Hacker Highschool

SECURITY AWARENESS FOR TEENS



LESSON 7
ATTACK ANALYSIS

HACKING IS LEARNING
www.hackerhighschool.org

ISECOM

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

Contributors

Pete Herzog, ISECOM
Marta Barceló, ISECOM
Chuck Truett, ISECOM
Kim Truett, ISECOM
Bob Monroe, ISECOM
Greg Playle, ISECOM
Marco Ivaldi, ISECOM
Rob Dodson

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. Hacktivist 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. Rasomware 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**.

Hacktivism (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 **hacktivism**.

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 hacktivism. This activity may be viewed as heroic to some people but considered anti-social disobedience by others.

The main difference between hacktivism 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, hacktivism 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 lead 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 **Ettercap**. 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. 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).

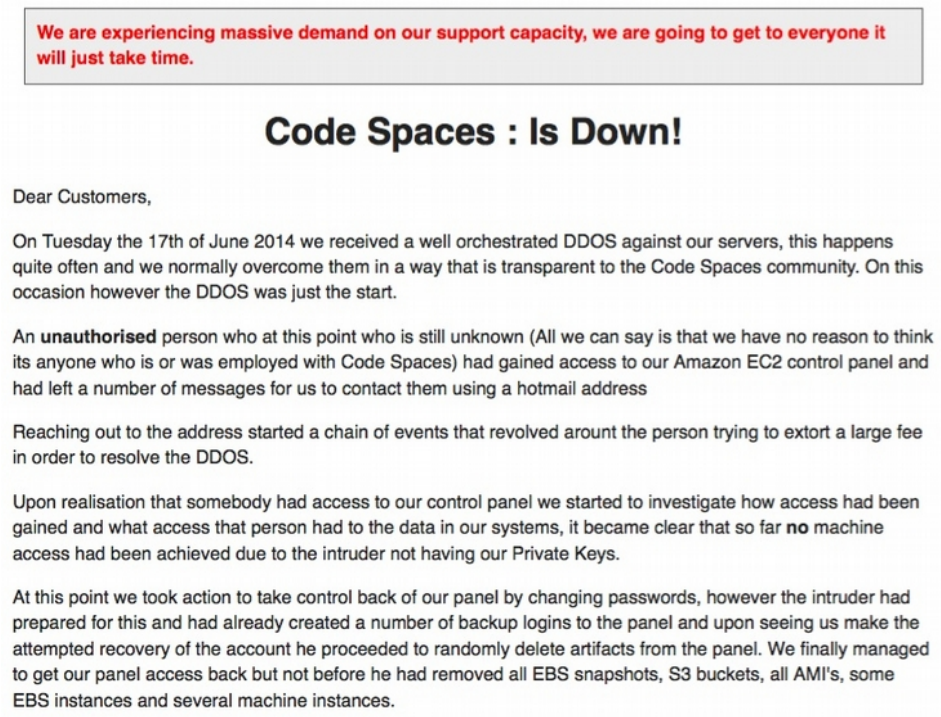
Exercises

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 arount 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 in 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 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 **tire 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 phone 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.winpacap.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 Xquarz 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.

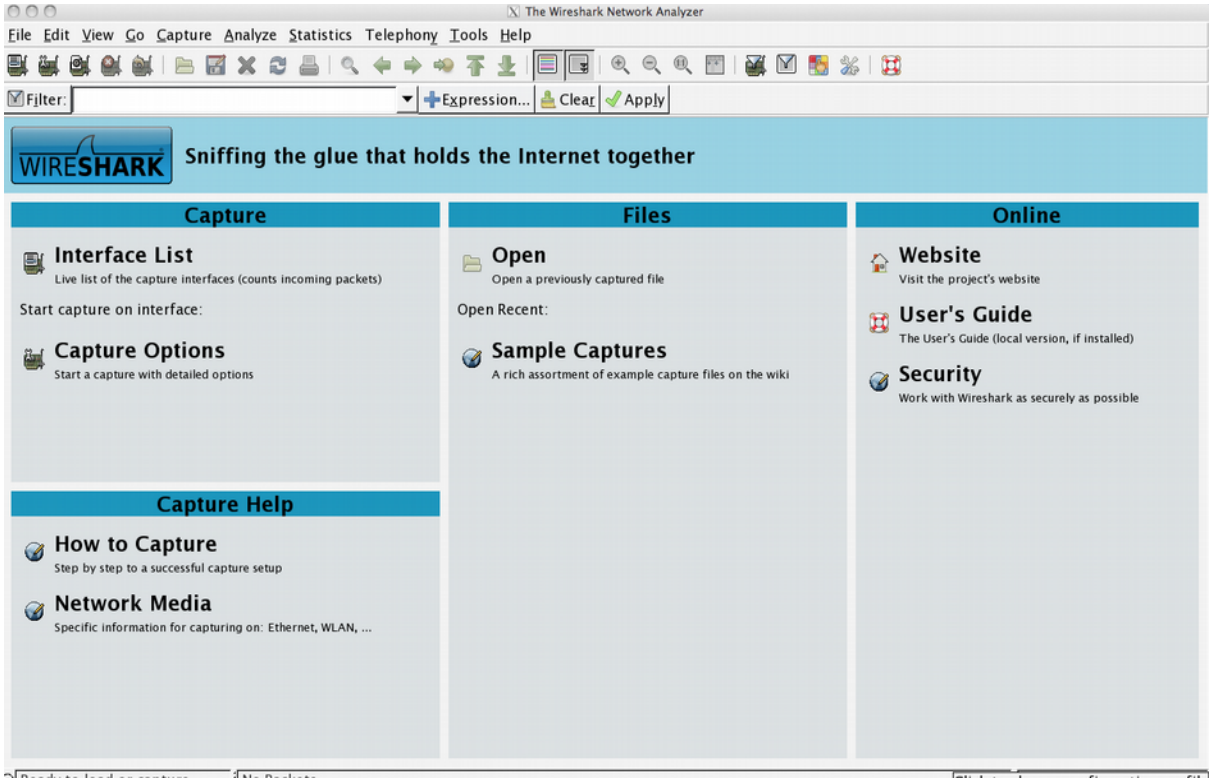


Figure 7.1 Wireshark

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.

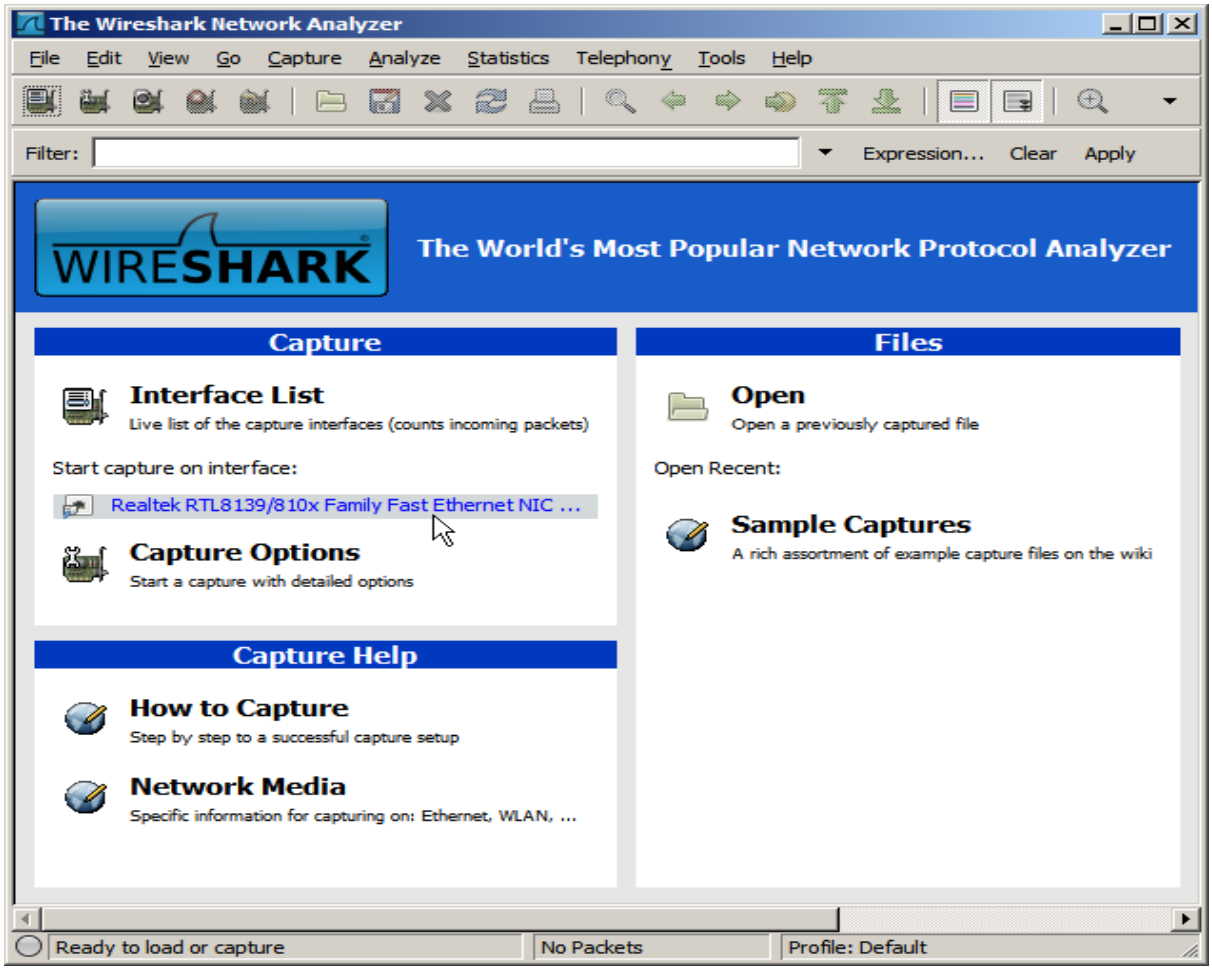


Figure 7.2 Wireshark Interface Selection

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.

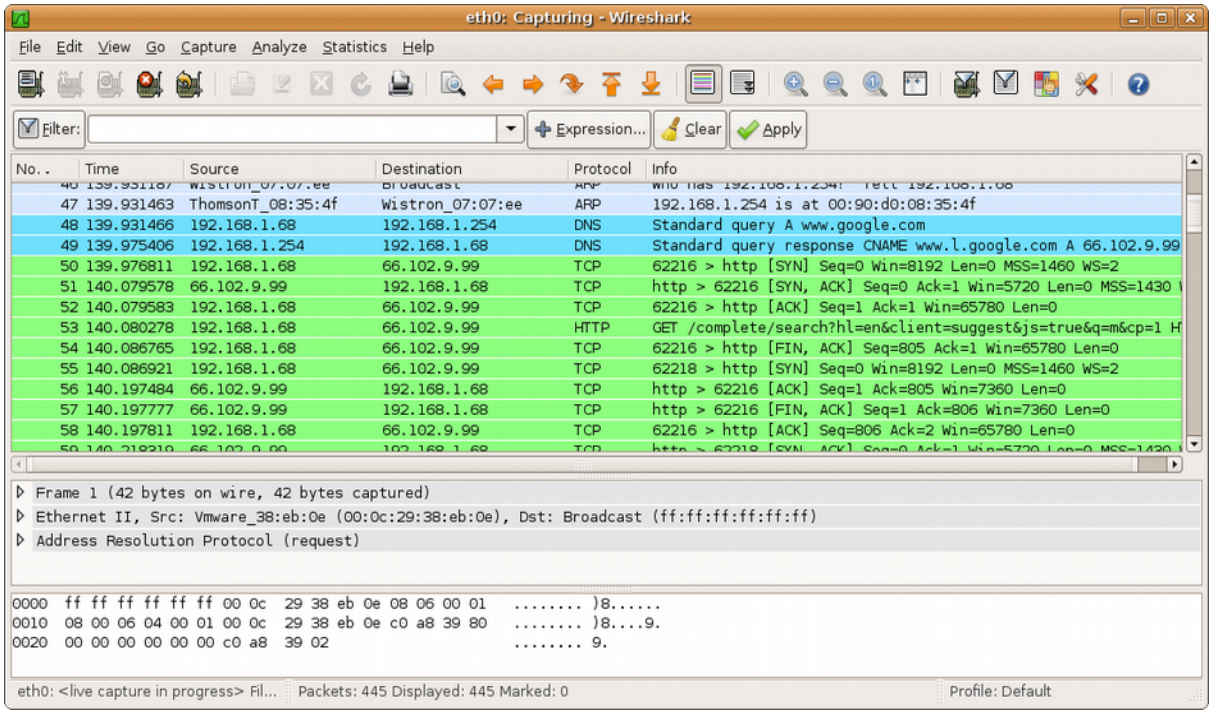


Figure 7.3 Capture

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.

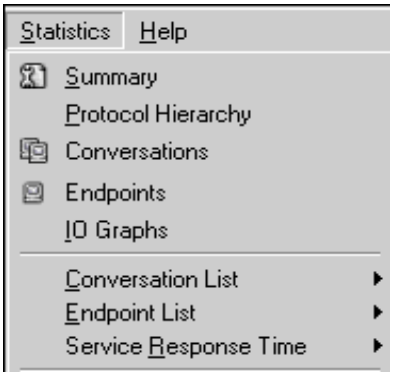


Figure 7.4 Statistics Menu

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

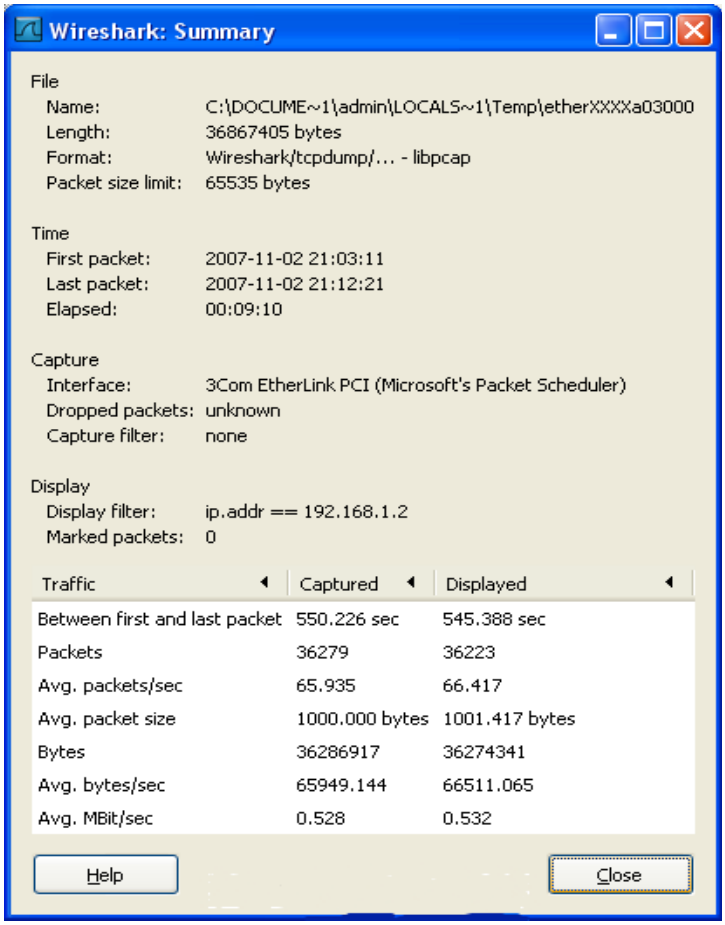


Figure 7.5 Summary

Protocol Hierarchy

The protocol hierarchy shows a dissection by OSI layer of the displayed data.

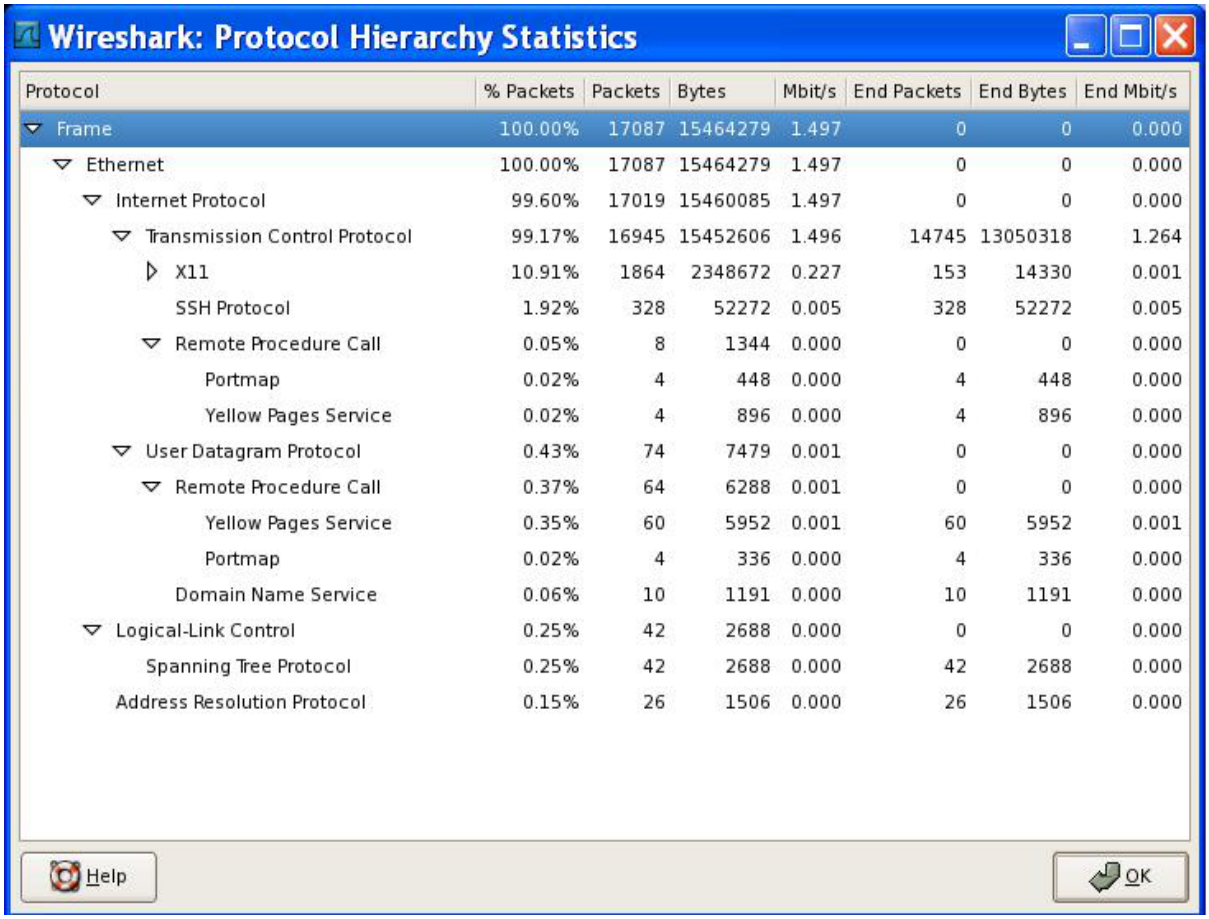


Figure 7.6 Protocol Hierarchy

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

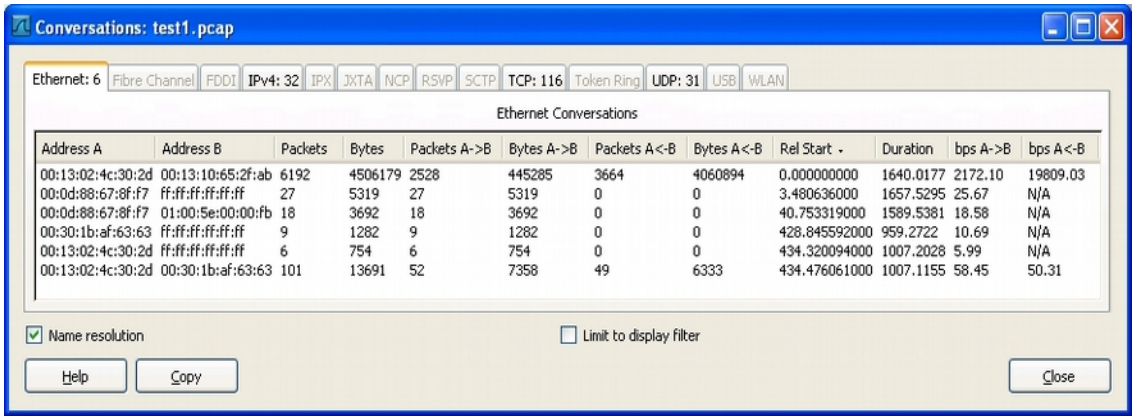


Figure 7.7 Ethernet Conversations

IP Conversations

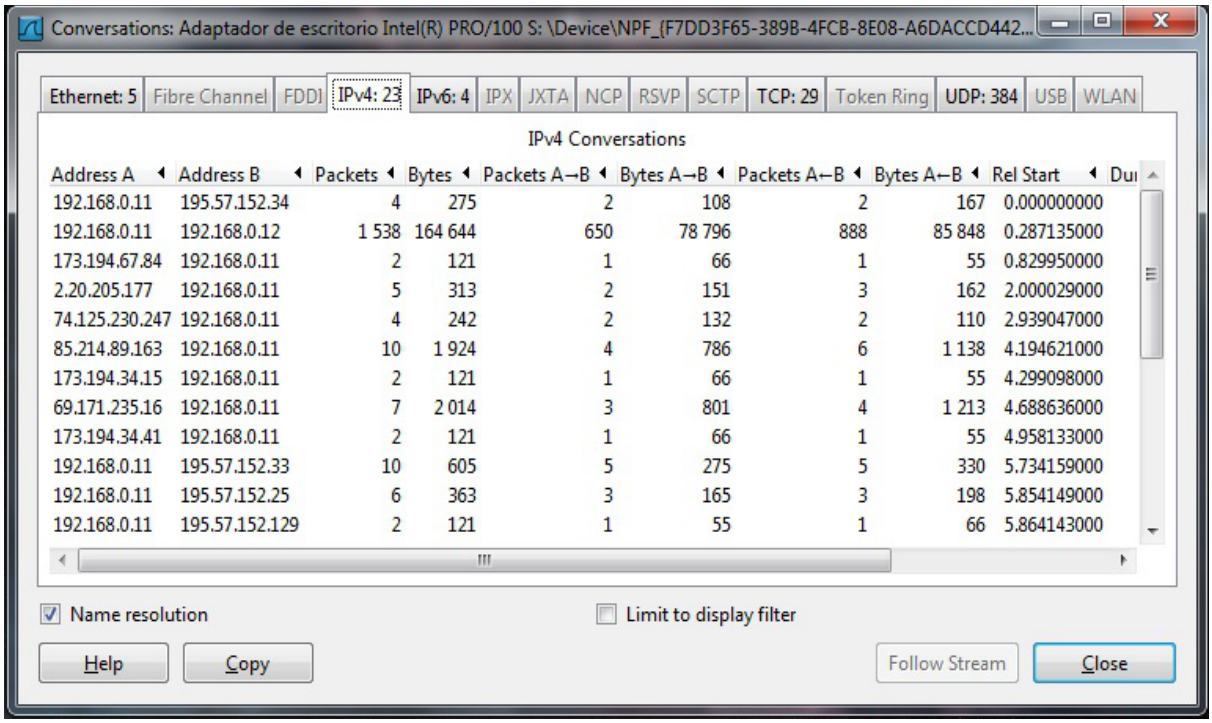


Figure 7.8 IP Conversations

TCP conversations

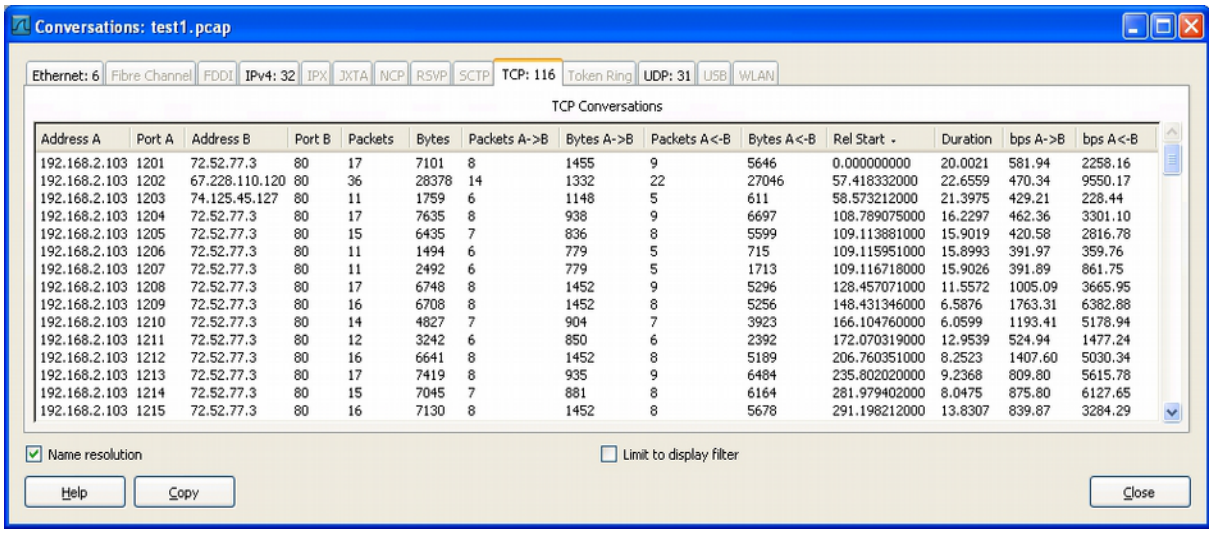


Figure 7.9 TCP Conversations

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".

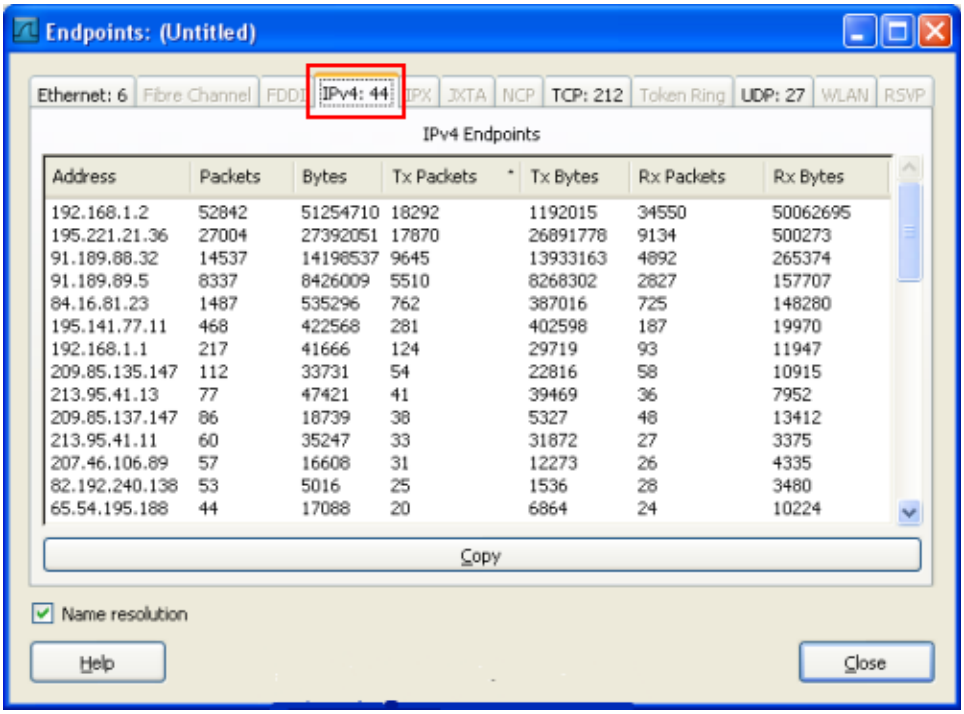


Figure 7.10 Endpoints

Which endpoints are consuming the most traffic? Why might that be?

Output

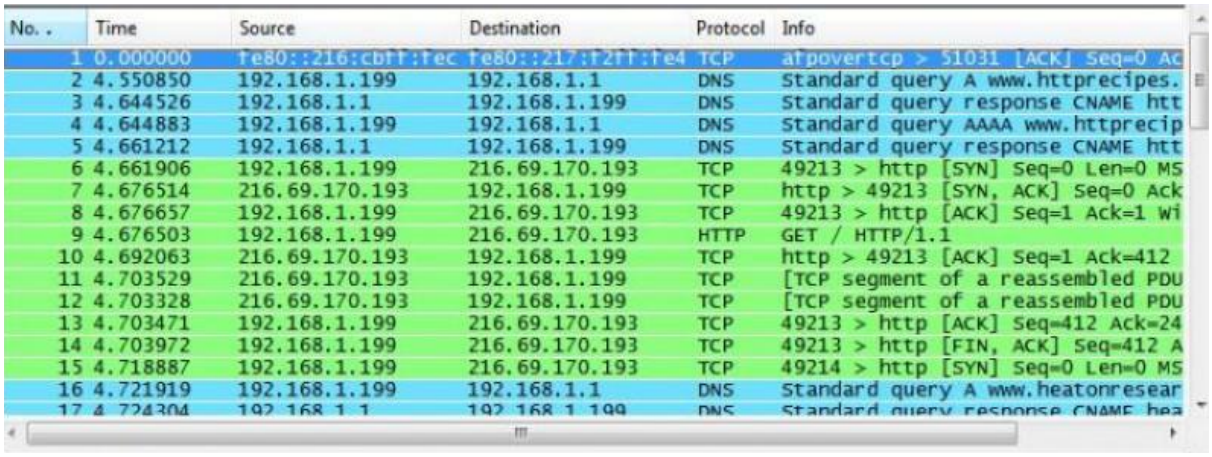


Figure 7.11 Output

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 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

ISECOM