In most cases, this is the **View Source** option under the **View** menu in your browser. Many browsers will also show source code if you press Control-U or Command-U.

### Exercise

10.22 Go to a web page you often visit. View the source code.

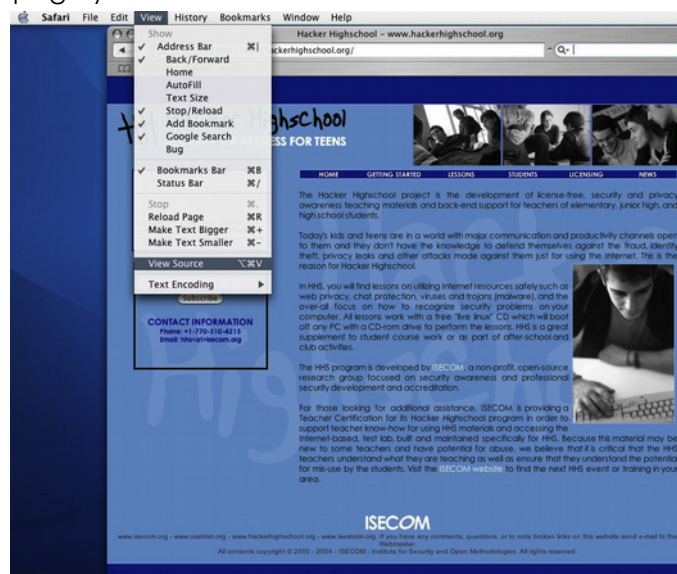


Figure 10.2: View menu

The results should be something similar to this:

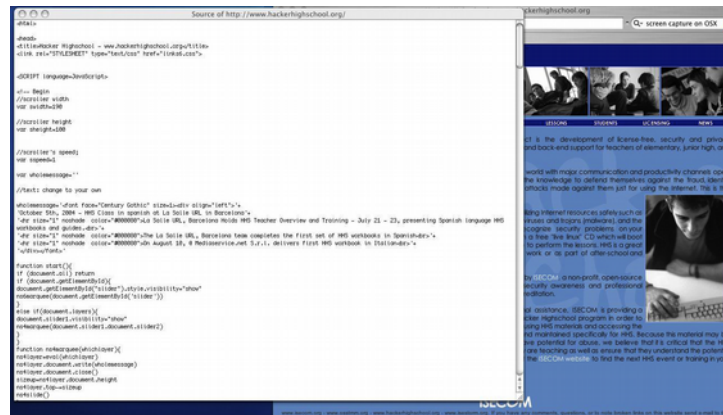


Figure 10.3: Source code

HTML code is visible to anyone with a web browser. This is why it is very important, when you create web pages, not to hide passwords or sensitive information in the HTML source code (including comments). As you can see, it's not very secret.

In most browsers you can also view the source by doing a right mouse click and select "view source". The web page can instruct your browser to behave different on a right mouse click. On some sites this is used as a (poor) security measure to prevent visitors from viewing the source. When you select view, view source, it always works on older websites. With some more modern sites, the page source is dynamically and only partially updated. You need additional add ons in Firefox to view such a dynamic source (install the Add-on "Web Developer" in Firefox and the select: View Source, View Generated Source).

### Linking Up With Hyperlinks

**Links**, or **hyperlinks**, are really the heart of HTML. The ability to link from document to document allows you to create **hypertext**, textual knowledge connected to related knowledge. A link, in HTML code, looks like this:

```
<a href="www.yahoo.com">www.yahoo.com</a>
```

This creates a link that looks like [www.yahoo.com](http://www.yahoo.com) on your website, and of course will send your visitor away to Yahoo.

Links can be followed and checked by **link checker** programs. These programs search HTML source code for `<a href="..."></a>` tags and then create a file or index of the links they find. Spammers use this technique to harvest email addresses or identify contact forms they can use to spread their mass emails. Link checkers can also be used to check your website for broken links that don't go anywhere. Due to the dynamic nature of the web, these are depressingly common even in relatively small sites.

### Exercises

- 10.23 Create a link to [www.hackerhighschool.org](http://www.hackerhighschool.org) that displays as "Hacker Highschool" on your web page.
- 10.24 Find and download a link checking program. Run that program against [www.hackerhighschool.org](http://www.hackerhighschool.org) and document how many broken links you find.

### HTML5

The new flavor of HTML, HTML5, brings a lot of security improvements. It doesn't require Flash to stream videos, which is a huge leap in terms of preventing unwanted tracking and the endless vulnerabilities of Flash.

On the other hand, it carries a whole stack of new problems with it.





**Cross Domain Messaging** or **Web Messaging** lets HTML5 escape the ugly hacks we've used in HTML4 when documents come from more than one source. The problem is, this kind of messaging requires a lot of trust – and some very secure coding – to ensure nasty things don't get into the middle of this process.

**Cross Origin Resource Sharing** is when a web server allows its resources to be used by a separate domain. Again, this is a cool way to **mash up** content from multiple sources – but this level of trust just begs to be cracked.

**WebSockets** provides asynchronous full duplex communication. That's a mouthful, but essentially it means your browser can bypass the usual security measures in return for pure speed.

**Local Storage APIs** let web pages store data on your computer. Did you know all contemporary browsers include a mini-database called SQLite? Now just ponder: what's stored on your computer in that database? All kinds of interesting things about you, right?

## Exercise

10.25 Find an application that lets you look into that SQLite database. Hint: you're looking for a “database browser.” Install it and see what's in there!

## Scripting Languages

Old-school coding in languages like C++ meant hours of deep coding, then compiling **binaries**, machine-language instructions that run very quickly. If you want raw horsepower, and have a whole lot of time on your hands, this is the world for you.

The rest of us will use **scripting languages** to write **scripts**, programs written in plain text that are interpreted at runtime by binaries (that you don't have to write) underneath. They don't run as fast as freestanding binaries, but with the very fast processors we have today, you may never notice the difference.

Scripting languages are great for dynamic web pages, but they also create a new avenue of attack for hackers. Most web application vulnerabilities aren't caused by bugs in any particular language, but by bad coding and poor web server configuration. For example, if a form requests a zip code but the user enters “abcde”, the application should return them to the form and point out the error. This is called **input validation**. Here are some of the most common scripting platforms today:

**Common Gateway Interface (CGI):** This is the granddaddy of scripting interfaces, and it's not really a language itself; it's a way to run scripts. **Perl** was one of the most popular scripting languages to write CGI programs in the early days, though it's not used much for web pages anymore. Perl is, however, very useful for hackers, and many handy tools are written in this language.

**PHP:** PHP is a very popular open-source scripting language that runs on the server before the page is sent to the user. The web server uses PHP to get data from databases, respond to user choices and build a dynamic page with the information the visitor wants. HTML displays static content; PHP lets you create pages that give the user dynamic, customized content based on their input. Web pages that contain PHP scripting usually have a file name ending in “.php”.

**Python:** Another popular language, Python is a competitor to PHP, and does many of the same things. Many web sites use both PHP and Python (as well as other languages), including Google.com, Yahoo.com and Amazon.com. Python scripts usually have the file extension .py. In the world of security, you ought to know as much as you can about at least one language. The flavor used today for security pros is Python.

**Active Server Pages (ASP):** Web pages that have a .asp or .aspx extension (ASPs) are database-driven and dynamically generated just like PHP or Python pages. ASP was Microsoft's first server-side scripting engine for the web. Its popular successor, ASP.NET, is built on the Common Language Runtime (CLR), allowing programmers to write code using



any supported .NET language, such as: C#, VB.NET, Jscript.NET, etc. If you love Microsoft you'll love writing code specifically for IIS.

**Java Server Pages (JSP):** It is a technology that helps software developers create dynamically generated web pages. JSP is similar to PHP, but it uses the Java programming language. To deploy and run, a compatible web server with a servlet container (such as Apache Tomcat) is required.

**Coldfusion** and **Ruby** have their own cult followings, and there are dozens of less-well-known languages that can do very interesting things.

**Javascript:** Javascript (which is very much *not* the same thing as Java) probably runs on more web pages than any language besides HTML. It's different from the scripting languages above, because it doesn't run on the server to generate a page. Instead, it runs in your browser after the page arrives. This gives you visual effects like fly-out menus, expandable and collapsible sections of pages and "live" interaction with the page. Practically every dynamic page uses Javascript somewhere, and it's the front line of defense for validating the information people submit via forms. However, note that client-side input validation only is not enough to guarantee protection against attack targeting dynamic web application parameters. One example of ways to abuse client-side scripting is to pull up a web page, fill in the form, and capture it with a specialized proxy. The proxy lets you rewrite the code in the returned page and send it back using bogus values. You could also do this by "saving" the web page, editing the code, then sending it back using netcat. Or you could simply disable Javascript in your browser, using either built-in controls or add-ons.

## Web Vulnerabilities

Giving someone what they ask for is simple; selling them something is a lot less simple. Online stores, companies selling products, bloggers selling ideas and personality, or newspapers selling news – all require more than just HTML-encoded text and pictures. Dynamic web sites that market products based on your preferences, show you alternatives, recommend other options, up-sell add-ons and make sure you pay for what you get require complex software. They're no longer static web sites, they're web applications. When we say goodbye to web sites and hello to web applications, we enter a whole new world of security issues.

### Lions, Tigers and Bears Oh My!

We mentioned earlier "complexity breeds insecurity". Now we are going to look at the true meaning behind that mantra. When the automobile was first invented, it was nothing more than a wheel, one simple wheel. Some folks thought that the wheel was just fine but other people wanted to improve on that wheel. These early hackers began to add more wheels together attached with an axle and a frame. Others added seats to the frame and still more added a fancy horn or installed a horse as the engine. As time went on, this simple wheel automobile slowly took shape into a fast multi-horse drawn carriage. Luckily, someone added a brake before too many other got hurt. Today, we have airbags, seat belts, V-8 engines, and really great sound systems in our cars without having to smell horse dung all the time.

The Internet's evolution has progressed in a similar fashion. People weren't happy staring at a green display screen so they added a color monitor. The old dreary ASCII text became flashing lights and splash colors with midi sounds giving way to 3-D surround sound. Each new upgrade to the Internet added a new level of complexity, another layer of vulnerabilities to consider.

Let's take a ride on the Vulnerability Train to see some of the threats in their natural habitat.



## SQL Injection - The Lion

A **SQL injection** isn't so much as an attack against a web site but rather an attack to gain access to databases behind the web site. The primary purpose of a SQL injection is to bypass the web pages. An attacker will want to gain either super user privileges to the databases associated with the web site or get the web page to dump database information into the attacker's hands.

SQL is the basic building block for databases. SQL commands will perform whatever task it is asked, including giving up passwords, credit card numbers, and so forth. All the attacker is doing is adding rogue SQL code (injecting) to any open form field on a web page.

For example, a web page asks a user if they would like to sign up for a monthly newsletter. The page will have open fields for the user to enter their name, email address, and whatever else the web builder wants. Each open field allows a user to input text, which is then stored in a database. A SQL injection simply requires an attacker to input SQL commands into the open fields. If the open fields are not protected (parsed) against allowing such commands, the attacker can easily type a request into an open field for a password list.

Here is an example of such a SQL request that has been slightly sanitized for your protection.

```
<?php
$query = "SELECT '1', concat(uname||'-'||passwd) as name, '1971-01-01', '0' from usertable;
        WHERE size = '$size'";
$result = odbc_exec($conn, $query);

?>
```

SQL injections are the most popular form of attack vectors. This attack can be rendered useless by correctly filtering SQL escape characters in user input fields or building your web site without using SQL. Don't forget about the URL field, that's a user input field too! Not to mention HTTP headers, cookies and more.

## Buffer Overflows - The Tiger

Think of a buffer as a small cup. This buffer cup holds a certain amount of data and will spill if too much data is added. When the buffer cup overflows, because someone or something tried to add too much data, the system behaves strangely. When a system behaves strangely, other weird things can happen. A **buffer overflow** is an attack (or accident) where too much data is forced into a buffer that was only built for a certain amount or type of data.

When we look to the same open fields used in SQL injections, we can insert massive amounts of gibberish into fields that were designed to handle 25 characters. What happens when we add 1 million characters to that same field? The web page goes a little crazy and can provide an entry point to an attacker.

Buffer overflows can be avoided by limiting the amount of acceptable data to each open field and ensuring rogue code/commands can't be run inside these user input fields.

## Cross Site Scripting (XSS) - The Bear

Our first two vulnerabilities were direct attacks against a server. **Cross Site Scripting (XSS)** is a client side vulnerability that exploits a user's trust to gain access into the web servers the client is looking at. A good example of this type of attack is when an attacker hitch hikes on a user's browser while that user is logged in to a security site, such as a bank. The attacker preys on the user's established trust between the user and the bank to gain access.



The attacker has various methods to piggy-back on users browsers. Some attackers will establish a **Phishing** web site or create a web site that looks identical to the one the user would visit. Since few people pay any attention to the URL field in their browser, it is fairly common for an attacker to lure a victim into allowing malware to be installed in their browser. Email is the preferred method to install malware since humans are, well, human. Java is another application that can be used to install malware from a malicious web site.

The attacker uses the trust of that user when they log onto an HTTPS site or any useful site the attacker would like to gain entrance into. One method you can use to protect yourself against this type of attack is to keep your browser updated and disable scripting in the browser "options". Based on your type of browser, there are add-ons available that can eliminate URL redirecting and stop any script that may appear dangerous.

### Exercise

- 10.26 Go to <http://scriptalert1.com/>. Set up at least one tool from that page. Be prepared to explain what it does, how it's installed or set up and what security issue it addresses.

### Bear Traps and Tiger Rugs

Each web application event and every web site you visit can bring potential harm to you or your loved ones. This doesn't mean that you should unplug from the Internet and go back to crayons in some remote cave somewhere. Each minute of each day new threats are presented and some are more successful than others. However, as a security professional, it is (or will be) your job to learn as much as you can about emerging threats and how to protect against them.

Honey pots can be used to lure bad guys into exposing themselves or track them as they probe your network. It is also common practice to apply maximum security and privacy settings to your firewall. You are using a firewall, aren't you? If you are not using a firewall, stop reading immediately and install a firewall this very instant. Once installed, max out the security and privacy settings. Your network router can be used as a firewall, as you should use both hardware and software firewalls. Don't forget the malware protection software (anti-virus).

Next on the list of self preservation is using anomaly detection software, otherwise know as **Intrusion Detection Systems (IDS)**. An IDS can be as simple as a program running in the computer's background that looks for changes to system files. More sophisticated IDS's will perform more sophisticated actions, like sounding alarms or rerouting an ongoing attacker into a honey pot. We'll talk more about these later on.

Some often overlooked aspects of security is cyber law knowledge, compliance requirements, auditing, and knowing what government regulations you need to adhere to. Yes, those do sound lame but having a basic understanding of these areas can prevent you from making the international news headlines. Jurisdiction is a constant problem when an attacker is based in another country. Knowing what you can do and what you can't do to pursue an attacker is valuable information.

### Exercises

- 10.27 What can happen if a web application doesn't perform validation on the data that the user inputs?
- 10.28 What are Cross-Site Scripting (XSS) and SQL injection? How does an attacker use them to steal data or gain access?
- 10.29 What security and privacy settings do you have on your own system? What about the rest of your family's computers?
- 10.30 Look up at least two different methods to mitigate SQL injections and describe them as a feasible addition to a web host. Tell us what they mean and how they work.



10.31 Do a Google search on the terms "inurl:search.asp" or "inurl:search.php". How many results do you get?

10.32 Find a search form at [www.hackthissite.org](http://www.hackthissite.org), and type in:

```
<script>alert ("hello")</script>
```

What happens?

Now, think: what happens if you try this on other sites? What kind of peril are you in if you do this?

10.33 Do a Google search on the terms "inurl:login.asp" and "inurl:login.php". Once again, consider how web sites invite attack simply by using these names.

10.34 Back on [www.hackthissite.org](http://www.hackthissite.org), find a login form and attempt to type in special characters (@#\$^&) for both the username and password. What happens? Do you think this kind of probing is logged?

### Using a Local Proxy for Web Application Testing

An external proxy isn't the only kind of proxy server. You can also install a proxy server right onto your own computer, where it can perform some of the same firewall and filtering functions as an external proxy. Proxies designed specifically for testing web applications can manipulate data requests to test how the web server will respond. That means that with a proxy testing utility (such as Burp Suite, SpikeProxy etc.), you can run various cross-site scripting attacks, SQL injection attacks and almost any other direct request attack. While some of these proxy server utilities have an automation feature, the best tester is a real person behind the keyboard.

Remember, these tools are designed for you to test your own web applications! Be sure to test only websites for which you have permission, or you will most certainly be logged, and possibly jailed.

### Exercises

10.35 Find and download **Burp Suite**. How do you install it? How do you run it?

10.36 Change your browser settings to point to the new proxy. This is usually localhost port 8080, but since these tools may be updated read the instructions to be sure. (Tip: the IP address 127.0.0.1 is the localhost address.)

10.37 Pull down the Help menu and select User Guide. Take a look at the documentation.

10.38 Now spend some time browsing the web site you're testing, in this case [www.hackerhighschool.org](http://www.hackerhighschool.org). Test out forms and visit every page. You're creating a recording, so try to be thorough!

10.39 Once you have surfed around with your browser, go back to the ZAP interface and save your session.

10.40 Right-click on the [www.hackerhighschool.org](http://www.hackerhighschool.org) listing on the left, and note the types of attacks you can run. Try them.

A proxy server can be a powerful tool in helping you determine how solid a web application is. For penetration tests or vulnerability assessments, a proxy utility is a good tool in your toolbox. But it's not the only one.

### Looking Behind the Curtains

There is a lot going on that's not visible when you request a web page. Long before a page loads, **headers** and **cookies** are flying back and forth, and if you know how, you can see them.

Cookies are usually harmless, and only hold things like a unique user ID, so that when you visit that famous online retailer, the site already knows who you are and what you like.



**Cookie Information - http://isecom.org/**

Collapse All  Expand All

**http://isecom.org/**

**2 cookies**

<b>NAME</b>	__utma
<b>VALUE</b>	98749830.2135635262.1315931269.1315931269.1316029972.2
<b>HOST</b>	.isecom.org
<b>PATH</b>	/
<b>SECURE</b>	No
<b>EXPIRES</b>	Fri, 13 Sep 2013 19:53:33 GMT

 [Edit Cookie](#)

 [Delete Cookie](#)

**Figure 10.4** Cookies

HTTP headers can be confusing. HTTP is Hyper Text Transfer Protocol: notice the **P!** It's a transmission protocol, a way to send signals down the wire.

Headers get transmitted invisibly with every web request, and usually you never see them. Here's what they look like:

```
http://en-us.fxfeeds.mozilla.com/en-US/firefox/headlines.xml
```

```
GET /en-US/firefox/headlines.xml HTTP/1.1
```

```
Host: en-us.fxfeeds.mozilla.com
```

```
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.7; rv:11.0)
Gecko/20100101 Firefox/11.0
```

```
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
```

```
Accept-Language: en-us,en;q=0.5
```

```
Accept-Encoding: gzip, deflate
```

```
Connection: keep-alive
```

```
X-Moz: livebookmarks
```

```
HTTP/1.1 302 Found
```

```
Content-Encoding: gzip
```

```
Cache-Control: max-age=604800
```

```
Content-Type: text/html; charset=iso-8859-1
```

```

Date: Fri, 23 Mar 2012 00:50:26 GMT
Expires: Fri, 30 Mar 2012 00:50:26 GMT
Last-Modified: Sun, 18 Mar 2012 10:39:59 GMT
Location: http://fxfeeds.mozilla.com/firefox/headlines.xml
Server: ECS (lax/2872)
Vary: Accept-Encoding
X-Backend-Server: pm-web01
X-Cache: 302-HIT
X-Cache-Info: cached, cached
Content-Length: 200

```

-----  
 http://fxfeeds.mozilla.com/firefox/headlines.xml

```

GET /firefox/headlines.xml HTTP/1.1
Host: fxfeeds.mozilla.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.7; rv:11.0)
Gecko/20100101 Firefox/11.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
[and many more...]

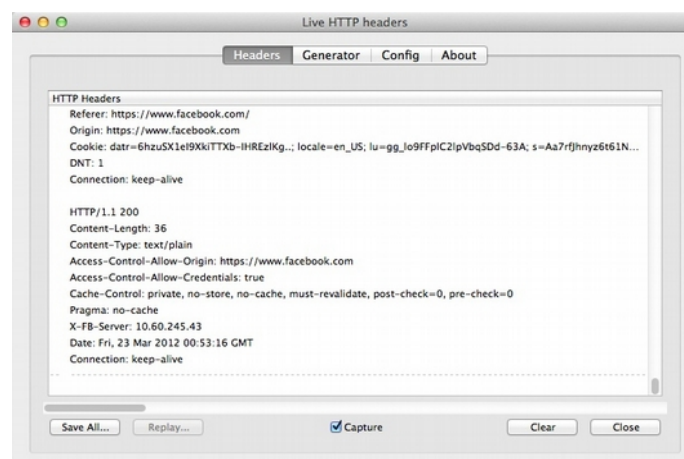
```

The HTML programming language (notice the **L!**) has a part of the document called the **<head>**, but that is *not* the same thing; it's where hidden things like scripts or meta tags go.

Even worse, HTML has **headings**: **<h1>**, **<h2>** etc. These are yet a third thing, bold headings for documents! We apologize for the confusion, but you do have to understand the difference.

So when we're talking about HTTP headers, we're talking about the hidden, behind-the-scenes chatter that happens before your page is even sent.

You can use tools like the Firefox add-ons **Live HTTP Headers** and **TamperData** to capture headers, which gets you a display like this:



**Figure 10.5:** LiveHTTPHeaders



You can use other tools, like **Add N Edit Cookies** or the **Web Developer** add-on, to capture the cookie information your browser is exchanging with the web site you're visiting. Notice that you can also edit that information. Finally, local proxies can also be used to intercept, analyze and modify cookies.

### Exercises

- 10.41 Find and install the LiveHTTPHeaders add-on for Firefox. How do you use it?
- 10.42 Find and install the Web Developer add-on as well. How do you use it to view and edit cookies?
- 10.43 If you think a web application uses the User-Agent for some purpose (why do they care what browser you're using?), check what happens if you remove it. Most applications don't anticipate a missing User-Agent, so when they inspect the content, they're referring to a missing variable. In most cases the application throws an unhandled "object not set" exception, which can include stack dumps or even source code snippets (for instance, with .Net apps with the debugging option on). Sometimes this gives very interesting results.

## Building Secure Web Applications

Have you ever considered building a web site? Selling something online? Building a web community? In some ways, it's getting simpler all the time. But in terms of security, it just keeps getting trickier.

Secure programming starts with the programmers. Each has his or her own logic and skills with particular languages. But there are accepted guidelines that a programmer determined to write secure code can follow, like the ones derived from the OSSTMM (<http://osstmm.org>):

1. Security should not require user decisions.
2. Every input and output in the application should have a business justification.
3. Quarantine and validate all inputs including the application's own content. Most especially, re-validate **all** the data server-side. Someone, somewhere, will try the trick of sending you bad data.
4. Tightly limit the systems and users the application trusts.
5. Encrypt data stored and in transit.
6. Create hashes of all the application's files, so you can check for unauthorized changes.
7. Make sure that all interactions occur on the server side.
8. Layer the security.
9. Invisibility is the best defense. Show only the service itself.
10. Build in triggers for alarms.
11. Promote security awareness in users and technical support people.

### Exercises

- 10.44 What kind of error messages have you seen online? Do you remember where you got them, or how? Could you do it again?
- 10.45 Name three ways you could build an alarm into a web application.





## Protecting Your Server

Consider what it's like to run your own server. You can set it up to do anything you like. Interested in Java or PHP? Joomla or OpenWiki? Photography, animals or both? That's the upside. The downside is that unethical people will try to do unethical things to your server.

You'll have to take some steps to protect your server. These include ensuring that your OS software is always up-to-date with patches, updates and service packs. And do the same thing with your web server software, your database software, your firewall, your antivirus, your intrusion detecting system, all of your installed software (which makes a good point: the less you install, the better!). There are many sources for correct configuration of servers (also called **hardening**), scattered around the Web. If you are running a server, search for those sources and read through some of them.

## Firewalls

**Firewalls** originally were fireproof walls in a building or used between your car engine and the occupants (old school stuff). We use the same words for systems (either hardware and software) that are designed to prevent unauthorized access to a network. Just as a firewall in a building can be a matter of life and death, network firewalls are critical to the health and survival of computer systems.

The key to understanding firewalls is that they're based on sets of **rules**, commonly called **ACLs (Access Control Lists)**. Imagine the rules for a security guard at a school: anyone with a school ID can enter, but no one else can; anyone can exit, except that small children have to be with a parent. The rules for a network firewall are very similar, for instance:

1. Nobody from the Internet gets to come in, except by invitation.
2. Anybody from inside the network can go out to the web, except when this poses a risk.

Simple as that! (If only it were.) In the digital world, we have to work with three things: MACs, IP addresses, and port numbers. You're familiar with ports and protocols from Lesson 3. Here's where you can use this information: if you're running a web server, you need to open port 80 to allow HTTP traffic. If you've got a secure website, you'll need to open port 443. *But only these ports.* Anything else is an open invitation to attackers.

Also, unlike a security guard, a pure network firewall basically isn't very smart. It doesn't look inside of packets, so it can't tell what's being transmitted. That means you could do clever things like paste the contents of corporate documents into forms on a convenient website, and steal intellectual property.

At least, you used to be able to do things like that. The newest generation of "**smart firewalls**" (which are actually very, very smart proxy servers) can indeed tell exactly what's in every IP packet. Woe unto you if you try to sneak a stolen credit card number past this kind of system, because you will be not just detected but identified.

Which brings us to the subject of intrusion detection.

## Intrusion Detection Systems/Intrusion Prevention Systems

Let's return to our school. It has good security guards at the doors and gates, but they could be fooled by a fake ID (and yes, you can fake or **spoof** your IP address). Also, the guards go home at night. We need another line of defense, in case somebody who shouldn't have access slips in: we need burglar alarms.

**Intrusion Detection Systems (IDSs)** are our alarm systems. They can detect the burglar, as well as call the police and ring a loud bell. In the Internet world this means noticing abnormal activity, realizing that something is wrong and sending email and text alerts.

Newer systems include **Intrusion Prevention Systems (IPSs)**, which are even more active: if an attack is detected, they can cut off traffic, block IP addresses and even launch countermeasures.



## Exercises

10.46 What was the original popular IDS? What is its nickname?

10.47 If you had to pay a lot of money for either one, would you want your server to have a firewall, an IDS/IPS or both?

## Secure Communications

What do we mean by "secure communications" on the web? The phrase sounds like something you'd hear in a spy movie, but actually it's critical to a lot of things we do online. Communicating securely requires a lot of **PAIN: Privacy** (and **confidentiality**), **authentication, integrity** and **non-repudiation**.

Consider your account at that really famous social network. Obviously, you have to have a user name and password so you can log in to the account. And there are all kinds of privacy settings you can tweak, especially about who can see your most private stuff, so you really don't want anybody else getting into it. Which means you're counting on **authentication** to give you some security. So, does authenticating yourself to the web site makes you safe?

No.

You might be completely deceived about the web site. Someone could send you a **phishing** email that looks exactly like a notice from that social network. Click that link, and it looks like you've gone to the site, but you've actually gone to a malicious site and gotten yourself infected with malware. Or you might make a simple mistake, like typing **whitehouse.com** instead of **whitehouse.gov** – and ending up at a porn site. (No, it's not there any more! But do check out <http://www.antiphishing.org>.)

This means authentication is a two-way street. You should have to identify yourself, certainly. But the site should provide authentication credentials too, and on the web, they come in the form of a **certificate**. When you go to a famous web retailer, and buy products online, your browser probably accepts the web site's certificate automatically, because it comes from one of the big-name, big-dollar certificate authorities like **Verisign**.

When you go to a less famous site, you may be prompted by your browser to accept or refuse its certificate. There is no more critical time on the web to read the details carefully! If you know you want to shop there, then perhaps you should accept the certificate. But if you have no idea why you're being asked to accept a certificate, then you probably shouldn't take it. The idea to understand is that a site's certificate provides its identification and authentication. You're trusting that ID.

So you've ordered that book (or whatever). You don't want the price changed after you order, naturally, or the number of books. And the retail website doesn't want anyone altering critical details like your credit card number. What you both want is **integrity**, which means that what you communicated hasn't been changed.

This is really critical when it comes to big-money transactions. No one should be able to change the dollar amount in the document you send to make an offer for a house – at least not without detection. And the way you do that is by having your computer calculate a **hash** from that offer document. A hash is the result of some complex math, but basically it's a big number that will be unique for every document.

On any Unix computer (which includes Linux and Macintosh), you can calculate an **MD5 hash** (that's the most common kind, although it's not considered that secure anymore and is therefore slowly being replaced by SHA-1, SHA-256 and similar algorithms) with a simple command. Let's say your letter is named Offer.txt. In a shell you can use the following command:

```
md5 Offer.txt
```



and you'll get back something like:

```
MD5 (Offer.txt) = d41d8cd98f00b204e9800998ecf8427e
```

The long string of numbers and letters (actually, it's a **hexadecimal** or **hex** number) is the hash. If someone changed so much as one period in that document, the hash would be totally different. Oh, so the hash you made before you sent the document no longer matches the document's hash now? Watch out! The document has been changed! And you might be buying a much more expensive house if you're not careful.

It's not just you that has to do the trusting in our real estate transaction. The people you send your letter to need to be certain you can't say, "I never sent that offer!" What they want is **non-repudiation**, which means that you can't deny making the offer. And if they accept your offer, you don't want them to repudiate their acceptance later. So how can we make sure of this?

Hashing does the trick here as well, though the process is more complex. You learned in another lesson about getting yourself a PGP/GPG key pair, which you can then use to create a **signature**. A signature works here because nobody but you can create that signature – and it will be different for every document you sign. At its root, that signature is just another hash.

## Privacy and Confidentiality

**Privacy** means keeping your information yours, or at least maintaining control over it. This may be as simple as protecting your credit card number, or as complicated as keeping your new love affair secret.

What you do on the web stays private, right? You know by now that's not true. Web sites gather information about you constantly, particularly the search engine giants. You sign away your privacy when you accept that free email account, or when you join a social network. Web sites that collect information are supposed to have a **privacy policy** that clearly spells out how they will use your information. If you visit a site that wants information but doesn't have a privacy policy, *get out!*

When you visit, sites set **cookies** on your computer. Cookies aren't dangerous, in the sense that they don't install malware on your machine. In fact they're very handy for remembering your book preferences on Amazon.com. But cookies can be dangerous because they track your every move. Be particularly aware of this:

When you log on to a website, you've given it permission to watch everything you do, everywhere you go, until you log back out! Even if you've left that site, if you didn't log out, it's still watching everything you do. Some, for instance that famous social network, track you even after you log out.

When you are done with a web site, to protect your privacy, *log out*. Ensure your browser is set to clear cookies, cache, temp files and history, then close the browser (not just the tab). Depending on the website and why you are there, you may wish to provide false information to avoid tracking. And you should be aware: some cookies (like Flash cookies) can survive "clearing" and can still be used to track you.

**Confidentiality** sounds very similar to privacy, but it's a different thing. If I don't know you have a crush on Betty or Bret, you've kept your privacy. If I know you're meeting behind the gym, your meeting is no longer private, but if I'm not watching it may still be confidential.

On the web, this means I may know you're visiting a site, but I can't read your communications because they're **encrypted** – scrambled so thoroughly you'd need a

supercomputer and millions of years to decrypt them. When you visit a secure (HTTPS) web site, that's exactly what's it's doing. And it's using its certificate yet again to provide the **key** for that encryption. Is HTTPS totally safe? No, but it's the best we've got for now.

### Exercises

- 10.48 Take a look at Google's **privacy policy**. Can you find it? Can you understand it? Compare it to the privacy policy of a retailer like Amazon.com. What about the privacy policy at social networking sites (SNS), such as Facebook?
- 10.49 Under these privacy policies, who can access your information? What can you do if the site violates its own policy? Can you take your information away?

### Knowing If You Are Communicating Securely

If you have a private conversation, somebody could still find out about it. And even if it's confidential, somebody could be looking over your shoulder. (Search for "spy camera" to see why this is important.) So how can you be sure your communication is secure?

On the web, there are two major signals almost all browsers will give you that you are on a secure connection (using **HTTPS**, in other words). The first is the web protocol you see (or type) up in your URL address bar. If it's HTTP (which will look like **http://**), it is not secure, repeat, not secure.

You only have a secure connection when the protocol is HTTPS (so that the address starts with **https://**). While it's not visible, HTTPS is doing encryption, and unlike HTTP's port 80, HTTPS uses port 443.

The other visible cue in some browsers is a small **padlock icon** in the URL bar (or sometimes in the lower right corner of the window). If the lock is open or missing the connection is insecure, and if it's closed the connection is secure. Hover your mouse pointer over the lock, and in most cases a small message will appear telling you how long an encryption key is being used. Forty bits is very flimsy, while 128 bits is fairly stout, and the current standard.

Some newer browsers are using a feature like a "site button" - an area you can click near the URL address bar - which gives you security information about the site, though you may have to dig down to find the key length.


This is a good place to point out that while we use the term "secure connection" in this section, there is ultimately no such thing. The SSL/TLS encryption used by HTTPS is flawed and definitely can be cracked, if someone wants to crack it badly enough. You can probably trust it with Amazon.com (for now), but you should not trust it with your life. Why do we put it that way? Because in some countries, people literally do risk their lives accessing the web, and they should do it through a more secure technology called a **Virtual Private Network (VPN)**.

### Methods of Verification

By now you've learned the basics of web security and privacy, and we hope you've thought about the things you could do as a hacker, or as the operator of a web site or server. Every decision you make will affect the security and privacy of real people - including you.

So how do you know if a server is secure, or a network, or an application? Should you trust the developers of the app or the OS? Will you be secure if you just keep your system updated?

The answers are, in order: Testing, No and No. We'll come back to testing, but when it comes to either applications or operating systems, manufacturers have proven over and over that you shouldn't necessarily trust them to get security right. (Read an **End User License Agreement** the next time you install software; you'll see that the manufacturer isn't



responsible for anything at all!) And there are famous stories of updates that broke more things than they fixed, or introduced new vulnerabilities. Not to mention the **backdoors** that were discovered in some applications and operating systems!

When it comes to testing, the very best thing you can do is to think like a hacker. That's what this course is about. In other words, take advantage of other people's work. There are several security testing methodologies available, the results of contributions from hundreds of professionals. Just remember that any computer, network or application will change frequently, so test early and test often.

Methods arise from different philosophies. The **IT Infrastructure Library (ITIL)** is all about system life cycles; the **ISO 9000** standard deals with quality management systems. **ISECOM**, the organization that brings you Hacker Highschool, also provides a security testing methodology built on a open-source contributor model using open-source tools: the OSSTMM.

## OSSTMM

The **Open Source Security Testing Methodology Manual (OSSTMM)** documents a widely-used and straightforward format for conducting security verifications. The individual tests in the OSSTMM aren't revolutionary, but the methodology itself allows anyone to conduct ordered tests with consistent professional quality.

The OSSTMM tests are divided into five sections:

- **Human Security**
- **Physical Security**
- **Wireless Security**
- **Telecommunications Security**
- **Data Networks Security**

You don't have to do every type of test; part of the methodology is doing only the relevant ones. This is where you can learn to be an expert: knowing which test to do, why to do it and when to do it.

To put it another way, the OSSTMM details the technical scope, but doesn't prescribe specific testing software. Instead, it describes:

- what should be tested
- the form in which the test results must be presented
- the rules for testers to follow to assure best results
- the **Attack Surface Metrics**, which put hard numbers on how much security you have

The OSSTMM is a document for professionals, but it's never too early to take a look at it and learn how it works. The concepts are thoroughly explained and are written in an easy-to-comprehend style.

## Exercises

10.50 Patching is a necessary evil, and web administrators need to constantly patch code as new vulnerabilities are discovered. Search for a case in which a new problem arose from installing a security patch. Imagine you're a system administrator: would you argue for, or against, patching your systems immediately when the patches are released?

10.51 You've discovered that a patch will introduce a new vulnerability. Should the patch still be installed? Would it matter whether you have the source code or not?

- 10.52 Go to <http://cve.mitre.org> and find the button or link to search CVEs. When you find the right page, do a keyword search on "Apache". How many vulnerabilities are listed? (And consider: how many patches do you want to install? Is patching a realistic solution, or is it a cat and mouse game?)
- 10.53 Download a copy of the OSSTMM and review the methodology concepts. What principles can you learn from this methodology? Could you do a security audit based on the OSSTMM?

## Feed Your Head: Understanding HTTP

Now that you have an understanding of how information flows, did you ever wonder how your web browser really asks for whatever it is you want to see online, and presents it to you?

What do you think the browsers ask for? What kind of information? We know that, depending on the website, we might have a few images, some animations, perhaps video. We also have text, and polls of different types. That is quite a lot of different things a web server has to send us, for us to properly enjoy a well formatted page, don't you think?

So, now you'll see EXACTLY what it is that your browser does, behind your back, so you can have a pleasant experience.

To help us in that task, we are going to install a Firefox extension called **httpfox**.

Httpfox monitors and analyzes all incoming and outgoing HTTP traffic between the browser and the web servers, so you get to see each individual request that is made.

You can download httpfox at <https://addons.mozilla.org/en-US/firefox/addon/httpfox/>

Restart your Firefox browser, and now on the bottom right side you get an extra icon that you should click in order to open httpfox.

Press the icon, and then open a new tab and visit a couple of different websites. You now have a list of all the requests that your browser made.

Clear all of it, close all open Firefox tabs, start httpfox again and open [www.isecom.org](http://www.isecom.org). Hit stop, and let's go through the requests.

The first thing that gets sent out of your browser, happens once you hit the enter key after typing [www.isecom.org](http://www.isecom.org), and that's exactly what we see in httpfox.

Your browser does a GET request (as you can see from the Request Header), in HTTP 1.1. We are also saying that we want a specific host in that address, host [www.isecom.org](http://www.isecom.org).

Consider the situation of a shared service provider. On one IP address, you could have hundreds of websites. This is possible exactly because of this "host [www.isecom.org](http://www.isecom.org)" part of the request. If this functionality wasn't present, we could only have one domain per IP address, because the browser would have no way of telling what it really wants.

Now, because not all browsers are created equal, our browser is going to introduce himself, so the server can be sure to reply with information that suits us.

Next, we are telling the web server the kind of content that we are willing to accept. In this case, our browser says it accepts code written in HTML, XHTML and XML and would like to receive the information in American English (en-us) and what kind of encoding. If you have been to the ISECOM website before doing this capture, you will



also see that we sent information regarding a cookie. A cookie is a small piece of data kept in your browser that stores information about a particular website. Almost every website you visit uses cookies. This information is used to serve you better (when it's not being used by the dark side of the force) because once you visit a website, you can be authenticated and so the website owner can greet you by your name, and present you with his content the way YOU want to see it. Think about the personalization that you can have on your online email application, for instance. All of this is for the GET request you instructed your browser to perform.

Now we will run through just three more lines. If you go to the "Content" tab, you will see the exact HTML code that was downloaded from the web server as a response to our previous GET request.

After the META tags, we got the following lines of code (at the time of writing):

```
<link rel="stylesheet" href="style.css" type="text/css"
media="screen" />
<!--[if IE 6]><link rel="stylesheet" href="style.ie6.css"
type="text/css" media="screen" /><![endif]-->
<!--[if IE 7]><link rel="stylesheet" href="style.ie7.css"
type="text/css" media="screen" /><![endif]-->

<script type="text/Javascript" src="jquery.js"></script>
<script type="text/Javascript" src="script.js"></script>
```

Look at the highlighted pieces of code. You will find that the next three requests our browser made were exactly to those files. Notice that if we had Internet Explorer 6 for instance, we would have downloaded style.i6.css instead of downloading style.css. Different browsers, different code.

The rest of the files downloaded are more of the same. The browser interprets the HTML code it receives and makes decisions based on what it receives on the contents it should request next, depending on your behavior (clicking, mousing over, etc.).

Think about what all this means from a hacker's perspective. Everything your browser shows you are interpretations of what it receives from the servers. So could we somehow intercept this traffic and change how a client interacts with a server and vice versa? Of course it does... in another lesson.

### Exercise

10.54 Aren't you curious what those .css, .js, and .swf files are? Try searching for those file extensions to learn a bit more about them.

## Going Deeper

While web hacking can go so much further because it's a huge, immense, monstrous field, you can't. You've reached the end of the line of what we can show you in this lesson. However, if web hacking is your thing, jump deeply into the [OSSTMM](#) and check out the hacking methodology for Web Applications.

Today's teens are in a world with major communication and productivity channels open to them and they don't have the knowledge to defend themselves against the fraud, identity theft, privacy leaks and other attacks made against them just for using the Internet. This is the reason for Hacker Highschool.

**The Hacker Highschool project is the development of security and privacy awareness learning materials for junior high and high school students.**

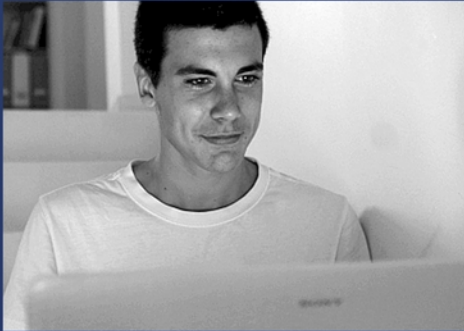
Hacker Highschool is a set of lessons and a practical means of making hackers. Beyond just providing cybersecurity awareness and critical Internet skills, we need to teach the young people of today how to be resourceful, creative, and logical, traits synonymous with hackers. The program contains free security and privacy awareness teaching materials and back-end support for teachers of accredited junior high, high schools, and home schooling. There are multiple workbooks available in multiple languages. These are lessons that challenge teens to be as resourceful as hackers, including safe Internet use, web privacy, researching on the internet, avoiding viruses and Trojans, legalities and ethics, and more.

**The HHS program is developed by ISECOM, a non-profit, open-source research group focused on security awareness and professional security development and accreditation.**



# Hacker HighSchool

## SECURITY AWARENESS FOR TEENS



## LESSON 11 HACKING PASSWORDS



HACKING IS LEARNING  
[www.hackerhighschool.org](http://www.hackerhighschool.org)

ISECOM

Creative Commons 3.3 Attribution-Non-Commercial-NoDerivs ISECOM

[WWW.ISECOM.ORG](http://WWW.ISECOM.ORG) - [WWW.OSSTMM.ORG](http://WWW.OSSTMM.ORG) - [WWW.HACKERHIGHSCHOOL.ORG](http://WWW.HACKERHIGHSCHOOL.ORG) - [WWW.BADPEOPLEPROJECT.ORG](http://WWW.BADPEOPLEPROJECT.ORG) - [WWW.OSSTMMTRAINING.ORG](http://WWW.OSSTMMTRAINING.ORG)



## WARNING

The Hacker Highschool Project is a learning tool and as with any learning tool there are dangers. Some lessons, if abused, may result in physical injury. Some additional dangers may also exist where there is not enough research on possible effects of emanations from particular technologies. Students using these lessons should be supervised yet encouraged to learn, try, and do. However ISECOM cannot accept responsibility for how any information herein is abused.

The following lessons and workbooks are open and publicly available under the following terms and conditions of ISECOM:

All works in the Hacker Highschool Project are provided for non-commercial use with elementary school students, junior high school students, and high school students whether in a public institution, private institution, or a part of home-schooling. These materials may not be reproduced for sale in any form. The provision of any class, course, training, or camp with these materials for which a fee is charged is expressly forbidden without a license, including college classes, university classes, trade-school classes, summer or computer camps, and similar. To purchase a license, visit the LICENSE section of the HHS web page at <http://www.hackerhighschool.org/licensing.html>.

The Hacker Highschool Project Project is an open community effort and if you find value in this project, we ask that you support us through the purchase of a license, a donation, or sponsorship.



## Table of Contents

WARNING.....	2
Introduction.....	5
Pins, Passwords and Personal Poop.....	6
I Don't Get It.....	6
Feed Your Head: CAPTCHA and Passwords.....	7
Methods of Authentication: Passwords.....	7
Strings of Characters: Something You Know.....	8
Strings of Characters Plus a Token: Something You Know plus Something You Have.....	8
Biometric Authentication: Something You Are.....	8
Game On: Hungry for Knowledge.....	9
Password Biology: SWAT.....	11
Strengths.....	11
Weaknesses.....	12
Advantages.....	13
Threats.....	13
Human Biology.....	16
Memory.....	16
Bad Habits.....	16
We Want Change.....	17
Feed Your Head: Bad Passwords Are Your Problem.....	18
Hashed Passwords and Salts.....	18
Hashes (MD5, SHA-1, SHA-2 and SHA-3).....	18
Salt.....	19
Key Derivation Functions.....	19
Super Duper Complex Cryptography.....	20
Password Cracking.....	21
Use the (Brute) Force.....	21
Dictionary attacks.....	22
Somewhere Over the Rainbow.....	23
Network-based Password Cracking.....	25
Wireless Password Cracking.....	29
Cracked Up.....	30
The Soft Side.....	31
Build a Great Password.....	31
Feed Your Head: Cor Issues.....	32
Size Matters.....	33
Other Methods.....	33
Check, Please.....	34
Change Up.....	35
Conclusion.....	36





## Contributors

---

Marta Barceló, ISECOM

Pete Herzog, ISECOM

Kim Truett, ISECOM

Chuck Truett, ISECOM

Jaume Abella, ISECOM

Greg Playle, ISECOM

Bob Monroe, ISECOM

Cor Rosielle, ISECOM

J. Agustín Zaballo

Mario Platt

Mark D'Angelo

Simone Onofri

Marco Ivaldi, ISECOM

**ISECOM**



## Introduction

---

To enter your home, you need a key for the locked door (unless you can pass through walls, which would be really cool). If you have a locker at school or in gym class, you either have a key lock or have a number combination lock to get inside the locker. If you want books from your library, you have to have a library card to prove who you are. You need drivers licenses, school IDs, and lunch passes to prove you are who you say you are and that you've got a lunch coming.

There are hundreds of ways we protect our stuff, using locks or guards, fences and gates or elaborate video surveillance cameras. Security professionals focus on physical security and digital security to maintain confidentiality, control access, to authenticate users, and to enforce non repudiation. The really good security professionals use combat attack dogs trained in at least ten forms of martial arts along with special training from the FBI, the CIA, MI5, and the KGB. These dogs are super vicious and will rip out your jugular if you don't rub their bellies first. If you have a dog treat, they'll play fetch for hours on end. But there's nothing as close to our hearts as our password.

In this lesson, we're going to take a hard look at how passwords work, what they can do, and what they can't do. Like any other tool, passwords have many uses but also have flaws if they are not used properly. Each day, passwords to accounts are compromised or stolen or just guessed by someone who shouldn't have access to that account. Having a weak password is much like leaving your house unlocked with all the windows left wide open.

Over the past forty years, countless studies have been conducted on passwords. Believe it or not users create and never change really crappy passwords all the time. Study after study shows how 99.9% of all Internet users have weak (easily cracked) passwords and use these lousy passwords for multiple accounts. Worse yet, we also choose bad login names. So, most of the users on the Internet have easy to guess passwords and we use the same login names. Isn't that convenient?

Hacking is easier and easier since more people add themselves to the Internet each day. What is worse (and things do get worse) is that names and passwords are the primary access control used by most websites. A long running survey by Microsoft shows that most common passwords are six to eight digits long, and user names are on average only six digits long.

Before we start hunting for passwords, we need to know quite a bit more about how passwords work. We also need to learn how to crack passwords and how to create strong passwords. This lesson can get kind of complicated because cryptology is involved here. If you don't do well with math or if complex algorithms make you nauseous, read slowly and breathe deep. Grab a paper bag in case you need to hyperventilate.

We are going to get into the weeds on passwords, digging deep down where the really fun stuff lives. Hold onto your seat belt because it's going to be a wild ride.



## Pins, Passwords and Personal Poop

You are unique, just like everyone else in the world. You look different, smell different, think different, walk and talk different than every person you've ever met. Sometimes you smell so different that others don't want to be around you. It is easy for you to see your own uniqueness and for you to recognize the differences in those around you. Once in a while you might confuse someone with somebody else but overall we humans are excellent at recognizing faces in a crowd. The problem is how do you prove you are who you say you are to a stranger? Even worse, how do you prove you are you to a computer?

Computers are not that good at being able to spot a single person out of a crowd like we are. Programs are getting better but they are no match for our matching abilities. When you meet somebody for the first time we usually introduce ourselves by our name, followed by some greeting or body contact like a hand shake, hug, kiss on the cheek or punch in the nose (depending on the situation). This introduction is based on your own culture and how those around you greet each other. Some cultures bow at the waist, others hug and kiss one another, while even more just nod and smile. This is how we introduce ourselves and start our first impression.

Deep inside our minds we are sampling their smell, the color of their eyes, the hair style they have, what sort of clothes they are wearing and a million other subconscious sensory inputs are taking place. Based on previous personal experience, we catalog each new contact and form an opinion about whether we like or don't like that person. This forms a trust bond in our emotions as to whether we are pleased to see this person or we really aren't sure about them. Computers don't have this ability, which is good for the some of us who are pretty unlikeable.

Systems need a way to determine if the user is who they say they are. We call this **authentication** which is part of **access control**. Yeah, this is where passwords come in. We need to verify your identification for you to access this device. Usually this is done in combination with a user name. Using just a password and a user name is single factor authentication. In the spectrum of security, we are looking for something that only you know, something you are and something you have. A password is something you know and your retina is something you are. If a token, a fob, credentials, digital certificates or an ID card is also required, than that is something you have. The same holds true for using your smartphone to verify a text with a code because your phone is something you have. Plus that provides two-factor authentication.

Biometrics approaches this by using your physical features to prove your identification through eye blood vessel patterns, finger prints, voice input, breath, facial features, earlobes, microbes on your body, and other aspects of your body that are unique to you. Passwords are a cheap and easy method to partially prove you are who you say you are. These are the basic flaws with passwords: they are easy to forget, easy to lose, easy to steal, easy to replace (which is also, oddly, an advantage), hard to remember, and too easily direct blame at the person who has their account hacked (which is also an advantage but not for password owner). Then again, if your biometric password is stolen, you can't just change your fingers.

### I Don't Get It

So basically, encryption is a two-way mechanism. You start with a plaintext written letter, encrypt it to cyphertext and send it to whomever you want. They receive it, decipher it with the key you gave them earlier and they read your original plaintext written letter.

Hackers realized years ago that to keep passwords really secure, you couldn't store them encrypted, because given enough time (and computing power) you can **crack** (decipher) them. So instead of encrypting passwords, systems started to use **hashing**.

Hashing involves taking something (a file, a password, a document) and running a series of operations on it, leaving you with a series of characters called a **digest** or a **hash**. Hashing is a **one way function**, because unlike encryption, what you get at the end of the

process is not reversible. This means that starting from the digest or hash, you will never be able to re-create the original data. (In theory.) But the same data, using the same hashing algorithm, will always generate the same digest. Not only that, but if you change so much as one letter in a 1000 page document, the digest will be completely different.

## Feed Your Head: CAPTCHA and Passwords

CAPTCHA has single-handedly kept more morons from posting stupid opinions on blogs and websites than any other technology ever and thereby keeps the Internet from crumbling under the full weight of the stupid part of humanity.

You know CAPTCHA. It's that box with hard-to-read words that you need to type in before posting to websites. It's interesting because it adds a new channel to authentication, the human channel, and it adds entropy for slowing down the attack by making a complicated extra step in the process. It's meant to prevent SPAMMERS and password cracking attacks. If a password is a key then CAPTCHA is a special twist of the key if I may. That's good. Too bad most people find reading mutilated letters off a paisley background more than enough entropy to slow them to a stop.



CAPTCHA is impossible for some and frustrating for the rest of the people. Too bad computers didn't have the same trouble. If we are at risk of the Matrix or Skynet it will be because of CAPTCHA. It's so annoying that if the Terminators wanted to end humanity they would put CAPTCHA on our Internet-connected refrigerators and watch us starve to death.

## Methods of Authentication: Passwords

The first step in obtaining proper access to any important information or secure area is to prove you should have access to it. In order to gain that access, you need to authenticate yourself to the system. Authenticating is just fancy word for proving you are who you say you are. You know who you are but strangers and networks need some proof that you're not pretending to be somebody else. Depending on the sensitivity of access, security folks look for two or three items. This is known as **multi-factor authentication**. We are looking for **something you know**, **something you have** and **something you are**.

Something you know is just a piece of information that only you should know. For example, only you and a few select other people would know your favorite color, favorite food, all-time greatest movie, or your password. This information could include the school you last



attended, the name of your pet, the sport and position you like to play, favorite music or book and so forth.

Something you have might be the token you were given for access to a network. It could be your key to the lock or even an identification card. This is usually something that you would physically have in your possession. You must have that on you to gain access to that network as part of multi-factor authentication.

Something you are is unique only to you. This is a big topic for biometrics because it can involve your fingerprint, which have been proven to not be unique and are easily spoofed. However, there are other things about you that are much more provable such as the blood vessels in your iris, the speed at which you type on a keyboard, your thought patterns, your behavior and all kinds of other aspects that make you who you are.

Passwords come into play as something you know. The logic behind using passwords is that only an authorized agent would know the correct password to gain access. (See any problems there?)

There are three main types of authentication:

### Strings of Characters: Something You Know

At the most basic level, authentication uses passwords, which are words or strings of characters (numbers and symbols). A keyboard or keypad allows entry of these types of passwords. These passwords range from the simplest – such as the three digit codes used on some garage door openers – to the more complicated combinations of characters, numbers and symbols that are recommended for protecting highly confidential information.

The problem with character based passwords is that the more complicated the password is, the harder it is to remember. The classic solution is writing the password down on a slip of paper next to the computer screen. If you're going to write down a password, keep the paper in your wallet or purse! People rarely lose their wallet or purse and if they do, changing your passwords will be a snap compared to replacing all their credit cards and their drivers license.

### Strings of Characters Plus a Token: Something You Know plus Something You Have

The next level in authentication (**two-factor authentication**) is to require a password plus a **token** of some type. An example of this is the ATM, which requires a card – the token – plus a personal identification number or PIN. This is considerably more secure, because if you lack either item, you are denied access. So make sure you don't keep your PIN in your wallet too!

There are all kinds of token based systems out there. One of the current attack trends is for attackers to attach a card scanner (a **skimmer**) in front of an ATM machine's card reader. When a victim inserts their card into the ATM, they enter their PIN. The skimmer reads the magnetic strip off the victim's card and also captures the PIN. The victim is completely unaware of this crime until their bank account is emptied.

The point here is that tokens can be hacked and have been hacked.

### Biometric Authentication: Something You Are

The third level in authentication is biometrics. This is the use of non-reproducible biological features, such as fingerprints or facial features, to control access. An example of this is the retinal scan, in which the retina – which is the interior surface of the back of the eye – is photographed. The retina contains a unique pattern of blood vessels that are easily seen and this pattern is compared to a reference. Biometric authentication is the most sophisticated and is considered safer, but it has failed. How about copying someone's fingerprint with a gummy bear, and then using it to authenticate as another person? (Been done.) And hey, if they just need your retina, that's not so tricky, is it?





But biometrics can also be invasive (retina scan are, iris scan are better), dangerous (think about germ propagation), and discriminatory (if a user has some fingers missing, an eye missing etc.).

A reality TV show demonstrated the use of clear plastic tape to copy a fingerprint and fool a fingerprint scanner. Lately, the whole idea that fingerprints were unique to every individual has come under fire. There have been several recent criminal cases where one person's fingerprint matched another person. Like most science based on human characteristics, there's still plenty of work to be done before biometrics can be considered fool-proof.

Let's also keep in mind that science can only prove that something is wrong, it can never prove that something is right. All theories are just theories until they are proven wrong. That is the job of science.

### Exercises

- 11.1 Is the SIM card in your cellphone an authentication token or a tracking device? Pull the SIM card out of your phone and see if you can still connect to another cellphone.
- 11.2 Going CSI, try to capture your own or a friend's fingerprint using clear tape. What additional materials do you need to get a "good" print?



### Game On: Hungry for Knowledge

Mokoa's stomach groaned next to the gold-fish tank. Several small orange swimmers looked at the teen with worry, hoping he wasn't in the mood for seafood. Gerbils across the aisle hid their apple and vegetable bits behind their water bottle. Even though the pet store was filled with food, none of it was human food.

Mokoa complained to himself that he shouldn't have eaten his lunch so early. It was late afternoon and he was running the store by himself. He also forgot to get a snack from his upstairs room before he started his shift. Normally this wouldn't be a problem because he could just run upstairs and grab a quick bite. However, every time he started up the stairs, another customer would enter the store. Every time.

It was one of those days where a certain customer would come into the store and just ask questions or want to see one thing after another, never buying anything. Mokoa had to treat them with respect and reply to every question or request, because he was the pet store owners grandson. And he was on shift.

But he was also very hungry.

The demanding customer asked to see the bottom bag in a pile of 50 lb. bags of dry dog food. The stack was ten bags tall. The teen pet seller smiled outwardly but truly wanted to throw a bag of food on top of the annoying customer. Using two ladders, Mokoa had just moved the eighth dog food bag when the customer changed his mind. He didn't want the last bag anymore. He now wanted the 50 lb. bag that was on the bottom of the new pile now.

As he stood next to the ladder trying to figure out if he could throw a 50 lb. bag on the customer and tell the police a convincing story, Jace stepped through the front door.

Trumpets blew. Heavenly lights flared behind her. Angels sung in chorus. Jace was here.

"Jace, you are the best thing I've seen all day," Mokoa announced throughout the store.

She squinted her light sensitive eyes, unsure if Mokoa was joking or had some sort of illness. Jace approached him to check his forehead temperature.

Mokoa looked at his best friend and said, "I'll be right back. Hold down the store for me,



will ya.”

He was upstairs in a single step.

Jace turned to the customer and asked, “What is this all about?”

By the time Mokoia returned to the downstairs store, his milk mustache was almost gone. He almost forgot about Jace, now that his tummy was close to being full. Mokoia held a plate with three cookies on it. As he looked around the store, he didn't see Jace but he did notice that the piles of dog food were neatly stacked in their original location. The nagging customer was gone so the teen grabbed a chair and started to sit behind the cash register desk.

“Hey, ouch! Move that thing. I was here first,” Jace yelled.

Mokoia saw Jace laying on the floor using her knapsack as a pillow.

“What are you doing down there,” he asked as he set the plate on the chair.

“Resting, where you been?”

“I needed some food. When you came in, I finally had a chance to grab a bite. Thanks for covering for me,” Mokoia said. “Where did that customer go? Did he buy anything?”

Jace gathered herself up and said, “Naw, he was just looking. I got him to put those bags away before he left, though. Hey, thanks for bringing me some cookies for my effort.”

“Hands off. They're mine. Don't touch or else,” he said in a defensive posture.

“Or else what,” Jace replied.

“Or else I'll do something,” Mokoia said.

Jace mocked, “You'll do something, I'll bet.”

Mokoia guarded the cookie plate and said, “I will and I'll take that bet.”

“What bet,” Jace knew Mokoia was losing this argument. Now was a good time to show him who is the brains in this team. “Tell you what, I'll bet you those three cookies that I can guess your laptop password in under 90 seconds,” she said.

Mokoia replied, “And what if you don't guess it in 90 seconds. What do I get out of the bet?”

Jace thought about for a second before she replied, “Then, I'll teach you how to crack passwords. Deal?”

“Deal,” he replied. He always wanted to learn some of her skills but she never taught him anything.

Mokoia grabbed his laptop and handed it over to Jace. She smiled back.

The teen hacker inspected the outside of the machine and began her lecture, “First off, guessing passwords isn't terribly difficult. The more you know about the victim, the better guesses you can make to unlock that password. For example, knowing your name is a step towards knowing your username. Many times, usernames and passwords are the same or very similar. People are lazy; they don't want to remember too many things.”

“With something portable like a laptop, a keyboard, or a phone, the password is sometimes written on the outside of the device. Users love to keep passwords on post-its under their keyboard or next to their computer screen. It looks like you didn't write your password on the outside of your computer so you did good there,” Jace said as she spied Mokoia for a reaction.

With the laptop turn on and ready for the password, Jace studied the screen. “I will need to know how many attempts I can try and guess your password before it locks me out. Most people disable this security feature and allow anyone to guess their password forever. I suggest you set password attempts to between five and ten attempts. In fact I remember telling you this when you bought this computer last year. Let's see if you



listened to my advice," Jace said as she tried a few basic passwords.

"Nope, you didn't set a password limit," she said with an evil laugh. Jace cracked her knuckles and started to whistle the tune from Clint Eastwood's movie *The Good, the Bad, and the Ugly*.

**Game continues...**

## Password Biology: SWAT

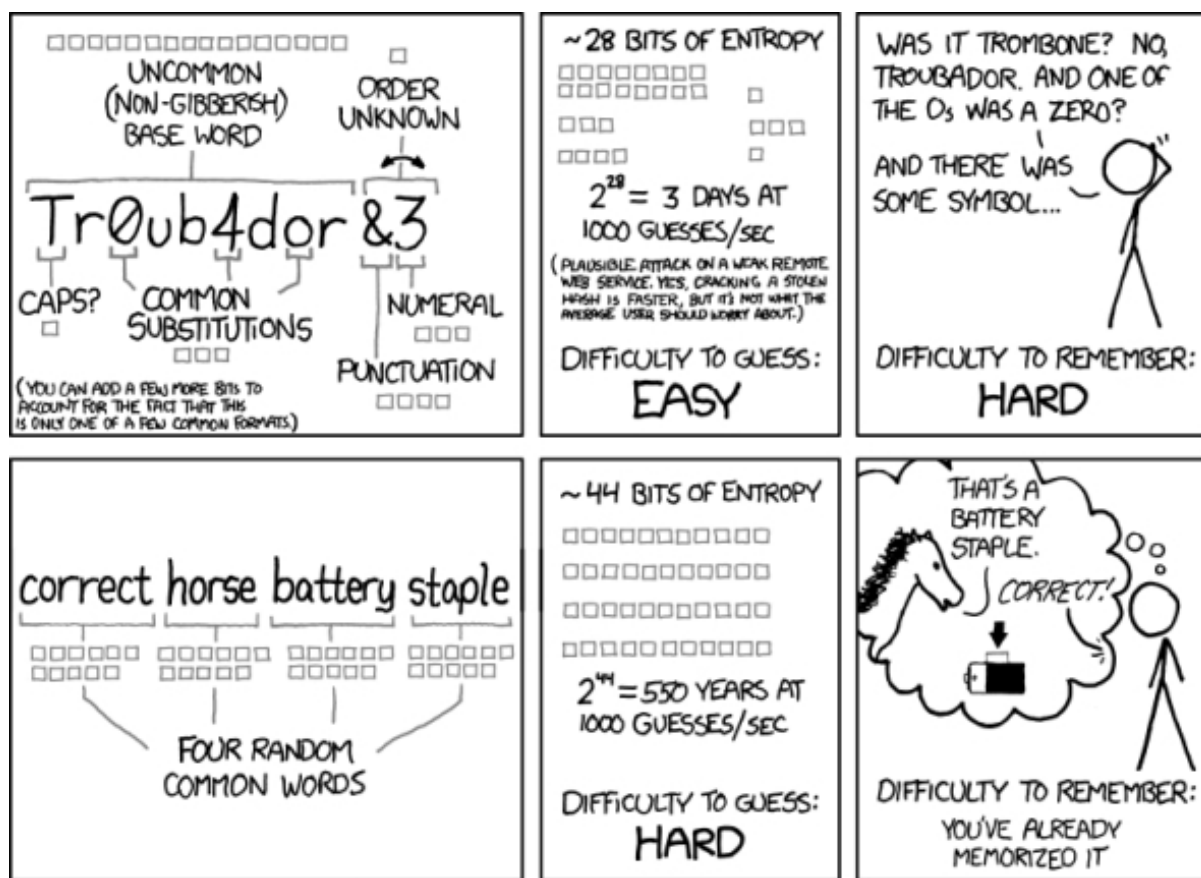
Passwords rule most of the digital world. In particular, passwords rule the entire Internet world. (When was the last time a website asked for your fingerprint?) To take a closer look at passwords we need a tool to open them up. The toolset we'll be using today is called SWAT. **SWAT** stands for Strengths, Weaknesses, Advantages, and Threats. Put on your lab coats and slap on those protective goggles because it's going to get messy.

### Strengths

Size matters when it comes to passwords. The longer the password, the more difficult it becomes to crack. At a certain length, a password becomes essentially too expensive to crack. Hackers are busy people, very busy. They want to direct their computing power toward easy passwords. If you have a password that is six digits long, an attacker will focus on you instead of someone whose password is twelve digits long. Cracking a password requires time and resources. For each additional character added to a password, complexity increases exponentially.

The word **entropy** describes the difficulty of cracking any password using brute force, pure guessing, using a collection of common passwords or just relying on dictionaries. This is why many systems limit the number of times you can enter an incorrect password before the system locks you out for a specific time. When it comes to passwords (because the term is also used in physics), entropy is usually measured in bits: the number of bits required to make up a password. For instance, an 8-character password requires 8 bits per character (in English, but not in all languages – some need 16 bits per character) so  $8 \times 8 = 64$ , meaning an 8-character password space has 64 bits of entropy. And to put it simply, the more entropy, the stronger the password.

Another major strength for using passwords is they are customizable for each user. Passwords are not stored, only the password hash is stored. If you haven't met MD5 yet, we'll get into the hash stuff a little later. Just be sure to know that any change to a password, such as capitalizing a letter in it, will produce a completely different hash output.



THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.

Figure 11.1: Password Strength (courtesy of <https://xkcd.com/>)

The XKCD cartoon above is famous in the password-cracking world because it's right in some respects (longer is better, easier to remember is better), but also wrong in some ways (using common words makes cracking easier, and using any simple pattern like all lowercase letters is catastrophically weak). The whole phenomenon has been discussed so much it's commonly known as "that horse-battery-staple thing."

### Weaknesses

Everything that was just covered as a strength is also a weakness. The size of a password adds additional security but it also adds complexity. Humans are lazy creatures; we want to make things as simple as possible for ourselves. From a security professional view, security should be invisible and effortless to the user. Security should not hamper or slow down work but should operate in the background. Anytime you add additional steps or requirements to the work environment, you slow down work performance. You might have noticed that security is not anywhere near that goal at this point.

As the need for stronger passwords are enforced by the boss, people will always find a way to get around such mandates. The average adult can easily memorize about seven things. Asking a user to memorize twelve characters means that the person will have to either create a password that ends with 12345 or one that they will write down. Worse yet,



they will reuse passwords over and over again. Think of what will happen when a user is told they must have a password that is eighteen characters long! You don't really want to be on either side of that conversation. That's like boss-level passwording.

On top of mandating a twelve or eighteen digit password, add in a requirement to change that password every ninety days (without repeating any of the characters from your previous ten passwords, welcome to the U.S. Department of Defense). You and you users are going to be overwhelmed with memorization. Go ahead, try to enforce this on your family.

Managing ten or twenty users accounts is fairly easy, but most big organizations have hundreds or thousands of users. Managing passwords and logins for larger groups becomes a full time job. Sooner or later it will be a highly hackable mess. From a password management standpoint, removing an account by deleting the user and their password is inexpensive and fairly easy to do. Any hacker knows that administrators often fail to do this, which makes for easy fishing.

Over the years several organizations have been caught storing passwords in clear text. One recent example was a major attack on Sony Entertainment in November 2014. Thousands of usernames and passwords were discovered and released to the public by attackers that showed Sony had stored this information in their networks as easy to read text. Then the Yahoo attack showed us how millions of passwords could be lost the same way. So your best bet is to not store passwords at all but rather store the hash values instead. Even then, the hash values need to be secure too.

## Advantages

How much does a password cost? Each password costs an organization a tiny bit of computation and storage (to hold it and look it up) but not much more than air costs you to breathe. Other forms of authentication can cost an organization a small fortune to install, maintain and operate. A lost password is easy to reset or replace. Passwords can be generated and distributed *en masse* in no time at all.

Users can be allowed to customize their own password as long as it meets expected security requirements. Passwords can't get lost like a physical token can. If a token goes missing, the token replacement requirements can be expensive and harmful to the longevity of the employee (you can get fired). Passwords can be replaced in a matter of seconds and a compromised password can be revoked easily.

If more than one person decides to share a password for easy access, that account can be audited and taken off-line. The password would be changed, as well as the login. Passwords can be traceable to the user if suspicious events need to be investigated.

Of course, this also means a criminal could hide her actions behind someone else's account. Isn't that handy? Unless you're the victim.

## Threats

The last tool in our tool set is Threats. There are many areas of concern to acknowledge when it comes to password threats. The easiest of these threats to attempt would be password guessing. Thanks to the Internet, there are many resources available for guessing passwords. For example, fire up your search engine of choice and search for "common passwords." Start compiling the common passwords from your results into a single location and before you know it you'll have a great password dictionary to assist you with password guessing.

**Social Engineering** is another great way to determine a users password. You would be amazed at how many people use the names of their children, spouse's name, pet names, anniversary or birth dates, favorite color, etc. for their password or as a combination to make up their password. All of this information can be gathered from someone through casual conversation or examining the articles in their office or cube. The best way to prevent against this form of attack would be avoiding the use of any information that may be common knowledge.



**Dumpster Diving** and shoulder surfing can also provide quite a bit of useful information about someone that can be used to guess their password. Being aware of your surroundings, using a screen protector, not facing your monitor towards a window and using a cross-cut shredder can assist you in protecting against those threats.

With the large amount of social media platforms now available, it's getting even easier to find out information about people. You can probably get someone's favorite color, birthday, kids names and spouse's name right from their Facebook page. Be cautious of the information that you share via social media and keep strong privacy controls in order to help reduce the threat of attackers phishing for information that can be used to guess passwords.

**Offline Password Cracking** is a threat to all of you who think your password is so good! If the attacker has the password file then they have all the time in the world to run dictionaries and rainbow tables against it.

**Phishing** may be the easiest and most popular way to get account details. In a phishing attack the cracker sends a link to a fake login page via email or chat. The unwitting victim clicks, sees a familiar login page and enters their credentials. Often from there they're redirected to the real login page, where they will be confusingly not logged in. Oh well, try it again. But by now the cracker has the victim's login details.

**Password Sniffing** means just waiting for you to type a password and grabbing off your system, network, or even right off your keyboard. While keeping your computer clean of key capture dongles that fit between your keyboard and PC or malware that waits for passwords will help, it won't do any good if the attacker is between you and the server you connect to. In that case, encryption is your friend. On web sites, enable **HTTPS Only** in your browser to ensure that the site uses an encrypted connection whenever data is exchanged. Otherwise it's too easy to grab your password straight from your connection to the server.

**Shoulder Surfing** is nothing more than someone looking over your shoulder while you type a PIN or password. Of course they can be standing three blocks away and have a really good camera lens.

**Stupid Friends** are exactly what it sounds like. Take care of sharing your password, also with friends. On some web sites terms of use – these are an **Indemnification control**, see the OSSTMM for details – you are responsible for all activities generated with your credential.

**Keyloggers, RATs** and other malware can set up a keylogger or RAT server on the victim. Your boss can do it too, using either software or a slick little piece of hardware that plugs in between your keyboard and your computer. The keylogger records every key stroke of the victim. Everything you type, the keylogger records and sends to the cracker. Some malware will look for the existing web browser's client password list and copy it to a remote cracker, making the passwords easily accessible if they aren't encrypted.

Consider this: when you answer **password reset questions**, do you use the real answer? We know we don't and we recommend that you do not either. Password reset questions are a great way for attackers to guess your password. Next time you answer one of those questions, think of something other than the real answer to use. For example, come up with a strategy that says, each time you see the question, "What was the make of your first car?" you answer it with the word "Chocolates." We all know there are a limited number of car manufacturers, so it would be pretty easy for a person to rip through a list of car makers. We seriously doubt anybody would be guessing "Chocolates" for the make of your first car.

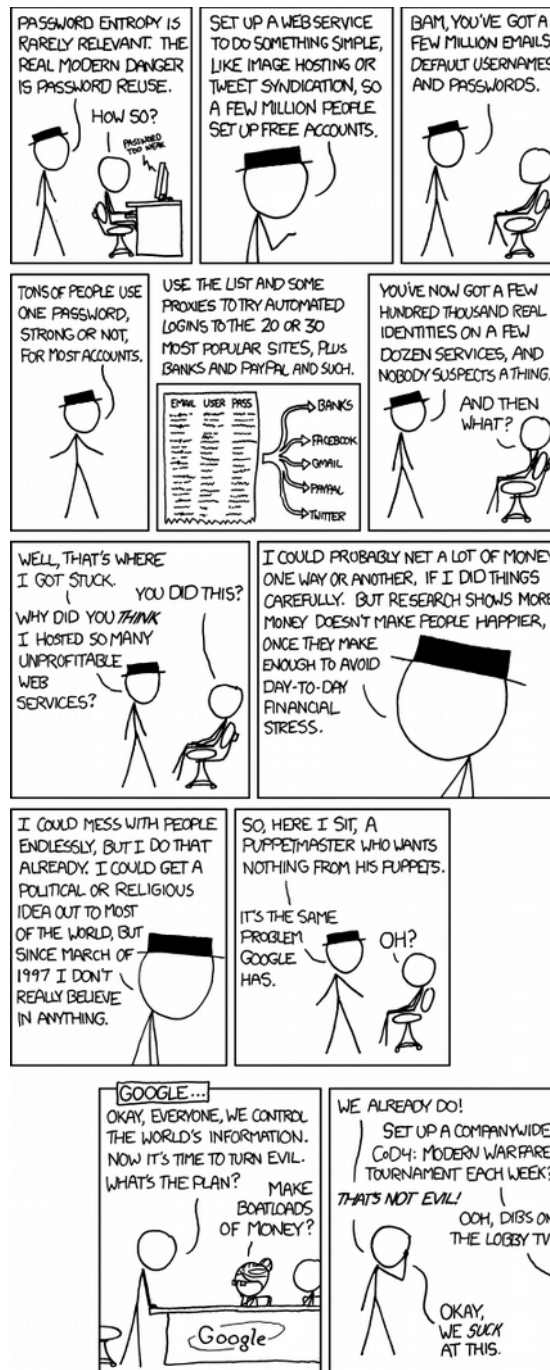


Figure 11.2: Password Reuse (courtesy of <https://xkcd.com/>)

In order to help decrease the number of password threats you are exposing yourself to, get in the habit of following best practices such as memorizing passwords instead of writing them down, not reusing passwords and creating strong passwords. The creation of strong passwords will be discussed later in this lesson.

Another important threat is the password reuse. A long time ago in a galaxy far, far away an important website was compromised via SQL Injection and attacker found passwords stored with weak hash algorithms and e-mail addresses users used to register. The attacker tried – with a lot of success – e-mails and associated password on various social network and posted malicious messages containing a malware. This is an important lesson: never use the same passwords on different services, are you using same passwords? Change them now! Have you difficulties to generate or remember complex passwords? Use hints in this lesson and a good password manager.



## Human Biology

---

### Memory

We store every bit of sensory information our body interacts with for a fraction of a second in the cranial cortex (your noggin). Some of that input is moved into our short-term memory, such as parts of a conversation with your parents or your math teacher showing you your grades. The sensory data that catches our attention, like the school bell ringing or your smelly socks under the bed, will stay in short-term memory throughout the cortex. This short bit of excitement stays in our memory for just a few seconds, perhaps a minute.

Studies from the Harvard School of Medicine indicate that the average adult can memorize nine random items, characters, numbers, or other unimportant items for up to a week. After that week, our capacity for memory drops down to seven items. If the collection of random items are not utilized, or rehearsed, these seven items slowly fade in less than three weeks.

If we do not attach some sort of meaning to random objects for later retrieval, even if its movement (because muscle memory rocks!), those items will not be moved into our long-term memory. This plays an important role when working with lists, tasks, combinations, events, recollection, and passwords. If we can associate new information with old memories or past events, those new memories will become locked in long-term brain storage.

Basically, humans can't be expected to memorize long strings of characters and random digits if there's no real meaning behind them. This is one of major faults with computer generated passwords when you are strictly forbidden to write them down. (Welcome to black ops.) How can you be expected to remember a string of eight, ten, or twelve characters when your short-term memory is maxed out within one minute? Try telling this to your mom when you forgot her birthday, again.

The problem is most networks have a password update policy that is almost impossible to comply with. These policies require the user to have a password of nine characters or more but not more than twelve. The password must be a combination of upper and lower case words, have numbers and special characters. The system stores your previous ten passwords and you must pick a new password every 45 days. This is where a password manager comes in.

One of the problems with such a policy is that people can't remember such long passwords, nor do they have the creativity to make new ones and remember them every 45 days. Plus, passwords usually start off with a capital letter and end with number or special characters. That makes cracking this policy even easier.

### Bad Habits

As our lives become more dependent on fast electronic devices and a speedier world, where we are expected to do more in less time, our attention span narrows. School teachers report that thirty years ago they could focus the entire class on a single subject for over an hour. Now, these same instructors are reporting that they have to change topics or refocus the students every eight minutes. Our attention span is shorter as we are getting accustomed to instantaneous stimulation ever minute of the day. The worse part of all this constant content, is that we think we can perform more activities at the same time, or multitask. College students and new workers who enter the world confidently proclaim their skills of conducting four and five tasks at the same time, with the same level of accuracy as focusing on a single task.

Research at the Massachusetts Institute of Technology (MIT) tells a completely different story. Three independent studies found that individuals who think they can perform many tasks at the same time, with a high level of accuracy, fail miserably when tested. The test subjects were carefully selected to ensure they had scored fairly high on academic tests and used smart-phones to tweet, socialize, research, email, answer calls and other activities we associate with the "tech generation."





Each test group was given one main task to perform such as writing a policy letter or typing a memo without errors. As the testing progressed, the test subjects began to receive messages, texts, requests for chat, emails that needed to be answered with researched information, and other multitasking activities at set intervals. The test subjects failed to perform a single task. Each task was full of simple errors and many tasks included answers that belonged to a different task.

The next phase of this study added one more critical element sponsored by the National Traffic Safety Council (NTSC), driving a simulated car. One in ten students were able to back out of a simulated driveway and move down the road one block before they crashed due to multitasking. Again, each task that was added to the main objective of driving a car was nothing more than answering a call, checking an email and looking up a traffic report on a smart-phone with a cup of water in the test subject's lap. Sound familiar? That's because millions of drivers are over confident in their ability to accomplish several tasks at one time. Our brains were not designed to handle the complexity brought on by modern technology. Instead of making life easier, it just makes our lives run at a faster pace.

## We Want Change

We are creatures of habit. We want to conform to our environment just as we want to be accepted by our peers. We also need to make our lives comfortable but interesting. This combination causes mankind to resist change, more specifically, abrupt change.

In the world of security we preach to our users the basic principles of using long passwords, using upper and lower case, changing passwords, and not using the same password for all accounts. However loud and long security folks bang this message into everyone's head, it's still hard to change habits. Resistance to change is what makes hacking passwords so easy, unless some incentive to use strong passwords is applied (like keeping your job).

Incentives can be either positive or negative. A positive incentive would be something as simple as a little banner telling a user that their password is strong or a pat on the back by their supervisor. A negative incentive would be getting fired because their account was hacked due to a weak password. If you're a security professional, you will need to find a balance between the two forms of incentives. If you're hacking the system, you can use them to your advantage.

Humans respond better and longer to a positive incentive than a negative one. Think about it this way: do you like being yelled at for having a bad grade or would you rather be given a cash reward for earning good grades? Cash, right? In the security arena, humans have been and always will be the weak link for a security environment. You will need to "sell security" to your peers, your friends, and your family. You can start by looking at the current password policy your organization has and work up from there. What – you don't have one?



## Feed Your Head: Bad Passwords Are Your Problem

Many places claim that a criminal hacker got into an account because the user made a bad password. That's just shifting blame. The truth is that the password problem isn't a people problem. It's an authentication problem.

The thing is that people are people. It's too easy (and cheap) to put the responsibility on just regular people for making uncrackable passwords despite security researchers (and anyone with common sense) knowing that really good passwords are too long and too hard to remember so they can't really exist in a realistic, usable state. But for legal and economic reasons, companies continue to make their users accountable for making and protecting their own passwords so that the server and service owners can't be blamed for giving access to the wrong person.

So how you prove who you are, how you sign up, make passwords, retrieve lost passwords, change accounts, etc. is in YOUR slippery hands.

But you only have a problem if you're careless with your accounts, share them, get phished, click on something you shouldn't, get malware on your system, or any part of your communications are hijacked. So no problem because that like NEVER happens! (Note: see every major criminal hack in the last 5 years and a book on sarcasm).

So yes, the current scheme of login and password that we all know and suffer under is a careful blend of legal deniability, fiscal sensibility, and a hefty pinch of what most parents of the world call "not thinking things through". You don't ask regular people to make irregular passwords. But it's done all the time so, you still need to know how to make unbeatable passwords and protect them.

## Hashed Passwords and Salts

Once everyone discovered that storing passwords in plain text was a bad, really, really bad idea, some smart folks came up with a better system. Instead of keeping passwords lying around in plain text, let's encrypt them. The problems with encryption are the computing power required to encrypt and decrypt the password, and safely storing the encryption keys. Ha ha, got your keys! Now we have ALL your passwords.

Next, some smart folks came up with the idea of using a one-way **hash** algorithm on a password to create a hash string. This hash value can't (in theory) be reproduced through reverse engineering, hence the name "one-way." Easy, right?

### Hashes (MD5, SHA-1, SHA-2 and SHA-3)

A hash is a mathematical formula that uses a password and sometimes random data (a **salt**) to return a single value. Think of a hash as a locksmith creating a key to fit one lock. The locksmith uses unique materials and special tools to make each key so that only one key will ever fit that one lock. This **hash key** is built using the password as part of the material and some added information (the salt) to make the rest of the key. The hash key or **hash string** will only match one particular lock. If you try to create a duplicate key, the fake key won't match because only one key will ever fit that one lock. It's pretty tight.

Let's try this another way. Take a can of colored paints, blue, green, red, yellow, brown, white, purple and mix some of each paint into a small jar. Don't worry about how much of each paint you add, just pour, mix, repeat. Whatever color that jar becomes with all those colors mixed in, is fairly original. It would be difficult to duplicate this process and make the exact same color.

Now, let's repeat this exercise with a bucket full of sand. In that bucket each grain of sand is slightly different from the next, and there are millions and millions of pieces of sand in there. This means getting another bucket full of the exact size, color, and shape of each



piece of sand in that original bucket would be almost impossible. When you are done with your first bucket of sand, throw it into the ocean. Now, try and locate an exact duplicate bucket of that first sand batch at another beach located on the other side of the world. A hash value is almost (yes, almost) impossible to repeat ever again.

Those grains of sand or exact mixes of color are what are used to formulate an answer, not the sand or paint themselves. If one grain of sand in the new bucket is slightly different from the original, the calculations will be wrong. Any one change to the data will result in a completely different calculation answer: The lock will not open.

Since nothing has to be encrypted and decrypted, it's easier to calculate a hash value when a user types in a password. Instead of looking up the user's password, the password's hash value is calculated and compared to a stored hash.

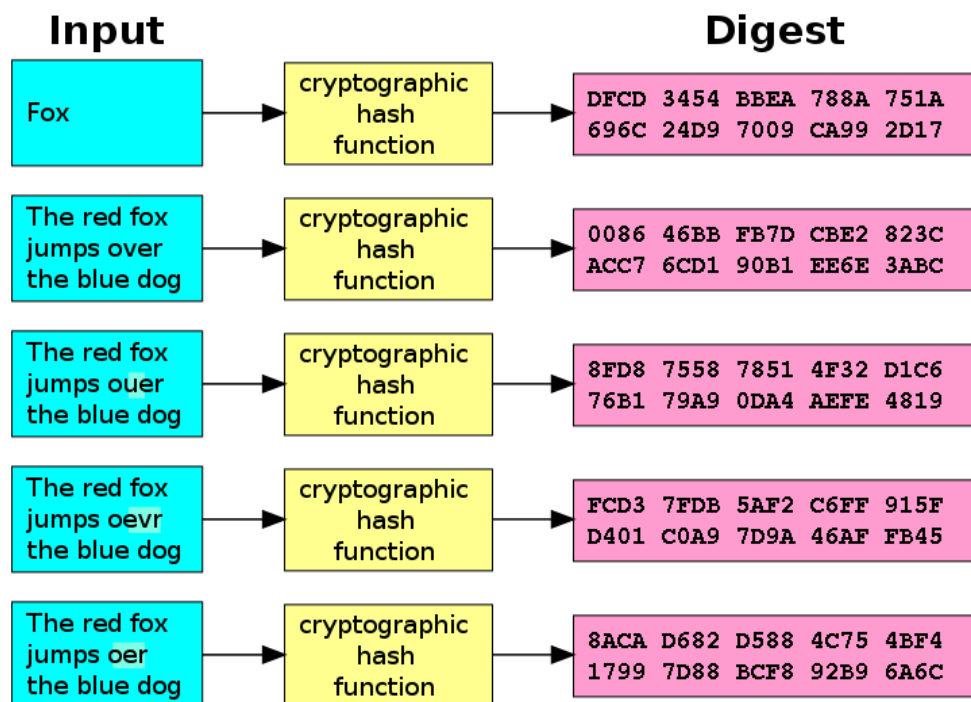


Figure 11.3: Cryptographic Hash Key Function. Notice how changes to Input alters digest (output). Courtesy of Wiki Commons.

### Salt

The random bits of data that are added to the hash algorithm, like a key for computations, is called the **salt**. A salt can be used to help create the hash value when combined with the original password, since it adds additional levels of complexity to the answer key. All this hashing and salting creates a one-way hash value. A salt is usually a string of random data from 48 to 128 bits long. The longer the salt, the higher the level of complexity and difficulty to crack.

Using a salt with the hash value makes many password attack methods slower, such as the good old dictionary attack and a friendly brute force attack. More on these attacks later in this lesson.

One of the attack techniques mitigated by salts is **rainbow tables**. With this technique an attacker pre-generates huge lists of hashes. In order to add complexity and prevent this attack, you can generate a specific salt for each user (so that an attacker would need to pre-compute a rainbow table for each user).

### Key Derivation Functions

A step forward in hash functions are **key derivation functions**. These are used to derive keys from a string, a salt and a (big) number of iterations in a formula something like this:



### DK=KDF(Key, Salt, Iterations)

where DK is the derived key, KDF is the key derivation function, Key is the original key or password, Salt is the cryptographic salt and Iterations refers to the number of times the subfunction is run.

The whole point is to make hash generation slow. But not too slow! "Fast" hash algorithms let brute-force and dictionary attacks run wild; slow algorithms make them too slow to be practical in many cases. Too slow algorithms introduce <http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-5229>.

## Super Duper Complex Cryptography

Take a deep breath and turn off your cellphone for this next section. We will do our best to avoid giving you nightmares or headaches about complex mathematics. If you want, you can just skip this section if you scare easily or are prone to fits of rage.

Cryptology means altering information or data to make it incomprehensible and unreadable unless you possess some secret knowledge. One of the ways we do this is encryption.

Back when passwords were encrypted, there were lots of places to get attacked. The effectiveness of encryption, usually described as its **strength**, ranges from very weak to extremely robust.

At the weakest, passwords were simply **encoded**. This produces a password that is not readable directly, but, given the key, can easily be translated using a computer, pen and paper, or a plastic decoder ring from a cereal box. An example of this is the **ROT13 cypher**. ROT13 replaces every letter in a text with the letter that is 13 places away from it in the alphabet. For example "ABC" becomes "NOP."

Even when using algorithms that can more accurately be called encryption, the encryption is weak if the key used to generate it is weak. Using ROT13 as an example, with the 13 place differential as the key, then ROT13 has an extremely weak key. ROT13 can be strengthened by using a different key. You could use ROT10, replacing each letter with the one ten places forward, or you could use ROT-2, replacing each letter with the one two places before it. You could strengthen it even more, by varying the differential, such as ROT $\pi$ , where the first letter is shifted 3 places; the second, 1 place; the third, 4 places; the fourth, 1 place; and so on, using  $\pi$  (3.14159265...) to provide a constantly varying differential.

Because of these possible variations, when you are encrypting any type of information, you must be sure that you are using a reliable method of encryption and that the key – your contribution to the encryption – will provide you with a robust result.

You must also remember that a good system of encryption is useless without good passwords, just as good passwords are useless without good encryption.

## Exercises

11.3 In Linux, in a terminal, create a new file using the touch command:

```
touch testfile.txt
```

Now put some text in it by using the redirect character (>):

```
echo This is a test > testfile.txt
```

Now use the MD5sum command to generate the unique hash or digest value of that file:

```
md5sum testfile.txt
```

You will get a long hash value back. Look at it carefully, then add to the text in your test file using the append operator (>>):

```
echo . >> testfile.txt
```

Run the md5sum command again. Compare the new hash value to the one you got before.

11.4 Here is a list of fruits encoded using the ROT13 cypher. Try to decode them:

- a) nccyr
- b) benatr
- c) yrzba
- d) jngrezryba
- e) gbzngb

11.5 Is there a web page that will allow you to decode the ROT13 encoded words automatically?

## Password Cracking

Password cracking for anything other than stuff you own is against the law. If it's your password, then it's your information. Once you password protect something, and then forget your password, you are stuck. Or in another common scenario, you're taking over management of a network and didn't get all the password information from the last admin. This is where password recovery becomes really useful.

Sometimes, for research or analysis, experts crack giant lists of password hashes, which is called **hashcracking**. What they're trying to find are **plains**: plain-text passwords. Methods for cracking fall into just a few categories. These include:

### Use the (Brute) Force

This is one of the most misused terms in password cracking. While in a generic sense you can say this term means trying millions of passwords until you find the one you're looking for, that's not accurate in our world. (Show people how 1337 you are by knowing what it really is.) Actually, brute-force attacks try every possible combination in the keyspace, or at least as many as it takes to find the password. They don't start with a list; they generate all the permutations on the fly. This means that brute-force attacks are very thorough on the one hand, but can take a very long time to run on the other. That's why experienced password crackers will try brute-forcing passwords of up to seven or eight characters in lower-case only, then switch to other attacks for longer and more complex passwords. You'd think people wouldn't use such simple passwords – but about ten percent of them did in a 2013 experiment by ArsTechnica.

In a brute force attack, you are basically taking a library of words, numbers, or characters and trying to match each one against the actual password. When you are looking for your socks, you can always find one clean sock but you can never find the other matching sock. So, you look through your room picking up every sock you can find and comparing it to the one clean sock you have. You are thinking to yourself, "is it this one, is it this one, is it that one" as you stumble to locate the other matching sock. Brute force works the same way, it just doesn't smell as bad.

Don't be fooled by the simplicity of this attack method. It may take up all of your computing resources and a tremendous amount of time before a match is found, if one is found at all. Be prepared to use another computer while the brute force attack is ongoing. Across the Internet you can find tools that will automate this process for you. This may be kind of them, or it may be a ruse to crack that password for themselves.

### Exercises

11.6 First you'll need a **hashdump** file, preferably a huge dump of millions of hashes for you to gleefully crack with your friends. Get lots of potato chips.

You're going to get a truly epic dump of password hashes courtesy of every user of LinkedIn in 2012. (Thanks, everyone.) Use your search engine skills to locate the file: we suggest a search along the lines of "LinkedIn hashdump and passwords" if that doesn't make it too obvious.

Download the file. Unzip it. Put it someplace sensible, somewhere that you can remember.

- 11.7 Now you need a tool that can put that hash dump to use. Go to <http://hashcat.net/hashcat/> and download **hashcat**. This is a serious professional tool, and you're expected to figure it out for yourself. Amazingly, this is in fact possible. On the same page, scroll to the bottom to the Help section and follow the Video link to a video site. There are excellent step-by-step tutorials that you can explore at your leisure, but for now look for one that's under five minutes in the first page of results. One we'll suggest is "Using Hashcat to Bruteforce Encrypted Hashes" at <https://www.youtube.com/watch?v=RMg2uaKxhdA>, which gives you a good example of the syntax:

```
hashcat-cli.exe -a 3 -m 0 -o cracked.txt -n 1 --bf-pw-min=2 --bf-pw-max=15 --bf-sc-buf=0123456789 hash.txt
```

where `-a` is the attack type, in this case 3 for brute force,

`-m` is the hash mode, in this case 0 for MD5,

`-o` is the output file for cracked passwords, in this case `cracked.txt`,

`-n` is the number of threads, in this case 1,

`-bf-pw-min=2` means that we're setting the minimum password length to two characters,

`-bf-pw-max=15` means we're looking for up to 15 characters,

`-bf-sc-buf=0123456789` means that the character space buffer holds the characters 0-9,

and `hash.txt` is the name of the file with all the hashes. Substitute the names of the real files you got in the above exercise.

You can get more information, like the numbers for different attack types and hash modes, with the command:

```
hashcat-cli.exe -help
```

See if you can crack some hashes. Particularly try different lists of characters in the character set buffer for lots of fun revelations.

## Dictionary attacks

These run through a series of possible dictionary words until one works as a password. This is slow for a different reason than brute force: each word in the dictionary has to be hashed, then compared to the hash you're cracking. Dictionary files, or **word lists**, are available all over the Internet, though the quality varies hugely. A simple word list might just list words alphabetically, but the probability of letters or words showing up isn't the same as alphabetical order, so these lists are always slow. An optimized or custom word list will put more-likely words first, meaning that the solution will probably be found more quickly. Master hashcrackers build their own word lists and guard them jealously. You won't find many or any of these online. You'll have to make friends with somebody who has them. But choose your friends carefully: some people are researchers (hackers) and some people are criminals (crackers). Don't learn the difference the hard way.

As mentioned earlier, humans tend to be creatures of habit. Users will make passwords that are easy to remember. Often, those passwords are common words found in a dictionary. Brute force uses several libraries to compare against the encrypted password, looking for a match. This type of attack works well if you are looking for your own lost password and you just needed a hint.

The key to success in using this attack is to use as large and as many different types of word lists (libraries). And by types we mean industry word lists, names, and especially languages because, you know, there's a few other languages out there besides English.



As a hacker or security professional you will need a fairly decent set of software tools, scripts and hardware to do your job properly. The interesting part of computer security is that hackers openly share their tools across the Internet, yet security companies charge big bucks for their security tools and knowledge. Open Source software is provided to anyone for free as long as proper credit is given and the software is not commercialized in anyway. Pretty cool, huh?

## Exercises

- 11.8 Now you need some word lists. Openwall offers several sample word lists at [http://download.openwall.net/pub/word\\_lists/](http://download.openwall.net/pub/word_lists/). Go there and download the all.gz file, which contains (you guessed it) all of the word lists.

Linux and OSX can unzip Gzipped files natively. In Windows you'll need an add-on compression tool like 7zip (<http://www.7-zip.org/>). Get it if you need it. Unzip all.gz and put the output in a logical place.

- 11.9 Now you need a tool that can put those word lists to use. This time we're going to use John the Ripper, so you can get experience with more than one tool. Remember that Hashcat could do the job too.

We'll start with software password crackers. **John the Ripper** at <http://www.openwall.com/john/> is at the top of the list for its success and longevity. John works on several platforms (across platforms) such as UNIX, Windows, OpenVMS, and BeOS. If you build a live CD or bootable USB drive, you must include John because it's that good. Trust us. John is an Open Source package and has been worked on over the years by teams of volunteers.

Get it. Install it. If you're using the Fedora Security Spin, you've already got it.

- 11.10 Open your new friend John and see what you need to do to crack passwords from your list.

## Somewhere Over the Rainbow

How do you speed up a dictionary attack? Well, how about pre-hashing all those dictionary files? That would sort of make sense, until you remember that there are lots of different hashing algorithms out there. What you really need to do is hash each password with several different algorithms, and store the results in a table with a column for each algorithm. These tables, with the plain and the hashes of different fixed lengths, looked like a rainbow to somebody somewhere, and that's how we got the name. **Rainbow tables** are an example of **space/time tradeoff**. In this case, the tradeoff is between huge (and we mean really huge) use of disk space, to save huge amounts of time during actual cracking runs.

In practice, creating rainbow tables of entire word lists is impractical precisely because of the issue of file size: you'll fill up that terabyte drive fast. So hashcrackers do something different, something very, very smart. They crack the hash by cracking small parts of it at a time, then assembling the parts. This is wildly clever and intensely mathematical. If you are a hard-core math geek you are gonna love this stuff.

One logical attack is to look for repeated values in calculations. Using a hash (with salt) the word "Stun" could look like **dr23 n9n2 8v84 2lwi**. So, why not look for a pattern in other hash calculations to see if you can find a match?

Rainbow tables do a nice trick. Instead of looking for a whole password (or whatever it is you're trying to decrypt), rainbow tables let you look for fragments or pieces of passwords. Rainbow tables aren't exactly tables; they are a collection of columns in which each column uses a different reduction function. If you assigned each a color you could think of the columns as a curve, or a rainbow. Hence the name rainbow tables. The whole idea is that if the calculated values of any two "colors" matches, you've found part of the password.



From a mathematical perspective, try and wrap your mind around a simple formula of

$$v+x-4= 162$$

both  $v$  and  $x$  could be any combination of numbers as long as the answer equals 162.

The rainbow tables already have hash values computed and any input you provide (to crack a password) is matched against each column (reducing the values) until a perfect match is found. As we saw with "Stun" above, the hash calculation would look like this: **dr23 n9n2 8v84 2lwi**. If you were to apply this dr23 n9n2 8v84 2lwi string into a rainbow table automated lookup generator, every two digits are analyzed and reduced to find a matching character. While this lookup is occurring, each column is reducing your input, thus speeding up the match finding process. Simple stuff, right?

All the rainbow tables are doing is looking for the answer  $v+x-4= 162$ . The  $v$  and the  $x$  values are the unknown factors and the tables have already been calculated to find that type of answer, just on a massive scale.

Since hash values are one-way, it is impossible to reverse engineer any hash string and expect to get a correct answer. Think of a hash calculation as a door that only opens in one direction and traffic can only flow in that same direction. It is highly likely that hash values will be the same somewhere along the way and that is how rainbow tables operate. They compare hash calculations that you provided against a massive collection of predetermined hash values. As the two digit set moves through the table, each column reduces the answer until a match is found.

## Exercises

### 11.11 Rainbow Tables

Get some rainbow Tables from <http://ophcrack.sourceforge.net/tables.php> for Windows XP, Vista or 7.

## L0phtcrack

L0phtcrack is a powerful commercial password auditing tool. Help can be found on the official website at <http://www.l0phtcrack.com/help/using.html>. See if you can afford it.


## Combinator Attacks

Unfortunately, rainbow tables have a weakness: they're easily defeated when systems use a salt. We'll talk about those below, but the gist is this: when the user creates a password, the system also creates a random salt (a string of a few letters to add to the password), combines password and salt, and only then creates the hash. Every user's salt is unique to them. The salt can be stored unencrypted in the same table as the password hashes, in theory. At least that's the way they used to do it. Obviously it's way too easy to steal both the hash and the salt, if they're in the same table, so best practices seems to tell people to store the salts in a separate table, or in a separate database, or on a separate machine. But that also increases the attack surface and makes new potential vulnerabilities through complexity. In any case, the added salt makes the original password much harder to derive from the hash, effectively defeating rainbow table attacks. Fooyey.

Okay, you broke our rainbow tables? Fine, we'll do something else smart. We specialize in smart. How about we try something like this:

1. First, we've all got machines with powerhouse graphics cards, right? Good: now,
2. Run a brute-force attack for lower-case-only passwords up to six characters, and
3. Repeat for upper-case-only passwords up to six characters. This lets us keep passwords of six characters or less out of our word lists; brute force can test literally all possible combinations.



- 
4. Then run a dictionary attack using, of course, highly optimized word lists. Let this run for at most an hour, so we capture the most likely passwords.
  5. Next, let's use that same word list again, but truncate all the words down to seven characters or less, then put a 1 at the end. Try again with a 2 at the end. Try again with a bang (!) at the end. Run all these trials only until cracked-password output slows down.
  6. Chop the words in the list down to six and try brute force again with up to two digits or special characters, then three. If you've got a hot graphics card try four, otherwise at the three or four digit length switch to digits only (no special characters). Appending the output of a brute-force attack to the words in a list makes this a **combinator attack**.
  7. Transform every password in the word list by making its first character upper-case. This is where people put upper-case letters: at the front of the password. Taking advantage of this kind of pattern turns a brute-force or dictionary attack into a **mask attack**, an especially deadly attack that reduces the keyspace dramatically so that cracking happens much more quickly.

We can run each of these attacks for only as long as they produce lots of plains quickly, then jump to the next attack. We want lots of plains today, not all the plains next month. We're doing research, or we're in a competition, right?

### Location, Location, Location

Say you're having a security contest with some friends. If you have physical access to your target's computer, **try looking around**: passwords are often taped to the bottom of keyboards, under mouse pads, posted on personal bulletin boards, in calendars or daily planners, wherever is handy. People tend to post their passwords in plain view but try to be tricky by writing the password backwards. It is not that people are lazy, they just to make their lives as uncomplicated as possible.

There are two commonly used methods to recover hash values in Windows. The first one is an injection using Local Security Authority Subsystem Service (LSASS) and the other is taking a direct reading off of the registry using SAM/System. LSASS reads the hash directly from memory, while SAM reads the local registry hives. There are other methods that can be used to gather hash values but LSASS and registry reads have a higher probability of success. Then the attacker can research and crack those hashes at his leisure off-line. That's especially handy because the attacker can gather several (if not all) the hash strings in a single breach.

### Network-based Password Cracking

What about those times when you don't have a password hashdump, but need to recover the password to a database, web application or some other online service? This is when you need a **network password cracking** tool.

#### THC Hydra

A speedy network authentication cracker which supports plenty of different services, Hydra is at <https://www.thc.org/thc-hydra/>. When you need to brute force crack a remote authentication service, Hydra is often the tool of choice. It can perform speedy dictionary assaults against more than 30 protocols, including telnet, ftp, http, https, smb, several databases and many more.



Figure 11.4: Selecting a target in Hydra

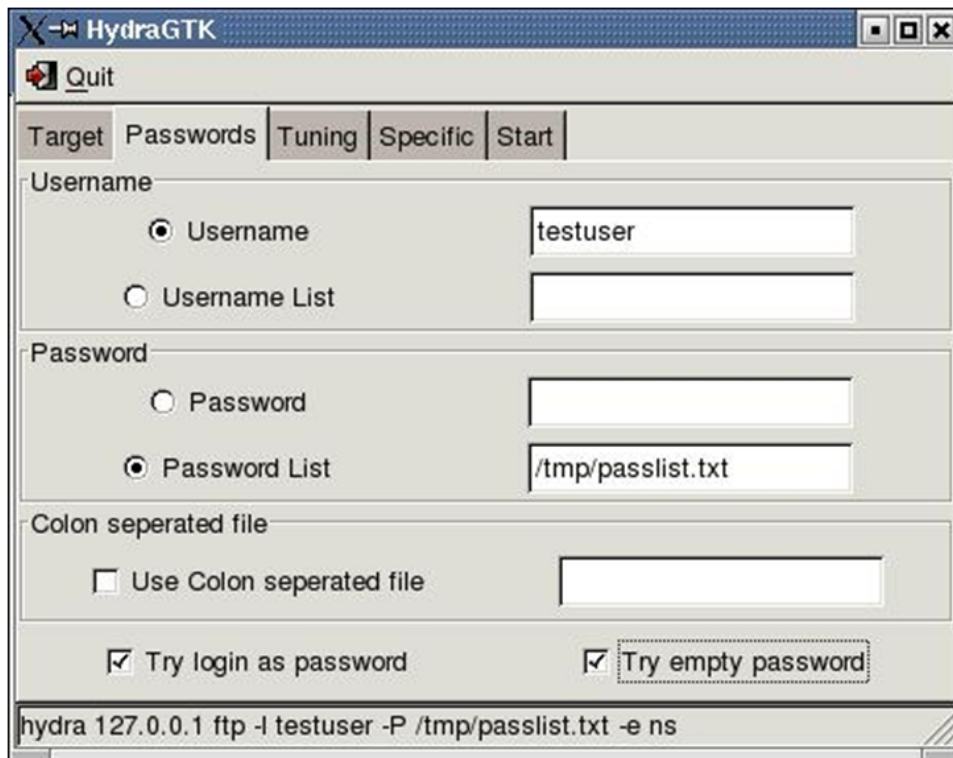


Figure 11.5: Setting up Username and Password lists in Hydra

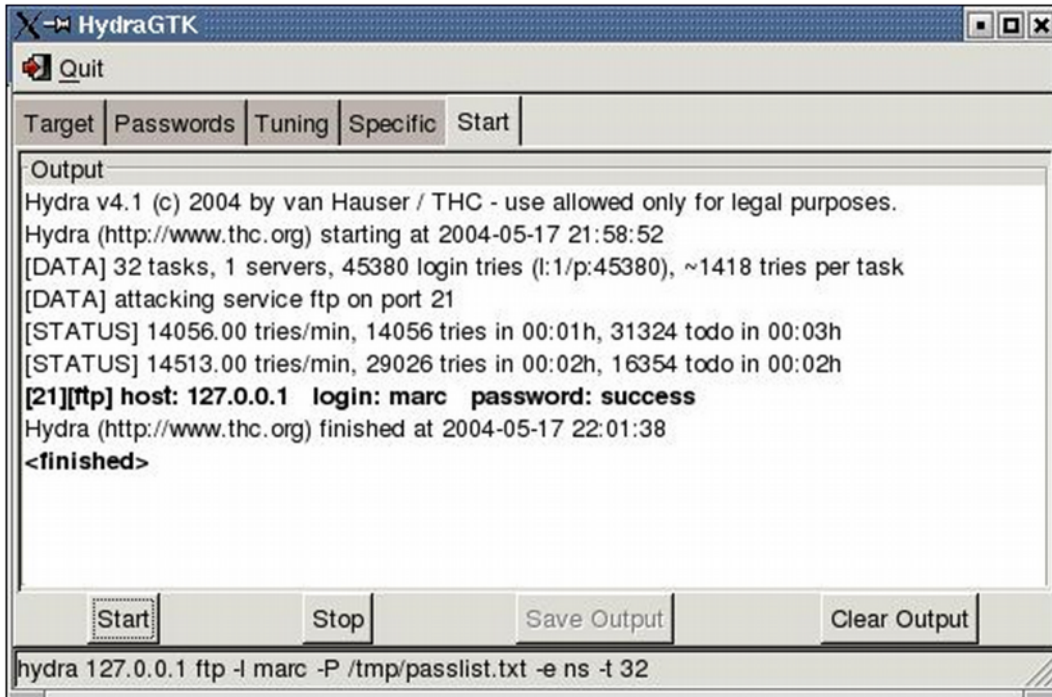


Figure 11.6: The results of a password cracking run in Hydra

### Brutus

Brutus is a network brute-force authentication cracker, available at <http://www.hoobie.net/brutus/brutus-download.html>. This Windows-only cracker tests network services on remote systems, trying to guess passwords by using a dictionary and permutations (see Combinator Attack above). It supports HTTP, POP3, FTP, SMB, TELNET, IMAP, NTP and more. Because no source code is available, this is not a truly open-source tool, but it's a very popular one.

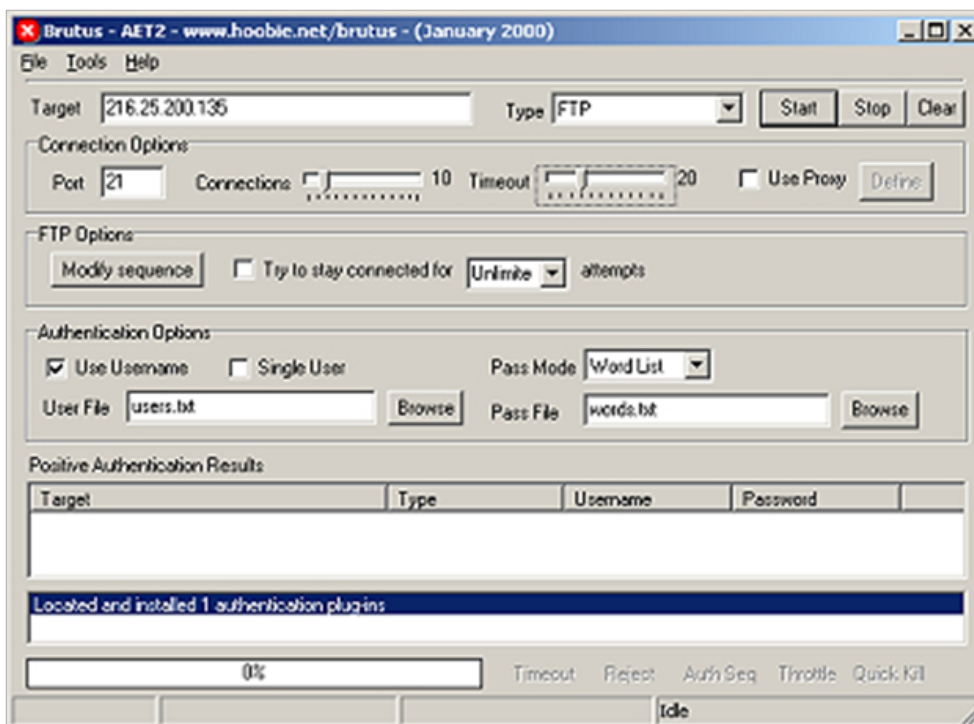


Figure 11.7: Selecting target and options in Brutus

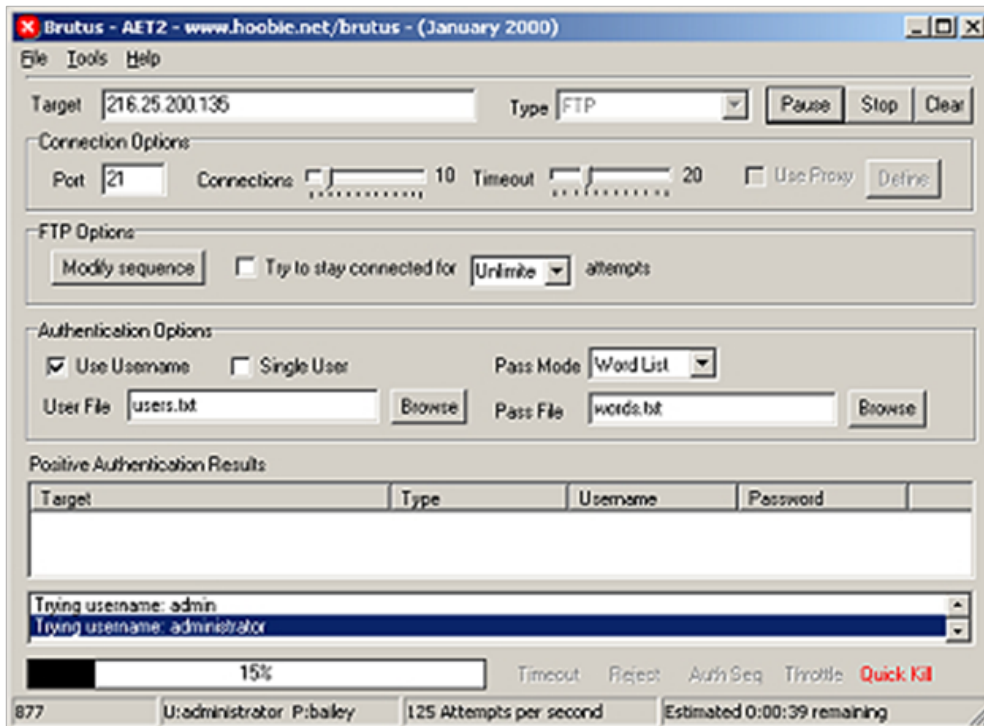


Figure 11.8: A run in progress in Brutus

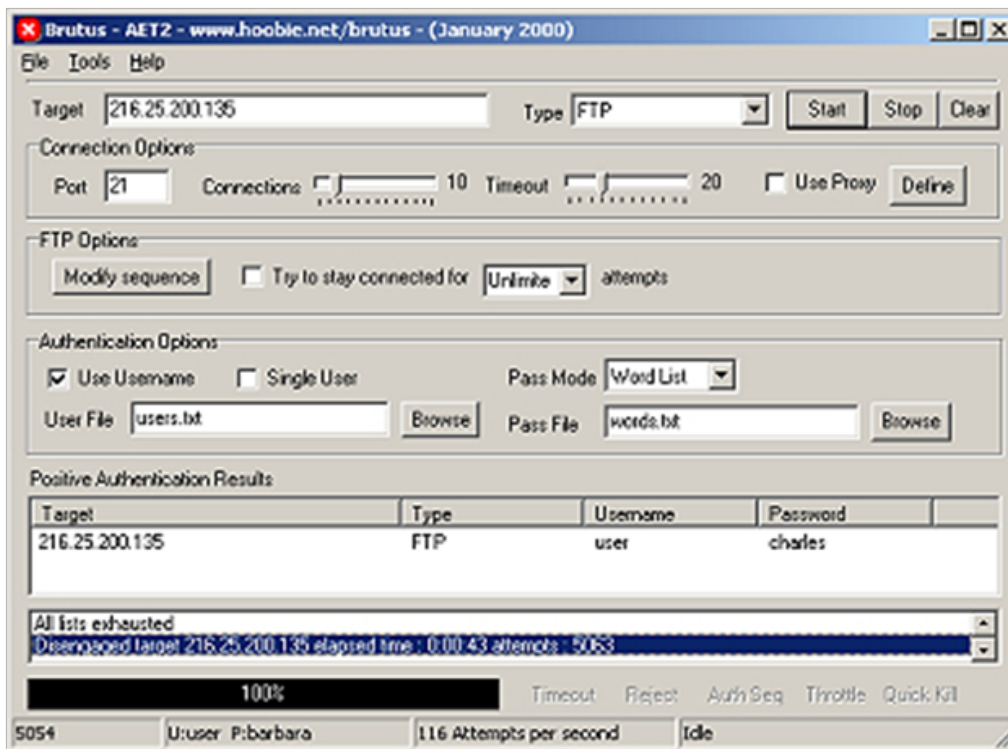


Figure 11.9: A completed test run in Brutus

## Wireless Password Cracking

Aircrack-ng - <http://www.aircrack-ng.org/downloads.html>

Once upon a time there was **Aircrack**, an open-source wireless security testing suite. It was good, but this field changes fast, so along came the next-generation suite (which is a fork, or split-off project, of the Aircrack project) called **Aircrack-ng**.

Aircrack-ng contains several tools: **airodump-ng** (a packet capture program), **aireplay-ng** (a packet injection program), **aircrack-ng** (for static WEP and WPA-PSK cracking), and **airdecap-ng** [decrypts WEP/WPA capture files]. The suite can recover a 40-bit through 512-bit WEP keys once encrypted packets have been gathered. It can also assault WPA or WPA2 networks using advanced cryptographic methods or by brute force.

It needs a wireless network interface controller with a driver that supports raw monitoring mode (injection), and can sniff 802.11a, 802.11b and 802.11g traffic. There are versions for Linux and Windows; the Linux version has been ported for **OpenWrt** (a popular open-source wireless access point operating system) as well as the Android, Zaurus and Maemo platforms.

```
C:\aircrack-ng-0.4.2-win\bin>aircrack-ng.exe -a 2 -w dict capture2.cap
Opening capture2.cap
Read 607 packets.

# BSSID          ESSID          Encryption
1 00:06:25:BF:64:99 cuckoo          WPA <1 handshake>

Choosing first network as target.
```

Figure 11.10: Cracking a WEP key with aircrack-ng

```
[00:00:25] 4090 keys tested (160.75 k/s)

KEY FOUND! [ passphrase ]

Master Key      : CC 9D 81 0B 93 70 BE 17 BD 60 18 2E D0 D9 11 EB
                  E7 51 BD 15 4D 92 30 87 3F BF FC 32 04 D2 F5 1B

Transcient Key : 7A C7 4A 43 65 48 0E 21 68 66 2D A8 01 FB 29 37
                  C5 2A A2 3A 78 8F 85 24 F8 A2 26 03 CA 62 43 88
                  03 F3 9B 7D 1F D0 D0 95 DC 83 51 54 69 CB 96 0A
                  24 36 82 C4 80 68 A2 1C A4 E4 9E 2C A7 28 D8 98

EAPOL HMAC     : C3 D0 6C 14 EC B7 74 20 62 05 A0 55 88 38 E8 DB
```

Figure 11.11: A successful WEP key crack

### Exercises

11.12 Are you familiar with null-byte on WonderHowTo.com? Get familiar. Go here:

<http://null-byte.wonderhowto.com/how-to/hack-wi-fi-cracking-wpa2-psk-passwords-using-aircrack-ng-0148366/>

Read this step-by-step tutorial. If your instructor can supply a wireless access point as a lab target, or you have your own, see if you can crack its key.

11.13 This article mentions another tool: coWPAtty. Find the linked article and read it. How does coWPAtty compare with aircrack-ng?



## Cracked Up

For serious password cracking you will need some additional tools. One of these tools is your average graphics card, known as a GPU for Graphics Processor Unit. These cards are designed for repetitive tasks unlike a CPU, which is designed to all kinds of tasks. Even though your computer, smartphone, tablet and other devices have GPUs built into them, those chips are unlikely to be programmable by the user. They are locked chips that only accelerate the graphics for that device. The type of GPU needed to crack passwords (or mine bitcoins) are higher end gamer cards. A couple of these cards in a computer can crack passwords dozens of times faster but also costs more money.

You will want to take a look at either nVidia Ge-Force or Radeon cards made by AMD. They are similar in speed and price with best results obtained by using several mid range cards in tandem. Performance increases significantly when multiple cards are used instead of a single high end GPU. Some experts argue that the AMD cards are superior to the Ge-Force cards based on the usage of integer instructions in Radeon GPUs. It will be up to you to research which card offers the best password cracking benchmarks if you are going that route. This also applies to the device that you intend to use for your work.

Computers have ample computing power because they have plenty of power available. This isn't the case when it comes to laptops, smartphones, tablets or minicomputers like Raspberry Pis and Beagle Bone Boards. In these cases, you will need to do your homework to locate devices that already have these chips installed. Minicomputers allow you to add on capes, shields, or boards where you can customize your own GPU but power will still be a major consideration. High end gaming laptops have this same issue because they are built for power and speed, not portability. It is the same with any device running a graphic or computing intensive program, you will need plenty of juice. This can be overcome by having a hot swappable battery or a portable battery charger like the Anker. That thing can jump start a car.

Older computers can make great password cracking machines with a few low cost modifications. First of all, you will need to replace the power supply with a much larger one because we've been talking about power for the past three paragraphs. Next, the motherboard will need to handle several fullsize graphics cards, at least three, more if possible. You won't need to worry about the CPU or other hardware because this machine will only be used to crack passwords or mine bitcoins. Heat and ventilation will be another consideration but that can be taken care of by drilling holes in the case, adding cheap fans, clearing out unnecessary hardware, or installing a liquid cooling system. Above all, keep the computer in an area that has good airflow.

If your computer motherboard can't fit the GPU cards, you can always build an additional box (old shoebox or a wooden crate) and add PCI extension cables along with power cords to link them to your machine. Linux OS will be your best bet thanks to the highly customizable interfaces available but Windows will work too.

Hardware may sound unlikely for cracking passwords, but there are amazing ways to use devices in other ways. Video cards were designed to display better graphics than your CPU can, and free up the CPU for other tasks. An interesting thing happened on the way to better video performance: a group figured out that NVidia Cuda and certain ATI Radeon graphics cards would crack passwords. Of course, this required designing software for the purpose of hacking passwords, but it sounded like a great challenge.

Certain video cards provide amazing data acceleration to break strong encryption. A video card really is just one data accelerator after all. With a little tweaking and fine tuning, hackers have stacked (parallel) bunches of video cards together. Students at MIT brought this data acceleration to a whole new level when they added the computing power of the Sony Playstation. You read it right. Using a parallel pattern (like stacking columns of bricks instead of one single column), the students demonstrated a machine that could crack several encryption algorithms that were once thought too complex to break.

This is what hacking is really about, taking technology and making that technology better or creating something entirely new out of it. Real hackers are more like inventors, always



trying to tweak this or that and building new ideas or gadgets out of other electronic or digital parts. Too often the word “hacker” is used when people mean “criminal.” If a carpenter breaks into your house with a hammer, is he a carpenter or a criminal?

## The Soft Side

Thanks to the ever increasing computing power of technology, password cracking has gotten much easier. People are often asked to provide an email address to create their username, which makes the first part of hacking an account that much simpler. Since most people use the same email account for all their work, knowing one email account means you probably know the username for every account that person has. Having the username, your next step is to figure out their password. We've talked about human conditions and the flaws of passwords where most people tend to use the same for every account they have. The average person has twenty five accounts.

One of the best software programs out there for password recovery is **oclHashcat** available at <http://hashcat.net/oclhashcat/>. This tool runs on both Linux and Windows and is aimed at either AMD or nVidia's graphics cards. Besides having control over almost every aspect of decrypting passwords, it also provides a watchdog to ensure you don't fry your GPU due to over temperature. Hashcat rightfully claims to be the fastest GPU enabled password cracker that is free. There are limitations such as it cannot crack Truecrypt 6.0 or above but other than that, the software has an amazing track record of recovering lost passwords.

New hash lists can be found dumped at Pastebin.com and other sites. The largest collection of plain text passwords was from a 2009 breach of an online gaming company called Rockyou.com. This is a database of 14.3 million passwords and can be found on several password collection sites. Hash lists have already been run against this massive collection and you can usually find both the password and MD5 hash list on the same site. If you look around long enough you will find over 100 million cracked passwords available online. One of them is probably yours.

## Exercises

11.14 Do some research into the pros and cons of LSASS versus gathering hash strings using registry hives. Which method would you use if the target server was running Windows 10 server? Hint: DEFCON.

## Build a Great Password

---

The best passwords:

- cannot be found in a dictionary (any language dictionary)
- contain numbers, letters and those odd symbols on top of the number keys
- contain upper and lower case letters
- are longer – literally, the longer the stronger
- can be remembered and not written on something that can be easily found



## Feed Your Head: Cor Issues

Cor's a heavy thinker in the security world and doesn't take facts as fact. He ponders them. Then he ponders them some more. So when Cor writes you to tell you he has some doubts, you can learn as much from his thought process on breaking down a problem as you can about what he's doubting.

When Cor wrote us at Hacker Highschool about the Password Lesson draft, he shared with us he thoughts on passwords. (We stripped out the formalities to get to the grit for you because I'm sure you really don't want to read Cor remarking on Pete's strange trust issues with origami or Cor's feelings about Pete always wearing the exact same shirt in all his conference presentations for the last 10 years. Well, I guess we did tell you after all. Never mind then.)

So pay attention to Cor's process and what he has to say about password complexity and biometrics as passwords:

"Strong passwords are long. That's it. They have to be long. This is the main thing. Further, to allow for more complexity, you should allow as many characters in the character set as possible and no restrictions. The restriction reduces the keyspace."

### *This Leads to the Bad Idea of Complexity Rules*

"It explains why required complexity reduces the password strength. In short, the requirements eliminate perfectly safe passwords because of lay-out. Now I have to say that complexity rules are invented for a reason. That reason is laziness (or stupidity or both). Many people choose weak passwords because they are easy to enter (123456, qwerty, qazwsx, 123qwe, 000000, etc.) or easy to remember (pet's names, kid's names, favorite brand, favorite sports team, dictionary words etc.). To prevent such "weak" passwords someone had the bad idea to require complexity in the passwords. Others had an even worse idea to like this and spread the idea. You can read the explanation below and do the math yourselves to find out that such rules eliminates thousands of billions of perfectly fine eight character passwords."

### *Bad Passwords are Still Better than Biometrics*

"How many easy-to-enter or easy-to-remember bad passwords do you think exists? If you add all simple keyboard sequences, pet names, kid names etc. you can think of and add the English, Dutch and Pakistan word lists together, I doubt if you even reach as much as a million. So how about getting rid of complexity rules and allow all combinations, except the ones on the black list? That black list doesn't have to contain a million existing words and names, it'll do when it contains the top 1000 most used passwords in the leaked password lists that are widely available nowadays."

"Even if you only put the top 100 most used passwords on the black list, the chances that someone brute forces the right password is less than when you use biometrics. You probably all read recently that a phone's fingerprint reader can be fooled by latex fingers (reproduced from fingerprints left on a cup). Realize that affordable biometric equipment had an accuracy worse than 1% in 2011. I suppose that with nowadays technology this increased to 0.1%. This is still pretty inaccurate. A single try on a three digit password gives the same chance to gain access. Therefore today, biometrics are only useful as a second factor in authentication. This might change some time in the future when accuracy has raised to a sufficient level. I'm not sure if that'll happen during my lifetime (and I intend to live at least for another half a century)."

*So what do you think about what Cor has to say? Is he right? Discuss this. Or don't and*



*just think about it.*

## Size Matters

The longer and more complex the password, the better the password. The best way to do this is to take several words that normally do not occur together to build a passphrase that you can easily remember. For example: This year, my favorite sports team has been playing especially poorly -- they're a "dog." Today, the builders started pouring the "foundation" for the house next door. This happens to be during the "summer." And perhaps I like to play "guitar." The passphrase becomes " DogFoundationSummerGuitar." Notice the mixed case, that is, both upper case and lower case letters.

Now let's add some more complexity by adding some characters and substituting, so there are no "dictionary" words. If we use digits 0 - 9 and special characters (shift-numbers), we can do substitution like zero-for-capital-O. Digit 1 for letter "l." @ or 4 for letter "a," either case. And, to make it better, change the rules for substitution. The first time there is a character "a," use "4." The second time, use "@." And so on. Mangling the passphrase this way, we might end up with:

```
g#ounD@t!on$umM3r&u17ar.
```

If that's too hard, you might split up the words and add numbers and punctuation marks, like this:

```
!Dog#Foundation$Summer&Guitar*.
```

Some people develop a reasonably complex passphrase they use every time they need a password, but to make it unique to that system, website or application, will prepend (add at the beginning) something related to that website. For example, you might want a long password for Facebook. People post all kinds of things to Facebook; they're a bunch of clowns. So:

```
Clowns!Dog#Foundation$Summer&Guitar*.
```

People tell you "don't write your passwords down." Unfortunately, you quickly have more passwords than you can remember. So: get a good encryption program. Choose a decent passphrase for encrypting this collection of files. The passphrase should not be similar to or correlated with your other passwords. Keep an encrypted text file with your user names and passwords. Only open it when you need to remember a password. Keep an encrypted backup of that file somewhere (like a USB drive), so if your disk crashes, you don't lose all your passphrases.

And don't call your password file "Passwords." Give it a different name like "Grocery List" or "urinary tract infection dates." Just don't call it passwords.

Or just use a good password manager.

## Other Methods

There are many password generators available on the Internet, but these will generate a nearly impossible to remember password.



Try instead to use a seemingly random string of letters or numbers that you can easily recall.

For example:

gandf3b! (goldilocks and the 3 bears!)

JJPL2c1d (john, jill, paul, lucy, 2 cats, 1 dog – the members of your household)

Or you could create your own algorithm to generate passwords. An algorithm is the designation of a step by step procedure for a calculation. Here's an example for you.

This algorithm for choosing a strong password will be the following:

- The last 2 digits of a birth year
- The first 3 letters of a first name, with the last one being in uppercase
- the symbol '!'
- 3 letters to define the service you could registering in

So, that is the algorithm. Let's build a password for Facebook and Gmail, shall we?

A password for Facebook, following this algorithm would be: 84maR!Fac

The password for Gmail, following this algorithm would be: 84maR!Gma

Easy? That way we just created a 9 character password that contains numbers, letters (lowercase and uppercase) and special characters.

This way you have strong passwords and a different one for every service. Of course, everything has a weakness. The password is only as strong as the protection for the algorithm and the length of the characters. If someone discovers your algorithm, it would mean that all your passwords would have been compromised or potentially stolen.

Search online for password managers. You have several free options that you can try out. The password managers are computer applications that you protect with a master password, and can keep all your other passwords in there so you don't have to remember all of them. That way, it's easy to have strong passwords for everything that is important.

## Check, Please

So, you think you have the best password ever created. Your password will withstand ten thousand years of attacks because you built it using the rules we've outlined above. You're a great student of Hacker Highschool, so you applied your knowledge to make an unbreakable password. Before you celebrate, you want to see how well it stands up to a password checker.

Like anything built for quality, you need to test your great creation. How do you do that? You can beat on it with Javascripts that you can trust to apply maximum pressure to your device. Below you will find one of the best password checkers available, and it's free. Go ahead, don't be shy. Hey, why are sweating so much? You look a little nervous.

One key to being a successful security professional is being able to test your own designs. Try and break into your servers or web pages. Locate the weaknesses and correct those vulnerabilities. The same principle applies to passwords. You have to test everything to its limits, no holding back, right? Other attackers aren't going to hold back either so why should you?

**Password Strength Checker:** <https://www.microsoft.com/security/pc-security/password-checker.aspx>





## Conclusion

It has always been difficult to bridge privacy and confidentiality with ease of use for any form of communication. Remember that the Internet was never designed with security in mind. The Internet was design for freedom of information flow to be its primary purpose, even in the event of a failure within the network. Fail-safes were built in to ensure data continued to reach its destination even in the event of a nuclear war. This massive network was built back in a time of flowers and peace signs everywhere but the Soviet Union and United States posed a serious threat to the world. It was called "Mutually Assured Destruction" or MAD because of the nuclear arsenal both countries possessed. But that is another story for another time.

Passwords were introduced when people wanted some sort or privacy on their digital communications. It was an easy and inexpensive band-aid to apply in Unix environments. The young Internet was built to connect university academics, researchers and DoD agencies who funded that research. A username and password allowed a certain amount of security to this fundamental network.

Over time, other commercial organizations and private entities plugged into the Internet, thus expanding the capabilities of this data sharing network. Each company wanted to keep their data separate from prying eyes so passwords were also introduced because they were already part of the Internet culture.

As far back as the 1960's academics and researchers brought up the vulnerabilities of passwords and many advocated for stronger authentication measures. Passwords were inexpensive and easy to use. Year after year password use never faded as it was supposed to. Most security experts fully expected passwords to be replaced with a stronger method. As you can see, that still hasn't happened. The culture of insecure practices continues because it's easier than fixing the problem.

Some powerful organizations have spent millions trying to devise ways to replace passwords with other systems. IBM created a database of personal questions that could be asked before access would be granted. Microsoft worked on using pictures that the user would recognize to gain account privileges. One company came out with voice recognition that was easily tricked using phone conversation recordings. Biometrics has been the Holy Grail of password replacement even though each attempt has been hacked using simple techniques.

In this lesson we showed you lots of information about passwords and how they work (or don't work). Words like "entropy" might not mean much to you at the moment but you will heard it again in your career as a security professional. Hacker Highschool is all about getting you prepared for the real world of cyber security the ISECOM way. You were shown the inner workings of CAPTCHA and how it can be bypassed. Multifactor authentication is another large word thrown around by security vendors to sound cool but it just means there are a couple of different ways to prove trust.

Hacker Highschool expects you to be asking questions about the world around you. We demand that you poke around and finker with things. Questions you should be asking when it comes to passwords include:

How are passwords stored?

Where are they stored?

Are they transmitted or is just the hash transmitted?

Are default passwords hardcoded into systems?

Is there a more intelligent method for brute force password attacks?

Do passwords reset when a system is upgraded or updated?

Be curious about how things have always been operating. Don't expect a book or the Internet to have all the answers. Sometimes you'll have to learn the hard way, by making a mistake. That is okay, though. Experience is gained through mistakes.



Throughout this lesson you were peppered with lots of **Feed Your Head** segments. This was done to highlight some of the highly technical (or really stupid) aspects of passwords. either way, you should have learned something new. We also discussed biometrics, attacks, hacks, habits, key functions, rainbows, wifi cracking, key sizes, hashes and all kinds of other words not usually found together.

Depending on how passwords are implemented, they can either be complex or rudimentary. It is your job to make systems more secure but not more difficult for the user. This isn't an easy task but one we know you are ready for.

Today's teens are in a world with major communication and productivity channels open to them and they don't have the knowledge to defend themselves against the fraud, identity theft, privacy leaks and other attacks made against them just for using the Internet. This is the reason for Hacker Highschool.

**The Hacker Highschool project is the development of security and privacy awareness learning materials for junior high and high school students.**

Hacker Highschool is a set of lessons and a practical means of making hackers. Beyond just providing cybersecurity awareness and critical Internet skills, we need to teach the young people of today how to be resourceful, creative, and logical, traits synonymous with hackers. The program contains free security and privacy awareness teaching materials and back-end support for teachers of accredited junior high, high schools, and home schooling. There are multiple workbooks available in multiple languages. These are lessons that challenge teens to be as resourceful as hackers, including safe Internet use, web privacy, researching on the internet, avoiding viruses and Trojans, legalities and ethics, and more.

**The HHS program is developed by ISECOM, a non-profit, open-source research group focused on security awareness and professional security development and accreditation.**

# Hacker Highschool

## SECURITY AWARENESS FOR TEENS



### LESSON 12

# INTERNET LEGALITIES AND ETHICS



## “License for Use” Information

The following lessons and workbooks are open and publicly available under the following terms and conditions of ISECOM:

All works in the Hacker Highschool project are provided for non-commercial use with elementary school students, junior high school students, and high school students whether in a public institution, private institution, or a part of home-schooling. These materials may not be reproduced for sale in any form. The provision of any class, course, training, or camp with these materials for which a fee is charged is expressly forbidden without a license including college classes, university classes, trade-school classes, summer or computer camps, and similar. To purchase a license, visit the LICENSE section of the Hacker Highschool web page at [www.hackerhighschool.org/license](http://www.hackerhighschool.org/license).

The HHS Project is a learning tool and as with any learning tool, the instruction is the influence of the instructor and not the tool. ISECOM cannot accept responsibility for how any information herein is applied or abused.

The HHS Project is an open community effort and if you find value in this project, we do ask you support us through the purchase of a license, a donation, or sponsorship.

All works copyright ISECOM, 2004.





## Table of Contents

"License for Use" Information.....	2
Contributors.....	4
12.1. Introduction.....	5
12.2. Foreign crimes versus local rights .....	5
12.3. Crimes related to the TICs .....	7
12.4. Prevention of Crimes and Technologies of double use .....	8
12.4.1. The global systems of monitoring: concept "COMINT" .....	8
12.4.2. "ECHELON" System.....	9
12.4.3. The "CARNIVORE" system.....	9
12.5. Ethical Hacking.....	11
12.6. The 10 most common internet frauds.....	12
12.7. Recommended Reading.....	14



## Contributors

Francisco de Quinto, Piqué Abogados Asociados

Jordi Saldaña, Piqué Abogados Asociados

Jaume Abella, Enginyeria La Salle (URL) – ISECOM

Marta Barceló, ISECOM

Kim Truett, ISECOM

Pete Herzog, ISECOM



---

**Universitat Ramon Llull**



## 12.1. Introduction

New technologies, while building a new paradigm that invades every human activity, also influence the dark side of these activities: criminal behavior of individuals and of organized groups.

For this reason, we have reserved the last lesson of HHS to analyze some aspects related to Legality and Ethics, analyzing several behaviors that could end in crimes and the consequences of these crimes.

## 12.2. Foreign crimes versus local rights

As noted above, the introduction of new technologies can result in the creation of new dark sides of activities: criminal behavior of individuals or organized groups. There are two main characteristics through which Information Technology and Communications (TIC's) are related to crime:

1. Technologies can give the possibility of renewing traditional ways of breaking the law. These are illegal activities which traditionally appear in the penal codes, but are now being attempted in new ways. Examples include money laundering and illegal types of pornography.
2. In addition, because of their own innovation, TIC's are resulting in the appearance of new types of criminal activities, and because of their nature, these new crimes are in the process of being added to the legislation of several countries. Examples include the distribution of spam and virus attacks.

Another characteristic of the TICs which must be emphasized is their territorial displacement, which affects the general surroundings but without any doubt affects other countries as well. Previously, areas of 'law' always had a clear territory regarding the judicial authority judging (COMPETENT JURISDICTION) and also regarding the law to be applied in the judging (APPLICABLE LAW). Both concepts are still noticeably geographic.

In summary, we can say that the TICs are global and essentially multi-border, while the law and the courts are limited to a specific state or territory. In addition, this disorientation is even more confusing than it initially appears. Although we are not aware of it, a bidirectional online communication between a user in Barcelona and a Web site hosted in an ISP in California can pass through more than 10 ISPs, hosted in a variety of remote points around the world. Facing this diversity of addresses and nationalities, it becomes necessary to ask *What laws of which country will be applied in case of litigation? Which of the possible countries will be the suitable court to adjudicate the case?*

The relatively recent European Council's agreement on cyber-crime was signed in November 2001 in Budapest by almost 30 countries, including the 15 partners of the European Union, the United States, Canada, Japan and South Africa. This agreement intends to restore the TERRITORIAL PRINCIPLE to define competent jurisdiction. The signing of this agreement is the culmination of four years of work that have resulted in a document containing 48 articles that are organized into four categories:

1. Infractions against confidentiality
2. Falsification and computer science fraud
3. Infractions relative to contents
4. Violations of intellectual property



Once the especially complex regulations and sanctions on criminal activity on the Internet have been described, consensus must be reached on three main areas of concerns or difficulties:

**1st DIFFICULTY: JURISDICTION CONFLICT.** Election of the most competent court for judging multinational and multi-border crimes. This problem is not definitively solved by any of the known judicial systems.

**2nd DIFFICULTY: CONFLICT OF LAWS.** Once the court has been chosen, the first obstacle that the court will encounter is choosing the law applicable for the case to be judged. Again we are forced to conclude that traditional legal criteria are not designed for the virtual surroundings.

**3rd DIFFICULTY: EXECUTION OF SENTENCE.** Once the competent court has determined a sentence, the sentence must be carried out, possibly by a different country than the country which dictated the sentence. Therefore, it is necessary to have an international commitment to recognition and acceptance of any sentences imposed. This problematic issue is even more complicated to solve than the two previous ones.

These complications were clearly demonstrated in the recent case of a hacker in Russia, who had hacked several US systems, and was invited to a phony US company for an interview. During the interview, he demonstrated his skills by hacking into his own network in Russia. It turned out that the interview was actually conducted by the FBI, and he was arrested. The FBI used sniffers placed on the interview computer to raid the hacker's computer in Russia and download evidence that was used to convict him.

But there are many unresolved issues:

- Was it legal for the FBI to examine the contents of a computer in Russian, without obtaining permission from the Russian government?
- By inviting the hacker to the US, the FBI did not have to arrange for his extradition to the US. Was this legal?
- Could the US convict a person for crimes that were technically committed on Russian soil?

Finally, he was convicted in the US, because he had used a proxy server in the US to conduct some of the attacks. He served just under 4 years in prison and now lives and works in the US.

#### **Exercise:**

Conduct a modified white-hat / black-hat discussion of at least one of these questions (examination of a computer on foreign soil; invitation or entrapment(?) to avoid extradition; conviction for internet crimes committed against a country from foreign soil).

1. First, have students focus on and list reasons why the chosen topic was probably legal.
2. Then reverse and have them focus on and list why the chosen topic was probably illegal.
3. After these completely separate discussions, see if the class can reach a decision.

Note – these questions are interesting for discussion. There is no right answers, and governments are still working to come to a consensus on these and other issues related to the international nature of these crimes. This exercise is purely for critically examining and thinking about internet crimes, as well as formulating a logical argument for an opinion related to internet crimes.



## 12.3. Crimes related to the TICs

The classifications of the criminal behaviors is one of the essential principles in the penal systems. For this reason, several countries must think of changes to their penal codes, such as Spain, where the effective Penal Code was promulgated relatively recently. The well known Belloch Penal Code was approved on November 23rd 1995 (Organic Law from the Penal Code 10/1995) and it recognizes the need to adapt the penal criteria to the present social reality.

Among others, we can classify potential criminal actions into the following six sections.

1. Manipulation of data and information contained in files or on other computer devices.
2. Access to data or use of data without authorization.
3. Insertion of programs/routines in other computers to destroy or modify information, data or applications.
4. Use of other people's computers or applications without explicit authorization, with the purpose of obtaining benefits for oneself and/or harming others.
5. Use of the computer with fraudulent intentions.
6. Attacks on privacy, by means of the use and processing of personal data with a different purpose from the authorized one.

The technological crime is characterized by the difficulties involved in discovering it, proving it and prosecuting it. The victims prefer to undergo the consequences of the crime and to try to prevent it in the future rather than initiate a judicial procedure. This situation makes is very difficult to calculate the number of such crimes committed and to plan for preventive legal measures.

This is complicated by the constantly changing technologies. However, laws are changing to increasingly add legal tools of great value to judges, jurists and lawyers punish crimes related to the TICs.

Next we will analyze some specific crimes related to the TIC's.

1. Misrepresentation: The anonymity of the internet allows users to pretend to be anyone that they want to be. As a result, crimes can be committed when users pretend to be someone else to gain information, or to gain the trust of other individuals.
2. Interception of communications: Interceptions of secrets or private communications, such as emails, or cell phone transmissions, using listening devices, recording, or reproduction of sounds and or images.
3. Discovery and revelation of secrets: Discovering company secrets by illegally examining data, or electronic documents. In some cases, the legal sentences are extended if the secrets are disclosed to a third party.
4. Unauthorized access to computers: Illegal access to accounts and information, with the intent of profiting. This includes identify theft.
5. Damaging computer files: Destroying, altering, making unusable of in any other way, damaging electronic data, programs, or document on other computers, networks or systems.



6. Illegal copying: Illegal copying of copy-righted materials, literary, artistic, scientific works through any means without the authorization of the owners of the intellectual property or its assignees.

**Exercise:**

1. Choose one of the topics above, and conduct the following searches:
  - Find a legal case which can be classified as the chosen type of crime.
  - Was there a legal judgment, and if there was, what sentence was applied ?
  - Why did the authors commit this crime?
2. Regarding intellectual property: Are the following actions a crime?
  - Photocopy a book in its totality
  - To copy a music CD that we have not bought
  - To make a copy of a music CD you have bought
  - To download music MP3, or films in DIVX from Internet
  - What if it were your music or movie that you were not getting royalties for? What if it were your artwork, that others were copying and stating that they created it?

## 12.4. Prevention of Crimes and Technologies of double use

The only reliable way to be prepared for criminal aggression in the area of the TICs is to reasonably apply the safety measures that have been explained throughout the previous HHS lessons. Also it is extremely important for the application of these measures to be done in a way that it becomes practically impossible to commit any criminal or doubtful behaviors.

It is important to note that technologies can have multiple uses and the same technique used for security can, simultaneously, result in criminal activity. This is called TECHNOLOGIES OF DOUBLE USE, whose biggest components are cryptography and technologies used to intercept electronic communications. This section discusses the reality of this phenomenon and its alarming consequences at all levels of the human activity including policy, social, economic and research.

### 12.4.1. The global systems of monitoring: concept "COMINT"

The term COMINT was created recently as a result of the integration of the terms "COMmunications INTelligence" and refers to the interception of communications that has resulted from the development and the massive implementation of the TIC's. Nowadays, COMINT represents a lucrative economic activity providing clients, both private and public, with intelligent contents on demand, especially in the areas of diplomacy, economy and research. This has resulted in the displacement of the obsolete scheme of military espionage with the more or less open implementation of new technologies for the examination and collection of data.

The most representative examples of COMINT technologies are the systems "ECHELON" and "CARNIVORE" which are discussed next.



### 12.4.2. "ECHELON" System

The system has its origins in 1947, just after World War II, in an agreement between the UK and USA with clear military and security purposes. The details of this agreement are still not completely known. Later, countries like Canada, Australia and New Zealand joined the agreement, working as information providers and subordinates.

The system works by indiscriminately intercepting enormous amounts of communications, no matter what means is used for transport and storage, mainly emphasizing the following listening areas:

- Broadband transmissions (wideband and Internet)
- Facsimile and telephone communications by cable: interception of cables, and submarines by means of ships equipped for this
- Cell phone communications
- Voice Recognition Systems
- Biometric System Recognition such as facial recognition via anonymous filming

Later, the valuable information is selected according to the directives in the Echelon System, with the help of several methods of Artificial Intelligence (AI) to define and apply KEY WORDS.

Each one of the five member countries provides "KEY WORD DICTIONARIES" which are introduced in the communication interception devices and act as an "automatic filter". Logically, the "words" and the "dictionaries" change over time according to the particular interests of the member countries of the System. At first, ECHELON had clear military and security purposes. Later, it became a dual system officially working for the prevention of the international organized crime (terrorism, mobs, trafficking in arms and drugs, dictatorships, etc.) but with an influence reaching Global Economy and Commercial Policies in companies.

Lately, ECHELON has been operating with a five-point star structure around two main areas. Both are structures of the NSA (National Security Agency): one in the United States, coinciding with their headquarters in Fort Meade (Maryland), and another one in England, to the north of Yorkshire, known like Meanwith Hill.

The points of the star are occupied by the tracking stations of the collaborating partners:

- The USA (2): Sugar Grove and Yakima.
- New Zealand (1): Wai Pai.
- Australia (1): Geraldton.
- UK (1): Morwenstow (Cornwell).
- There was another one in Hong Kong before the territory was returned to China.

### 12.4.3. The "CARNIVORE" system

The second great global systems of interception and espionage is the one sponsored by the US FBI and is known as CARNIVORE, with a stated purpose of fighting organized crime and reinforcing the security of the US. Because of its potent technology and its versatility to apply its listening and attention areas, CARNIVORE has caused the head-on collision between this state of the art system, political organizations (US Congress) and mass media.



CARNIVORE was developed in 2000, and is an automatic system, intercepting internet communications by taking advantage of one of the fundamental principles of the net: the dissemination of information in "packages" or groups of uniform data. CARNIVORE is able to detect and to identify these "packages of information". This is supposedly done in defense of national security and to reinforce the fight against organized and technological crime.

The American civil rights organizations immediately protested this as a new attack on privacy and confidentiality of electronic information transactions. One group, the Electronic Privacy Information Center (EPIC) has requested that a federal judge order the FBI to allow access by the ISPs to the monitoring system – to ensure that this system is not going to be used beyond the limits of the law.

In the beginning of August 2000, the Appeals Court of the District of Columbia rejected a law allowing the FBI to intercept telecommunications (specifically cell phones) without the need to ask for previous judicial permission, through a Federal Commission of Telecommunications project that tried to force mobile telephone companies to install tracking devices in all phones and thus obtain the automatic location of the calls. It would have increased the cost of manufacturing equipment by 45%.

With these two examples, we see the intentions of the FBI to generate a domestic Echelon system, centering on the internet and cell phones, known as CARNIVORE. The project has been widely rejected by different judicial courts in the US and by Congress, as there is no doubt it means an aggression to American civil rights, at least in this initial version.

The project is being rethought, at least formally, including the previous judicial authorization (such as a search warrant) as a requirement for any data obtained to be accepted as evidence in a trial.

#### **Exercise:**

A joke related to these COMINT systems is found on the Internet. We include it here for class discussion of the ethical and legal implications:

*An old Iraqi Muslim Arab, settled in Chicago for more than 40 years, has been wanting to plant potatoes in his garden, but to plow the ground is a very difficult work for him. His only son, Amhed, is studying in France. The old man sends an email to his son explaining the following problem:*

*"Amhed, I feel bad because I am not going to be able to have potatoes in my garden this year. I am too old to plow the soil. If you were here, all my problems would disappear. I know that you would plow the soil for me. Loves you, Papa. "*

*Few days later, he receives an email from his son:*

*"Father: For God's sake, do not touch the garden's soil. That is where I hid that . . . Loves you, Amhed. "*

*The next morning at 4:00, suddenly appears the local police, agents of the FBI, the CIA, S.W.A.T teams, the RANGERS, the MARINES, Steven Seagal, Sylvester Stallone and some more of elite representatives of the Pentagon who remove all the soil searching for any materials to construct pumps, anthrax, whatever. They do not find anything, so they go away.*

*That same day, the man receives another email from his son:*

*"Father: Surely, the soil is ready to plant potatoes. It is the best I could do given the circumstances. Loves you, Ahmed."*





### Exercise:

Search for information about the Echelon and Carnivore systems on the internet, as well as their application on networks and TICs systems in your country to answer the following question:

1. What does the term "ECHELON" mean?
2. What elements form the ECHELON system?
3. What elements form the CARNIVORE system?
4. Search for an example of controversy attributed to the ECHELON system and related to famous personalities.
5. Search for an example of the application of the CARNIVORE system related to a TERRORIST known worldwide.
6. What is your opinion about the "legality" of such systems?

## 12.5. Ethical Hacking

Besides talking about criminal behaviors, crimes, and their respective sanctions, we must make it very clear that being a hacker does not mean being a delinquent.

Nowadays, companies are hiring services from "Ethical Hackers" to detect vulnerabilities of their computer science systems and therefore, improve their defense measures.

Ethical Hackers, with their knowledge, help to define the parameters of defense. They do "controlled" attacks, previously authorized by the organization, to verify the system's defenses. They create groups to learn new attack techniques, exploitations and vulnerabilities, among others. They work as researchers for the security field.

Sun Tzu said in his book "The Art of War", "Attack is the secret of defense; defense is the planning of an attack".

The methodology of ethical hacking is divided in several phases:

1. Attack Planning
2. Internet Access
3. Test and execution of an attack
4. Gathering information
5. Analysis
6. Assessment and Diagnosis
7. Final Report

One helpful tool that Ethical Hackers use is the OSSTMM methodology - Open Source Security Testing Methodology Manual. This methodology is for the testing of any security system, from guards and doors to mobile and satellite communications and satellites. At the moment it is applied and used by important organizations such as:

- Spanish Financial institutions
- the US Treasury Department for testing financial institutions



- US Navy & Air Force

### Exercise:

Find information about Ethical Hacking and its role in IT security companies.

Search for information about the OSSTMM and methodologies.

Search for information about "certifications" related to the Ethical Hacking.

## 12.6. The 10 most common internet frauds

Listed below is a summary from the US Federal Trade Commission of the most common crimes on the Internet as of 2005.

1. Internet Auctions: Shop in a "virtual marketplace" that offers a huge selection of products at great deals. After sending their money, consumers receive an item that is less valuable than promised, or, worse yet, nothing at all.
2. Internet Access Services: Free money, simply for cashing a check. Consumers are "trapped" into long-term contracts for Internet access or another web service, with substantial penalties for cancellation or early termination.
3. Credit Card Fraud: Surf the Internet and view adult images online for free, just for sharing your credit card number to prove you're over 18. Fraudulent promoters use their credit card numbers to run up charges on the cards.
4. International Modem Dialing: Get free access to adult material and pornography by downloading a "viewer" or "dialer" computer program. Consumers complained about exorbitant long-distance charges on their phone bill. Through the program, their modem is disconnected, then reconnected to the Internet through an international long-distance number.
5. Web Cramming: Get a free custom-designed website for a 30-day trial period, with no obligation to continue. Consumers are charged on their telephone bills or received a separate invoice, even if they never accepted the offer or agreed to continue the service after the trial period.
6. Multilevel Marketing Plans/ Pyramids: Make money through the products and services you sell as well as those sold by the people you recruit into the program. Consumers say that they've bought into plans and programs, but their customers are other distributors, not the general public.
7. Travel and Vacation: Get a luxurious trip with lots of "extras" at a bargain-basement price. Companies deliver lower-quality accommodations and services than they've advertised or no trip at all. Others impose hidden charges or additional requirements after consumers have paid.
8. Business Opportunities: Taken in by promises about potential earnings, many consumers have invested in a "biz op" that turned out to be a "biz flop." There was no evidence to back up the earnings claims.
9. Investments: Make an initial investment in a day trading system or service and you'll quickly realize huge returns. But big profits always mean big risk. Consumers have lost money to programs that claim to be able to predict the market with 100 percent accuracy.



10. Health Care Products/Services: Claims for "miracle" products and treatments convince consumers that their health problems can be cured. But people with serious illnesses who put their hopes in these offers might delay getting the health care they need.

**Exercise:**

Think about the following questions and discuss them with the rest of the class:

1. Do you think that you could have been a victim of some of the crimes mentioned throughout the lesson?
2. Here is a quote from an ISECOM board member: "In order to have the proper background to evaluate the security readiness of a computer system , or even an entire organization, one must possess a fundamental understanding of security mechanisms, and know how to measure the level of assurance to be placed in those security mechanisms. Discuss what is meant by this and how you could prepare to "evaluate the security readiness of a computer system". Have these lessons given you enough materials to get started?
3. [optional exercise for personal consideration (not general discussion)]: After analyzing the comments in this lesson, you may find that there are technological activities that you have heard about, or that you may have even done, that you never considered to be illegal, but now you are not sure. Some research on the internet may help clear up any questions or confusion that you have.



## 12.7. Recommended Reading

<http://www.ftc.gov/bcp/menu-internet.htm>

<http://www.ic3.gov/>

<http://www.ccmotwanted.com/>

<http://www.scambusters.org/>

<http://compnetworking.about.com/od/networksecurityprivacy/l/aa071900a.htm>

<http://www.echelonwatch.org/>

<http://www.isecom.org/>