CS 136 Lecture Notes
W 1 M Lec 3-29-16

- Familiarize with how to develop secure software

- Not a lot of instruction on how to build software where it doesn't have security flaws

- Give a little bit of information for securing a system.

- If you have a laptop, desktop, tablet, smartphone, etc., how do you go about securing your system?

- Go through some introductory material and see what is the range of material we will be covering.

- How the class will be graded and the class webpage?

Topics to Be Covered

- A great deal of what we do in computer security relies on cryptography for its basis.

- If you don't understand cryptography, you won't understand how to deal with these systems.

- We won't learn anything useful about how to build a cryptographic algorithm.

- Security models

- How can we hope to build a secure mechanism if we don't know what we are doing?

- Secure programming

- Secure protocols

- How do you achieve security effects across the network.

- Threats and countermeasures

- Operating systems security

- Typically the lowest level

- How do we secure the operating system?
- Security analysis and forensics
- I have set up my system and I want to see if it is secure.
- How would you look at that system?
- It can vary from a single computer to a whole vast network.
- Bad things tend to happen and we should take a look at the bad things that happened.
- Did the attackers get into every single machine?
- This is forensics (looking at evidence left behind)
- Malware
- A related topic deals with common types of attacks
- How we defend against such attacks
- Privacy
- Privacy and security can go hand-in-hand
- We need to see if the computer is secure as well.

Exercise Topics

0. Access control and permissions
0. Exploits
0. Analysis of attacks and forensics
0. Man in the middle attacks
0. Botnets: collection of malicious nodes that require a whole lot of resources (DDoS)

Class Web Page
http://www.lasr.cs.ucla.edu/classes/136_spring16

Introduction to Computer Security

- What threatens our goals of computer security?

Why Is Security Necessary?

- If people were always nice, we wouldn't need computer security because they would never shut a machine down.
- We are moving a whole lot of money around on the Internet
- A great deal of crime has moved from hold ups of the drug store to break ins on a computer system
- Many great advantages for criminals
- A lot of important info is handled by computers
- Private medical records
- Companies strategic plans and product designs
- Most of which are reachable from any place in the world.
- Computers are really neat and can do a lot of great things for us.
- Things run wonderfully except when they don't.
- Our cars are increasingly controlled by computers
- Typical automobile has a BUS with 40-50 internal computers
- They now need computer networks!
- Vehicles are going to be autonomous
- Self-driving vehicles
- Based on sensors in the cars and information that arrives from the outside world.
- Computers make the decision i.e. accelerate, hit the brakes, turn right.
- All well and good as long as the computer's integrity is NOT compromised!

- If we just turn the driving over to the computers, they will do a better job than people.

- This is true as long as people don't try to attack and seize control over the computer.

- Power grid

- Moving the power around the world efficiently so it isn't wasted so nothing is lost.

- Complicated problem!

- Computers are the solution!

- It is tedious to go out in the middle of the city, so why not have a guy in an office connect via a network and do it remotely.

- People on the network can send packets of data and make it do things if we are NOT careful.

- An attack in the Ukraine, presumably by Russian separatists where they attack the power grid remotely.

- Send computer messages and computers can be brought down.

- This is just a sample, we are throwing more and more control to the computers.

- Anybody can cause those computers to do something unless we are very careful about what we do.

- Unless we are careful, our world will work very badly when someone wants it to work very badly.

History of the Security Problem

- Early on, there was NO computer security problem

- Computers were rare and it was a very important device and it was almost impossible for anyone to get to the device.

- The only security problem was if we have guards at the door or locks that were good enough.

- Back when Reiher was an undergrad, all universities had some computer system that could be accessed by their students.
- CS students would break into the system, get caught, and get hired!
- 1979: Reiher took a computer security class and had a hard time finding bad problems in computer security.
- No issue with identifying theoretical problems, but there was never a practical issue with computer security at the time!
- Nobody thought computer security was important at the time!
- However, it later became clear there would be a problem!
- Fortunately for now, we have not had a real disaster i.e. 9/11 as a result of computer security.
- Bad things did happen, though, i.e. companies going out of business due to DDoS attack
- Identity theft and phishing
- At least one case (Stuxnet) where there was serious real world implications for Iran's centrifuges.
- Video showing how to get an electric transformer to smoke and burn.
- We connected our power grid to the Internet, which allowed it to be exploited.
- Underground business in cyber thievery
- A whole lot more spending nowadays on cybersecurity
- More jobs!

Some Examples of Large Scale Security Problems

- Malicious code attacks

Malicious Code Attacks

- Ones that are very successful are addressed by large companies like Symantec

- When a new virus comes out, they will analyze the virus and deal with it.
- If you go to places like Symantec's website, you will learn a great deal about each virus, but no one keeps a completely exhaustive list of every virus known to man.
- This is a serious problem dealing with malware and there is no hope that they can use past techniques to deal with malware.
- Initially, there were no cell phones, but now we have smartphones (portable computers)
- Not as powerful as the best server machine, but you can still do a lot and it has a full OS and full networking capabilities
- It now becomes a vulnerable platform that can be attacked!
- We can now buy or rent botnets that consist of smartphones.
- If you want to rent it, pay me this amount of money and the botnet is yours to use as you please.
- In both cases, it can be possible to attack it in both cases.
- Other cases refers to the underlying OS
- There may be a security flaw in one OS and not on another OS
- Many complicated and special cases.

Apple vs FBI

- The terrorists in San Bernardino left behind an iPhone
- Locked with a password
- If you guess wrong 10 times, the iPhone locks up and you cannot make any more guesses
- If you don't guess properly, you cannot decrypt the data
- After 10 guesses, they would permanently lock the iPhone and never get out the data.
- This was the issue for the FBI

- Make many guesses and we don't want the data wiped out if we make too many wrong guesses.
- What would work on this particular model of the iPhone doesn't work on other versions of the iPhone.
- Apple is trying to put more security into the hardware.
- They don't want **anybody** breaking into the system.
- The government doesn't like this at all.
- In many cases, the specific model of the iPhone and the specific OS matters a lot.
- The FBI has found someone who could unlock the iPhone, so they are no longer sueing Apple.
- This is not an issue going away and this will be a major public policy debate for the new few years.

Distributed Denial of Service Attacks (DDoS Attacks)

- Attackers gets control of many machines remotely and he finds someone he doesn't like
- I would be happier if you cannot do something on the Internet
- They would launch an attack on this site
- You don't need a vulnerability; you just need a lot of traffic!
- This is very common today.
- Multiple DDoS attacks across many different targets.
- These start after Reiher talked about the slide.
- This is done by people who have an axe to grind.
- DDoS attack on companies people don't like.
- Criminal enterprise stealing money from a bank

- Rifle money and you are a little concerned that the bank has some professional computer security network folks.

Vulnerabilities in Commonly Used Systems

- All computer software has terrible security flaws
- 802.11 WEP sucks!
- Stop using WEP and use WPA instead!
- There have been critical vulnerabilities in iOS, Windows, Linux kernel, glibc, and Oracle Java implementation
- This is just in the last 2-3 months!
- Applications also have vulnerabilities
- Adobe Acrobat has many problems!
- These are the poster childs for crappily written software from a security perspective.
- Android OS has issues
- IE, Microsoft Office, VMWare, etc.
- There are people who write security systems, but even these have security flaws!
- OpenSSL (almost every website uses this) and Comodo Internet Security
- Inevitably, software will have flaws, but we should still do better even then.
- Some issues are that legacy code cannot be refactored easily.
- Adobe cannot throw away Acrobat since their legacy code cannot be redeveloped easily.
- Microsoft bit the bullet and redesigned Windows OS from scratch!
- Adobe isn't willing to do this yet, so they will continue to get monthly reports for flaws

Electronic Commerce Attacks

- People are attacking on the Internet because that is where the money is!
- Criminals are now attacking these things:
- Credit card number theft (phishing)
- They ask you to provide things such as credit card number on a seemingly authentic website.
- People aren't stupid; criminals are just really clever and can trick people easily.
- Identity theft (phishing is common here)
- People take out a loan against your house under your identity
- Then you have to repay that loan
- This is painful process to fix!
- You need a lot of personal info i.e. DOB, home address, SSN, etc.
- SSN was created in the 1930s and we didn't expect it to be used to exploit identity theft.
- These aren't random; there is an algorithm and they need to be in a lot of places that aren't very secure.
- Loss of valuable data from laptop theft
- People don't care about the hardware; they want your data!
- There is a chance that you have a lot of confidential info such as private information to be saved in the web browsing cache.
- Manipulation of e-commerce sites
- Extortion via DDoS attacks
- People want you to give them money.
- If you want to go back into your business, you would have to pay $10,000.

- Common on betting sites like in Great Britain

- If you want to, go to British or Irish sites and bet on the outcome of the Presidential race.

- Another example: A week before the World Cup finals, the betting sites will become rich from traffic unless someone launches a DDoS attack on you.

- You can harness the power of someone else's cycles on the cloud, but what they do may not be what you want them to do!

- Pay attention to the cookie's information since computers cannot interpret if something like a refrigerator is assigned a value of $0.03

- It is tempting because you don't have to worry as much about bandwidth.

Some Recent Statistics

- 2015 Verizon report found over 2000 data breaches

- In 60% of cases, attackers were successful within minutes and only 20% of organizations found the breach within a few days.

- Losses amounted to over $800,000,000

- Ponemon Institute 2014 survey showed 94% of healthcare organizations lost data in the past two years

- Practically all healthcare organizations lost data due to attacks!

- National espionage, they want to see what your first plans are as long as they can stay in and try to come back out again.

Ransomware

- If I have control over your computer, I can go out to your disk and encrypt all the data

- After I encrypted all your data, you cannot read it without a key, so they may agree to pay up a bunch of money to find out what the key is.

- Hospitals are particularly vulnerable since they need timely data so patients do not die.

Cyberwarfare

- Do not underestimate people on the other side.
- They are really clever and in some cases really persistent.
- Cyberwarfare is an area where you see these bright people.
- Things involving the military, national security plans, etc.
- Other nations have wanted to know this information for their own gain.
- We are beginning to see attacks for this purpose.
- We are relying on the network for particular things i.e. people in Nevada bunkers controlling drones in Somalia.
- It would be beneficial for people to be able to interrupt that traffic using cyberwarfare.
- A lot of militaries are worried about cyberwarfare.
- DDoS attacks on Estonia and Georgia by potential Russian hackers
- Stuxnet was likely written by the US with Israeli help
- Designed specifically to wipe out centrifuges from one place in one facility.
- Cyberspying happens all the time.
- Recent report that an Iranian hacker was trying to take control of a dam in New York state.

Something Else to Worry About

- Are we going to lose something important?
- Use data mining to find terrorists.
- NSA has taps everywhere!
- Edward Snowden revealed that the NSA has listened in on a vast amount of stuff.
- It turns out that this isn't quite true; the NSA can find people like drug dealers.

- Call up the FBI and say we have information from doing data mining.
- We weren't supposed to do that and tell you, so we recommend you do a background check on this guy.
- Some people might argue that this is illegal.
- Whenever you type something into Google, Google remembers.
- Sometimes this is good. You searched this before so you probably found this interesting.
- You have to remember how Google makes its money
- From advertisements!
- Google knows so much about you!
- Google can give you names of people interested in finding a niche product.
- Facebook, Twitter, etc., all these social media platforms know a lot about you.
- We are in danger of losing a lot of privacy that we used to have.

Why Aren't All Computer Systems Secure?

- Why aren't all our computers secure?
- Due to hard technical problems
- Some major issues that have to be addressed and are really hard to solve.
- Cost/benefit issues
- Security costs
- Almost anything you do to increases security often has a monetary cost
- Costs more to develop and will cost you something.
- What is the benefit for that cost?

- Nobody ever attacked you, so you never get that benefit.
- It can be an attack that is relevant to that security mechanism, so it might not help in other scenarios.
- Example: Protecting against DDoS attack when you get hit by a SQL injection.
- Many users are oblivious and do not take any measures to deal with their own personal security.
- These securities need to be built in a way to work without their cooperation.
- Ignorance also plays a role
- Can we expect a billion people to be computer sophisticated?
- That is a very unrealistic expectation.
- Most people at the end of the computer systems will never understand computer security and probably do not even care.

Legacy and Retrofitting

- Since we didn't care, we didn't do squat about it.
- We were doing influential things like designing the Internet, designing programming languages, and choose operating systems
- We are still using programming languages and operating systems from that era as well as the Internet, but no one had security in mind at that time.
- Retrofitting security works very poorly and it is NOT a successful technique.
- We have all these things and we cannot fix their security.

Problems With Patching

- We find a problem in the code and we write some other piece of code to augment the bad code.
- We then tell everyone to download the new patch.

- When an important bug shows up, you have to fix it quickly. This is the result for the patch.

- When someone submits a bug report, it does not necessarily deal with that particular circumstance.

- Even a wonderful patch will have possibly 5,000 users out there, and we want to get that patch to all 5,000 users.

- It is generally hard to get the patch out to everyone and it is not guaranteed that everyone will apply that patch.

- IoT devices are little tiny devices that get attached to things like toasters, light switches, etc.

- These devices are not intended to be upgraded, so they are not built with the ability to patch them.

- If they have a security flaw, too bad!

- We want organic security: built into the system to address these problems.

Speed is Increasingly Killing Us

- Attacks are developing very quickly

- It is easier to adapt attack than defense

- Malware is increasingly spreading faster

- With a properly designed worm, you can compromise every computer with a flaw within 10 minutes.

- If there was a flaw in Windows, you could get hundreds of millions of computers in 10 minutes.

- US DoD computers targeted at least 43,000 times in the 1st half of 2009

- Background noise: your computer is under attack always as long as it is on the Internet

- The vast majority of attacks are exercising vulnerabilities that you have on the computer.

- You can still have automated software running all the time and they ignore all that kind of stuff since it happens all the time.
- It was common wisdom that if you took an unpatched Windows machine and got on the Internet, your computer would be compromised within a few minutes.
- Apply patches as quickly as possible for a new computer.
- DoD generally has terrible security compared to the Military or the NSA.
- These departments have to do periodic studies and they generally get report cards of D's or F's

Security and Protection

- Security is a policy
- No authorized user may access this file
- Protection is a mechanism of how you achieve a policy
- Compare your identity against those who have access permissions

Vulnerabilities and Exploits

- A vulnerability is a weakness that can allow an attacker to cause problems.
- Vast majorities of all vulnerabilities are never exploited.
- An exploit is an actual incident of taking advantage of a vulnerability
- People tried to use the vulnerability and have the attacker take advantage of this.
- Sometimes, exploits refer to performing an action.
- They can mean the code or methodology.
- Otherwise, it refers to the actual incident.

Trust

- You do certain things for people that you won't for others.

Problems With Trust

- Set of detailed rules like any other program
- If we want trust to be something built into computer security, we have to have a variable to express what is trust.
- How do we express trust though?
- Trust may not always make sense within the realm of what you are doing.
- What made you trust that person?
- It is often based on identity.
- You would do things for your mother that you wouldn't do from some random lady on the street.
- What if trust changes?
- You have to change your software to address this.
- All electronic browsers are authenticated via certificates.
- You can go into your list of certificates and you will not have heard of a vast majority of those entities.
- A few years ago, there was a Dutch company that screwed up, and Iran was able to convince that some websites were Twitter, FB, etc.
- This caused censorship and stemmed from the result of trusting the Dutch company.

Trust Is Not a Theoretical Issue

- At the base, there is some trust problem and you shouldn't have trusted them.
- You can get a piece of information coming from a program.
- I know what the info is and it is a request to run a routine.
- In this case, you are in a bad situation because it is doing something that you did not intend it to do.
- It can do all kinds of things on your behalf.

- All of this relates to issues of trust.
- Phishing is a trust problem
- You will use your privileges as a human user to do all the things that the user tells you to do.
- System shouldn't have let you do this, but you are the owner, so the system trusted you.
- Any exploited vulnerability goes back down to misplaced trust.

Transitive Trust

- See animation
- If A trusts B, and B trusts C, then A trusts C.
- Should you though?

Examples of Transitive Trust

- If you have a web server in your database, say you want to customize the web page.
- The database trusts the web server, and both are running on the same system.
- What happens if there is a security flaw?
- The data base will also trust the user, so the web server will tell the database to do it.
- Bad things can happen to due to transitive trust.

What Are Our Security Goals?

- CIA
- Confidentiality
- I want to keep secrets. If I want it kept secret, it should remain secret
- Integrity

- I have a very important piece of data. If the attacker can change the data, things I will do will be wrong.

- Availability

- If there is a system that should be available to users, we must be sure the system is available and it is impossible for an attacker to prevent things from happening.

Thinking About Threats

- Normal service: information source goes to an information destination
- How do they interfere with this service in various ways.

W 1 R Lec 3-31-16

Security Principles, Policies, and Tools

- Talk about some of the tools i.e. access control mechanisms

Outline

- Security design principles
- Basic concepts
- Security policies

Design Principles for Secure Systems

- These are helpful for examining and evaluating the security of systems.

Economy in Security Design

- We don't want to spend a great deal of effort building security into the system.
- We don't want a high cost of entry, so they don't want to learn a great deal that they don't already know.
- We want our assurance to come relatively cheaply so we can verify the security effects that we were hoping for.
- We want little or no overhead for ordinary options.
- It should only do then what needs to be done.
- Identify what needs to be achieved and build mechanisms that are precisely that.
- Good idea to build simple and small security mechanisms.
- Probably will cost less and it will probably be cheaper to verify.

Complete Mediation

- There are various actions a user might want to perform
- We want to check other elements of conditions in which the action is to be performed.
- Every time an action is performed, make an independent check to see if the action should or should NOT be allowed.

- Every time we access a remote server, check the data in the file, etc. we should have a condition to check if this should be okay!
- It can be quite costly and we sometimes need a compromise for true, complete mediation.

Open Design

- Widely regarded as being important to NOT rely on the secrecy of the design of your system.
- Don't achieve security goals by having no-one know how it works.
- If the attacker cares enough, he/she will be able to find out eventually what the system is doing and hack it.
- "Security through obscurity" says it is all a secret, and since you don't know the secret, you can't get it.
- Try to probe and get information about the system.
- You can obtain knowledge of how the system works.
- We assume the attacker knows every single detail of how the security works.
- Does NOT imply we publish all the details of our security systems.
- If we don't punish them, there is an extra step to attack our system.
- Even if they do, they still should be unable to break into the system.
- Sometimes, it is highly beneficial to publish all the details.
- Everyone understood everything about the system before it was put in place.
- It is the case that obscurity can provide *some* security, but it is very brittle.
- Once a system disappears, the security also disappears if that is the only factor.
- Cryptographic keys are changed frequently and they are easy-to-change.
- If you have a system whose security relies on a secret and you cannot change it easily, then you have lost all security related to that secret

Separation of Privileges

- You want the privileges one
- Old DOS -> can run any data you want or even a old version of Windows.
- Once you have logged in, you have complete access to everything on the system.
- It was hard to achieve important security effects.
- We get much greater flexibility if we separate the privileges.
- This implies that we are able to set separate access controls on each file.
- Separate read and write privileges
- Instead of mechanism for the user, it means privilege to do a certain action.
- Each privilege corresponds to one specific action.

Q. What is a mechanism?
A. Tool used to work with your system i.e. reading and writing.

Least Privilege

- If we have a process or user or remote system of some kind and we want it to perform some actions, and we want to give that party the exact privileges it needs to perform the exact action.
- Do NOT give it complete privileges for everything.
- We can allow you to open a file for read and perform the permissions for read.
- This does NOT give you exact privileges for write.
- This means that if the program is compromised, the attacker who compromised the program will NOT be able to write the file.
- You have to know exactly what security related entity is possible for the exact task.
- Once we are done, take the privileges away and don't give it more for things that shouldn't be done.

Least Common Mechanism

- Different security mechanisms protect different parts of the system.
- You want to compartmentalize things for different access writes.
- Separate mechanisms for mechanisms so we are fairly sure that we cannot misuse things that belong to someone else.
- When we couple different parts of mechanisms, there is a possibility that we will encounter security leakage.
- If more than one user has the same privilege like writing to X, that is potentially bad because it is easier for an attacker to crack that system.
- Spreads out the mechanism for a user. If too many users have the same mechanism, then the attacker can exploit it.s

Q. What is coupling?
A. Sharing access rights among different users.

Acceptability

- All the security approaches will interact with other entities.
- Interact with human users and if the human user does NOT properly know how to use the human mechanism
- The user doesn't like it, so he found some other way to avoid using the mechanism.
- Potentially dangerous thing where we don't have a protection mechanism in place.
- If the user cannot find anything acceptable, he will look elsewhere.
- Typically, our mechanisms probably have to be providing high-quality protection and not getting in the way of legitimate things.
- An example of a mechanism is when you have these security warnings that are invalid for the following reason.
- The typical user is in no position to understand what a certificate was to reasonably examine any information that could be presented to him.

- What is going to happen is the user takes the path of least resistance and it gets in the way of doing something.
- If he says deny, he is clearly not going to do that.
- He will take an insecure action and if given advice, he will ignore the problem
- If it is permissible, whatever mechanism better not veto that access.
- If they do, this is the security system is one we will get rid of.
- More of a marketing scheme: we want people to use our security products!

Fail-Safe Designs

- If a system isn't sure whether or not to give access, **default to not giving access**
- Unless you have good reason to, do not give access.
- Do NOT let any packets through.
- If packets are alright, we specify a set of rules to let them through our firewall.
- If we haven't thought about characteristic packets to work with, by default, we never wrote a rule to say let it through.
- If it did have a security implication, then it would be safe.
- Since I don't know, drop the packet.
- You have to be able to identify everything that is okay.
- Challenging to allow things
- Sooner or later, someone will try to find the perfectly legitimate thing and then someone will complain.
- Okay for it to happen once in a while.
- If it is happening regularly, we will run into that accessibility problem!
- Similar to computer scanning and if something goes wrong, no security is lost.s

Paradigm

- Tradeoff in achieving a good level of all of these principles simultaneously.

Thinking About Security

- Developed by Bruce Schneier
- How am I going to defend my system?
- What assets am I trying to protect?
- What is at risk?
- What bad things can happen?
- Am I worried about it being stolen?
- Do I want to make sure the system is available for use and what can I do to protect it?
- Come up with a security solution and ask "Is this going to help against these risks"?
- We use strong encryption (good for addressing a # of risks, but not so good on other risks)

- In those cases, we are going to not really be addressing the problem.
- If we have security solutions and we want to reduce the risk to a suitable level, have I introduced new security problems for my solution?
- Intrusion detection systems that are capable of being attacked.
- If you add the intrusion detection system, it can potentially be a detriment and the addition of that supposed security improvement has actually worsened your security.
- What tradeoffs do I require?
- Have a biometric access mechanism only for people who have the right retinal scan.
- Put their eye to it and determine if they are legitimate users in the building only if the retinal scan matches.
- It may be cumbersome!
- What if someone tampers with the authentication process? How do we recover from that?
- We will move on to talking about security policies.

Security Policies

- One of the overlooked elements is having some kind of definition of the policy you want to achieve.
- A security policy describes how a secure system should behave.
- There is an important distinction between **policy** and **mechanism**
- We want this happen, NOT this is how we achieve it.
- What we want to do vs. how we do it.
- If we do NOT know from a fairly clear way, we really do not know what we are doing and what effects we are trying to prevent.
- Throwing darts at a dartboard from great distance.
- End goal, not the algorithm to solve the problem.

Informal Security Policies

- A bunch of different users and users should be able to access only their own files.
- They can access our system only if they are legitimate users
- We don't want anyone else to be masquerading as that person.
- I don't want rogue executables showing up on my system.
- Only system administrators should be able to install and change executables on the machine.
- These kinds of informal policies seem relatively clear here.
- You put a bunch of mechanisms in place and it will allow users to only read their own files.
- You have to ask if that mechanism do what you wanted it to do.
- With an informal policy, it is hard to make firm statements on what you can do.

Formal Security Policies

- You can say exactly what you want to have happen.

- You can reason about it in a formal way via proof methods i.e. mechanisms actually achieve the policies that I said I wanted.
- Match to a particular model that you want to do.
- Bell-La Padula model
- Ensures secrecy of data for a particular class of systems.
- If you go back to informal policies on previous slide, it is difficult to express many of those in a formal way.
- If you can, it isn't that easy to do reasoning and it can be quite challenging.
- You may not get all the desirable effects.

Some Important Security Policies

- Bell-La Padula: read down, write up -> military model
- You can read down information that is secret because you are responsible for more secrets.
- It is acceptable to write up because we aren't reading in that case.
- Biba Integrity Policy: read up, write down -> warehouse selling widgets
- You know you are more knowledgeable so they trust you more.
- You don't want to get bad info from lower employees.

Bell-La Padula Model

- Derived from the military.
- Various people working in armed forces will be trusted to different degrees.
- Want to achieve the effect that very sensitive information can be seen by only a few people.
- The actual Bell-La Padula Model has some complexities that are too advanced.
- There are two important terms used in the Bell-La Padula Model
- Clearances
- Classifications

Clearances

- Subjects: anybody who can do something i.e. people or programs.
- Have some clearance describing how trusted they are and what level of sensitivity
- Description of how much you can trust the subject.
- In U.S. military use, common terms are unclassified, confidential, secret, top secret, etc.

Classifications

- Each object is assigned some classification
- Each entry that gets sent will have a classification
- Describes how sensitive the object is.
- If someone is cleared for secretive information, the implication is from an informal point of view.

- An unclassified clearance is NOT allowed to see secret information.
- We don't want secret information to be seen by people who have unclassified clearance, but we want it to be seen by those with top-secret or secret clearance.

Goal of Bell-La Padula Model

- Prevent any subject from ever getting read access to data at higher classification levels than subject' clearance.
- Concerned with the contents of the objects and not just the objects themselves/

Bell-La Padula Simple Security Condition

- Subject S can read Object O iff l_O <= l_S
- Subjects have clearances
- If you have top-secret clearance, you get to read top secret objects
- If you only have secret clearance, you do NOT get to read top-secret objects

Why Aren't We Done?

- We care about the data that is in the object.
- We care about the information that it holds.
- If we just use that previous condition and know restrictions in the Bell-La Padula model
- If he is careless, he could write that information to a confidential object.
- This implies that if someone else is in the system, that other subject can go to this confidential object and open it and read it.
- He could end up reading top secret information!
- **This is bad!**
- Thus, we have a second property

The Bell-La Padula *-Property

- S can write O iff l_S <= l_O
- **Prevents write-down**
- Take high-classification information and write it to a low-classification object.
- We want to prevent that and this property is meant to prevent that.
- Top-secret user has read top-secret data as he is allowed to do by the security condition.
- He is top secret, so it is confidential.
- He has a clearance that is > classification of that object.
- He cannot perform that write in this case.
- It is NOT safe for him with the knowledge that he has to write into this low classification object.
- We can prove that the system that meets these properties is secure.

Bell-La Padula Example

- General is in charge of the battlefield
- Most trusted person so he has top secret clearance and can see everything
- Scouts are secret
- Can provide better information of what is going on in the battlefield.
- We don't want them to have all secrets because they might be captured.
- Soldiers are classified
- Can be captured in war, so we don't want to give them too many secrets.
- They won't inadvertently divulge secrets to the enemy.
- In order to run the battle, the general must know what is going on.
- Keep track of all the geographic feature and weather conditions.
- Where are my troops and where are the enemy's troops?
- This is really important information, so this must be labeled **top secret**
- Let's say the enemy in red moves a new tank into the battlefield.
- They then say let's write that onto a situation awareness map.
- If we go through the Bell-La Padula security conditions, it is okay for secret subjects to write to top-secret objects
- This is okay because they are NOT reading information!
- Sooner or later, the general goes and looks at the map and sees a new tank.
- Permitted to do that because he has a top secret classification and he can perform the read.
- He may decide that he has a unit that is perfectly positioned to attack that tank.
- Those units are represented by classified soldiers on the map and attack the red tank.
- They will presumably perform the attack that is ordered to be performed.
- Bell-La Padula says top secret subjects cannot write to a classified object because he might inadvertently write info that is too high up.
- **Does NOT allow write down**

So How Do You Really Use The System?

- Need some mechanism to change orders to be merely classified.
- Not going to do this in any automated fashion.
- General can change the classification of a data object to be sent to the soldier.
- The problem is that if you do that, you have lost the formal guarantee that you got from Bell-La Padula.
- There is no sensitive information at a higher level of clearance than what I am labeling in the reclassification process.
- Data can leak and the whole goal of Bell-La Padula is lost if we are not careful.
- You could have different classes of information.
- Explicit operation -> tells the higher power that they have permission to write.

Integrity Security Policies

- Devoted to a single security goal
- Secrecy
- Certain information will not be seen by certain other parties.
- Doesn't do anything to guarantee particular data objects will guarantee integrity.
- Attackers will be unable to properly change a data object.
- Secrecy is the most important property for the military.
- In other cases, integrity is equally or more important
- Commercial systems.
- If a business depends on having correct information, then we have a problem. We want to make sure that the inventory is correct.
- We don't necessarily care if it is secret but it must really be in the warehouse.

Example: Biba Integrity Policy

- Shares some similarities between Bell-La Padula but has some differences.
- Subjects are things that can do things.
- Objects are things that hold data.
- We can also have a set of ordered levels
- Levels of sensitivity for information.
- We don't have those here, so we care about integrity levels.
- High integrity data: we want to ensure it is not improperly changed.
- For every subject, we are going to assign an integrity level.
- The basic concept is that high integrity levels are assigned to people who we trust won't make mistakes.
- If he says there are no widgets in the warehouse, then there won't be widgets in the warehouse.
- Random guys will be initially given low integrity levels.
- Generally speaking, high integrity users are trusted to ensure the program runs properly.
- We want to make sure data at high integrity levels are less likely to screw up.
- It is important for this information to be kept correct.

Biba Integrity Policy Rules

- s can write to o iff i(o) <= i(s)
- If we have a piece of information that is highly critical and must be correct, only people we trust a lot are able to write it.
- Some of our programs are things we trust very much because we are sure they are correct.
- Each of these programs is a subject just like a human would be.
- s_1 can execute s_2 iff i(s_2) <= i(s_1)
- A guy with extremely trusted integrity levels isn't allowed to look at an inventory handed to him by a clerk hired yesterday.

- Why do we need the write rule and read rule?
- We don't care about improper data.
- Allows someone of integrity *i* to only read objects that have a higher integrity.
- That way they can be trusted because they know more information and thus have a higher integrity.

Hybrid Models

- It is not always a case where we care only about secrecy or integrity.
- Chinese Wall model
- Used by financial firms i.e. financial analysis or auditing of companies.
- These companies that do these financial services have many different clients.
- Work for external corporations.
- Concern that if you work at Microsoft and you audit Microsoft's books, it could get leaked to IBM for example.
- One thing that we can do is to say if we follow a set of rules, you can make strong guarantees that information has NOT leaked properly from one set to another.

The Realities of Discretionary Access Control

- Most users never change the defaults on anything.
- Individuals users can say how they want to get things done.
- If we are going to use this kind of discretionary access policy, there are some things that we need to be aware of to normally be true.
- Most users never change the default access permissions of anything.
- Users will not go in and customize access permissions on **most** things
- If they cannot do what they ordinarily want to do, they will change that though.
- The reason this happens is that most users don't think about access control
- They don't generally care; even if they did, they don't understand the mechanisms used to perform access control.
- You should NOT rely on discretionary access control to protect information.
- If you are the one setting the access policy and you understand what you want to do, maybe it will be fine.

The Problem With Security Policies

- Hard to define properly
- Quite difficult to formalize.
- Even if you get to that point, it is hard to take the mechanisms to achieve the policy or understand the implications of this policy.
- There may be some unusual circumstances that you didn't even think of.
- Hard to scale out!
- You should define policies and try your best to achieve them.

- This is better than what most people do.

Tools for Security

- What tools do we have available to achieve the security effects we would like to achieve?
- We will have a brief outline of the types of tools:

Physical Security

- Lock up your computer
- Sometimes, this is a good answer since you make it impossible for the attacker to get physical access to the computer.
- This has become less true since most computers are attached to networks
- Networks poke a hole in the locked door.
- The attacker can access the computer by using the network.
- Another issue is that many devices nowadays are mobile devices (smartphones)
- Very hard to say we have any degree of physical security.
- The owner will have the device in his hands at all times and he can do essentially anything from a physical perspective from that device.
- Common to do jailbreaking on smartphones that disable features.
- If we cannot guarantee physical security, it is hard to achieve other measures of what we want to achieve.

Access Controls

- Policies in this previous class were related to access control in various ways.
- Let authorized parties do what they wanted to do.
- Rather tricky to achieve.
- Hard in a network environment.
- Once the data is out of the control of your computer, it is essentially gone.
- This is particularly important in the context of networks.
- We are essentially saying we lost access control to that version of the data.
- We cannot guarantee data will be taken from him.
- For every file, you have an ACL.
- What if I say I want to deny this user an access of any files?

Encryption

- Algorithms that hide content of data or communications
- Sender and receive know a secret that is unknown to everyone else
- If you know the secret, you can decrypt the message and find out the true contents.
- If you don’t know the secret, it is nearly impossible/very hard (which is ideal).

- Helps provide secrecy of data but can have important aspects of security and authentication.
- There are some problems where it doesn't matter if the data is encrypted or not and we cannot solve encryption problems.

Authentication

- We will allow certain people to do certain things and NOT allow other people to do other things.
- Example: Professor Reiher can do certain things, and we will determine by the basis of idea who can do what.
- Multiple methods to perform authentication.
- Many rely on cryptography, but not all of them do.

Encapsulation

- We run into circumstance where we do something for somewhat trusted but not entirely trusted entities.
- We can do that action in an encapsulated environment.
- Nothing else can be done and he cannot break out of the environment to do things they didn't approve of.
- Virtual machines are one form of encapsulation but it is extremely challenging to break out of encapsulation.
- Limited access to certain resources.
- Analogous to least privilege to ensure that attackers cannot exploit weaknesses.

Intrusion Detection

- We need to be aware that we should expect all of our security methods to fail under certain circumstances.
- Even if we are right, in certain cases, the assumption is certainly not true.
- In some cases, we are just fucked.
- From that point onward, we have no guarantees and we are shit out of luck.
- It would be better to say if mistakes were made and there were conditions we couldn't handle, we detected a problem and we can at least take corrective actions and remove the intruder or close the hole.
- Try to improve the security situation.
- Damage control.
- Reactive mechanism, NOT preventative.
- It detects when things have happened and lets you take remediative actions.
- We cannot prevent all bad things from happening all the time.
- We need ways to detect problems and recover.
- It has to pretty much be automated and we need algorithms that examine state and information.
- We cannot rely on human beings to do certain things because they are too slow.

Common Sense

- A lot of problems arise because people don't think.
- Didn't think about implications of actions.
- Even great security tools will NOT perform well if users use it badly.
- I just need to find some person who isn't thinking carefully and I just need to fool him.
- If he is not using common sense, he will fall for it.
- Common in the design of systems and the use of systems.

Access Control

- We can get a fairly safe computer by encasing it in concrete and dumping it in a trench.
- This is NOT the ideal situation we want!
- We want some people to get access, but not everyone to everything.
- A particular thing that might happen should or should not be allowed.
- Talking about this in a mechanical sense.

Goals for Access Control

- Complete mediation
- Every time an action is performed, we want to see if it is okay.
- Least privilege
- Give him a set of access privileges that are pertinent to that action and no more than that.
- Useful in a networked environment
- Almost everything we do has network implications.
- Scalability
- We are beginning to deal with systems with vast amounts of information.
- Acceptable cost and usability
- Ensure that it is not too expensive in too many dimensions.
- Human beings should understand what is being done and why or why not they are allowed access.

Access Control Mechanisms

- Access control lists and capabilities are more common
- Access control matrices but not too common and won't be discussed in detail.
- Role based access control
- A different way of looking at the problem.

The Language of Access Control

- *Subjects* are active entities that want to gain access to something
- *Objects* are things that can be accessed
- *Access* is any form of interaction with an object.
- An entity can be both a subject and object
- It may be a program that is responsive to users or other programs.

- Other people will try to access the program.

Mandatory vs. Discretionary Access Control

- Mandatory access control is a policy that is enforced
- It doesn't matter if he is talking about his own data.
- Individual users cannot override it even for their own data.
- There are mechanisms which you can make that change, but it is part of the mandatory access control policy.
- Discretionary access control provides a set of tools for each user.
- You can as the owner set whatever access control policy for the data.
- Tell me what you want and I will change the policy.
- I am not going to force you to do anything!
- Most systems work with this policy and this is what we see in Unix, Linux, Windows, etc. (most common OS)

Access Control List

- Some resource we want to protect -> put it in a list!
- There will be an entry on the list that specifies the user or program
- There will be other entries and each can have a set of access permissions
- If we want to use this to control access, we figure out what his mode of access is and we see if he is listed on the access control list.
- Look up entry and see if he can access the entry in the following way.

ACL Example

- Use the OS to control access to files
- If we have 3 different users, we will have our subjects be users.
- User A is allowed to read and write to the file in question.

An ACL Protecting a File

- Subjects A, B, and C
- Associate an ACL specific to this file.
- This ACL has three entries each with their own unique write permissions.
- Subject A says I would like to write
- We look up in the ACL the permissions and after confirming, we allow it.
- Subject C wants to read
- We look up in the ACL and since it is not permitted, we deny this request.

Issues for Access Control Lists

- How can we trust the requester is who he says he is.
- We need protection to overcome the access control list and make use of improperly changed access control lists that we didn't want him to have.
- With access control lists, we go through every single object in the system and look at all the files.
- We have to examine an access control list for every single object in the system.
- For extraordinary situations, that is perfectly reasonable.

- We need to figure out what resources to access and this is NOT always very practical.

Pros and Cons of ACLs
+ Easy to figure out who can access a resource
+ Simple to change permissions
- Find entries you want to change and just change it.
- Hard to figure out what a subject can access
- Can only change access rights when you get to an object
- If I want to do things for an ACL, he will have complete rights to access everything that I can access.

Capabilities
- Doesn't have a list associated with every object.
- Instead it is associated with every subject.
- Like having a bunch of tickets to get in different doors.
- Each key opens a different door.
- If you have a capability that lets you access something, the mere fact you have it lets you access that sort of thing.

Properties of Capabilities
- Must be unforgeable
- If you want to access something, you need the right capabilities under it.
- Not talking about actual physical keychains.
- We are referring to computer systems that are stored in bits.
- Do NOT give users direct access to capability lists
- Have the OS control this.
- Only OS can add or take away capabilities from a particular subject.
- Not as effective in a network system.
- An interesting aspect of capabilities is that you can set up your capability system based on an original capability.
- This has an interesting characteristic!
- If I want to do something for some other guy and I don't want him to get my full set of things, I could generate a new process and give him only a subset of the capabilities that I have.
- It only can do things that we limit the capabilities for, and we can determine exactly what capabilities we allow.

Capabilities Protecting a File
- Subjects A, B, and C along with a Capability Checking module and capability lists for every subject.
- They are on the OS side of things and they cannot arbitrarily alter the capability list.
- Gain the use of capabilities but he cannot change them.
- Subject A wants to Read X
- First goes to Capability list, sees a capability permitting read

- The capability checking approves and says it is relevant.
- Lets him Read X
- Subject C wants to Write X
- Looks for a capability relevant to File X.
- In a real OS, it would simply reject the call at this point.
- In theory, it goes to Capability Checking module and denies access.
- Let's assume we have a set of different machines (color coded)
- Capability lists live with their subjects.
- We cannot necessarily trust the machine where Subject A lives
- Not necessarily a machine under our control.
- How can we tell if this is a good capability?
- Bits need to be responded to and allow him to do what he has to do.

Q. What is the advantage of using a capability list over an access control list?
A. Access control list is for each object. Example is ls -l (you can see the permissions for each file, directory, etc.). Capability list is per subject i.e. user, administrator, etc. It is usually more scalable to modify capability lists because you group things that way, rather than modifying each file, directory one at a time.

Revoking Capabilities

- Accounts receivable database with a user Fred.
- The key indicates that he has the capability to access the Accounts receivable database.
- User Nancy also has a key getting to the Accounts receivable database.
- Assume Fred moves to a new job and doesn't require to access the Accounts Receivable database.
- We need to take away Fred's capability without changing Nancy's.

Options for Revoking Capabilities

- Bundle of bits -> working on a multi-machine environment and it needs to destroy the capability.
- How do we know there is no other copy of that capability.
- We could also say we need to remember it is NOT a valid capability anymore.
- Keep a list at the Accounts receivable database.
- If Fred wants to do it, we can deny it, and we need to check on use.
- Another choice is generation numbers
- Remember just one # for the object and we need to remember capabilities for earlier generations.
- Nancy will be able to access the Accounts receivable database and we might need to change her capabilities.

Pros and Cons of Capabilities

+ Easy to determine what a subject can access
+ People like thinking of capabilities

- Create a subprocess with limited properties from the original audience.
- Need to go out to every single subject and check their capability list.

- Investigate every single subjects.
- Revocation require extra mechanism and is painful.
- Cryptography is needed and tends to be expensive.

Distributed Access Control

- ACL's still work OK under certain assumptions.
- Global namespace for subjects and no one can masquerade.
- The guy who shouldn't be given access might be allowed access.
- We need unforgeable capabilities and make sure the wire is NOT tappable.
- Create a capability that is cryptographically based
- Helps prevent capabilities from being formed out of thin air.
- It doesn't prevent the copying of capabilities
- Even if it is just a bundle of bits, anybody with IP access should not be mishandling it.

Role Based Access Control

- Every user works in different roles at different times.
- If a user is operating in one role, we want to assign him different permissions that are suitable to that role.
- This has proven to be a powerful, useful concept.

A Simple Example

- Example of **Role Based Access Control**
- Install new software
- Fred is a system administrator and he needs to do many powerful and dangerous things
- Simultaneously, he is also a normal user who has a normal email and has to look up information about when his paycheck is coming in.
- Fred should operate under **one role** while doing system administration.
- We need to give him a set of suitable access permissions for particular tests.
- He can update the C compiler as a system administrator, but he cannot look up videos of cute cats on the network.
- Otherwise, he must explicitly change roles and we will take away his access permissions.
- Another analogy is myself (Matthew Lin)
- I am Circle K's technology chair and I am a normal user. When doing CKI related duties, my role is technology chair. When looking at personal emails, I am a normal user.

Continuing With Our Example

- Fred first reads his email.
- Fred changed his role and changes his role to sys-admin
- Evil malware in friend's com for example
- Evil malware in Fred's email cannot "upgrade" the compiler

- If he got an email trying to change the C++ compiler, he would NOT have had permissions to change it.

Changing Roles

- Need on explicit notion of roles
- We have to have some action that allows individual users to change from working in one role to working in another.
- Preferred that the data is a little more unique.
- There has to be something Fred does that won't be affected by evil software.
- Often done via password when changing from one role to another.
- Could be a password on a per roll basis.
- Other system administrator's probably have a different password

Practical Limitations on Role Based Access Control

- Run into problems via disjoint role method privileges.
- There will be system administration overheads
- Setting up every access control list in a timely manner.
- If you change one thing, you have to change other things as well.

Reference Monitors

- Widely available in all our systems and in some cases, it will be part of add-ons or will be built into the system.
- There will be some actual code in the network actions
- This code is called a **reference monitor**
- Correctness is very important and we want the code to never get the answer wrong.
- We also want proper placement of the code and in every path to gain access to an object.
- If there is a path that lets a subject go through the reference monitor, then we aren't going to have Access Control to check the reference monitor.
- We want it simple and reasonable flexible to perform different security options.

W 1 Dis 4-1-16
CS 136
Josh Joy
jjoy@cs.ucla.edu

Q. How many people can be using Deter at once?
A. For the first exercise, it is only 19. As we go out further on the course, we will increase the # of nodes.

- Controlled by ISA (Berkeley and USC). They host this infrastructure.
- Better to get work done early because there can be some downtime.
- There will be contention for these machines.

Q. Why do we ssh twice?
A. Firewalls regulate connections going in or out. Based on your user id, these are then going to be mapped to these resources requested.

- You can swap out to take breaks and save progress, but terminating experiments deletes it completely.

Difference between swap out and terminate?
A. Swap out is persistent and this is like a private EC2 and creates a template for instructions on the infrastructure. User specific settings are your assignment and these can be replicated and stored.
If you want to go to class or get lunch, swap out so you save your work.

When you terminate, you tell Deter to throw away their resources.

Q. Do you turn it in on Deter or CCLE?
A. Submit things on CCLE

Grading is as automated as possible and they run the scripts and do manual grading

- Course notes and Deter notes are not necessarily hand in hand.
- The lab assignments are pretty detailed, so you can go ahead and finish these ahead of time.
- The deadlines do NOT allow for late submissions (don't fuck around!)

California Data Breach Report

- In California at least, any security breach (> 100,000) has to be reported to the Attorney General
- Nearly three in five Californians were victims of a data breach in 2015 alone.
- UCLA Health
- Have I been pwned?
- UCLA UCShip
- Affects pretty much anything (banks, doctors, students, spas, etc.)
- Malware: software like a Trojan horse or a virus where if you send an email to someone, it runs some sort of background script that collects your data
- Physical breach: Veteran Administration (VA) system get medical support from the VA system.
- Centralized: the data is stored on one main node
- Distributed: every individual owns their own data
- There are 25 million records and all the VA records got stolen (weakness of centralized)

Q. Impact of a credit card breach vs medical breach
A. Credit card breach will be used to straight up for money. In the U.S., it isn't really an issue for credit card to have a PIN.

For medical data, people can use identity theft so medical information has a lot of information that only you should know. They can also hold information hostage, so it is a ransomware attack. You have to pay millions of dollars to get information back or else they will die.

Ransomware: encrypts your entire hard drive and you cannot find anything unless you pay for your key.

- Medical expenses are very high; people take this data and make fraudulent purchases for large expenses.
- Bigger issues because it takes months or even years to detect this kind of fraud.

Security is asymmetric: the attacker ALWAYS has the advantage.

- Much easier to attack than it is to defend

Transitive Trust

- Does NOT hold for security guarantees

Examples of Transitive Trust

- Certificate authorities: there are a relatively small group of authorities that assign certificates to ensure that this website is one you can trust.
- Uses a chain of trust: trust starts in each of your browsers to validate your bank.
- Q. Are there any issues with this?
- A. HTTPs checks the browser's store to see if it is validated or invalidated. Browsers will tell you if a certificate authority is trustworthy and embed the instruction.
- Stuxnet Example
- What prevents malicious updates? It is signed by some sort of certificate.
- In the case of Student, one of the certificate authority's was compromised and it was infiltrated. It masqueraded a Microsoft update!

Security Goals

- CIA
- Confidentiality: Ensuring access is only given to the intended audience.
- Put an intended message in a box and only intended recipients can open it. Defined by an **access control list**
- Integrity: Integrity checks (you can use a checksum!)
- Some sort of hash or checksum and perform the same verification yourself to ensure someone hasn't tampered with it.
- Availability: DDoS attacks
- Deny people who can access the service

Social Engineering Attack Example

- Ubiquity Networks Inc.
- Small company where everyone works through email.

- How do you communicate with your family i.e. does the CEO travel a lot?
- CEO scam is where criminals either hijack or impersonate the email of a senior member of the staff within the organization.
- Someone fell for the CEO scam and wired the money to a criminal account.

Possible solutions: attach a PGP and a signature for your email.

- Ky-Cuong signs everything i.e. encrypting tax documents with his dad
- Mac has a built in feature
- Ky-Cuong uses it when he sends anything that is kind of important.
- Q. Is there any tradeoff between security and usability?
- A. Extra steps for extra security.

Bell-La Padula Example

- Top secret is stronger than secret (secret can write up)
- Public cannot view this
- Read down means you can read anything that is below what you are currently at.
- Your subordinates can write messages to you.
- Q. Why is this useful?
- A. This prevents data leaks, so someone who knows any secrets cannot tell normal people.
- Anyone writing secrets (assuming no backdoors) will not have to fear information being leaked to the public.

Biba: If this was some general or commander-in-chief, you want to give some instructions and be careful about giving the memo and not making it super classified.

Apply ACL (access control list) to provide authorization.

- Authentication is to see if someone is who they say they are.
- Classification of secret or top secret can be thought of as permission

Q. What is the issue of these mechanisms?
A. ACL breaks very easily. It defeats the purpose of the security system and you want to issue these and you don't want to constantly be changing things.

- Lecture 2, Page 24
- If top secret needs to communicate with secret, there needs to be an explicit operation.
- Situations will demand that there are capabilities that need to be changed, and there has to be a system administrator to grant you privileges.
- Usually requires an explicit operation with an administrator that has to be trusted.
- Chinese Wall model: used in financial sectors
- Buyout of other companies. Before they merge, they have a "Great Wall" separating things.

Q. Difference between access control and capability?
A. ACL is defined on each file and if you log in, each user will have a mapping. Every file or every folder has an associated access control list. Capability refers to an entire subject (user, system, etc.), so it groups files/folders together.

Issues for Access Control Lists

- You need some sort of authentication
- How do you protect the ACL from authentication
- Root users can for sure access the chmod command
- Some other users can have permissions (you might need another ACL for this)
- You need need some form of integrity check
- Generally an Operating System (OS) problem
- Linux file system uses ACL

Pros and Cons of ACLs

- + Easy to revoke or change access permissions
- - Hard to figure out what a subject can access
- In a real world system, this is hard because the file system can become VERY LARGE!

Capabilities

- Everyone has a token if they are allowed to do something.
- When you try to read or write a value, it is hard to authenticate if everyone has a token for that.

Stored as a tuple:
ACL -> {permission, principle}

- It is hard to revoke issues
- If they don't log in for a while, you have to revoke on use.

There will be some sort of structure to this capability

- There can be some elements where the resource is the same.
- Metadata will be unique for each user.
- There need to be some sort of semantics in the token to protect against this.

ACL

- ACL is handled transparently; there are some tradeoffs to using it.
- High-level idea of capabilities so if you want to revoke, it won't be so easy to do this.

Pros and Cons of Capabilities

- +: Potentially faster than ACLs
- +: Easier to determine who has access to the ACL

- -: Hard to determine who can access an object because the capability can be quite spread out.
- -: Greater overhead because it requires an extra mechanism to allow revocation.
- -: Need cryptography in networks to prevent forgery of capabilities.

Role Based Access Control

- Abstraction based on ACL
- Easier to classify individuals and we need a subset of permissions rather than dealing with complexity.

Q. Lecture 2, Page 28: Why do we also need the read role for Biba?
A. Biba's write down is used to issue some order. We don't have read up because no one would be able to write in this case.

Q. Why can't we read up? Why cannot everybody just read everything?
A. Intuition based on corruption. If you have 3 pieces of data, you want to maintain the highest piece of integrity. The highly trusted person can write on top of everyone else because they can have the most trusted information. Comes from someone else trusted than they are.

- Biba can only write down because you don't want someone writing above their authority. For read, you only read up because you don't want the manager to say they trust all the stuff below.
- In the military, they are going to be using all sorts of access control and it is going to isolate a lot of things.
- Soldiers can restructure all these things for the model.
- We can use this system and we know we are going to communicate only with these subordinates.
- You have to think about what guarantees to provide in this case.

ACL is very complicated and you can see it is easy to rationalize and reason if you have to contain to this model.

- It is NOT easy to reason who has guarantees and who has access.
- In the military, they want strong guarantees.

W 2 T Lec 4-5-16
Thinking About Threats

- What are we trying to do with computer security?
- Protect against threats
- Confidentiality
- Integrity
- Availability
- Essentially an attack on a normal service that should be offered.
- Service should be an a transfer of data from info source -> info destination

Interruption

- Information source would like to get info to destination but an attacker blocks the data

Interruption Threats

- Denial of service
- Attacker doesn't want the service from being offered.
- Not so much you want to deny service but you don't want a piece of information getting to the receiver.
- Since the request never gets asked, the information never gets sent.
- Sometimes a threat to confidentiality and integrity, as well as availability.

How Do Interruption Threats Occur?

- Overload a shared resource.
- Far more demand than availability
- Things tend to get dropped.

Interception

- Appropriate places to deal with data refers to an unauthorized third party.
- An unauthorized third party gets information and he isn't supposed to get it and he does.

Interception Threats

- Legitimate request going on and the attacker has been able to send a request to get to him.
- Giving someone info they aren't supposed to get.

How Do Interception Threats Occur?

- Eavesdropping
- Hearing things in a communication channel that they aren't supposed to hear.
- Communication channel is anything that allows data to move from one place to another
- A bus
- Shared memory
- This means anything can be eavesdropped
- Masquerading
- Pretend to be legit party that gets info
- Break-ins
- If you consider anything an info channel, if you can get into your disk drive, you have interrupted the communication process and you can consider even a break-in to be an interception threat.
- Illicit data copying
- What if you keep data you aren't supposed to keep

Modification

- Info source sends data but then an unauthorized 3rd party pulls the data and changes it.
- Blue went out, red comes in
- NOT good!

Modification Threats
- Unauthorized party modifies data
- Data could be stored in the server or sent to the client

How Do Modification Threats Occur?
- Get info as it is transmitted across the wire.
- Change it and put it back on the wire.
- You can also masquerade
- Hi, I am a good guy, when the real good guy asks, he gets the changed data.
- Flaws in applications allowing unintended modifications
- Exceedingly common i.e. buffer overflows
- One thing about Computer Security is it draws a great deal of creativity

Fabrication
- Unauthorized third party just tosses something at the destination
- The source never sent this info!

Fabrication Threats
- Can cause improper behavior or other bad behavior
- Can cause him to deceive other people.
- Throw a bunch of garbage to whoever you are attacking
- Say I am going to ask someone else to send the data to my target.
- If you know about IP spoofing, you can create a request to a DNS server that says tell me the whole table.
- If the person giving the whole table is the guy he wants overwhelmed, that is the goal of the attack.
- You want bad things happening to someone else.

How Do Fabrication Threats Occur?
- Fake your IP address
- Bypass protection mechanisms
- Duplicate a legitimate request and you kept a copy of it and you want to throw that request back into the system.
- If you haven't designed carefully, you will accept that request.

Destruction Threats
- Blow up the source and it is gone.
- Once the source is gone, it won't provide data to anybody obviously.
- Can imply destroying physical hardware

- Back in 2007, someone demonstrated you can attach a power generator to the Internet
- Because it was electronically controlled, if they knew how the machinery worked, they could cause the generator to burst into flames.
- NOT good!
- Link in the slides below that shows the process happening.
- Student used to destroy centrifuges to enrich uranium.
- Prevent availability of power generator and we don't want databases available.

Active Threats vs. Passive Threats

- Passive threats don't do anything except listen
- Doesn't modify or create fake requests.
- Threats to secrecy
- Active threats are far more aggressive
- Refer to ALL properties
- They allow you to do anything.

Social Engineering and Security

- A lot of problems have a social engineering aspect
- If I cannot break your technical mechanisms, maybe I can influence people to do things they shouldn't do.
- When there is some sensitive stuff, there are people who are allowed to do it.
- If you can convince one of those people to do what he shouldn't do, you will have gotten someone who has the right to do it to do it for you.
- You might get an info saying something about your account and ask you to send your password
- Never do that!
- Always trying to steal your password!
- You want to be positively sure you are dealing who you think you are dealing with.
- These attacks are cheap, easy, and effective.
- Work quite well since if you sent these attacks to millions of people, they will send back info you will want.
- You can have an absolutely perfect technical system, but a good social engineering attack will blow it all away.
- One careless person can undo all the great security work I have done.

Social Engineering Example

- Phishing
- Visit a website and the attacker will send you plausible info that he ha set up to look like the real website.
- He wants you to go there, think you are dealing with the actual website, and input all those information.
- See how careful he has been making it a correct website.

- It is necessary to have a victim who falls for it but many people including many intelligent people can fall for it.
- There has been a telephone scam and I got the IRS scam.
- If there is a great sense of urgency, it is probably NOT real at ll.

How Popular is Phishing?

- Anti-Phishing Work Group reported 65,000 unique phishing sites in December 2015
- Attacking 406 different brands like banks, Internet companies, etc.
- It isn't really our fault that people fall for these things but is there any way to make it less likely.

Why Isn't Security Easy?

- Not an easy problem.
- Hard to provide security for what you wish to provide.
- Much different from other CS problems.
- How do you ensure reliability in the face of components that could fail.
- If you write the code right, it is NOT always a situation you can fix.
- This human opponent is likely to be more clever and/or persistent.
- We are trying to do something that has been historically proven difficult
- Historically since the beginning of the human race
- **Hard to share secrets!**
- This is really very hard!
- Difficult problem long before anyone thought of computers.
- Cavemen secret of where to find the best berries.

What Makes Security Hard?

- Mistakes are costly.
- Do we have to wait for bug-free software?
- That is impossible.

How Common Are Software Security Flaws?

- SANS is a professional organization and they used to publish a weekly compendium of security flaws.
- All the popular commercial software
- About 1500 security flaws per year
- Focused on popular software that had real security implications they could see.
- There is reason to believe there were more than 1500 security flaws.
- NOT done anymore because it is unreasonable to patch 1500 security flaws (about 30 a week).
- They still publish a weekly email but it usually only has on the order of 4 or 5 serious problems that are major issues.

Security Is Actually Even Harder

- The problem we have is that the computer is NOT the only point of vulnerability
- The attacker has to look for something else.
- They are going to be building a new version of the OS and we have to figure out who is on the programming team.
- System administrators
- Installation can be installed badly so that people can break in.
- Smart attackers will go for your weakness like in tennis.

A Further Problem With Security

- Security costs
- Computing resources
- # of cycles spent per second.
- People's time and attention
- Investigate the alarm to see if there is an actual attack.
- Money
- Buy a new piece of hardware or software or hire a new computer security consultant
- Mechanisms have to be used the right way
- You want security to work 100% effectively and with 0% overhead, inconvenience, or learning
- They don't require anyone's attention and users don't have to learn a single new fact.

Another Problem

- Most computer practitioners don't really know secure programming practices
- Let's say you get a job with Microsoft
- They will have you put on a month or so in classes teaching you to write secure code.
- NOT so great for startups!
- Take whatever they get out the door.
- If they don't know what to do, they won't get secure programs.
- Few sysadmins know how to configure security!
- Programmers who might have gotten a Bachelor's Degree who know things about computers and computer systems.
- We have a billion users out there; very few fall into the sysadmin group.
- We are stuck with users who know absolutely nothing about computer security.

The Principle of Easiest Penetration

- Intruders are resourceful and will try to use any possible method to break into a defense.
- People will brag they will use the most modern cryptography, but if they access garbage software as a back door, it will render the modern cryptography useless.

- Attackers are lazy enough to say they don't want to attack strengths; they will target your weaknesses!
- Protecting your house so you put in a big steel door with the most modern strong lock on it.
- What if someone wants to break into your house? They will attack the fragile window.

But Sometimes Security Isn't That Hard

- If you have something useful for the next hour, then protect it only for an hour.
- The Principle of Adequate Protection
- Protect only items that are valuable. The amount of protection should correspond with the item's value.

Conclusion

- Security is important but hard.
- Security expert's work is never done, at least for very long.
- Job security is excellent!
- Security is full-contact computer science
- Most adversarial area in CS
- Dealing with genuine human opponents
- Very interesting, difficult, and will never be solved!

Introduction to Cryptography

- CS 136 is NOT a class in cryptography
- However, it is a major tool used in Computer Security
- You have to understand certain things about cryptography
- Understand the basic ideas and principles
- Won't tell you how to build a good cryptographic algorithm

Introduction to Encryption

- About keeping secrets
- Convert it from easily readable form to a form that is hard to understand.
- You don't want it done in a one way operation.
- Take an obscure secret and make it easy to read again.

Encryption

- Process of hiding info in plain sight
- We use encryption because we have a feeling that the attacker will get to see our data.
- We don't want him to use our data even though he can see it.
- The theory is the attacker gets hold of the secret data and cannot decipher it.

Encryption and Data Transformations

- Data is sent in bits or bytes

- Take one bit pattern and transform it into a different bit pattern
- Usually in a reversible way

Encryption Terminology

- People who have worked with cryptography have characterized it in terms of sending a message
- Not always what you are doing with cryptography
- When we are talking about this at a conceptual level, it is if you discussed it as a message.

Roles

- Sender is S (Alice)
- Receiver is R (Bob)
- Attacker is O

More Terminology

- Encryption is the process of making message unreadable by O (the attacker)
- Decryption is the process of making the encrypted message readable by R (the receiver)
- If done right, R can do the decryption and O cannot!
- A system performing these transformations is a *crypto system*
- Any of these terms work in the same general, conceptual space.

Plaintext and Ciphertext

- Plaintext is the original form of the message (P)
- The secret in its first form
- Use a crypto system to convert the plaintext into *cipher text (C)*
- The theory is that the cipher text is hard to read.

Very Basics of Encryption Algorithms

- Generally speaking, most of the algorithms use a *key*
- Used for decryption and decryption algorithms
- Commonly referred to as K
- Secret
- If you don't have the key, you cannot decrypt, but if you do have the key, decryption is really easy.
- This is the goal!

Terminology for Encryption Algorithms

- Encryption algorithm E()
- C = E(K, P)
- Decryption algorithm D()

Symmetric and Asymmetric Encryption Systems

- Symmetric crypto systems are several thousand years old
- P = D(K, C)

- Expand this to say P = D(K, E(K,P))
- Asymmetric crypto systems are about 50 years old
- Uses two different keys
- One to encrypt, the other to decrypt
- C = E(K_E, P)
- P = D(K_D, C)
- Take cipher text and don't use the same key to decrypt
- Practical security on the Internet depends on this system.
- Done in the background for this.

Characteristics of Keyed Encryption Systems

- One characteristic is if you take the same encryption algorithm but you change the key, you get a really different cipher text.
- Ideally cipher text 1 and cipher text 2 are going to be very different.
- We want the same to apply for decryption!
- If it is slightly different from the one decryption key you are using, it won't be close to the plaintext that we started with.
- It should be hard to obtain the plaintext if you try to decrypt without the key.

Q. If the key has to remain secret, but we need crypto to share secrets out, how do we get the key?
A. Big problem in computer security. You cannot do it! Bootstrap everything and start out with everyone knowing a secret and use that secret to create other secrets.

- Initially sounds pointless but the way this works is that we use a small secret for a long time.
- If I want to send you 500 GB, you can use 4000 bits of shared secrets to share the info.
- Extremely important point
- Diffe-Hellman is the next best thing

Cryptanalysis

- The process of breaking someone else's crypto system.
- Process of trying to break a crypto system.
- When designing a crypto system, they always bear this in mind.
- All crypto systems will be subject to cryptanalysis
- In order to build one, you have to understand how cryptanalysis works.
- The approaches people use because it informs you of how to use cryptography.

Forms of Cryptanalysis

- Analyze an encrypted message and deduce its contents.
- If you use one key and you send millions of messages, they will all be encrypted and you can try to look at each individual message.
- The better way is to use all the other info to figure out what key you used.
- Once you know the key, you can read everything encrypted as if you were the intended receiver.

- If you can break the key, you can read the messages directly.
- Based on crypto-rules, if you see a flaw, you can either get the key or information out, and then you can read everything.

Breaking Cryptosystems

- Most crypto systems are breakable from at least a theoretical point of view.
- Just a matter of contest -> Principle of Adequate Protection
- These cannot be broken by anybody.
- The chances are that it is pretty good if the amount of effort it takes is exponentially large.
- It must act as a deterrent
- The job of the crypto system designer is to make the cost infeasible

Q. Very expensive, how do you measure how hard or expensive it is?
A. Depending on the exact cryptosystem, there may be other ways of doing it.

- Based on what we know about math today, if the number is too big, it takes more effort to do.

Types of Attacks on Cryptosystems

- Cipher text only
- We don't know anything about message content, we just have an encrypted data piece
- Hard!
- Known plaintext
- If you have the plaintext, why do you need to break this?
- Chosen plaintext
- Throw some plaintext that you encrypt for me
- Differential cryptanalysis
- We generally want to try to get the key, and from that point on, we have what we want.

Cipher text Only

- You don't know anything except that you have a plaintext
- The classical way of doing this is working with probability distributions, patterns of common characters, etc.
- Hardest type of attack
- Attacks from the 17th and 18th centuries
- Some of these ciphers have not been decrypted.

Known Plaintext

- Somewhat easier
- Let's say you know someone took an IP packet and at the beginning, there was a source and destination IP address
- Perhaps you know what one of those addresses was, but at least you have some background info to crack the cipher.
- This is a much easier way to break cryptography

- If you don't know the algorithm, it is still rather hard.

Chosen Plaintext

- Here, you are trying to break someone's cipher and they will give you back the encrypted result.
- If you are very clever, by choosing the plaintext in the right way, you can figure out a whole lot about the cipher.
- Differential cryptanalysis
- If you save the data, you cannot do anything with it yet, but you can alter it slightly and deal with the data.
- Keep doing this and if you have a good cipher, you can determine things based on characteristics of the cipher.

Q. In what cases would we have something like that?
A. Sometimes it requires a certain level of cleverness. Smart cards that do this kind of thing.

- Credit cards that have a chip in it and then you can say I will feed you data and you have to send it in an encrypted form.

World War II Anecdote

- US was having a certain amount of difficulty in the Pacific and they had a feeling that the Japanese would attack another American base.
- They could read a certain amount of what the Japanese sent out.
- The Japanese used code words for things like Pearl Harbor
- If you don't know the code words, even if you broke the cipher, you don't know what the hell they were talking about.
- They were able to figure out that the Japanese would have a major attack on some possession of the US
- There were a few plausible possibilities, so what they did was they sent out a message about information that the Japanese could crack at Midway.
- Back came a Japanese message with a chosen plaintext attack
- In this case, America knew the Japanese would attack Midway, so they were prepared and this was the turning point on the war.

Algorithm and Ciphertext

- It is easier if you know the algorithm and the cipher text
- No longer the case if there are cryptographic algorithms like the NSA
- It is usually AES, otherwise, it is RSA
- We almost always know what algorithm is being used.
- The first thing you do is in plaintext, you talk to the guy you are trying to communicate with.
- You start encrypting things with AES
- We grew to use AES and this is a very common way to perform cryptanalysis
- Generally speaking, he cannot get more cipher text using that key and that algorithm.

- One thing he can do is say that he knows this algorithm uses a key, so he will try every possible key and move on to the next possible keys.
- *Brute force attacks*: Tries every possible key (no intelligence required)
- Given N possible keys, on average, you will get a solution after N/2 tries. After N tries, you are guaranteed a correct solution.

Timing Attacks

- Most commonly based on knowing access to the device performing cryptography
- You have to be able to watch algorithm performing encryption and decryption
- Observe how the algorithm is operating.
- Some algorithms perform different operations based on what the key is.
- If you know that XOR takes less time on the hardware in question than shift right, then you know something if you observe this going on.
- Use info on the timing to deduce the key.
- Successfully used to break smart cards
- A lot of transportation systems worldwide use these smart cards and they have value on it.
- People encrypt information on the smart card and if he is the attacker, he will put it under an electron microscope.
- Watch with the electron microscope and see how many microseconds it takes to do things.
- After a certain period of time, you will get the key and do whatever you want.
- You can then create a smart card that has the right key and put $200,000,000 on my smart card and ride it free for life.
- Riding the subway for free is not worth the cost of an electron microscope.

Basic Encryption Methods

- What can you do with bit patterns?
- Take one bit pattern and substitute it
- Mono alphabetic
- Poly alphabetic
- Shuffle the permutations

Substitution Ciphers

- Substitute one or more characters in a message with one or more different characters

Example of a Simple Substitution Cipher

- Lecture 3, Page 24
- Gradually transform one character at a time and then you have the same thing on the right.
- Everything is substituted to the previous letter

- s became r, z became a

Caesar Ciphers
- Translate each letter a fixed # of positions in the alphabet
- Reverse by translating in the opposite direction

Is the Caesar Cipher a Good Cipher
- Only 2000 years ago
- Too simple now
- Fails to conceal many important characteristics of the message
- This makes cryptanalysis easier
- See what is in the plaintext that hides the transformation
- Lets you deduce what the plaintext says.

How Would Cryptanalysis Attack a Caesar Cipher?
- Letter frequencies
- Some letters occur more frequently than others
- ‘e’ is more common than ‘x’
- Caesar ciphers translate all occurrences of a given plaintext letter into same cipher text letter

More on Frequency Distributions
- “e”, “t”, and “s”
- True even in non-natural languages
- Certain characters occur a lot in C code like ‘0’

Breaking Caesar Ciphers
- Identify (or guess) the kind of data
- The more cipher text you have available, the more reliable the technique.
- A very long message will almost certainly match.
- A guy wrote an entire novel without the letter ’s’ but this is very uncommon.

Example
- Lecture 3, Page 31
- Count the # of occurrences of each character in the message
- Apply frequencies and compare with the most common English letters (“e”, “t”, “a”, “o”, and “s”)

Applying Frequencies To Our Example
- m, r, s, and z probably translate to four out of five of the common English letters

Cracking the Caesar Cipher
- We know some of these letters almost certainly translate, so let’s try a couple and see.

- +1 isn't good because it only maps to a => b
- -1 works well because it maps to four of them.
- Cracked the Caesar Cipher!

More Complex Substitutions

- We could have a more complex method of conversion
- Always convert letters to the same symbol and not just an offset.
- Table always works the same way:
- E -> Q, B -> A, etc.

Are These Monoalphabetic Ciphers Better?

- Only a little better
- We can use the same techniques to figure out the most common letters in this case.
- We can make the opposite assumption and determine which are the most uncommon letters.
- Anyone with the slightest knowledge of cryptography can break it in a matter of minutes.
- Experienced cryptographers can probably just read it.

Codes and Monoalphabetic Ciphers

- Codes are sometimes considered different than ciphers
- "Transfer $100 to my savings account" -> "The hawk flies at midnight"
- No character to character transformation
- Notice patterns and see if there is a common message being sent.
- If you do this often enough, the attacker figures it out

Are Codes More Secure?

- Codebook is needed (if found, it seriously hurts the defender!)
- Make sure you dispose of codebooks because if the enemy finds out, you are screwed!

Polyalphabetic cipher

- Don't have a single translation for any given plaintext character.
- A common substitution is having one for odd positions and another cipher for even positions

Example of Simple Polyalphabetic Cipher

- Move one character "up" in even positions, and move one character "down" in odd positions
- This can confuse matters because now some of the 'd' refer to 'c' and some of the 'd' refer to 'e'

Cryptanalysis of Our Example

- Consider all even characters as one set and consider all odd characters as another set

- Do basic cryptanalysis and figure out it isn't all that hard to figure out transformations.
- We can guess and people who do cryptanalysis can do this.
- Have good guesses and keep trying things until they break it.
- Might requires several guesses to find the right problem

How About For More Complex Patterns

- A more complex transformation can e quite good if the attacker does't know the choices
- Known methods for breaking things.

Methods of Attacking Polyalphabetic Ciphers

- Kasiski method tries to find repititions of the encryption pattern
- Russia from the East and the Germans from the West constantly invaded, so the Poles were good at encryption
- Index of coincidence predicts the # of alphabets used to perform the encryption
- Divide things into sets and use this information!

How Does the Cryptanalyst "Know" When He's Succeeded?

- See the output value!
- If you get an syntactically meaningful message, you are probably right!
- If not, you probably did NOT get the right decryption
- Most popular cryptographic methods say we have a large # of keys, and the only one likely to translate to something non-garbage is the right key!

The Unbreakable Cipher

- Substitution cipher
- Theoretically and practically impossible to break
- You cannot guess the key
- Brute force will not work.

One-Time Pads

- You have a bunch of characters and use new substitution alphabet for <u>every</u> character

Example of One Time Pads

- If you use a one time pad, you have some choice of using the encryption in any way you want.
- XOR is an example

One Time Pads at Work

- 010 is our actual value
- Flip some coins to get random #'s
- 001 is our randomly generated value
- XOR 010 with 001

- Produces 011
- If you do NOT know the key, you cannot break the message because it is random.

What's So Secure About That?

- Any key is possible.

Why Is the Message Secure?

- Let's say there were only two 3-bit values that made any sense
- It can be 010 **(our actual value)** or 000 **(another arbitrary value)**
- All you have got is the encrypted message.
- Could the message decrypt to either or both of these?
- Yes, this is assuming we chose the key bits at random.

Q. If you can narrow it down to a couple of messages, could your chances still be pretty good?
A. He can look at possible messages and presumably, he has some way of knowing that based on other prior knowledge before the decryptions.

Q. Does the person making the original message have a different key?
A. It is theoretically possible to break; it is NOT necessarily useful.

- This is NOT a practically useful thing to do.
- Next thing to useless.

Q. Why does it have to be one-time?
A. You will find a pattern eventually in this case. We would then have to figure out the algorithm that shifts our pattern.

- The moment you reuse one bit of the key, I am screwed.
- As long as you generate random #'s, you should be good.
- This is what randomly showed up.

Security of One-Time Pads

- If the key is truly random, it is provable that it cannot be broken without the key
- If I can secretly transmit one bit of key, why bother using this at all?
- Key distribution is painful.
- WW II is the epitome of cryptographic examples
- Britain would drop agents into occupied France and each agent would have a one-time pad printed on a silk handkerchief.
- Rather painful because synchronization of keys is vital
- Gestapo could be knocking at the door and the silk handkerchief could have 1,000 bits on it.
- Synchronization needs to be exact, and they had a bunch of young women who would determine if a message was screwed up or not.
- Occasionally, they could decrypt an improperly sent message.
- Hard to generate random #'s

- They had to get random #'s from somewhere so they came up with the housewife effort.
- Enlisted some of them with a Bingo cage with a bunch of little balls.
- Have them spend an hour a day pulling out a letter and writing it down.
- Generates a random key.
- They had great mathematicians like Alan Turing and they were able to see patterns.
- Unconsciously, each housewife had a favorite letter and their hand would migrate to those letters.
- Solution: put a black cloth around to prevent any discrimination

One-Time Pads and Cryptographic Snake Oil

- Companies usually bullshit and claim they have "unbreakable" cryptography
- Usually they are selling one-time pads
- They distribute the pads across the network and they encrypt it using some other method.
- When we decrypt it, then they have read our one-time pad.
- Generate a pad with a non-random process.
- They can have a genuine one-time pad that they reuse and it is just a big, long substitution cipher.
- One-time pads are common in embassies
- This works out because they have sent by a secure courier with a suitcase bolted to his hand and he is protected.
- He keeps an eye on the handcuffed case and sends a CD with the one-time pad and as long as you are careful, you can use the one-time pad for a CD's worth of material.
- A lot of embassies use this and this only works because there is a secure distribution channel.
- Gets to one place to another with a high degree of certainty.
- Used for most important/secret things they don't want opponents to see.

Permutation Ciphers

- If you don't want to substitute, you can permute and scramble the existing characters

Characteristics of Permutation Ciphers

- Doesn't change the characters

Columnar Transpositions

- Write message character in series of columns

Example of Columnar Substitution

- Lecture 3, Page 60
- If you write it a certain way, it doesn't look like what I started with.
- Trivially easy to crack this cipher.

W 2 R Lec 4-7-16

- The other choice instead of substitutions is **permutations**
- Scramble the bit pattern from one place to another.

Example of Columnar Substitution

- Read down from the columns to produce some cipher text
- Looks cryptic in a form that is just a line of text.

Attacking Columnar Transformations

- If we suspect we are using columnar transformations, we need to figure out if it is a substitution cipher.
- Ratios of letters will still be the same and map to themselves.
- If it comes out to the same distribution as normal English (or any other language), then it is PROBABLY a substitution cipher.
- One thing he might try if he is dealing with an unsophisticated opponent is try to figure out how manny columns there are.
- There are certain probabilities of certain characters appearing together i.e. “th”, “en”, etc.
- On the other hand, “tz” is very rare.
- Diagrams and trigrams are moved apart, so we need to figure out how they were shuffled apart.

For Example,

- The $ sign seems suspicious, and we see how many characters until the 1
- Let’s count, there are 6 characters in between each
- This implies there was 6 columns in the transformation

Double Transpositions

- Do it twice!
- This shuffles things even further, so how do you crack it?
- The $ and the #’s are no longer the same distance away.
- This gets harder and harder with the more different types of transpositions you do.

Generalized Transpositions

- Any algorithm can scramble text and you can do it any # of times you do it.
- If you are using permutation, you are using a key that describes how you do these transpositions
- Key would consist of the transposition information
- Rules of the cipher tell you how to shuffle the bits.
- Frequently, you shuffle the bits multiple times.
- The more times you do it, the more the actual movement and it gets harder for the cryptanalyst.

Which Is Better, Transposition or Substitution?

- Strong modern ciphers use elements of both
- Transposition scrambles text patterns
- Substitution hides certain characters/bits
- If you combine both correctly, it is much harder for the attacker to crack.

Quantum Cryptography

- “Quantum is very technical and seems advanced
- Basic idea is to use quantum mechanics to do key exchange
- One time pad -> very painful to exchange a key
- If you do it fast and generally enough, they could NOT be eavesdropped on because of the physics characteristics.
- People have built equipment to exchange keys and perform cryptographic operations over a mile or more.
- They aren’t really doing what we want to do, and making use of the quantum properties i.e. quantum indeterminacy or quantum entanglement in a perfect way.
- This means these implementations can be attacked or cracked because they are still flawed and under experiment.
- These are based on NOT quite correct assumptions on the quantum properties.
- NOT ready for real world use
- Requires heavily specialized equipment and great training to use this equipment.
- If you need to always communicate over two points, maybe one day you can use quantum cryptography
- We have nothing anywhere close to that, and it cannot be used for general purposes in the near future.
- Quantum computing based on quantum mechanic principles.
- JPL, In principle, can do things very fast (faster than Moore’s Law will ever allow)
- It can do brute force attack on a key very fast.
- If they get quantum computing working, we would be able to brute force most keys and crack it in a reasonable amount of time.
- We are talking about things like bringing the temperature of the computer down to the vicinity of absolute zero
- Very specialized setup and probably not going to be in your private basement!

Lecture 4

Desirable Characteristics of Ciphers

- We want it well matched to requirements of application
- Amount of secrecy required should match labor to achieve it.
- One time pads don’t usually require that much work/secrecy

- Relatively few places where we are motivated by the cost of setting up all the specialized equipment.
- Freedom from complexity
- The more complex algorithms, there are tradeoffs of being able to crack the cryptography
- Caesar cipher is simple, but easy to crack
- We want simple algorithms: we are going to do computation and we want to be able to perform it.
- They are easier to analyze and figure out what is going on.
- We can determine if there is or is NOT a flaw.
- If there is a flaw, you want it to be found as soon as possible, preferably by someone who tells you he found it.
- If you have a simple algorithm, you can hope that someone will try to analyze the algorithm and see how it behaves.
- Only by the most arcane analysis could you find that flaw, but it may take a long time and you may not know when they find it.

More Characteristics

- Simplicity of implementation
- This is important for hand ciphering
- Also important for computer implementations
- There have been released implementations where there are bugs and when closer attention was paid, it wasn't doing the right thing!
- It was significantly less secure
- If you have a simple to implement algorithm, you probably won't have a problem
- Errors should NOT propagate
- If a bit flipped for some reason, this resulted in some small part of the encrypted data NOT decrypting properly.
- A single error would wipe out all of that data.

Yet More Characteristics

- Cipher text size should be same as plaintext size
- They all have exactly the same size at the start and the end.
- There are some slight differences for padding issues
- Branch of cryptography coming up in the last few years with interesting properties
- Lets you perform operations on the encrypted data
- You don't decrypt it, you just perform an operation to be the encrypted version + 1
- When you decrypt it, just add 1 to it and it will come out right and you don't get to see what the # is.
- The problem is that the cipher text becomes much LARGER than plaintext size.
- Currently, these implementations are NOT practical.
- You have some plaintext, you try to run the algorithm and get cipher text.

- You want the relationship between plaintext and cipher text to be very UNCLEAR.
- You want to get rid of character frequencies and this is why mono alphabetic substitution is NOT so good.
- Small operations will show that there is a relationship between cipher text and plaintext.
- Encryption should maximize *diffusion*
- If you have 1 MB of data and you want to produce 1 MB of encrypted data, the first byte should have an effect on the first bit of every piece of the data.

More Characteristics

- Errors should NOT propagate
- If there is a bit flip in the cipher text, all of the plaintext will get changed (potentially)
- This is what engineering is about!
- You run into problems where you have 2 things you want to achieve
- You cannot get perfectly good values for both characteristics (tradeoffs)
- Do you want reasonable error propagation or do you want to maximize diffusion?

Streams and Block Ciphers

- Stream ciphers take one symbol of plaintext and immediately convert it into one symbol of cipher text.
- One at a time, they convert symbols
- Block ciphers work with a larger chunk of data, then they convert the block.
- The process rinses and repeats

Stream Ciphers

- Lecture 4, Page 7
- Feed in one character at a time like getc in C++/C
- Works on one symbol at a time (typically a bit, but could be a byte)

Advantages of Stream Ciphers

- Could be fast
- The moment you get a piece of data the size of one symbol, you can encrypt it
- You could also decrypt it immediately.
- Could have low error propagation
- Depends exactly how you do things
- Change errors in one of your symbols will depend on that symbol.
- Depends on cryptographic modes.

Disadvantages of Stream Ciphers

- Low diffusion
- Each symbol is separately encrypted.

- You can overcome this by using cryptographic modes!
- More vulnerability to insertions and modifications
- Attacker can send encrypted symbols and the guy who is decrypting them will NOT notice it has been inserted.
- Not a good match for many common uses of cryptography
- Payload is available all at one time, so you can very easily encrypt the payload at the same time.

Sample Stream Cipher: RC4

- Often in a stream cipher, you get a different key for every symbol that we encrypt.
- Key is different every time, so we don't have a single alphabet.
- No ability to exchange one time pads, so we want the sender and receiver to be able to reproduce.
- Having gotten your changing key stream, you can later use XOR
- If we presume the key stream is unpredictable, this is just as good as a one time pad.
- XOR the next byte and continue the process byte by byte.
- To decrypt, if you apply XOR twice, you get the same thing that you started with.

Creating an RC4 Key

- Originally byte 0 contains 0, byte 52 contains 52, etc.
- They key can be any size you want (at least 1 byte)
- Fill a 2nd array of the same size as the key
- If you chose 127 bit key, you will repeat it twice.
- 2nd array can shuffle the bytes in the 1st array
- Keep doing this based on the key that you have gotten.
- Swap two of the array bytes each time.
- Numbers need to be shuffled around a bit and then you do it again and keep swapping.

Characteristics of RC4

- This is 10x faster than DES!
- Very fast algorithm
- Significant cryptographic weaknesses
- Unless you have swapped a whole bunch of times, it is easy for attackers to figure out what is going on.
- You need to use your secret key to swap around bytes a few hundred times.
- It is real easy to implement this incorrectly.
- RC4 used to be the favorite algorithm to encrypt malware.
- Nowadays, it has been deprecated
- People who design standards i.e. TLS standards have said this is still an option but we strongly advise you NOT to use it.
- We probably don't want to use RC4 nowadays!

Block Ciphers

- Take a block of data (some set amount of data of a single fixed size) and we get a block of 12 characters and put them into the encryption box.
- Out comes the encrypted version, which is NOT done the same way there.

Advantages of Block Ciphers

- Good diffusion -> a block of certain bits and several pieces of data to work with at a given time.
- Better immunity to insertion
- Cannot insert something in the middle to an attack
- You can insert an entire block, which is a potential problem.
- Practically all the ciphers we use are called block ciphers
- Data gets encrypted on your disk drive.

Disadvantages of Block Ciphers

- If you know data is coming and you want to do cryptographic operations, you are doing a certain amount of encrypting it.
- This slows down performance
- Can cause bad error propagation

Cryptographic Modes

- We have a 1 MB file and we are going to want to encrypt it with the same cipher and same key.
- AES is going to encrypt in blocks of 64 bits
- We need to divide this up into pieces and perform separate encryption options on the entire block.
- Given that we have 64-bit chunks, how do we encrypt?

The Basic Situation

- A record that says what the account # is and what the account holds
- Let's say we have a block cipher of size 7 and we need to use the same key to encrypt everything
- Start encrypting data and each thing gets encrypted in an ordinary fashion.
- This can cause two things to be encrypted to the same form, and this predictably produces the same result.
- Happens because we used the wrong cryptographic mode.

Another Problem With This Approach

- We are going from one bank to another
- A guy has records of how much money they have in their account
- Sends encrypted versions because he doesn't want fake things to go into place.

- Sends another and eventually they all get delivered and everything is wonderful.
- Unfortunately, there can be a bad guy with an account at the bank.
- He knows there was one particular genuine deposit and this got encrypted by the bank.
- If he listens in, he figures out this is his deposit, so he can take the bits and send another message to the bank.
- At the other bank, it DOES decrypt properly, so he can deposit $550 into his account by stealing it.
- The problem is that everything looks fine!
- One thing you discover is that it is really hard to use cryptography right
- More subtle than people think and it is easy to make mistakes even if you are an expert in cryptography.
- People who develop TLS discovered a serious problem with a major flaw that attackers can exploit.
- Open source software and everyone could have looked at that code.
- Anybody running the Apache web server was using it.
- Relatively blatant issue called **insertion attack**
- Replay attack: replayed perfectly correct data, resulting in a bad thing happening.

What Caused the Problems?

- Each block of data was encrypted using the same key.
- In a totally deterministic fashion, this is NOT a good thing for our ciphers to do this.
- Ciphers are expected to behave deterministically if they work time after time.
- If they wren't like that, we would have problems encrypting.
- We used the wrong electronic mode
- If you don't do anything smart, you are probably using ECB mode

Cryptographic Modes

- A way of applying a particular cipher
- Block or stream
- Different ways to use the same cipher
- Don't change the cipher itself!
- Usually, cryptographic modes are combinations of whatever cipher you are using.
- Used when we know we are encrypting more than one piece of data and we want to use the same cipher, same key.
- Then, we do some simple operations using feedback.

So What Mode Should We Have Used?

- Cipher Block Chaining (CBC)
- Takes a bunch of related, encrypted blocks i.e. different packets and it ties them together in some fashion.

- Hides two plaintext blocks that might be identical.
- Helps prevent insertion attacks
- Usually cryptographic modes are 3-letter acronyms

Cipher Block Chaining Mode

- Encrypting X+1 blocks
- Save the encrypted block for a moment or two.
- Take encrypted version and combine it with the next plaintext block and save the encrypted copy.
- For block X+1, XOR the plaintext with the cipher text of block X
- He needs to pass it through the cipher he is using and it isn't an encrypted form of the plaintext XOR'ed with the encrypted block of X.
- Each block's encryption depends on all previous blocks' contents.
- Good for error propagation purposes.
- Decryption is similar because you get plaintext XOR'd with cipher text of previous block.

What About the First Block?

- You can keep saying don't send it for the first block
- What if you keep sending data that has the same message of rate same first block
- Germany in WW II had certain encrypted headers
- This happens a lot with standardized file formats
- Header info could be exactly the same, and you end up revealing to your attacker that it is a JPEG file
- We have to do something but be aware there is this problem.

Initialization Vectors

- It is not going to be doing anything with the block and you cannot XOR it with the first block
- Use the initialization vector which is **IV**
- Ensure that even if you are encrypting the same text, it will encrypt the same every single time.
- We create a random string and XOR the random string with the plaintext of the first byte of data
- Do encryption of that XOR'ed combination.
- Bootstrap yourself so you have an encrypted block.
- Guy at the destination end will NOT be able to figure out the first bit of data unless he knows the IV
- You don't encrypt the IV! You send him a plaintext version of it.
- This allows him to get the plaintext in a secure fashion.
- CBC is used on data that is going to the same place.
- If we were going to be sending a long stream of data, we want to use cipher block chaining on different participants.
- If we are sending something to different people, they will tend to NOT get mixed together.

- Each time it gets sent, every bit will be different.

Encrypting With An IV

- First block of the message with 8-bit blocks in this case.
- Create an initialization vector which is just a random #
- XOR them together and get a different block.
- Encrypt message and send IV plus message
- Get second block of message
- Use the previous encrypted block and perform XOR between encrypted block 0 and plaintext block 1
- Then, encrypt and send second block of message repeatedly.
- No need send 1st block again, but you will need to send the IV

How To Decrypt With Initialization Vectors?

- We are doing cipher block chaining, and using the key we distributed, it will decrypt to P, which is NOT the actual plaintext we want.
- What we want is P XOR IV to get the original plaintext.
- Unless we get the IV sent to us, we won't be able to get the original plaintext message.
- If we were going to use the key time after time after time, it would be undesirable to repeat the process.
- At the beginning of the transmission, we will send the IV in plaintext.
- This means that anybody can hear the IV and they hear the real IV.
- They hear the genuine IV and potentially change the IV.
- P XOR IV will result in a plaintext.
- The 2nd encrypted block of text needs to use encrypted first block of text to get out the true plaintext.
- NOT exactly what we do in real cryptography but we use a similar mode to this.

An Example of IV Decryption

- Lecture 4, Page 27
- Very first thing is to send a message to the receive with an IP header, encrypted data, and an initialization vector.
- Pull out initialization vector in plaintext (white block)
- Encrypted blocks are represented as darkened blocks
- Decrypt it and out comes first block of encrypted data
- XOR it with the plaintext IV and out comes the original plaintext.

For Subsequent Blocks

- Do a similar thing except we get the encrypted version of the previous block.
- Do a decryption and do our XOR to get the previous plaintext of the block.

Why is it okay to send IV in plaintext?

- What can an attacker do with the plaintext of the IV?
- He could take the IV and XOR it with the first block of encrypted data
- It won't do anything because he will get garbage.
- It is still under the umbrella of the encryption
- If you change bits in encrypted data, the attacker won't be able to predict what effects will happen.
- It won't look anything like the original version of the plaintext.
- Cannot use the IV to decrypt, but he doesn't have the key, so he cannot do anything.
- If he changes the IV, the receiver will get the wrong IV.
- He will get garbage, but we are assuming the attacker can grab the message and change the contents.
- If what the attacker wants to do is screw up the receiver's ability to receive the message, he can flip a bit in the cipher text and this can result in garbage plaintext.
- Gains no advantage by fiddling with the IV
- Before I encrypt, I am XOR'ing 0's with something else
- Every single one of the 10 blocks of 0's will XOR to something different

Some Important Crypto Modes

- Electronic codebook mode (ECB)
- Here is a phrase, "Transfer $100 to my bank account" translates to "The hawk flies past midnight"
- Blocks of 0's always encrypt to the same thing with the same key
- Like book lookup
- Ciphers block chaining mode (CBC)
- Some problems with CBC:
- Difficulty in implementations of TLS
- Does NOT work that well when the attacker can use a chosen plaintext attack
- Caused a serious problem for many web browsers
- Somebody was using a cryptographic mode but NOT precisely right for that particular browser version
- You should always tries to use someone else's implementation because it is hard
- Everyone will likely screw up their cryptography.
- This is a very subtle and difficult problem
- If you can avoid doing it yourself, it would be much better for you in the long-run.

Uses of Cryptography

- A fabulous tool and it is easy to make mistakes
- If we did not have cryptography, we would have a tremendously difficult time to secure our system.
- Hide secrets
- Use it for authentication

- Used for prevention of alteration

Cryptography and Secrecy

- Provide other forms of secrecy
- If you know a secret and you don't want to prove you know the secret, how can someone be confident that you are telling the truth?
- If you can describe your secret in terms of certain NP-complete problems, you can use routes of confidence and prove that you know the secret without giving him any information unless he can solve the NP-hard problem
- Zero-knowledge proofs (NOT useful in this class but interesting)

Cryptography and Authentication

- Who is requesting this operation and we won't do it for others.
- In certain cases, we can know this by requesting things in an encrypted form
- If we can ascertain that I was the only one who could have created it, then we can both create that operation!
- Encrypt it with a key that only my partner and I know
- Either I created the message or he created the message
- There are other cases where this isn't quite so clear cut.

Using Cryptography for Authentication

- If you have three parties and it matters which of the two remaining parties asked, then we no longer can confidently provide proof for authenticity

Cryptography and Non-Alterability

- Get better integrity of data
- Data should only be changed when we have an authorized party changing it.
- At the very least, we want to know if someone has been fiddling with our data.
- If we can encrypt the message, we will be able to tell that someone fiddled around with it.
- If we don't know the difference between good stuff and garbage, we try to compute the checksum.
- **If we get garbage data, our checksum will NOT work!**
- If you don't care about secrecy, you can achieve the same effect at a much cheaper cost by encrypting the checksum.
- If someone flips the bit and change it in any way they want, we will decrypt our checksum to the checksum of our data.
- If they match, everything is fine; otherwise, there is a problem.

Symmetric and Asymmetric Cryptosystems

- Symmetric - both parties use the same key
- Asymmetric - encrypt and decrypt with separate keys

Advantages of Symmetric Key systems

- Encryption and authentication performed in a single operation
- A lot faster than asymmetric key systems
- No centralized authority

Disadvantages of Symmetric Key Systems

- Encryption and authentication are done in the same operation
- Non-repudiation is hard
- I appear to have done something but I claimed I didn't.
- There is a problem based on giving Bob the file.
- "I never asked for that file and I never got the file"
- That is a lie but here is the encrypted message.
- "You created that message because you wanted to screw me over"
- Who could have created the message?
- Bob could have created it or I could have created it.
- There is NO evidence but the cryptography provides no evidence of who did it.
- Allows people to repudiate what they encrypted.
- This effect can be mitigated with the use of servers but it is very complicated

Scaling Problems of Symmetric Cryptography

- Lecture 4, Page 39
- The number of keys you add will exponentially increase as more and more people start to pop out.
- It gets quite complicated and you have to get the keys in some fashion.
- All of it needs to be bootstrapped off something.
- Needs to be hand-delivered out of band.
- Does NOT use the network channel you are trying to protect.
- **Out of band is bad**
- You want out of band communications to be minimized.
- An original courier can distribute an initial key and try to have each person pass down the key information
- The problem is that we don't trust everyone because the Internet is a cruel and untrusting place.
- Hence, symmetric cryptography is NOT scalable.

Sample Symmetric Key Ciphers

- We do most our cryptography using symmetric keys
- The Data Encryption Standard (DES)
- The Advanced Encryption Standard (AES)

The Data Encryption Standard

- Accepted as a good cryptographic cipher for many purposes and saying that it was good.

- Blessed by the NSA and they actually required changes in the IBM implementation
- NSA said to make the following changes and those changes were made.
- Researchers who were investigating DES wanted to look at what the NSA did so they could break the cipher.
- They looked at the NSA's changes, and what they probably did was strengthen the cipher against differential cryptography.
- Apparently NOT unknown to the NSA
- They strengthened it and made it a better cipher
- NSA Doesn't want anybody else cracking the cipher
- The key being used here was 56 bits long
- 2^56 options
- NSA could use brute force on 2^56 if they wanted to.
- NSA knew they were the only one with this capability, so they wanted to strengthen it so they can hold their advantage.
- Uses multiple *rounds*
- Things where you repeat the same operations in slightly different ways.
- Do the same operations the same time but in different ways.
- Cryptographers tell you flaws in DES
- A few keys you shouldn't use because it is relatively easy to crack the cipher.
- DES has 56 bit key with 2^56 options, which is NOTHING nowadays.
- Anybody with the time and persistence can try 2^56 options nowadays.

The Advanced Encryption Standard

- U.S. government needed something with a longer key size than DES to prevent brute force attacks
- NSA creates cryptographic algorithms for the military
- Open source the hell out of this and we will have a competition to tell everyone in the world that we want a new cipher that is going to have a longer key size.
- Tell us what you think we should use.
- Out of those, they chose 5 that looked pretty good and they wanted people to try it.
- Many other people and pounded on those 5 ciphers and examined weaknesses in the ciphers.
- Each person looked at the algorithm and calculated statistics
- The one that was best was created by a bunch of Dutch researchers called Rinjdael
- Has two possible key sizes
- 128 bit keys
- 256 bit keys

Increased Popularity of AES

- Gradually replacing DES
- Many RFCs describe how to use AES in IPSec
- Linux has had AES for quite some time.

- Most commercial VPN's, Windows, and Mac OS use AES.

Is AES Secure?

- Nobody has discovered a breaks on AES.
- What if we had only a few rounds?
- Some tracks work by using fewer rounds.
- Attacks that get keys quicker than brute force
- 2^126 options is NOT practical but still, that is a bit disturbing.
- What looks like an impractical attack yesterday can become practical tomorrow
- tracks on crypto only get better over time, not worse.

Public Key Encryption Systems

- Cracking AES will allow you to do an immense # of extremely valuable things
- If you can crack AES, you can crack the world
- Fortunately, it appears nobody can.

Public Key Encryption Systems

- You can encrypt the plaintext with the decryption key, and this means you have the decrypt with the encryption key.
- If you encrypt with one key, you have to decrypt with the other.

History of Public Key Cryptography

- Diffie and Hellman came up with it in 1976
- Hellman developed a later algorithm
- RSA developed in 1978 that is still used even to this day.

Practical Use of Public Key Cryptography

- Very tight relationship between private key and public key.
- Create the keys in pair, which is computationally expensive.
- Never share your private key to anyone else, but you tell your public key to other people.

Q. Everyone creates their own public/private key pair?
A. Yes, only the guy who knows the private key can decrypt

Authentication With Shared Keys

- Problem with repudiation

Authentication With Public Keys

- Everyone knows my public key so he can decrypt it.
- It decrypted with my private key and everyone knows the public key, I must have created it
- This assumes I have not divulged my private key.
- This allows people to authenticate their messages because it was created with my private key.

- This is vitally important for all security of the Internet.

Q. When the other party uses the private key and sends it over, the other person who gets the message wants to check if it is legit, is there any way to prove this?
A. You take what you have decrypted it and only I could have sent that.

Scaling of Public Key Cryptography

- Lecture 4, Page 50
- Everybody creates one key pair and they can communicate to everyone in the world.

Key Management Issues

- Somehow or another, people have to know everyone's public key.
- Back in the era of the business card, they would have their public key.
- Not very practical at the time
- One of the weaknesses of public key cryptography.

W 2 Dis 4-8-16
Panama Papers

- Crapton of corruption - Dylan
- Some sort of company that is untrusted with illegal companies
- It is not illegal to have a shell company
- Government officials are required to report these holdings
- Google and Apple have shell companies
- Majority of these holdings were held for money laundering and tax evasion
- Prime minister David Cameron: A lot of properties in the U.K. were bought with illicit funds
- U.K. is making bombastic statements but now he is hiding from security

Security and Privacy
Q. Is there any way to protect this? Apple has a competitive advantage?
A. Instead of being an individual, you want to announce what you researched or learned.

- There are also services that you set up on your network that monitors outgoing packets.
- Crypto provides small guarantees that makes it impossible to leak something.
- Any sort of probabilistic approach cannot guarantee that there will be error.
- The volumes of millions or billions packets will start to add up and can be used as an attack form.

- Libraries had a rule where you can read without being logged what you are doing.
- Persistent threat that you are being monitored means people are being more careful and less curious about what they do.

- People are starting to watch what they are saying (affects freedom of speech)
- If you have done internships, you might know about hierarchy and bureaucracy and how this slows down innovation.

Deter Lab
Q. Why would you want to use a symbolic link?
A. An alias or pointer to something. It is useful to make running or accessing files easier.

Q. What effect does it have on the permissions? Does it give you any ability to pass on permissions in the file?
A. In your OS, we should have learned the difference.

Q. Differences between a hard link and soft link?
A. Hard link uses an inode, while soft link is a pointer.

Q. What if you delete a hard link vs soft link?
A. Hard link will delete the original one, while soft link doesn't delete the original.

Criticisms
- Very simplistic models like Bell-La Padula vs ACL
- Unix permissions are pretty simple -> user, group, other
- Fast and simple to understand
- Android phones have gazillion types of permissions but generally, people don't really care or understand them
- Turning off location services on iPhone
- One tries to find your location via cell-tower, another one is for geolocating other users
- There are some secure microkernels that come out like App Armor that provide sandbox modes that control accesses to certain file system reads/writes.

sudo alternatives
- Idea of least privilege -> user is going on through everything in their life, they don't want too many permissions, but there is an option to sudo if you want to have absolute full permissions
- Stemmed as a result of multi-user machines
- Then, you needed some sort of administrator and the systems we use today are from multi-user administrations
- We have to mess around with shudders file and do some sorts of operations that are defined in the spec.

Software Tools
Q. Difference between adduser and useradd?
A. adduser uses the skeleton and one of them is POSIX and the other is not. adduser is more high-level than useradd.

Piazza Questions

Q. Not being prompted for password (my question)

- Use the password command
- Why isn't the password being prompted by default?
- It can be operated as a daemon, so these services can be non-human services. In this case, they will not have a password set.
- What happens when you start Apache
- Privileged ports are those below 1024 (most notably port 80)
- What happens if the service is running as root?
- Any process that is spawned has the permissions of its parent, so you can do whatever you want.
- EngineX user is isolated to whatever this web process needs and they can provide additional guarantees
- This particular user can start as root or init, and this starts the actual service and it begins with this user that is NOT root.
- Spawns a child process that is owned by the system group.
- The process will be owned by the system, not by the root, but it is able to get the root privilege (port 80)

A. There won't be a password prompt by default because it is a system user.

Q. Home Directory Component #12 Question (Frank's Question)

- Why is there execute on directories?
- Executing a directory is just being able to cd into it.
- This needs execute permissions

Firewalls

- You set up a network and no one is trying to attack it.
- Academic and government researchers were originally using the Internet
- Someone decided to write a worm and this affected the entire Internet.
- FTP opens up different connections

Q. Stateless vs Stageful

A. IP_UCLA and Deter decides to lock out this individual

- Blacklisting: Drop all packets from a bad user
- Stateful will look at IP address because we want to prevent DDoS attacks, while stateless does not keep track of the IP address or other info
- For Deter, if you incorrectly log in 3 times, you get blacklisted
- If you want to build a stateless firewall, you have to look at the port and maintain some sort of database.
- Enforce this kind of restriction and you just need

Q. What do we need to do to mitigate DDoS attacks?

A. Take advantage of some sort of vulnerability in the infrastructure.

- Cheap way of doing 1 attack, send one request and this turns out to generate 100 requests (2 orders of magnitude greater)

- How do we know we are under attack?

Q. How can Deter recognize an attack?
A. Look at the time between these events. If there is a large span of the attacks with large frequency, it is probably not an attack.

Tor
- Popular websites are blocked by CloudFare
- They use IP and make claims that are deemed malicious, so they blacklist everyone
- Majority of Tor traffic is insignificant to what you see on the Internet.
- Keep digging down to what kinds of actions are performed.
- Traffic shaping: read-only, so people are not leaving comments (writes) or you can look at how the traffic originated.
- In order for a DDoS attack to be successful, you have to have some kind of advantage.

DPI (deep packet inspection): Identify if it is email, HTTP, Snapchat, Facebook, etc.
- Determined by packet sizes, etc.

iptables
- If you have to use this in the real world, just be careful and keep in mind if the rules are in a saved state.
- You can put them in an iptables that can be run at start time though
- If you do mess up, you cannot get back in.

Q. Difference between reject and drop for iptables?
A. Both prohibit a packet from passing, but reject sends an ICMP destination-unreachable back (leaks some knowledge), while sends no response. For an actual attacker, DROP packets, but if it seems authentic and friendly, REJECT the packet.

Q. Difference between telnet and ssh?
A. telnet is plaintext while ssh is encrypted. You generally want to use ssh if you want to make sure everything is confidential.
- telnet is a quick and dirty way to diagnose and debug.
- Just another toolkit you can consider

- Two factor authentication is a feature you want, and these examples are likely to be very old.
- telnet is generally used for setting up some infrastructure.
- The best way to learn this is to use examples and it is better to do it yourself.

Q. Difference between TCP vs UDP?
A. TCP has connections and is more reliable, UDP is stateless.

Read over the spec with a fine-tuned comb and some things may not be so intuitive so you just have to ignore those types of things.

- Go through this lab literally and if something is not available, ask away on Piazza
- Be sure to use the scripts so be careful when checking permissions
- Read through the spec once or twice before starting!

Lecture Review

- Encryption is one way to achieve confidentiality
- Only authorized parties can read.
- One time pad is one-way traffic
- Hashes -> cryptographic hashes should be irreversible
- Easy to compute but hard to reverse.
- Hashes are useful for checking integrity and passwords.
- You don't want to store the passwords themselves because if it gets breached, that is not good.

Q. Why not encrypt the password rather than hash?
A. If you encrypt and have the same length, the attacker can know the length of the password which makes it easier to crack.

- Hashes are typically very fast, while crypto is typically very slow.
- There is really no reason to do the decryption and do these things.

- If you are relying on a secret, someone could eventually figure out the algorithm. If people can see the algorithm and still NOT get in, that is the most secure version.
- No unauthorized person should be able to read the data.
- You have one scheme and each end up being broken.

Symmetric and Asymmetric Encryption Systems

- Symmetric uses one key
- Asymmetric uses a public and a private key
- You can use public keys to encrypt and your own private key to decrypt.
- It is a distribution issue where if there are 1 million users on the network, you don't have to store 1 million keys

Q. What does HTTPS use?
A. Asymmetric key to get a symmetric key.

- Symmetric keys are faster while asymmetric keys guarantee authenticity.

Q. If you use symmetric keys for authentication, what is the problem with that?
A. Leads to non-repudiation.

Q. What does non-repudiation mean?
A. You cannot prove that the party created a message. You cannot deny that they created a message either.

Q. Is there a way to do authentication using symmetric keys?
A. Very difficult, you need specialized hardware. Create an unique identifier that is appended to messages. You need some authentication server and a 3rd party mediator that has additional enhancements.

- Individuals can be distinctly identified here.

Q. How do you do authentication through public key crypto?
A. You send a challenge message encrypted, and the receiver decrypts the message and sends back a response.

- If Alice wants to send a message, sign using a private key that only Alice has, and anyone can verify it because they have a public key.
- Think about it like comparing checksums, whoever is the attacker cannot modify the checksum to be what they want because it is encrypted.
- Compare checksums and they will NOT match.

Q. What does a one-time pad do?
A. Each bit is encoded with a new key so you are generating an almost impossible thing to crack since there are no patterns.

- Come up with some clever scheme to permute a message.

Q. What are two ways to hide a message when you decrypt it?
A. Confusion and diffusion

- Confusion: If you look at the cipher text, you should be confused what it is saying
- Typically, there is some leakage of information that can exploit character accounts
- Q. If we try to do the Caesar cipher scheme, what issues are going to be there?
- A. Frequency analysis can be used to identify patterns

Q. What are some good properties we want to achieve to avoid frequency of these characters?
A. Change the way you add 1 or subtract 2 values. Randomize the pattern of switching values.

- You want to think of theoretical patterns first. Generate a random key the size of a message that is unbreakable.

Q. Why is the one-time pad unbreakable?
A. Multiple configurations that leads to this same conclusion.

Q. Probabilistically, what does this mean?
A. Every plaintext is equally likely to be obtained from the cipher if you have a one-time pad because of the XOR operation

Stream and Block Ciphers

- Stream is done one bit at a time, while a block is a chunk of an arbitrary length.

Advantages of Stream Ciphers

- Errors start to add up

Q. Where does the error happen?
A. In the encryption method or in the wireless channel.

- In this case, we are using the encryption for our communication.
- You want to do checksums for these messages and we want to do error propagation in this case.

Block Ciphers

- Q. Does a stream cipher use an IV?
- A. NO!
- The issue is that it has problems with eavesdroppers. The same key is used all the time, so you can alway recover information using English frequencies.
- You will see some structure, so it isn't that good of a stream

Take a block and XOR it and feed this encryption to the next message

- There is a lot of structure and you cannot use it to break into how it is encrypted.
- This method lets you use the same key over the entire encryption method, so you never use original data.
- Q. Does this provide good diffusion?
- A. YES! It does.

Stream ciphers have low diffusion because it processes one character or bit at a time.

- It cannot randomize the final, entire encrypted message at a time.

Block ciphers are good for protecting against insertion

- Problem with block ciphers is that it is slower, but security is a necessity so grin and bear with it.
- Problem of rerandomizing things will affect entire blocks

RSA + AES

- RSA done with for the initial three-way handshake, then AES is done for the rest.

W 3 T Lec 4-12-16
Key Management Issues

- In asymmetric cryptography, we have to separate keys
- Encryption key
- Decryption key

- The way we use public key cryptography is that one key is kept secret, and the other key is kept public to the rest of the world.
- Public key is known by everyone
- Private key is known by only the party who created the key pair
- He can use the private key to encrypt the message, you can decrypt it with the public key, so you don't get any secrecy benefit.
- You get an authenticity benefit, and only he knows the private key.
- You can verify this using the public key to verify that he knows the message.
- This is working though on the assumption that you know whose public key it is.
- How do we make sure that people know whose pubic key this is?
- If we say this is Microsoft's public key, but it actually matches a hacker, then you won't achieve the effect you wanted to get.
- The original solution was simply publish your public key.
- There are problems with this though!

Issues of Key Publication

- You need a high degree of assurance that a person is really who they say they are.
- You could send an email message and ask for confirmation
- You can share it on USB's
- Problem is that these solutions do NOT scale!
- We need a way to get keys out to someone who needs them.
- This is very much the weak link of every cryptographic system in the world.
- There is a compromise in this algorithm or that algorithm and it may give into a brute force attack.
- Hackers try to obtain the key in a way that they shouldn't be using it.

RSA Algorithm

- If you are using cryptography on the Internet, you are using public key cryptography
- You are also probably using the RSA algorithm.
- Very widely used and built into all of our web browsers!
- Old enough algorithm that it was originally patented
- The patent has expired so now it is in the public domain
- Withstood a lot of cryptographic analysis
- The reason you run into issues saying that a key is no longer secure is because of the hard problem of factoring large numbers
- Based on hard mathematical problems, use this info to create a public/ private key pair
- RSA is based on the hard mathematical problem of factoring extremely large #'s
- Known to be computationally intensive.

- You can apply those algorithms to any large numbers you want but this is computationally intensive
- This is where RSA gets its security

RSA Keys

- There has to be a relationship!
- When you create RSA keys, you create a pair of keys at the same time.
- You run a special algorithm that takes a couple of 100-200 digit prime numbers.
- You do some function on a couple of them and this gives you the public and private key.
- The relationship between the public and private key is a complex relationship
- They have to be related since you have to encrypt with one and decrypt with the other.
- If there is a simple relationship, you can deduce the relationship, but RSA isn't a simple relationship.
- Figuring out what the plaintext is is supposedly equivalent to factoring the product of two 100-200 digit prime numbers
- You only have to protect something according to its value (less important things don't need to be as protected)

Comparison of AES and RSA

- Both are in wide use everyday.
- AES is a much more complex algorithm
- Shifts things and does XOR's, table lookups
- Shuffles things around
- You get pages of code that implements AES
- RSA is really quite mathematically simple
- If you know a little bit of exponentiation, you can understand RSA
- Every single operation is simple from the point of view of the computer
- We can do shifts, XOR, and addition that are simple to do in hardware very quickly.
- Multiplication and division take a lot of time to do in hardware
- Exponentiation is really hard because it is a series of multiplications in hardware
- Computationally, RSA is 1000 times more expensive in hardware, and 100 times in software
- RSA key selection is complex and it will take quite a while
- If you want an RSA key pair, you will have code that does it for you.
- Takes minutes to do even if you don't have anything to do on your computer.

Is RSA Secure?

- No one has been able to prove RSA's security

- Some variants of what RSA does and you can prove that invariant problem.
- The security of RSA depends on complexity of factoring very large #'s
- What is the key size that I am using?
- How big are my keys?
- For AES, the key size determines security because that determines the difficulty of a brute force attack
- Nobody does brute force attacks on RSA because it is stupid
- If you have the public key, you can do the factoring to derive the private key
- This is a lot quicker than brute force, so it means the key size does NOT have the same meaning

Attacks on Factoring RSA Keys

- In 2005, a 663 bit RSA key was cracked
- A 768 bit key has been cracked in 2009
- RSA company used to give prizes for cracking their algorithms, but they stopped because it was very obvious that any key size would eventually be cracked
- This operation can be parallelized fairly easily.
- Things up to 2048 bits are NOT fully secure.
- People who really care nowadays use 4096 bits!
- 256 bits in the absence of quantum computers is sufficient as long as there is no other attack besides brute force.
- Why don't we just go with a 16K key?
- The cost of encrypting and decrypting depends on key size.
- If you go to a 16K key, you will get more security but it takes forever to do the encryption and decryption
- Tradeoff: How much computation time are you willing to spend vs the amount of security you get?

Elliptical Cryptography

- Pretty much all of the public key algorithms depend on mathematical problems
- They aren't dependent on the same mathematical problem each time though
- There are equations that describe the ellipsis and we can use those properties to get a public key algorithm
- Get a # related to the encryption chosen and generally speaking, you can get by with much smaller keys and it is similar to the key size issue for RSA
- The problem is harder based on the fact that this is the elliptic curve problem vs the factoring problem
- A lot of mathematics behind it so it was widely studied
- Looked for efficient ways to solve problems related to elliptic curves
- NSA is pushing it and wants people to adopt it!
- Often, people do NOT feel safe when the NSA is preaching a certain amount of things.

Combined Use of Symmetric and Asymmetric Cryptography

- Symmetric cryptography is NOT good at authentication
- Instead, we combine the two types in particular ways.
- When you are doing browsing on the Internet, you are using asymmetric and symmetric cryptography
- Asymmetric cryptography “bootstraps” symmetric crypto
- The problem is getting symmetric cryptography to both sides of the Internet
- The only way to do this is through the Internet
- Diffe-Hellman algorithm!
- The first party shouts out a random #, the second party shouts out a different #
- Everyone else who hears it happen cannot discern the public key generated.
- Party A and Party B can use some kind of medium and only they will know it!
- They need to communicate across the Internet so why don’t we use Diffe-Hellman?
- Guarantees that you are sharing a symmetric key with one and only one party.
- If you are shouting out, you can recognize the voice, but on the Internet, you cannot recognize voices.
- This can be a serious problem, but it can be solved via **public key cryptography**
- I am sharing a symmetric key and I know who I am sharing it with (Microsoft)
- Now you have a symmetric key shared between Microsoft and me, so we don’t need to use expensive public key cryptography
- First, you exchange a key based on RSA, and after exchanging that key, you use AES from that point onwards.

Combining Symmetric and Asymmetric Crypto

- Most papers on cryptography talk about Alice and Bob
- Lecture 4, Page 60
- Alice has her own public and private key pair
- Bob has his own public and private key pair
- Alice knows Bob’s public key, and Bob knows Alice’s private key
- Alice wants to share the key only with Bob, and Bob wants to be sure it is Alice’s key
- C = E(K_S, K_EB)
- Nobody else will be able to get the K_S key
- We will do one more step and use Alice’s private key to send it to Bob
- If everything is going well, she can encrypt it twice and you can undo the cryptographic algorithms correctly.
- Bob can decrypt the message and pull out C

- He will decrypt a 2nd time and use his own private key, so C can be encrypted with his public key.
- Out comes K_S, so this looks fine.
- Unfortunately, there are some serious problems here.

Q. What happens if it comes from an intruder and the message is NOT encrypted from Alice?
A. Bob treats it as encrypted and the attacker didn't know the private key. It is all random bits, so that would suggest that you cannot tell if it belongs to Alice or an intruder.

- It won't be K_S because it wasn't encrypted with his original public key.
- He is going to get K'_S, everything encrypted with the key will NOT decrypt properly.
- They will NOT be able to effectively communicate, and thus Bob can detect something went wrong.
- We don't want to wait that long to figure out an attacker was posing, so we include some information in the message we encrypt, so we know something went wrong there.
- Assuming everything is fine, we should get something there saying "Hey, I am Alice!"

Digital Signature Algorithms

- How to we know the electronic document is what it should be?
- We want to ensure integrity and we don't want attackers fiddling around with our data.
- If we care about authenticity and integrity, we need to do something that signs the data, and this can be an analog to signatures on checks and contracts.
- Analogy with the digital signature, which we want it to be a bunch of bits.
- Copy bits time after time, and in the physical world, if I can copy your signature perfectly, I can copy it anywhere!
- One of the challenges is to make sure it is NOT forgeable and you don't want to reuse it on another document.

Desirable Properties of Digital Signatures

- It has to be verifiable as well and people have to in some reasonable way to check that.
- Peter Reiher signed this document
- Repudiation is a problem with symmetric cryptography before!
- We don't want this for digital signatures and we want it to be relatively cheap to compute and verify
- We want it non-reusable and we want to make sure each signature can be used for the purpose intended.
- Very different from real world signatures
- Not everyone can produce the same handwritten signatures, so we want to do something the same each time.

- We want every signature to be unique, and a signature from one document cannot be created for another document.
- Go to trusted authority and get his help
- This is BAD!
- Who is trustworthy enough to be trusted by everybody?
- If we can talk amongst ourselves but not anyone else, we are NOT the trusted authority
- If I have put my name on a contract selling you this building for $1 million, we don't want anyone to change the document and have the signature look the same.
- You should be able to tell if you have been fiddling with the cryptography!

Encryption and Digital Signatures

- If only I know it, only I can decrypt it and only I could have created it.
- No one else knows the key so no one else can check it.

Signatures With Shared Key Encryption

- Never going to screw anybody but also be perfectly honest and perfectly careful.
- If I want to sign things, I want to create a shared symmetric key for the trusted 3rd party and myself.
- Create a shared key and I encrypt my document with that shared key and give it to the guy who is supposed to share the document with me.
- Trusted 3rd parties should know the key and have him decrypt it.
- This is the signature on the document that you think it is.

For Example,

- Lecture 4, Page 65
- Alice encrypts the document and sends both copies to Bob
- Bob gets the unencrypted one, but is this really indeed the document that Alice created
- It claims to have a signature and the trusted 3rd party can compare it to the plaintext.
- I know that because I trust him and he says so.
- Has some major **disadvantages** that need to be avoided

Signatures With Public Key Cryptography

- Everyone knows my public key, so we have no trusted 3rd-party required.
- Bob knows Alice's public key and determine it was created by Alice.
- We will run into research papers where everything looks great and this depends on proper key distribution

For Example,

- Lecture 4, Page 67
- Alice sends both copies to Bob and we can compare them and see the result.

Problems With Simple Encryption Approach

- Public key cryptography is very computationally expensive
- Can be a very expensive process to check the signature and send it.
- If I were to send both, I would double my cost of sending things.
- He can get the useful version by decrypting it but if he doesn't care, he has to pay the expensive decryption costs.

Cryptographic Keys

Outline

- Genuine practical problem

Introduction

- You can have a very strong encryption algorithm or a wonderful security protocol for how you use it.
- Once your opponents get hold of your keys, you are done.
- NO security at all!
- You really have to be careful about your keys,
- Whenever you are using cryptography, you have to make sure your keys are secure.
- If the attacker gets your keys, then the security of your system depend on how hard it is for him to get the key.

Key Length

- The strength of the algorithm depends on the length of the key
- The longer the key, the stronger your cipher tends to be
- All things being equal
- Ciphers that are crap will NOT help

Are There Real Costs for Key Length?

- Sometimes you are doing encryption in hardware
- Certain # of gates that will reuse the hardware
- Speed you get depends on parallelism and things depend on cycling from gate to gate to gate.
- More expensive to build that chip and takes up more space in the die.
- Some algorithms have simple key lengths and DES works for 56-bit keys
- You cannot have 64-bit DES keys or 128-bit DES keys
- AES works for 128 bits and 256 bits, but it does NOT work for 512 bits or 200 bits
- Those are your choices if you go with AES
- Sometimes attacks are NOT brute force, and key length will make no difference

Key Randomness

- Once you know one of your keys, you can deduce all of your subsequent keys

- A brute force attack on cryptographic algorithms assumes that you chose your keys at random.
- If you didn't, if the attacker can figure out your method, he doesn't have to do a brute force attack.
- If he can figure out how good your random # generator is, he won't be doing brute force on you, but rather he will attack your random # generator

Generating Random Keys

- Don't use rand()!
- Shitty idea
- Have a method that approaches true randomness
- Rather tricky and relies on exotic hardware
- There are physical processes believed to be truly random
- How many cosmic rays believed to fall at a given time.
- Count the cosmic rays and that will be your random #
- Most of us do NOT have hardware that readily counts cosmic rays
- Difference between genuine randomness and statistical properties of randomness
- You don't need actual randomness, you just need something with the same statistical properties
- You also want it non-reproducible
- You don't want to repeat it and get the same random #
- Avoid reproducibility!

Cryptographic Methods

- Start with a cipher and encrypt it
- Take any old data and out will pop a bunch of data that looks random
- Start with a random # and perform a cryptographic hash on it
- Produces a shorter # (128 bits, 256 bits, etc) that looks random
- If it is a good cryptographic hash, then the new # looks pretty random and people can gather one random # from the cosmic ray hardware.
- If we have the random #, plug it into the random hash and if we want a random hash, we take output from the result and keep doing it.
- Each time it goes into the hash, it will have a different #
- Take old key and hash it with the algorithm
- Then you get a new key each time!
- Obviously, you need a good hash algorithm
- If you have your hash algorithm or key broken, then you lose this system completely!
- Once you have his old keys, you can look at the next million keys (nothing for modern computing hardware)!
- Certainly, you would be able to determine the keys in the future!
- Avoids **perfect forward secrecy**
- If you have the result of an earlier cryptographic algorithm, if I have that, it doesn't help whatsoever!

- Revealing what happened in the past does NOT affect what you cracked in the future.

Perfect Forward Secrecy

- Normally the case where you cannot regard a key as being secret forever.
- There is a pretty good chance that a key will get compromised
- The entire security of your system would then disappear.

Random Noise

- We want to create our keys so how would we do that?
- We think there's a random process so we want to count that.
- Prefer to have something with the hardware in question to deal with these random processes
- Nowadays, we have flash memory but back then, we had hard disk drives
- It was not engineered to a perfect tolerance.
- Variation based on the manufacturer of the disk drive.
- We could determine that it wasn't really random but from an external point of view, it would look pretty random.
- Look at variations in disk drive delay
- Poke inside that box and get an answer.
- I know you are giving me things on millisecond granularity, I want microsecond granularity!
- Keystroke delay
- Each person types at a different rate, so consider the time between he typed the first character and the second character
- Use keys on computers so it is NOT a bad idea and we can look at what the computer is gathering.
- High-Quality, random # generation processes use this in real systems.
- Assuming you are correct about the randomness, then you have perfect forward secrecy.
- Random data does NOT help you generate the next key.

On Users and Randomness

- Users can provide entropy
- They will be relatively easy for users to crack
- If you ask the users to do something, it will probably NOT be random!
- What have I gotten that is random and why is it random?
- It is probably better to find a package and verify your results rather than doing it yourself.

Don't Go Crazy on Randomness

- We want to make sure it is NOT reproducible, which is really bad!
- We want to make sure there is NO obvious patterns
- Odd-even, odd-even for example
- If there are unknown patterns, it will be hard for attackers to figure out though

- Unless the attacker is extremely motivated, he probably won't take the effort to crack it
- Few attacks are made on key randomness
- The exception is when people have done key randomness very badly.
- Once you figure out an attack, it is easy to exploit!
- There might be only 32 or 64 bit keys you can use.
- If you have a process that isn't random, you are probably NOT going to practically make use of that fact.

Key Lifetime

- Very hard to get a good key
- Why don't we use it forever?
- That leads to a question of how long we should use a given key?

Why Change Keys?

- The longer you use a key, the more likely that someone will find out the key is used.
- Attacker will get it somehow if you use it often
- Much harder if you only use the key for a minute than if you use the key for a year.
- 300 GB of encrypted data; if only I knew the key!
- He becomes highly motivated to get the key.
- If you change the key every minute, he will get the same data encrypted but with a whole lot of different keys.
- The value to him of doing something extreme is a whole lot less, so he is less likely to do it.
- The longer a key lives, the more resources opponents will devote to figuring out the key.
- The more you use a key (the more plaintext you encrypt), the easier the cryptanalysis is!
- This point may or may not be valid for AES
- As a general principle, the more you give the opponent to work with, the easier his cryptanalysis
- Don't give him a lot to work with!
- **A secret that cannot be readily changed should be treated as a vulnerability**
- Stick with that key forever and this is what happens to key secrecy
- If you can change the secret at a moment's notice, you can always change the secret.
- It is a bad idea to embed a secret key into a piece of software that you are distributing.
- It doesn't matter how it is scattered. Sooner or later, you will be figured out and you cannot change it in the hardware.
- Server knows for a fact that it has gotten its information.
- You cannot change it and the attacker perhaps can.

Q. Why can’t you change the key in the software?
A. What I was not saying is that it's hard to use software to change a key. As usual, it's easier to do something in software than hardware, as a rule.

- The situation I was talking about was where one had distributed a piece of software with an embedded key. Speaking generally, it's hard to ensure that all copies of a piece of software one has distributed are updated in any way, so changing the key consistently in a piece of software would be similarly difficult. If you are changing a key you've embedded in software, it's probably because the old key was compromised, so you can't afford to keep backward compatibility by accepting both the old and new key, but if you don't accept the old key, probably all the un-updated copies of the software no longer work (since they don't have the right key).
- Usually you want to write software that obtains the current key (or participates in generating a new key), rather than software that actually contains the key.

Practicalities of Key Lifetimes

- Shouldn’t we change our keys all the time?
- This unfortunately is impractical.
- Store it in an encrypted time and read secret data out of the file.
- Encrypt it once with a particular key
- It could be years on the disk where it uses the key and perhaps, the encrypted data will live for a long time.
- A common practice is to say a new key for every session
- It generates a new symmetric key that you have not used before!
- If you use Skype and you have your Skype sessions encrypted, Skype will generate a new encrypted key every time.
- If done properly, every time you generate a session, you will have a mechanism in place and you know when the session ends.
- Throw away the key at the end of a session.
- We have a problem of key distribution:
- How do we bootstrap this whole process?
- Use the key to distribute other keys.
- We are using the public keys that are already distributed and we cannot change the keys there.

Destroying Old Keys

- It is a good idea to get rid of keys when you are done with them.
- Not beneficial to save keys that you will never use agin.
- If you were writing a piece of software and you wanted to generate a key for each session, it is bad to keep a log of all the sessions
- Throw away keys once you are done using these keys
- It will make it more likely that an attacker will deduce one of your old keys and deduce your own data.
- Destroy a key securely and this starts to coincide with Operating Systems
- Computers aren’t that good at forgetting!

- If you have a process running, somewhere in the memory, you have to store the key.
- At some point, the page of memory got swapped to disk and it is sitting out on disk.
- What happens when it swaps back?
- It did NOT zero everything out.
- This is even more true if you put it into a temporary file.
- They are just sitting somewhere in the free block list without the key actually being wiped out.
- Keys have NOT been figured out where the data is still living.
- There are all kinds of places where old data sometimes lives longer than you think it does.
- There have been very modern attack methods and what such message is the “cold boot” attack.
- When we turn off computers, we assume that RAM totally disappears, but this is NOT totally true.
- From an electromagnetic point of view, values in RAM gradually deteriorate.
- The colder it is, the slower they deteriorate.
- At the temperature of liquid Freon, it will last for a few minutes.
- If an attacker can spray liquid Freon, he can deduce what is going on in memory.
- This is NOT a common attack but it is possible!
- This is a real physical access attack with and you have to move pretty quickly and the data gradually deteriorates.
- This is just an example of the data living in places where you did NOT think it was still living.
- Chances are that the cloud operator cannot guarantee that your key was actually deleted.

Key Storage

- Without the key, you cannot read the encrypted data!
- If you lose the key, you now have a disk full of random data.
- As useful as a blank disk but with unreadable crap that takes up space
- You probably have the key in your memory and worst comes to worst, you lose it.
- Tear down the session and create a new key.
- This can be a very serious issue!
- You have to store keys under some circumstance.

What Is Key Storage?

- Save the key in a safe place
- Prevent encrypted data from becoming unreadable.
- Ransomware: Breaks into your computer, encrypts the data, and saves the key on the attacker’s machine
- All of your important data is encrypted and you don’t know the key.

- Give me $10,000 and I'll tell you the key.
- If you have intentionally encrypted your data, you damn well better get the key from somewhere.

Where Should You Store Keys?

- Somewhere attackers cannot access it.
- If the machine is hacked, you probably are afraid your machine will be hacked.
- If he hacks your machine, you have NOT done much good.
- Presumably, you want to get to those encrypted files yourself.
- Put it on a removable device i.e. a flash drive

How Do You Get Keys There?

- Each time you generate a new key to be saved, you want to transport it to the key server
- He just listened to your network so you need a high-degree of associativity
- There must be a lot of channels so the attacker doesn't know where to look

Key Secrecy

- NOT everyone in the real world does this!
- Especially the case with public key cryptography

Some Problems With Key Sharing

- You want the same public/private key pair on all 5 machines
- Multiple users sharing the same key
- Needs to be in a convenient place
- Not a problem unless someone breaks into your office.
- If it gets backed up, anyone who gets hold of the tape needs to get hold of your private key.
- Where is my private key?
- Is it accessible by only one person?

Why Do People Do This?

- Certificates cost actual money
- They have a private key associated with them.
- If you want to save money, you can buy one instead of ten, but you would have to share the public key ten times.
- Sometimes, people do NOT know where the private key is.
- Example:
- RuggedCom's Rugged Operating System had its private key embedded in executable
- They probably could have generated an update to the OS that was recognized as legit and anyone with access could have installed their own version on the power plant.

To Make It Clear,

- **PRIVATE KEYS ARE PRIVATE!**
- Never share your private key ever!
- Doc what is necessary to switch your private key if compromised
- Widely distributed executables are insecure
- Don't think you are clever and you can hide things in an executable
- There are people who can easily read machine code hexadecimal or binary
- The entire security of PK systems depends on maintaining the secrecy of your private key.
- If you don't maintain the secrecy, it is useless and expensive.
- You must NOT allow private keys to ever be divulged.

Q. The computer that screwed up, why did they put it in the executable in the first place?
A. The guy on this side of the power plant had to authenticate the message and then they wanted it convenient to sign with the private key.

- People using this stuff really just don't understand computer security.
- If people don't understand something, they cannot use it properly.

Key Management

- You have to handle key management properly.

Desirable Properties in a Key Management System

- Secure
- Fast
- Low overhead
- Scalable
- Adaptable
- You don't want to be limited to one encryption algorithm i.e. AES
- You want it to handle any application as well as encryption algorithms

Users and Keys

- Some are generated by applications he is running.
- Some keys for certain operations need to be maintained for a longer time.
- What happens if a machine is compromised?
- People cannot expect to remember these keys
- The average person cannot remember random keys
- Let's hash keys from passwords/passphrases?
- Users typically choose the same password or passphrase time after time.
- Even the slightest difference will result in a different key.
- Smart cards have their own capabilities
- Plug the key into the smart card and ordinarily it is NOT in the machine.
- Key servers
- Store and distribute keys

Key Servers

- Highly trusted machine
- Should know keys for many people

Security of Key Servers

- Good attackers will go after your key server
- If you run it, then the attacker will try to go for the key server
- You have to protect it really well!
- If you have chosen to run a key server in your own environment, devote a machine to its work!
- Absolutely nothing else should run on the machine
- Probably going to be administered by having sysadmin do things locally.
- Watch it carefully
- Keep an eye for attackers and looks or evidence that they are succeeding.
- Keys that you need for long term use but chances are good you will be done with a key.
- You don't want the disk drive to contain a bunch of deleted blocks
- Revocation: A particular party could do something in the past but I changed my mind.
- Difficult problem that we want done properly.

Single Machine Key Servers

- Handle passwords but they also handle keys.
- Keychains or password vaults
- Stores user's keys or passwords for various web sites.
- How do you protect them?
- Attackers will look for files to try to scrape keys or passwords
- To prevent this, try to keep your key protected.
- Users can create a passphrase and obviously, we will have an encrypted version of this stuff somewhere.
- Make sure you choose a good password because they can get all of your passwords.
- Ensure that if you walk away from your computer after you log in, it wouldn't be able to see what is in your password field which hopefully the attacker doesn't know.

Security Issues for Single Machine Key Servers

- You need to make sure whatever you use stores its info in encrypted form.
- Once you have gotten in, you will be able to get your passwords and you can get info from those that wish to communicate with you.
- If your machine is compromised and they find your password, attackers can be you anywhere on the Internet.

Certificates

- Another technology that are really vital for true Internet security
- Authentication is the property of being able to determine a particular party is who they claim to be.
- Certificates are generally used with public key cryptography
- Any certificate you work with is to bootstrap public key cryptography
- Essentially a bundle of bits that can be infinitely copied
- Bits can be changed in the bundle.
- They can create a second copy that isn't included in the bundle.
- Supposed to be able to prove who you are you claim to be.
- I provide the security to someone else as proof.
- Certificates can be used to get a symmetric key from Point A to Point B
- In practice, this is the bulk use of certificates

Public Key Certificates

- Every user generates a public, private key pair
- When a public key gets published, how do we know for sure that the person who published it is really who he claims to be.
- Certificates are intended to solve this problem!
- There will be an actual copy of your public key and it is digitally signed!
- You cannot have self-signed certificates!
- Instead you have trusted authorities who digitally sign your public key.
- Trusted authority has signed certificate that says your key belongs to Peter Reiher!
- Verisign is a company we all hopefully all trust
- If the public key belongs to me, anyone can be Peter Reiher since they can use his private key.

Implementation of Public Key Certificates

- Takes exquisite care that the person is usually who he claims to be.
- Once the authority is satisfied, they take the information and perform a digital signature of that information.
- The private key belonging to the authority can be given by a bundle of bits
- If we have a key server online, we go to it and the key server no longer acts as the certificate.

Checking a certificate

- If you know Verisign's public key, you can check the signature and say "Sure enough, this certificate contains a public key that belongs to Peter Reiher"
- Now you know my public key, you have reason to think it doesn't belong to someone else
- Even without network connectivity, you can check the authenticity of that certificate since you have all the resources to get.

W 3 R Lec 4-14-16

Public Key Certificates

- Major companies on the Internet: How do I ensure I am communicating with the right person
- What we need to do in order to have proper use of public key cryptography is that we have to tie an identity with a particular public key
- Presumably, Amazon.com's public key means it should know the private key and hence, it can create encrypted entities
- The common type of certificates are **public key certificates**
- Bundle of bits that tie together an identity with a particular public key
- Relationship between public key and entity is true
- Somebody who you trust is going to sign that relationship and this public key belongs to Amazon.com
- If we can do this properly, we need to obtain a certificate and contain the public key, which is signed by someone you trust

Implementation of Public Key Certificates

- Everybody who would like to have a public key will go to one or more of these authorities
- This request is for a company with some money.
- You are Amazon.com and this is your public key
- Give you a digital signature proving that you are Amazon and this is your public key.
- Bundle of bits
- All bundled together!

Checking a Certificate

- I would like to communicate with Amazon but I am not sure you are who you claim to be.
- Get your certificate, only useful if you have the authority's public key
- If they went to VeriSign to get their certificate, you could use that public key and get the public key of Amazon with a high degree of security and assurance.
- You have VeriSign's public key and now you know that VeriSign had the opinion that this particular public key belonged to Amazon and you have reason to believe it.
- You have already got the public key and you don't need to do anything else.

Scaling Issues of Certificates

- Who do you trust?
- Not everyone trusts the same people.
- You want different authorities signing certificates
- If you have multiple authorities, when I go to a website, I will get a certificate signed by one of those authorities
- I need the public key of whoever did the signing

Certification Hierarchies

- Start with someone at the top that we all trust.
- When we talk about Internet communications, we translate names to addresses starting at the very top
- Let's say we do have an authority who we all trust
- Create multiple signing certificates, so we can do a recursive operation.
- Instead of containing just the public key and the identity of who signed it, why don't we have the ability to have a recursive certificate with multiple public keys and signatures.
- You may or may not know who has the public key and signed it.
- We can say "Hi, I am the universal authority and we can have multiple signatures on the certificate"
- Universally trusted authority says this is the public key and the second one verifies this is the key.

Using Certificates From Hierarchies

- Get a new certificate
- Look at it and say "Here is the signature for Amazon, but I don't know the public key of the guy who signed"
- With a certification hierarchy, we will contain a 2nd signature for a higher-level authority
- If you know the guy at the top, you can recurse down through the signatures

Extracting the Authentication

- Use public key at higher levels to get to the next level down
- Typically, this is only done once because you are not going to throw away the public key in the future
- The first type you go to Amazon, you go through a lot of shit, but it gets better in future uses.

An Example

- Lecture 5, Page 44
- Web browser with a certificate in it.
- Determine the public key of someone she wants to communicate with.
- She actually wants to have secure communication with green (learn the public key of green)
- Should Alice believe he's really the green site?
- How does the green site authenticate himself?
- The green guy goes to the yellow guy and says please give me such a thing
- The yellow guy is a lower level certificate than green, so he has to have something already to prove that his public key is the actual yellow guy's public key
- These certificates say this is the yellow guy and that signature could be the purple guy, which is the higher-level certification
- Even further on, somewhere, there is this higher level certification purple that the yellow guy used.

- Yellow guy gave the certificate to the green guy, in addition to the signatures, this certificate authority also proved his own identity.
- Green guy's public key with a signature proving it.
- Why do we believe that?
- The purple guy is willing to state with high confidence that this is who he claims to be.
- Whenever the green guy wants to, he can send this info to anyone who wants to learn his public key.
- Alice has never heard of yellow, but she has heard of purple and has its public key
- Alice doesn't have the public key of green and yellow except what is in the certificate, but she can bootstrap down with purple to assure that yellow's identity and green's identity are right.
- Purple verifies yellow
- Yellow verifies green!

Certification Hierarchies Reality

- A bunch of independent authorities
- A lot of different public keys sitting around available to you.
- When you download a browser, in the image you get along with the executable code, there are a bunch of public keys that people decide are trustworthy.
- If you didn't have a browser, you couldn't do web browsing.
- Assuming your browser is a modern, commercial one, all this stuff is built in.
- When you download Firefox from Mozilla, I have confidence that they are absolutely right on who I should trust.
- They have given me 150 public keys, where some belong to big names like Microsoft, Google, and VeriSign.
- From the first lecture, trust is very important because trust allows us to determine whether we perform an action or not.

Certificates and Trust

- We need to determine if it is a trustworthy website
- We need to talk to a bank and do financial translations with the bank
- This bank's public key is the following
- Cryptography works upon checking
- We believed that Trustysign.com is who the bank claimed to be!

Q. Can the yellow guy sign other people's certificates?
A. If you set things up properly, he could and when you issue a certificate, there are multiple things you can say in this certificate.

- I believe that this guy is a trustworthy person and he himself should be able to issue certificates.
- This is if the chain of trust goes back to someone you trust.

Potential Problems in the Certification Process

- Transitive trust happens in real life, so why do we accept all this trust?

- Taking these things on faith.
- Trustysign.com issued a certificate for this bank
- Things change, all of this stuff with public key cryptography is based on the secrecy of the private key.
- Sometimes, things go wrong and private keys get divulged.
- People can pretend to be you and the public key that you trusted is actually being used by the hacker.
- We want to be able to invalidate certificates!

Trustworthiness of Certificate Authorities

- At the moment a certificate was issue, what was done to ensure the right thing to do?
- Did it have the president of the bank show up and indicate documents that had the incorporation of the bank?
- Did someone phone up Trustysign and did they say “sure, why not?”.
- We have no idea what Trustysign.com did to confirm security clearance or not.
- Companies that make a business of selling certificates
- You can buy certificates and it can cost you a tiny little bit of money
- You can be very damn sure and these will cost you a lot of money
- He did things that verify that the person claiming the certificate is really who he claims to be.
- In reality, no one ever looks at certificates
- Don’t worry about the degree of trust and say it is fine.

What Does a Certificate Really Tell Me?

- Certificate authority (CA) means that at some point, I will tie together a public key with this identity
- Doesn’t tell you why it thought it was a good idea, it just tells you a boolean if it thinks it’s good or not.

Showing a Problem Using the Example

- Lecture 5, Page 50
- Alice has gone to a facility and there is nothing wrong with this purple certificate
- She has never heard of yellow, so she doesn’t know the procedures yellow uses.
- Red guy is evil
- What if red guy says yellow will accept any old trash and you can get the certificate?
- Yellow guy doesn’t give a damn and gives out certificates each time someone gives him money.
- Yellow has a signature from the purple guy
- Purple guy has NOT done anything wrong and the yellow guy who he claims to be.
- Unfortunately, “Green” is the actual evil red attacker.

- Purple trusts yellow, yellow trusts green (red, **this is the bad step**)

Another Big Problem

- Adobe had its private keys compromised
- They fucked up bad!
- Several of their private keys are compromised and whoever took advantage posed as Adobe
- Before Adobe lost those private keys, they probably issued correct certificates
- How do we know which attacker stole what?
- We are shit out of luck in that regard.

Revocation

- A general problem for keys, certificates, ACL
- Revoke every certificate issued to that private key
- How do you revoke things (take stuff away)
- Capabilities
- Given the right to do something, how do we take away the key?
- There won't be just one place to check this, everyone in the world can determine this.
- We have to be sure it is safe and efficient!
- If we have a certificate in the past and discover later it is untrustworthy, we need to get rid of it.
- There are certificate revocation lists associated with web browsers
- We can configure browsers to check a certificate's validity.
- It goes to a special website that maintains certificate revocation lists.
- Checks to see if it is okay or not okay.
- If not okay, reject the certificate!

Revisiting Our Example

- Lecture 5, Page 53
- How does Alice make sure she is not accepting red's certificate

Realities of Certificates

- What if the discovery is made between when she downloaded the link?
- All your OS and browsers come with a set of "pre-trusted" certificate authorities
- There may be a hierarchy but it will be a fairly shallow hierarchy
- Usually does NOT go down too far where you have stepped outside your organization.
- If what you get coming in is NOT signed by someone you trust, then the safe thing to do is throw it away because it is NOT valid.
- In many cases, people don't understand what the hell is going on!
- There is NO good reason to believe all billion users understand certificates
- Too complicated to expect everyone to understand

- If the box pops up, it doesn't necessarily mean that something bad has happened.
- When you get a certificate, it could be valid for a certain amount of time.
- After 5 years have passed, we can pop up a box and whoever got the certificate forgot to get a new one.
- No security problem at all!
- Someone didn't bother to et a new version of the certificate
- First time you ssh to a new site, you get a message warning of man in the middle attacks but you accept it anyway
- Probably, you are communicating to the right person but I just don't have their public key yet.
- In some cases, you will be screwed because the only reason you are talking to the wrong person is because they want to attack you.

Q. Microsoft was risk-averse outside their corporations. Is there liability if Microsoft's certificates were bad?
A. It depends on the degree of the liability flub.

Q. When we do our hw when we ssh, when we ask "if we trust it", if we say no, what happens?
A. We cannot communicate to the actual website!

- You either trust them or you don't, or you communicate with them or you don't.

An Example

- Lecture 5, Page 55
- Many certificates on Firefox
- Same is true on Safari, IE, and Chrome
- Comes pre-configured with over 100 certificate authorities

Firefox Preconfigured Certificate Authorities

- Big names:
- Microsoft
- RSA Security
- Verisign
- No names
- Unizeo Sp. Z.o.o.

The Upshot

- If you have never heard the certificate, you have no reason to trust them.
- But your system's security depends on them

The Problem in the Real World

- Used to be a theoretical problem, but now very practical problem!
- DigiNotar lost their private keys!
- People who found them began to use them on their public keys

- Done for things like Twitter or Facebook
- Until compromise was discovered, everyone trusted their certificates.
- Somebody who is highly technologically sophisticated looked at the packets and determined that there is malice.
- Reason to believe the compromise was the Iranian government.

Effects of DigiNotar Compromise

- Attackers could redirect users to fake sites
- Attackers could eavesdrop onuser's
- Forward thing to the real Twitter website and it is a man in the middle attack

How Did the Compromise Occur?

- DigiNotar had shit security
- Had old versions with various vulnerabilities
- Weak passwords
- No auditing of logs to see if anything weird has happened.
- Those private keys were the companies major asset and they dun goofed.
- DigiNotar was absolutely awful at security and that was their job!
- You probably didn't even know if you trusted DigiNotar beforehand!

A Browser Solution

- Done by the FF people and Google people using certificate key pinning
- My browser in going to be aware that Google doesn't want random people to sign an key in the list
- Google has a small subset of those who have approval to sign the certificate
- If DigiNotar has this in place, stolen private keys would be caught by Google since DigiNotar would NOT have been on the list.
- Does NOT scale well because you have to encode this information in the web browser.

Q. How many places are am I going to go too?
A. This isn't a very scalable process but at least you have control over what you can see and not see.

The Core Problem With Certificates

- If you set yourself up as a certificate authority, you can create self-signed certificate authorities.
- You can get a certificate coming in from almost anyone
- Typical user is NOT going to figure out if they should or should not trust that certificate.
- Hence, they will probably trust it since the options are either trust the certificate, do what you want and don't trust the certificate, don't do anything

Should We Worry About Certificate Validity?

- Stuxent used certificate frauds

- Someone managed to convince Version to hand out Microsoft certificates to other people
- Microsoft was NOT happy about this!
- Attacker usually go to the place that is weakest, but this part is usually NOT the weakest link.

Should I Trust Crypto At All?

- There are a lot of reasons to believe the NSA and other government agencies can read encrypted messages.
- They have somewhat compromised key handling but didn't break the crypto algorithm themselves.
- Put something in the message to get in without brute force attacks.

Some Practical Advice on Crypto

- Bruce Schneler said to trust crypto.
- Big companies could have been influenced by people to put in weird cryptography.
- U.S. can do a lot of things to coerce companies
- OpenSSL is an example of a previous security device that had bugs in the programming
- It was there for years and people relied on the OpenSSL library!
- Since then, a number of companies that rely on OpenSSL have started to pay people to build OpenSSL and prevent security leaks.
- Standards are especially true because you can have a fair degree of assurance.

Security Protocols

- Designing secure protocols
- How to design things on the Internet
- Key exchange protocols
- We need to have the keys in the right places
- We have to assume they are in the right place, so how do we do that?
- We will have some common security protocols in this network, and many problems are general problems for key exchange.

Basics of Security Protocols

- Encryption is pretty strong
- Attacker won't break the crypto
- How do we achieve a message exchange to achieve the given result securely.
- Many difficulties are general
- Arrive in some circumstances that aren't network related at all.

Security Protocols

- Achieve some security result like setting up a secure channel between two parties

- Somewhere in there, we will start encrypting stuff

Types of Security Protocols

- Invoke the help of a trusted 3rd party
- Some of the steps in the protocol that are arbitrated will involve the trusted 3rd party.
- Adjudicated protocol
- Only two people talk to each other and exchange messages back and forth.
- If some party thinks something went wrong, look for evidence and tell if someone did something properly or cheated.

Participants in Security Protocols

- Alice and Bob are involved in all security protocols

And the Bad Guys

- Eve is an eavesdropper
- Mallory is actively malicious - he is a fuck
- Sometimes Alice or Bob might cheat

Trusted Arbitrator

- Trent is a disinterested 3rd party who is NOT in favor of any particular participant.
- We work on the assumption that this trust is well-placed.
- We presume he has NOT been compromised and work with this presumption.
- If things didn't go well, we will have to deal with an untrustworthy trusted authority.
- Often simplifies protocols but it could possibly add overhead.

Goals of Security Protocols

- Not a generic protocol, but rather it can have one effect.
- If you have N parties, we want a secure vote so we know what the majority of N parties wants to do.
- If it is not what you want to do, you need another protocol.
- Minimalism is an important secondary goal
- Send as few messages as you possibly can
- Send as little data as possible
- We do NOT want to include a lot of extraneous data for security purposes
- We are going to use encryption and we want to encrypt as little stuff as possible.
- Why don't we just encrypt everything?
- There is a performance cost but we need to understand why we are doing things in the first place.
- What is it about that protocol to be kept secret?

- When people design protocols, we should look exactly what we are sending and determine if the field needs to be encrypted or not.

Key Exchange Protocols

- Usually when doing communication across the network, you want to use a fresh key!
- Skype calls, new remote logins, etc.
- This means the keys have to be at both the sender and receiver.
- Nobody knows the keys at first and trusted 3rd parties won't know the key.
- How do we create the key securely?
- It has to be a wonderful random key.
- We want to get the key to participants securely, quickly, even if they've never communicated before.

Key Exchange With Symmetric Encryption and an Arbitrator

- For the trusted arbitrator Trent, they both trust him.
- Alice has keys she shares with Trent and Bob has keys he shares with Trent.
- Only Alice knows her keys shared with Trent
- Only Bob knows his keys shared with Trent
- How do Alice and Bob get a shared key?

Step One

- Lecture 6, Page 12
- Symmetric cryptography
- Alice requests session key for Bob
- Plaintext in this form
- Anybody can read it, saying I want a session key for Bob.
- At each step, who knows what?
- What does Alice know?
- Alice knows her key
- She knows she asked Trent for a key (could be represented as a state)
- Assuming the message gets delivered, Trent knows that Alice probably sent a message
- Bob doesn't know shit right now.

Step Two

- Trent creates new session keys for her (K_S)
- Nobody can deduce the key yet.
- Trent encrypts that key with the key he shares with Alice, and he also encrypts that key with the key he shares with Bob
- Alice knows there is an encrypted version of the session key
- Generated by Trent
- She presumes she has a key usable by Bob

- Keys only get compromised if you screw up in the protocol in this assumption

Step Three

- Alice can't use that other piece, but Bob can use it.
- The key arrives at Bob with this encrypted form
- Does Alice gain any knowledge?
- She gained the knowledge she sent it off to Bob, but nothing more.
- Trent's knowledge has not changed.
- Bob now knows the shared key though!
- It looks pretty good and we are supposed to get a shared key, K_S
- Eavesdropper Eve could NOT have derived this key.

What Has the Protocol Achieved?

- Session key was transmitted using keys known only to Alice and Bob.
- Because both keys came from encrypted passages, Alice and Bob know that Trent participated in this protocol
- Kind of garbage results in this protocol

Problems With the Protocol

- Mallory can screw with you

The Man-in-the-Middle Attack

- You have people who wish to communicate and an active, evil participant where you mediate the message and alter things and insert own copies of messages.
- The goal is to participate in the protocol and learn what the secret key is without legitimate participants knowing that he has done so.
- I, the man in the middle, will read all the encrypted messages even though I am not a legitimate participant.
- Security is still as solid as can be.

Applying the Man-in-the-Middle Attack

- Mallory has a key shared with Trent and Trent has no reason to believe Mallory is evil.
- Mallory can intercept Alice's request
- Mallory creates a message saying "Alice requests session key for Mallory" to Trent
- This is possible because the original message was sent in plaintext.
- Trent sees a message in plaintext so nothing seems out of the ordinary
- He will do exactly as he is asked to do!
- What do we actually know at this point?
- Alice, Trent, and Bob think nothing is wrong and it is the same as the previous protocol!
- What Trent thinks he knows is NOT the truth!
- We see that the knowledge we thought we had is NOT actually correct.

- What we know and what people think they know is actually 2 different things.

Trent Does His Job

- Alice wants to send something back to Alice
- Exactly what you expect (a brand new session key)
- What does Alice know at this point?
- She knows she asked Trent for a session key to talk to Bob.
- In comes a message with two encrypted session keys (which is true!)
- The difference is that K_M is for Mallory, NOT Bob!

Alice Gets Ready to Talk to Bob

- She is going to send the session key E_(K_M) (K_S) to Bob
- Mallory can now masquerade as Bob
- She can grab each one and decrypt each one

Really Getting in the Middle

- While Mallory can create meaningful responses, Alice can deduce that Mallory isn't behaving like Bob and then somehow, you have attacked me.
- Mallory doesn't want to be detected.
- Mallory can ask Trent for a key to talk to Bob
- Mallory gets K_S1 used for E_K_M and E_K_B
- Mallory can get out the second encryption key and let me decrypt it and see what I have got.
- Bob has an encryption key from Trent, and Trent has done exactly done what he was supposed to.

Mallory Plays Man-in-the-Middle

- Alice has a big secret and she wants to encrypt it with a shared key with Bob
- Mallory obtains Alice's big secret and decrypts it.
- He goes one step further by reencrypting it to Bob
- Bob gains trust and decides to take Bob's big secret
- Mallory intercepts it and reencrypts Bob's big secret with a key is sharing with Alice.
- Alice and Bob get their expected secrets, but Mallory also see it as well.
- Trent won't create key for three people because that would cause trouble.

Q. If Mallory happens to let an actual message from Alice to Bob slip, what would happen?
A. It would depend on how Bob reacts. It can be a legit error, or it can be a suspected man in the middle attack.

- NSA may be listening in on the message between your people.

Defeating the Man In the Middle

- Fundamental reason is that Trent doesn't really know what he is supposed to do.
- Does it actually represent the true intentions of people involved in the communication?
- Stuff comes back from Trent to Alice and she got a key that allowed her to communicate with Mallory
- Trent didn't know what he was supposed to do and Alice didn't check to see if she had to do the right thing.
- Trent would know what he is supposed to do and if we included the identity of the person, Alice would be able to see the identity of other participants.

Q. Does Trent need to know Alice's key beforehand?
A. Yes, he does need to know her key beforehand.

Applying the First Fix

- Instead of sending this in plaintext, we are encrypting it.
- Mallory can still have this technical capability but Mallory can't read the request.
- Mallory also cannot forge or alter Alice's request
- Mallory doesn't know K_A so Mallory cannot create that message!

But There's Another Problem

- She has asked for something that Trent verifies is indeed proper.
- The reason we need to further fix this is for replay attack.
- Take something legitimate the first time and send it again.
- Perfectly good the first time and this results in bad things happening.
- Mallory copies down a bunch of protocol messages and encrypts them again.

Step Two

- Trent does what he is supposed to do and the only thing that is different is that Trent encrypts the request.
- What can Mallory do with his saved messages?
- All the messages are encrypted either with K_A or K_B
- He can only encrypt it with K_A or K_B
- Are these messages of any use to him?

Mallory Waits for His Opportunity

- He can replay the previous keys back to Alice

What Will Happen Next?

- It looks like it is a new session key and we send the other copy to Bob and Mallory will let it through.
- What if Mallory had cracked K_S
- What he can do now is since he knows K_S, if he can force Alice and Bob to use K_S again, he can read everything they are sending.

Q. Would K_S expire?

A. You shouldn't use K_S again to be safe.

What Will Happen Next?

- Mallory could fool Alice into talking to him.
- Mallory has copied down all the encryption values and saved it for later.
- Later, he figures out that Alice talked to Bob.
- Mallory can intercept messages and decrypt things so Alice and Mallory cannot communicate.
- He can set up the other half of the communication.
- Alice is getting stuff that would allow her to talk to Mallory
- If the identity of who she is talking to is included, you could defeat the man in the middle attack.

Key Exchange With Public Key Cryptography

- If we know public keys already, how would we do that?
- Alice and Bob can exchange public keys!
- Alice will generate a session key and send it to Bob
- Bob gets something that goes secretly and he gets authentication with a key that only Alice knows

Basic Key Exchange Using PK

- Alice tells Bob her public key, and vice versa
- Alice generates a secret key and encrypts it.
- This will guarantee that Alice is talking to you, and this is a key only you and I will know

Man-in-the-Middle With Public Keys

- Mallory intercepts the message and lies
- This is a message that Mallory claims is the public key if it really isn't

And Bob Sends His Public Key

- Mallory gets the session key and reencrypts the other keys.

Alice Chooses a Session Key

- This is precisely the kind of problem that certificates are intended to overcome.
- Why did we believe this message?
- We don't have any certificates in this case.

Combined Key Distribution and Authentication

- Diffe-Hellman does NOT do authentication
- As you should be aware of from the examples, it could harmful to share a session key with someone you aren't familiar with.

Needham-Schroeder Key Exchange

- Uses symmetric cryptography

- Trusted authority
- Much more complicated protocol

Needham-Schroeder, Step 1
- Alice shares a key with Trent, Bob shares a key with Trent
- Alice generates a random number R_A that is new tech time.
- Alice then sends a message to Trent saying she wants to talk with Bob

What's the Point of R_A
- Quality of random # generator isn't important
- You want to defend against replay attacks
- Based on this unique number (**nonce**), you can identify if you have a replay attack.

Needham-Schoreder, Step 2
- Create a package and encrypt with Alice's key
- Bob's identity and another encrypted package is sent into here.
- This particular part of the package is encrypted with K_B
- Alice's identity can also be included.

Q. Why do we have all this stuff?
A. R_A is included to prevent replay attacks. When this comes back, R_A here can match R_A there, or it doesn't!
- If this doesn't match that, throw it all away and reject the session.
- Bob is the intended recipient and Alice is going to verify that it is a new key exchange customized to talk to Bob.

- Bob cannot be replaced. Mallory can intercept this message and input a key for Mallory.
- He cannot fiddle around with data assuming he doesn't know K_A

Needham-Schroeder, Step 3
- Alice has K_S so she knows she is talking to Bob
- She also knows it is a fresh message because the nonce matches
- At this point, once Bob decrypts the message, we are done.
- We are NOT done!
- We will see in the next class!

W 3 Dis 4-15-16
- My question about the ballot box!
- You need to use the execute permissions to execute the directory!
- tips user
- Yes, it should follow the specifications

Exploit labs
- Write upwards into the stack and jump into the location to execute the program

- Stack smashing
- The simplest case is if you make the program crash via segmentation fault
- A more interesting attack is one where you can compromise the system
- You might know this part of the buffer is a function pointer that allows you to execute some code.
- If we take advantage of a web server running as root, you effectively gain access.
- Avoid using C functions because they aren't as secure

Difference between strcpy and strncpy

- strncpy sets the amount of characters you want to read
- More precise and helps prevent this kind of attack
- strcpy looks for the zero byte

Pathname Attacks

- Different from firewalls because you are setting up an active system.
- If you remember the ACL, it can be very complicated and Linux permissions may not be paid attention to.
- Consider the *group* & *other* permissions, not just *user*.
- This is a little abstract and this model is a lot better with the Apache example.
- If Apache runs on port 80, it needs to run on the root.

We want a web server that runs with www-data!

- Port 80 is on this system less than 1024.
- Called pathname vulnerability because you will get access to the entire system.

Pathname Vulnerability

- Needs access to certain types of sensitive files, and because of this, if it is NOT properly locked down, you can gain access to the files.

URL Shorteners

- Takes a long URL and makes it very short
- What they do is create a sharing service and see if it is somewhat private or safe.
- Do a simple brute force attack and very quickly iterate through all potential possibilities.
- Download a valid URL or document.

SQL Injection Attack

- Whenever you are inputting values, it is often sent to a SQL database.
- You can input arbitrary SQL commands that are NOT sanitized, which means you can gain control of the root access.

- There is a lot of sanitization functions in the backend that does proper escaping and validation.
- A lot of the API's will execute just one statement or verify these statements.
- Use least privilege to minimize powers

Advantages of using telnet?

- Quicker

CRLF vs. LF newlines

- CRLF is Windows/DOS where you have a carriage return followed by line-feed
- LF is Unix where you have a line-feed
- Be cognizant of the way it parses

WARNING

- You will be formulating HTTP requests so you need to know the text has the proper encoding

/etc/shadow

- Important file that is an example of a pathname vulnerability
- If this is NOT properly secured, then we can gain access to passwords even if they are hashed.

Keep in mind there are two services and you will do this against the web server.

- Most people don't use encryptions but rather financial transactions
- Should we have backdoors or not?
- New encryption bill that is being proposed.
- They want to create a bill that puts a backdoor into every encryption that is created.
- This means that it is like potentially having access to anyone's data whenever you want.
- Getting a warrant for no reason.

No one is above the law

- Controversial debate over the power of Congress to enforce this.

RSA encryption algorithm

- Rivest Shamir Adleman
- Despite initial fears among the law enforcement that encryption would lead to impediments in anti-terrorism work, reports from DOJ show no federal wiretaps encountered encryption in 2002
- Most times they don't need to break the encryption

Why does Encryption work?

- You can decrease the # of tries you need to find the key
- You can create a fantastic new algorithm where everything thinks it is good but you know the secret of how to get in.
- RSA -> everyone knows this algorithm (easy to compute the product of two numbers, but if they are prime, it is hard to factor)
- Parameters can mean that it is hard to find who picked them.

DH seed

- Spend a couple of years to crack it and everyone can use the same seed.

Lecture Review

- It doesn't matter how strong your encryption algorithm is if someone gets your key.

Key Length

- For RSA, it is recommended to have a bit-length of 4096 bits (quite a long time to crack)
- Josh had to put up a web server previously and 2048 is generally enough but 4096 is definitely more secure.
- 2048 is secure against normal people, but the NSA can hack your web server.
- Consider the threat model and make claims against security.

Are There Real Costs for Key Length?

- 4096 is slower than 2048

Key Randomness

- Attack either hardware or random # generator

Generating Random Keys

- Don't use rand() because there are patterns in rand
- Let's say you are interning and they want you to implement something random
- What would you use instead?
- If you are doing low-level in Linux, you can use random and urandom
- random blocks
- If you are using C++, you can use the bcrypt library
- These would have strong properties for security.

Perfect Forward Secrecy

- A lot of messaging apps claim they have perfect forward secrecy, so you can use the same key for a couple of messages
- Called a session so we are NOT going to derive this for any messages.
- You cannot go back in time to recover the messages.

Diffie-Hellman

- Transmitting shared secrets is like the chicken and egg problem
- How will you submit the shared secret?
- The way it works is with public numbers and they agree on the same number.
- Public numbers do some sort of special computation using the public keys and using the properties of modulus.
- Vulnerable to man in the middle attacks
- Mallory is going to intercept the data and decrypt it and reencrypt it
- Diffie-Hellman used for perfect forward secrecy

Entropy
- A measure of randomness
- Time between different events
- Sometimes, sample the voltage fluctuations of temperature.
- Disk drive delays that are used holistically speaking that have strong degrees of randomness
- If it is reproducible, we can just play it back.

Why is the certificate authority used in Firefox?
- Verify public keys of sites you have visited
- What will happen if you go to visit the website?
- What is the flow of exchange here?

How Did the Compromise Occur?
- Security company called DigiNotar paid little attention to security

A Firefox Solution
- Certificate key pinning

Heart bleed and Certificates
- Current evidence suggests Heartbleed can expose certificates
- You can essentially steal any information and this allows anyone on the Internet to read the memory of the systems

Arbitrated protocols vs Adjudicated protocols
- Arbitrated always has a 3rd party involved
- Adjudicated has a trusted 3rd party coming in after the main stuff has been done

Key Exchange With Symmetric Encryption and an Arbitrator
- You cannot do authentication with a symmetric key and anyone who has the key can do this.
- Symmetric Encryption: no way to do Diffie-Hellman using only symmetric encryption

Step One

- Alice will make the request and Alice knows who Bob is.
- Trent is responsible for generating this.

Step Two
Q. Why are there two encryptions here?
A. Alice can open one and Bob can open the another, but NOT both.
- Both are secure.

But There's Another Problem
- What is a replay attack?
- If you can include the identity, what can Mallory still do?
- You can replay messages that were sent previously to exploit something for a key you may have broken.

Mallory Waits for His Opportunity
- How is he going to use this to his advantage?
- Trent sends the message encryptions and what is Mallory going to be able to do?

Basic Key Exchange Using PK
- Alice's PK gets the result from Mallory and it looks like they are talking to each other

W 4 T Lec 4-19-16
- Key distribution is an important thing with security protocols
- Symmetric cryptography to do this (man in the middle attacks)
- Public key cryptography
- Situations in which we were concerned about replay attacks

Needham-Schroeder, Step 1
- We need a trusted authority Trent who we all trust for key exchange
- Mallory is also a customer of Trent, so he can work with Trent but he cannot force Trent to do something against the rules.
- Alice wants to talk to Bob and exchange a key securely.
- Alice shares a symmetric key with Trent, and Bob shares a symmetric key with Trent
- Alice sends a nonce to Trent and perhaps it isn't necessary for Alice and Bob to be put together.
- We don't want people to understand what they are saying.
- The nonce is sent in plaintext.

What's the Point of R_A
- R_A is a random # chosen by Alice and it is unique, so it has never been used in a protocol before.
- This means that anything involving the nonce would not be valid because it does NOT include the nonce.

- We have to be careful though because the attacker can modify the nonce to a previously used one in a replay attack!

Needham-Schroeder, Step 2

- Alice knows she has requested a key to talk to Bob and she remembers the nonce
- If all goes well, Trent now believes that Alice and Bob would like to communicate securely and R_A is a nonce for this distribution protocol.
- Trent creates a big old package with a secret key that Alice and Trent shared.
- Only Alice and Trent could have possibly created this key
- We assume the cryptographic operations cannot be overcome.
- Nobody but Alice or Trent could have created this, and Alice didn't create it, so only Trent could have created it.
- Within the stuff encrypted in Alice's key is another package that Bob and Alice share.
- The nonce is NOT included at the beginning of this protocol because Bob doesn't know the nonce!
- The nonce ensures freshness, which Alice gets a guarantee that this is a fresh protocol and NOT a replay.
- Alice is going to get back a message to talk to Bob and this could be replaced with a nonce.
- What would appear here is Mallory, not Bob!
- This is one of the reasons why the initial message that went from Alice to Trent could be in plaintext.
- Could be checked in an encrypted message later on and it would have been okay later on to send this message in plaintext.

Needham-Schroeder, Step 3

- Indeed, Trent is going to need K_S and Alice needs an encrypted package of Bob's key.
- Bob gets the package and he does have K_B
- He is running the protocol
- Bob decrypts the message and realizes that Alice wants to talk to him using Trent's encrypted package
- They know who they are talking to because of the encryption.
- Unfortunately, we are NOT done!
- New replay problem that we haven't dealt with yet.

Needham-Schroeder, Step 4

- Bob generates a new nonce and he has never used this random # in Needham-Schroeder before.
- Encrypt the nonce in a brand new session key to the person he thinks he is communicating with (Alice)
- She doesn't know what nonce Bob would have chosen so this doesn't tell Alice anything.

Needham-Schroeder, Step 5

- Alice subtracts 1 from the nonce and reencrypts it i.e. E_K_S(R_B-1)
- Given the encrypted version, they will be unable to predict what the encrypted version of R_B-1
- Bob knows he didn't create it
- Alice could have created and she did create it.
- Trent could have if he remembered the key, but Trent is trusted to not do this!

What's All This Extra Stuff For?

- Alice knows she's talking to Bob because Trent told her so
- Mallory canNOT jump in later because the only time that K_S is sent is when Alice encrypts her key or when Bob encrypts his key (Just twice!)
- K_S is used so Mallory cannot jump in unless he knows Alice's key or Bob's key

Mallory Causes Problems

- Mallory really likes what happened with Alice and Bob communicated the last time
- Legitimately, someone put $1000 to Mallory's account.
- If Mallory was watching using the established key, he could make a copy of all these messages.
- Can Mallory later replay the conversation?

Mallory Waits For His Chance

- Mallory at this point grabs that message and intercepts the message to avoid Trent
- Instead, he takes Trent's old message and sends it back to Alice.

What Will Alice Do Now?

- Everything looks pretty good here, so let's go ahead with the protocol.
- Mallory is going to step aside for a bit to allow Alice and Bob to actually communicate.
- Bob won't create the nonce because the protocol is done without nonces.

So What's the Problem?

- Trent created it at some point in the past and we are assuming Mallory has NOT cracked the key.

Mallory Steps Back Into the Picture

- Mallory decides to intercept all the messages and first sends an old message 1 that was encrypted with K_S to Bob.
- Bob believes he was supposed to get a proper message from Alice and falls for the dupe.

- Mallory then sends E_K_S (Old message 2) to Alice
- This strategy probably would NOT work if Alice and Bob were people
- If Alice and Bob were banks, this replay attack can actually work!
- Mallory can replay Alice and Bob's old conversation

How Do the Random Numbers Help?

- Alice's random # assures her that the reply from Trent is fresh

- Why Bob Also Needs a Random #
- Let's say Alice doesn't want to talk to Bob

So What?

- Mallory cannot understand what is being sent or received
- However, Mallory can fool Bob into thinking he is talking to Alice
- He might be able to get the $1000 deposit without talking to Alice
- Bob wants a high assurance that Mallory didn't replay an old message
- He wants to be sure this is a fresh session and that Alice is really there and wants to talk to me.
- Bob needs to send Alice something never sent to her before.

Q. On Step 3, when Alice was sending a message to Bob, how did he know it was a session meant with him?
A. No one knows anyone else's key, so he knows it is linked to him.

So, Everything's Fine, Right?

- If you use the second nonce, then Bob's encrypted version is sent back to Alice and he chose a different R_B the last time.
- Why do we have Alice decrypted and we want to ensure the session key and send back the nonce itself.
- Anyone could have copied it down and it is something else related to something we sent.
- Mallory can decrypt it if sending R_B
- If Mallory is saving an old session, we might be in trouble

Mallory Cracks an Old Key

- Mallory compromises 10,000 computers belonging ot 10,000 grandmas to crack K_S
- Mallory wants to screw Bob over and sends K_S to Bob
- Bob generates a brand new nonce over encrypted, but unfortunately Mallory knows K_S
- This means Mallory can answer Bob's challenge
- Based on the assumption that Mallory can get the key

Timestamps in Security Protocols

- One method to handle this issue is the timestamp
- Sitting in one of the backup tapes could have been a copy of K_S
- This kind of thing can happen but it takes time.

- Takes a while to break into a facility and figure out what you need to get.
- Any of these key distribution sessions are good for only a particular amount of time.
- If OTOH someone is trying to do a replay attack, we need to detect a really old session.
- Timestamps are different from nonces but time keeps moving forward.
- Definitely NOT random
- Monotonically increasing set of values
- Related to true real time
- Related to the amount of work that an attacker can do to crack a key.

Using Timestamps in the Needham-Schoreder Protocol

- Trusted authority includes timestamps in his encrypted messages to Alice and Bob
- This is based on a global clock that everyone looks at

Using Timestamps to Defeat Mallory

- Package that went in the past and there was a timestamp T_X
- This was pretty close to when Alice would like to talk to Bob
- It is no longer close to that time but rather far in the future.
- Now Bob checks T_X against his clock
- If it is too old, Bob throws it away

Problems With Using Timestamps

- They assume a global clock
- Everyone can read this clock
- In distributed systems, you realize that global clocks are hard to achieve with granularities that you need.
- You can run a time synchronization protocol and maybe the attacker will attack this first.
- Attacker will cause your protocols to be out of sync and you have to say that there is some period from the start to the moment at which it becomes illegitimate.
- We have to be careful about the time for that.
- If we make it too long, there is a possibility that the attacker can get hold of the key.

The Suppress-Replay Attack

- Assume two participants in a security protocol
- If sender's clock is ahead, then this attack is feasible

Handling Clock Problems

- Usually we have clocks that are pretty synchronized
- GPS and each satellite is synchronized very closely
- Makes it hard to attack their synchronization protocol.
- Good reason to believe that it is the proper clock value.
- There is a synchronized clock assuming we have the ability to use GPS

- Even if you have the hardware, there are situations where you cannot achieve GPS
- If you are in a big building with a lot of steel, you might not get signals.
- Use a different protocol since all comparisons will be to the same clock
- Use your clock value and whenever we need to check a clock value, we can design protocols with that effect.

- If we had K_S and also the time stamp, if Mallory deduces K_S, Mallory cannot change the time stamp!
- Timestamp works according to the idea that Bob waits to identify the time stamp instead of acting immediately. If the timestamp is in the past, he throws it away because he has done it before.
- If this requires that you send an encrypted message to Trent, then nobody but Trent could have created that message.

Q. Why doesn't Mallory crack K_B and K_A rather than K_S
A. Could be embedded in hardware to make it difficult
- We need to avoid relying on secret keys somewhere.
- This means that if you have such a key, you need to exercise caution in protecting it.

Is This Overkill?
- Never sent K_A or K_B across the wire in this case.
- Why are we bothering with all this nonsense?
- Requires special nonsense and abilities.
- Sometimes, we can achieve only limited effects.
- If replaying the old session doesn't do you any good, who cares?

Why Should We Care?
- Bad guys can be very clever!
- If you change how you use the protocol, the software developer can extend the program as we implement extra functionality
- We have to consider how new features can impact vulnerability
- Always choose the protocol that is NOT vulnerable!

Something to Bear in Mind
- Particularly, for generic problems, we can have man in the middle attacks on machines
- We need to understand at a high conceptual level how to deal with these problems.

Lecture 7: Authorization and Authentication
Introduction
- Access control is predicated on knowing who someone is
- If you don't know those things, you cannot make good access control decisions

- Authentication determines the identity of some entity
- In order to have authentication, we need to have the notion of identity
- Whoever is coming to me belongs to one of those identities in that space.
- We need a high degree of confidence to understand who it is.

Authentication vs. Authorization

- Authentication is determining who you are
- Bill, Bob, Alice, Bob, Mallory; who are you?
- Authorization
- Okay you are Bill. Can Bill write to this file?
- Without good authentication, you cannot make good authorization decisions!
- The reason we do authentication at all is that we are going to do authorization later on.

Proving Identity in the Physical World

- This has been around for all of human existence
- How do we figure out who someone is?
- We can use physical recognition to know who we are
- We know each other well enough to understand how our body shape looks from behind.
- Even in the real world, we would sometimes run into problems

Other Physical Identification Methods

- Identification by recommendation
- You don't know Bill but you know Jane.
- Hi, here is Bill, we believe her!
- Identification by credentials
- Hi, I am Bill, he pulls out his driver's license.
- Comes from a trusted source that tells you who someone is.
- Identification by knowledge
- I happen to know that Bill knows a secret I know, tell me the secret.
- If you know the secret, then you are Bill.
- Identification by location
- Apply for your driver's license
- Why did you believe that the person behind the counter is authorized to give you a driver's license
- They didn't show you any credential, so why did you believe them?
- The location (behind the counter at the DMV) gave you assurance that they are who they are.
- All of these can be faked and have some weaknesses
- These all have cyber analogues used in computers and networks

Differences in Cyber Identification

- Whoever we are trying to identify may not be human.
- Sometimes, person doing the identification isn't human.

- Often, there is no physical presence required.
- Amazon has a big facility up in Oregon
- Nothing involving physical presence is going to help us here.
- We believe it no matter what.
- We could have a situation where we thought someone had a particular identity and we realized later we were wrong.
- We could have figured out based on their behavior that this isn't who they thought they were.
- If the computer figured out you are Bill, they would believe it.
- This suggests we have to be careful about identifying people on computers

Identifying With a Computer

- Computers are clearly NOT as smart as humans
- We have to define carefully and precisely the steps we are going to use.
- Computers do NOT do facial recognition as a person
- Computers, however, can do fast mathematical computations much quicker than any human can do.

Identifying Computers and Programs

- Let's say we need to identify something
- If there is no person at the other end of the communication, how do we do this?
- If we are going to authenticate a program, keep in mind they are real easy to duplicate!
- How on Earth will I identify that this is original vs this is a copy.
- Computers are NOT very flexible and that is all they can do.
- Computers are NOT flexible with backups i.e. if I forget my driver's license but bring my passport.
- They are good at computation
- Two machines can authenticate each other to do computations
- We cannot expect to do computations in the human world since we aren't as sharp at that.

Physical Presence Optional

- Authenticate something at the other end of a network
- Have a remote site and see that we can pass bits across a wire.
- Characteristics that networks have
- Active wiretapping
- Come through with us and we have to deal with it.
- Everything we are using to authenticate is reduced to digital information (bits)
- Doing facial recognition across the wire
- Camera takes a picture and the bits get sent across the wire.
- We can take that set of bits and represent the person's face.

- Somebody else can send another set of bits without having ever taken that picture at all.

Identity Might Not Be Rechecked

- Human beings are good at recovering from mistake, but computers are NOT so good at this unless we explicitly identify exception cases.

Authentication Mechanisms

- We want to ensure if we can allow those entities to do certain things or not.
- You can authenticate based on something you know
- Passwords
- Something you have
- Smart card, security token, badge
- ATM cards
- Something you are
- Biometrics (eyes, voice, fingerprints)
- Inherent to you and NOT anyone else
- Physical characteristics of your body
- Somewhere you are
- Usually identifying a role
- Location based

Passwords

- Most familiar form of authentication
- Authentication what you know
- Long predates computers
- Armies used to his thousands of years ago.
- User knows a secret other people don't know
- We can check the secret and authenticate him as being that user.

Problems With Passwords

- They have to be unguessable
- They also have to be hard to memorize
- If you are working with authentication across a network, once the secret has been provided before, attackers can listen in and attack.
- Why would anyone set up a system like that?
- Practically all websites have log-ins like this!
- We need to be aware of the description of a problem and make sure no one does that.
- Passwords are susceptible to sniffers.
- Unless the passwords were wrong, brute force attacks can work on them.
- Nowadays, computers allow you to keep trying until we get the password right.

Proper Use of Passwords

- Should be sufficiently long
- Should contain non-alphabetic characters
- Harder to guess the right password
- Should be unguessable
- Use a random password with no particular pattern
- Passwords should be changed often
- Never write passwords down
- Never share these passwords!
- Difficult to achieve all these simultaneously

Passwords and Single Sign-On

- Many systems keep working until you log out and you need to provide your password again.
- We want complete mediation to check every operation.
- Maybe, we were sitting in front of the computer the whole time.
- We can still be accepting that as you!
- Vulnerable for a lot of websites -> when there is a copy, we still need to treat it as Peter Reiher.
- Tell Amazon that I am Peter Reiher and send a copy of that cookie.
- It would be a pain in the ass to include your use ID and password every time you have an HTTP request

Handling Passwords

- Everyone will get a password and we need to check to make sure it is right.
- Keep checking and we need an approach to do this properly.
- Do something that says this is the right password or the wrong password.
- Keep a copy in the file somewhere and compare it using a string comparison
- If the file ever gets divulged, people can figure out the information in the file.
- The OS can check the password and this will be good enough.
- The perfect type of function for this purpose is a one-way function.
- Hash functions are of this character!
- There are some specialized hash functions that are good at this.
- It is very difficult or impossible to deduce anything about the original bit pattern.

Q. When you are storing, you want to encrypt it first, how do you use it in the future?
A. User Reiher, here is a hash of his password. We type in the password and we run it through the one way function to what we store.

- Checksum

Q. Wasn't there an attack on the password hashes?
A. Not the best way to do this, but better than unencrypted passwords.

One Way Functions

- Data A -> Data B
- Hard to convert Data B back to Data A
- Often done as a particular type of cryptographic operation

Standard Password Handling

- Marx Brothers' Family machine
- Lecture 7, Page 20
- Somewhere on the machine, you have stored a file and a hash of their passwords.

Is Encrypting the Password File Enough?

- What happens if an attacker gets hold of a copy of your password file?
- They have the encrypted version and if the encrypting function is good, he cannot just take the unencrypted password as is.
- This vulnerable to a dictionary attack!

Dictionary Attacks on an Encrypted Password File

- They run words in the dictionary and check against everybody!
- It matches Karl's encrypted password
- Now we know that Karl Marx's password is that.
- One hash function used by all Linux machines!
- They can steal the knowledge of what your files have.
- What are the chances that people used aardvark as your password?

Q. Why aren't there mechanisms trying to prevent brute force attacks?
A. The attacker has taken the password file on his own machine and this won't work very well over the wire if he is smart.

- Serious problem

- Hash function is deterministic, meaning people with the same password will hash to the same result.

Dictionaries

- Real dictionary attacks don't use Webster's
- Rather, they use probability of words being used as passwords.
- Include more common words than less common ones
- Set up procedures i.e. their username backwards
- Their username with a '1' after it
- Check common names and a list of the 5 most common men's names in countries in question.
- Generally speaking, people doing this stuff improve their dictionaries.
- Certain amount of secrecy that attackers use in keeping their own dictionaries.
- This kind of thing happens all the time
- Get hold of a copy of someone's password files

A Serious Issue

- All Linux machines use the same one-way function to encrypt passwords
- You would never again need to check the password against the function
- When you steal the password file, any matches in the dictionary?
- This would mean that you could create the largest possible dictionary of hacked passwords.
- People designing Linux knew how to address the file.

Illustrating the Problem

- Dictionary attack sees that “beard” always encrypts to something, so we have passwords going to the same guy.

The Real Problem

- Darwin and Marx chose the same password and it hashes to the same thing
- This means that you could create this huge dictionary and get the hash of each one of them and we would then have to use pattern matching

Salted Passwords

- Best practice where we have to code up such a thing.
- Take whatever plaintext password someone has chosen and generated a brand new random #
- Combine them in some reasonable way i.e. XOR or concat
- Run it through the one-way function and use it for a one way function.
- This random # does NOT need to be secret
- Guy is going to come to your computer and run it through your password.
- You stored his password and you will need that random # each time you try to check his password.
- You have to store it and store it in plaintext.
- You’re going to have the key sitting somewhere on the system, so people could probably have privileged access anyway.
- He could decrypt all the salted passwords anyway, so he didn’t gain much of anything.
- Doesn’t lose any security from plaintext.
- The fact he knows half of what you hashed should NOT give him any help figuring out what the other half is.
- If you have a crummy hash, you should obviously change it.
- It is alright to store the # of plaintext; you just need it to be pretty random.

Did It Fix Our Problem?

- Darwin and Marx both choose a password “beard”
- Karl Marx takes one flavor of salt and runs it through the function
- Charles Darwin makes a different flavor of salt and it yields a different result.
- The corresponding result is different, so the attacker has NO reason to figure out the password is “beard”

Q. Is there a value of having a larger size for salt?
A. Yes! Harder combinations are harder to check. With a single bit, 2^1
Two bits, 2^2
Three bits, 2^3
….
32-bits, 2^32 times the work

What Is This Salt, Really?

- We are seeing something where we see a random # and we are using it to get uniqueness
- We want to make sure it is unique each time and we want to see why this is okay to leave it in plaintext in this file.

Modern Dictionary Attacks

- People often choose crummy passwords
- Let's say he has stolen a password file, if he has stolen your file, he can eventually figure out your encrypted password.
- If you have 1,000 people, he will do a separate dictionary attack on each of these people.
- Unfortunately, it is feasible for these modern dictionary attacks to work because of the computational advances of modern computers.
- Choosing a good password obviously matters a lot.

Password Management

- Limiting login attempts
- Protecting your password file
- How should we deal with forgotten passwords?

Limit Login Attempts

- If you are lucky, you can still run a dictionary attack on your login procedure.
- Here is an opportunity to detect a dictionary attack
- Once the attacker has guessed incorrectly a certain amount of times, we have to judge if it is a dictionary attack or if someone forgot their password.
- After a few attempts, they will either reset the password or say I forgot my password
- Lock account
- Until some special action is taken.
- Slow down
- Humans probably are looking at a dictionary, or a program could be spewing out passwords.
- After the 1st guess or 2nd, slow down and make it take longer!
- It is very sensible to say to slow down after a few guesses.
- Watch the results carefully!
- Is this where he is guessing weird arbitrary passwords?
- If it is the latter, allow a bit more to happen in this situation!

Limited Password Guesses and the FBI

- The recent FBI iPhone case hinged on this concept
- They would like to get into the iPhone and get any information about the shooters.
- They had no idea how to know the password, except the shooters who were dead.
- It would slow down more and more and eventually, it would delete an encryption key
- Past a certain point, all the data would be lost.
- The key was stored on the machine to decrypt it when you needed it.
- After a certain # of guesses, the key would be deleted.
- All the encrypted data would be deleted because no one would have the key any longer.
- AES cipher with a 128 bit key.
- Most people cannot crack this!
- The FBI cannot keep guessing here because we would eventually lose all the data.
- They also wanted people to login over the wire and enter the password via keyboard.
- Tedious to have an FBI agent type on the keyboard, so the FBI wanted Apple to change the Operating System
- FBI had little to do with a dictionary attack

Encrypt Your Passwords

- Unix systems have been doing this for 25-30 years.
- Pretty cheap and easy to do this.
- You would think everyone does this, or anyone who was technologically savvy.
- Yahoo lost half a million unencrypted passwords in 2012
- It does take some cycles to encrypt passwords as they come in.
- The more hardware you have to buy and less profit for your company.
- It is a whole lot more secure.

Protecting the Password File

- It is okay to leave the encrypted version for the password file?
- NO!
- You can lose all your passwords pretty quickly.
- Set up access permissions so the only thing you can do is access the login program.

Other Issues for Proper Handling of Users' Passwords

- Check it to see if it is right.
- You have an unencrypted password; you want to get rid of it ASAP
- This is partially an issue of how you store your file
- This is partially an issue of good programming

- Get rid of passwords and don't leave them in temp files or anywhere else
- People would put a log message into files each time a user was wrong.
- You typed in your password and got it wrong by one character.
- Once an attacker gets a log file, they are probably off by one key so he can try a few hundred things and he can get your password.
- It should NOT be possible to print or save an unencrypted password
- Do NOT save a password in an unencrypted form
- Do NOT move it in plaintext and if the server is compromised, this won't matter.
- Attacker will remove your well-written software.

Wireless Networks and Passwords

- Wireless networks are often unencrypted or encrypted with WEP (which is bad!)
- Websites used to request and transport passwords in the open
- Send an HTTP request over the network in plaintext and including the password
- Anyone raising the antenna would be able to hack into your password.
- If you are using passwords on your website, use HTTPS throughout.
- Once you login, you aren't going to request his password again.
- Send a cookie to user Reiher and this would re-verify you with your next HTTP request.
- When you are are NO Longer sending the password, he can send in his own request this used to happen a lot.
- Firesheep is a plugin for your browser and it would give you a copy of all the cookies.
- This brought the issue to the attention of many websites.
- You need to make sure you not only encrypt the original login session, but also cookies to verify the user.
- Usually use HTTPS to provide encryption on top of HTTP
- You have to pay encryption costs and have more hardware for you.

Handling Forgotten Passwords

- A bad idea is to store plaintext password and when a user sends them on request, send it back to them.
- Better idea:
- Generate new passwords when old ones forgotten
- There are still sites that do this!
- It happens frequently but lets generate a new password for you!
- The website will generate a password and give it to you somehow.

Generating New Passwords

- Generate a great random password with the full character set with no meaning whatsoever.
- The only problem is you have it in the website and there is a network to cross to send it to the user

- How do we get it to the user?

Transporting New Passwords

- Engineering solution to end it via email.
- Going across email is often unencrypted.
- This almost always works out fine without problems
- There is one serious problem with this
- Sometimes, people change their email provider.
- AOL email account and now people have a Gmail account
- An attacker who has figured this out can crack the AOL account and there is nothing there I care about.
- He will go to your Amazon account and generate crappy attempts on your password.
- We will go to AOL.com and we now look at your Gmail account because you don't remember.
- Compromised the AOL.com account and people have had loss of passwords and accounts this way.

Password Proliferation

- Practically every web site wants you to enter a password

Using the Same Password

- +: Easier to remember
- -: Much less secure
- When an attacker gets one of your accounts, he gets ALL of your accounts
- He will go to every possible account in the universe and try to figure out your information
- All the other information will be compromised
- None of them are nearly as bad as Yahoo and they had an unencrypted password file

Using Different Passwords

- +: Much more secure
- -: How many can you actually remember?
- Many of us cannot remember our passwords in that way.
- You can do a little something to customize it for each site at the end

Other Options

- Use a few passwords
- Low-Security sites get low-security passwords; high-security sites get high-security passwords
- Write down your passwords
- Cyber attackers cannot get your hand-written passwords but your evil co-worker could work.
- Use algorithm customized to sites

- Password vaults are another option and they are the practical option
- Login page that has a password vault and put in the info for your site.

W 4 R Lec 4-21-16

- Windows -> Get rid of QuickTime
- Apple is NOT supporting QuickTime for Windows anymore.
- Open House for undergraduates and bring up any concerns tonight

Choosing Password

- We had finished talking about security protocols dealing with man-in-the-middle attacks, clocks and how to use them properly.
- We talked about determining if they are who they claim to be.
- Authentication based on things you know, things you have, etc.
- Passwords have properties with methods of authentication for computer systems.
- It is necessary for ordinary users to choose passwords and there will be an automated password put in.
- Store the password somewhere online and the password could be obtained to log in even if you aren't the party you claim to be.
- Keys could be embedded in the software, which is a horrible idea.
- Theory is that people remember their passwords and feed it back into the system.
- How do you go about choosing your password?
- You want it hard for anybody except the guy who chose the password to use it.
- It has to come to his mind quickly when he needs it.
- Is it something a person will actually remember.

How Long Should Passwords Be?

- Attackers use dictionary attacks and sometimes brute force attacks on passwords.
- The more characters, the more possible passwords you could have chosen from.
- Hackers have a program that checks against the database, so they have computer speeds that enable quick guessing.
- As computers get faster because of Moore's Law, databases are easier to crack.
- You can divide up your machine into 8 pieces and parallelize the hacking process!
- Given realistic expectations of databases and modern computing, it has been determined that 15 character passwords are pretty safe.
- Often, this is problematic on older sites because you try to put in your 15 characters and it says only 8 characters allowed

Some Password Choice Strategies

- Use first letters from a phrase that you can remember

- It is okay to use common words if you use several of them and have nothing to do with each other (apple tiger)
- One thing to increase the complexity of the passwords is to NOT use letters!
- Sites that care about your security wants you to throw in numbers and punctuation on your keyboard/
- Attackers are building up dictionaries out of procedures that he thinks people commonly use
- Replacing “l” with “1” or “0” with “o” is NOT super useful.
- It is also a bad idea to limit the non-alphabetic characters to the 1st and last characters.

Password Lifespan

- You shouldn’t use foreign language dictionaries because people know that too
- Don’t use names or common Proper Nouns
- Letters + symbols need to be memorable ONLY to you!
- It isn’t going to be too useful if you forget your password each time
- Ideally, all your passwords should be changed frequently.
- Eventually, they may guess it by trying several thousand passwords on thousands of machines.
- If you change your password, it will render their attack useless because they have to repeat the dictionary attack
- Inconvenient because you have to remember the password each time, which can be a pain in the ass!
- There is going to be tradeoff between convenience and sticking with the old password.
- If you have had passwords you have used for several years, you should probably change them!

Challenge/Response Authentication

- System asks a question that it believes that person should know
- If the answer is right, the computer believes it has authenticated the person.

Differences From Passwords

- Challenge/response systems ask for different challenges every single time that a user comes to the system
- In practice, these systems work with a certain set of information
- These are questions like “what is the street you lived on when you first entered school” or “what is the make of the first car that you drove”?
- Moreover, when you think of these questions, they aren’t the best questions in the world.
- A lot of this information isn’t private, and people can figure out what street you lived on

- When you set up these accounts, you should choose questions you don't forget!
- They don't typically give this out except banks can go to credit reporting agencies.
- Very crummy and insecure method of doing authentication
- NOT a totally separate authentication mechanism!
- When you forgot your password, many websites essentially ask you a challenge question!
- This is an important means of authenticating you.
- The security is based on the challenge response system
- When it asks the challenge response question, it asks something like "What is your best friend's name in elementary school"?
- This is NOT the level of security that you want.
- People are seriously compromising the system.
- All of the other stuff people are doing is window dressing by using mis-authentication

Q. Does it make more sense to have a two factor thing? Do I need password and challenge/response systems?
A. This isn't really that important but having two factor authentication in general is a very good idea.

- Probably doing this over the network most of the time!
- Anyone sniffing the network can find out what website you are communicating on.
- Saw that you eat a particular encrypted thing and was sending all sorts of encrypted stuff
- If they can replay the encrypted stuff, bang, they are in.
- You want to use nonces to customize encryption every time.

Challenge/Response Problems
- Small handful pieces of information about you that they use to create challenge/responses

Identification Devices
- Authentication based on something you have.
- A start card or hardware device that a computer can scan
- Authentication to a computer by providing the device to the computer
- Information from these devices is used to authenticate a person
- People have to use this device in some fashion.

Simple Use of Authentication Tokens
- Often just a thing that has a battery and a display and it might pop up a #
- Different # every few seconds
- Read # off the token and try to read the person in.

- Create a pseudorandom # generator for each token and embed a copy associated with that user and use relative synchronization of clocks
- Since pseudorandom # generators are deterministic processes, as long as the clocks are fairly synchronized, the computer and the authentication will be running at the rate.
- Sometimes, you can do this by typing or connecting the authentication device to the computer.
- You can do it via wireless as well but then there is the issue of spoofing over wireless.
- What happens if the thief steals your token?
- You canNOT authenticate, but he can!

Authentication With Smart Cards

- Lecture 7, Page 55
- User takes smart card like a credit card and inserts the card into the computer he is using
- Smart card takes the challenge that is cryptographic and sends back a response
- If someone is sniffing in the network, he can replay E(challenge) but this won't help them!
- Sniffing the new # across the network and you have a minute where you can authenticate yourself to the other computer.
- Usually, it takes a few seconds to to authenticate and login.

Some Details on Smart Cards

- What if the computer is compromised,
- If you did cryptography on the computer, this means the computer had any secret keys available
- Answer any challenges required do the authentication
- It is only possible to create the encrypted version of the challenge using the smart card.
- Sometimes, ifs we want to be really careful, the user can enter a password that activates the smart card.
- Otherwise, the smart card won't share the smart card.
- The smart card will need an entry data options and if that smart card leads to your accounts, it is less secure
- Don't rebuild the wheel; use technologies open for smart cards!

Problems With Identification Devices

- Use two factor authentication
- If you use your smart card in an unintelligent fashion, it is relatively easy to crack it
- Anyone who buys a little hardware can read what is on your mag
- They require special hardware!
- Engineering workaround for this problem is to use a smartphone
- Two ways to communicate:

- Over the Internet
- Over the wireless, telephone network
- Send a challenge to your cell phone
- NOT as strong as true smart card.
- Law enforcement agencies can set up their own cellphone tower and grab cell phone calls before they get to the real telephone tower.
- If you are talking about your bank account and you never have more than that, you probably won't be attacked this way.
- This is NOT a problem for an average, typical bank account.
- Principle of adequate protection - you protect things to that level
- Extra inconvenience to authenticate yourself to the bank
- Wait for the code they give you on the cellphone
- Takes time to input and receive the code message

Attacks on Smart Cards

- Based on fake ATM machine attacks
- In order for this to NOT be an effective attack, the card should NOT respond to fake or tampered terminals
- If you have a recently added credit card, you might have a chip built on it. Reiner definitely does!
- Make sure this is secure and NOT easy to crack.
- European Chip & PIN as means of defense
- Only extra protection is that it is difficult to create a duplicate of the chip

Q. Why don't they add the pin capability?
A. Pain in the ass for the users and your grandma has forgotten her PIN.

- Everyone else in line gets pissed off
- Small transactions is willing to accept a certain amount of fraud in exchange of user convenience

Q. For ATM machines, are they translated back to attackers or no?
A. Sooner or later, whoever came through and put new cache into the ATM machine can figure it out.

Another Form of Attack

- Guy who legitimately has that piece of hardware is trusted, but is that really him?
- Other situations in which you use smart cards
- Terminals for rapid transit systems i.e. subways or metros
- It is to your benefit to have a free, $1 million payment to ride
- The guy who has that card in his hands must NOT be able to fiddle with it.
- Difficult thing to do because there is a limited way of getting info from the smart card and info out of the smart card.
- Inputs and outputs must be the only possible options for this to work.
- Criminals usually don't play by the rules
- Particular PINs that transfer information for the smart card.

- Electronic devices can create magnetic symbols
- There have been attacks on this when someone comes up with a brand new version, it is in aide use.
- Academics will get hold of one of those cards and will take advantage of this for weeks and months later.
- They have either tried to observe what happens on the smart card when legitimate things are going on, and by deducing what is going on, they have figured out how to make the smart card misbehave.

Authentication Through Biometrics

- Authentication based on what you are
- Some quantity that can be measured by a person's physical characteristics
- Things like fingerprints, voice patterns, retinal scans, etc.
- A way to authenticate yourself is to present yourself to the system and is capable of measuring whatever the biometric is.
- Biometric converts to binary and compares it with stored values
- We cannot require perfect matches, but they have to be "good enough"

Problems With Biometric Authentication

- Typically requires very special hardware
- More a property of behavior and have some degree of evidence that people have their own typing pattern.
- Give someone a brand new phrase and people you are who you claim to be.
- People give it too much respect when there are a lot more possible attacks
- If we are measuring vocal patterns, it can cause false negatives i.e. voice recordings and authenticating if you have strep throat!
- Indeed, there are NOT biometrics but individual patterns.
- People can customize their browsers in various ways so that their browsers are individual and d
- What happens with your system if it gets cracked?
- If an attacker figures out what your retinal scanner looks in a bit pattern, he can crack both, and you have NO more eyes to look.

When Do Biometrics (Maybe) Work Well?

- Need a clean recording of the device with a clean version of the biometric value
- Whenever someone tries to authenticate themselves, it might NOT have the right orientation and at least you have something accurate to compare against.
- Biometric readers are themselves secure.
- It is gone across the network in a binary form, and if we have everything encrypted, we can feed you back the exact binary pattern.
- Just replaying a pattern and everything gets reduced down to bit patterns

Q. Would we ever hash the binary pattern or can we NOT do that because it is NOT always perfect?
A. There is an issue of imperfection and we want to gran things before they get hashed.

- Hash can be just as good as the binary pattern itself and we can feed you the hash.
- Throw in a challenge and combine them and hash them.
- We have to make sure what we do to read the biometric is a secure design that cannot be interfered with.
  - Fingerprint reader on your smartphone protects your smartphone.
  - If you lose the smartphone, then it loses the importance of the fingerprint reader.
  - You either have the smart card, or you don't have the smart card
  - It is rare for biometrics to be used as the SOLE form of authentication
  - Usually, biometrics is used in conjunction with a password or passphrase.

When Do Biometrics (Definitely) Work Poorly?

- Finding "needles in haystacks" i.e. using facial recognition to catch terrorists in airports
- Are terrorists going to come into your facility and say I am a terrorist so I want you to get a clean reading of me to arrest me?
- Hell no, your photo quality is going to be shit and you cannot tell anything from it.
- Leniency and comparisons can cause you to misidentify things
- Kindly old gentleman gets confused with an actual terrorist.
- You scramble your police and want to arrest the guy and investigate him.
- The minute they took to investigate someone, the more false positives can come up.
- Biometrics won't work well if we have low-quality readings or getting the reading that is supposed to authenticate them.
- Anything you do a cross the network makes this a lot easier.
- How much difference will there be for bit patterns that emerge to determine if there is a match?

Characterizing Biometric Accuracy

- Two kinds of problems
- How many false positives?
- I took a reading, compared it to something, and I was wrong
- Made a positive decision there was a match and I was wrong
- Hw many else negatives?
- I took a reading and i thought it was right, but the result turned out to be wrong
- Biometric reader has sensitivity factor
- You have to be very careful about reading if the user isn't actually the user.
- As we increase sensitivity, it gets more and more likely that there is some error due to something like dirt smearing the camera lens or a minor shaking

- No false positives and no false negatives
- We want to increase or decrease sensitivity to find the perfect balance.
- We pretty much have to come up with a compromise
- The Crossover Error Rate (CER)
- When you read about a biometric, using the readers we have created, the error rate is some %
- Generally, the lower the CER is, the better the system.
- We don't like false negatives very much because the guy owning the phone didn't read it.
- He gets pissed off and doesn't buy your iPhone anymore.
- If this is the doorway that leads into the nuclear missile silo, you don't want to let the wrong guy in.
- Depending on what you are doing, you might want better false positives or better false negatives.

Some Typical Crossover Error Rates

- Data is 15 years old
- <Read the chart>
- Things improve over the course of time
- In some cases, there is a significant probability of error and if we try to protect a $20 transaction, it could be a problem depending on the context.

Biometrics and Usability

- Tradeoffs between false positives and false negatives
- For consumer devices, false negatives are really bad
- For high security devices, false positives are really bad.

Didn't Carnegie Mellon Just Perfect Facial Recognition?

- What if 2% of the time you are wrong?
- Will it be a disaster for your system or will it be tolerable.
- We might want 5 nines (99.9% right)
- Not anywhere close on any of these facial recognition things.

Authentication by Where You Are

- This can sometimes be useful in ubiquitous computing
- Put something on door to your lab that automatically opens if someone who comes to your lab comes to the door.
- Detect a signal and there will be information to detect his actual physical location and we didn't want the door actually open.
- Detect where he is if he is standing in front of the door
- NOT used all that much!
- Is someone in a position where I want to apply a service of some sort?
- Somebody just needs to be there but he also needs a 2nd factor of authentication
- This can be used in a few cases but it is NOT very common.

Lecture 8

Outline

- What doe the OS protect?
- Store data in more or less a permanent form

Introduction

- OS provides the lowest layer of software that users can see
- Close to the hardware and often able to give commands to the hardware.
- Not only have access, but they have complete access
- If the OS is NOT completely protected, he cannot do what he damn well pleases.
- Usually, if you have a security flaw, everything above it is unsafe.
- This clearly means that OS security is absolutely critical.

Why Is OS Security So Important?

- The OS controls access to application memory
- That memory you are using is controlled by the OS
- The OS controls scheduling of the processor
- If it doesn't like your process, it doesn't run!
- If the OS doesn't do any of these securely, your program is totally screwed.
- Almost all security systems assume you have a secure OS at the bottom.
- If you want to secure your database system, you are working on a secure OS.

Single User vs. Multiple User Machines

- Some computers are still multi-users (specialized servers like web servers)
- However, while you may have a single user machine, we could run many processes on it simultaneously
- Code can be downloaded on your machine and in comes the code.
- Your browser does this all the time without bothering to ask you about it.
- In actuality, you are doing things on behalf of a lot of other parties.
- It is increasingly the case that we have embedded machines.
- Computers built into other kinds of devices
- Many computers are built into car nowadays.
- They are doing things on behalf of the human being.
- There is no human user being taken care of here.

Trusted Computing

- Computer that was truly isolated and I don't worry about mistakes, don't worry about security
- If one program wants to use data in a certain way, that is okay.
- We are going to start off talking about OS security via trusted computing
- OS needs to be a secure OS, so how do we do that?

- Build a bunch of code and get a degree of assurance that it has the security properties we want.
- If we install this OS, does it achieve that degree of security on the point of the OS
- Only if that is the OS I am running!
- What if I am actually running a different OS?
- Then I am NOT getting the security guarantee that I thought I was getting
- What we booted should NOT be something an attacker fooled us into
- There have been attacks on boot loaders (things that determine the OS you boot)
- This is a significant problem and if you think about this from a point of view of trust, then you get to the presence of hardware.
- If there is a problem with the security of the hardware we are running on, there is a compromise and we have to change it.
- We are stuck with levels all the way up, so we know that this version of Linux has properties we want and we want to ensure security of what I am running.

How Do We Achieve Trusted Computing?

- From the bottom up
- We need hardware we can count on
- Ensure the boot loader is safe!
- Then, we can have the boot loader to ensure it is the right operating system.

TPM and Bootstrap Security

- Step we need to worry about first is that we have to ensure we are running the right boot loader
- Trusted Platform Module (TPM)
- Main purposes is to give you a high degree of assurance that your OS was loaded with a particular bootstrap loader
- If an attacker changes the hardware, you are kind of screwed
- Intended to be tamper proof, but there is no such thing!
- There is tamper-resistant hardware, but it is IMPOSSIBLE to create a tamper-proof hardware.
- Attacker could figure out how to misuse hardware using an extremely sophisticated knowledge of chip design and equipment that at a microscopic level would slice off a layer of the chip
- After several months of effort, he broke into the chip and this more or less destroyed the chip in the process.
- Prove that OS was booted with a particular bootstrap loader
- Use cryptographic techniques to ensure that it is the proper bootstrap loader
- Hardware takes cryptographic hash and if it is right, then that hardware will take that code and run it as the bootstrap loader.
- This hardware does a few other things like providing secure key storage and crypto support

- You can use the keys via the crypto here
- Good way to save your private key
- TPM will do signatures for you but you won't be getting your private key back out again!

Q. Does this mean you could lock someone out of their computer?
A. If you could change what is bootable, then yes! In particular, if you alter the OS so there is no acceptable OS is on the machine, it won't boot anything!

- You could always go to a removable device and boot off a flash drive. Hard to really keep someone out this way but pain in the ass.

TPM and the OS Itself

- We mostly care about if we got the right OS
- The bootstrap loader can play the same game with the OS
- Hash cryptographically the OS and check to see if it is the right OS
- Store a signature for the OS that you want to boot
- We have been told you want to boot OS A and get the code for that version of the OS and run it through the cryptographic hash
- Done through TPM's hardware and then check if the hash is right.
- You have the version of the OS you want to run.
- We are all set to go and let's load the OS and get going.
- Once you have gotten to this point, you know for close to a fact that you re running the correct version of the OS.
- Your OS is sitting in memory and if there is some way for the attacker to go into RAM, then tough luck.
- We checked at boot and the hardware doesn't check again.
- We need to guarantee that we booted to the right thing!
- If the OS really cared, it could use TM to verify applications as well.
- See if that's really the right version that should run.

Transitive Trust in TPM

- We trust the app because the OS says it was trustworthy.
- The bootstrap reaffirms it is trustworthy
- The idea extends downward because you have believe the TPM hardware is doing the right thing.

Trust vs. Security

- TPM doesn't guarantee security; we associate TPM with trust!
- It just guarantees you are running the OS you thought you were using.
- There is a little bit of protection with tampering against software, so it is NOT helpful against SQL injection or phishing attacks

Status of TPM

- Hardware widely installed
- Macs have this feature but it is probably NOT used in the standard MacIntosh

SecureBoot

- Microsoft decided to build a similar thing and called it SecureBoot.
- If you the hardware manufacturer wants to run Windows, you need SecureBoot and you want your hardware to accommodate this.
- When Microsoft first proposed this, did they want to enforce you can only boot Windows?
- It is possible to build Linux systems instead.

Some Details of SecuretBoot

- Microsoft insists that you support these features and it only boots systems with pre-arranged digital signatures

Hardware Security Enclaves

- The reason there is a difficulty breaking into the terrorist iPhone is due to security enclaves!
- This hardware is supposed to be tamper-resistant.
- They are looking at it for login security.
- When you provided a password to open up your iPhone, it takes that password and a tamper-resistant key that could NOT be extracted.
- Used if you successfully logged in to encrypt remaining data on the phone
- Compare the times you logged in unsuccessfully and produced a number.
- Lock up the phone and prevent access to that guy I was giving it to.
- Apple is NOT open about their security enclaves
- People at Apple know how they work and some people like the NSA have an idea of how it works.
- The security enclave is supposed to say you only have 10 attempts to login before you get logged out.
- Somewhere else, there will be a counter the keeps track of when to throw out the key.
- If counter is secured in RAM or flash memory, they could say that every time we make a log in attempt, we would still be checking on every attempt.
- We could try it as many times as we want!
- Does the security enclave send the key out to main memory and run out the algorithms in main memory?
- See what the key is and get around this whole process.

Authentication in Operating Systems

- One issue that has come up a lot has been a government policy issue
- It says “Every company that deals with software, and if the DOJ wants you to decrypt this encrypted data, then the company MUST do so.” This implies if that bill passes, it would be illegal to create a security enclave and we cannot give you the key.
- It is locked away in hardware that we ourselves cannot access.
- You have to bypass the security built into the system to decrypt data I wanted to encrypt.

- If the bill passes, people know there has to be a way in somehow and they will find a way!
- This would mean we would NO longer have secure systems.
- There is a great deal of push back and unhappiness in this bill from the high tech company.
- There are people in charge of the NSA who say we need secure cryptography who argue about this!
- They don't enforce that there must be a way to break into everyone's cryptographic operations
- Certain people in law enforcement claim they will never be able to stop crimes unless they can break encryption.
  - If this bill happens, you're kind of fucked.
  - Not too many people who are computer security field favor this for obvious reasons
  - Reiner sure as hell does NOT favor this, I don't fucking favor this!

- Back in the 1990s, official wanted to push a cryptographic chip called Skip Jack. Done in such a way that your key is embedded into the message.
- Embedded into an encrypted way and take a warrant to each one.
- Go to your encrypted message and read the encrypted data.
- Savagely attacked by everyone in the field of cryptography.
- Criticized everything about this idea

W 4 Dis 4-22-16

- Classical examples of digital authentication
- Passwords
- Users pick dumb and/or trivial passwords
- Companies use very simple passwords and they reuse passwords
- You have to be who you say you are!
- If I know Person X, there are a lot of characteristics I can identify, but passwords are flat and it isn't too feasible.
- Doesn't make too much sense if you think about it.
- Identify who you are and do this on a computer.

Other Physical Identification Methods

- Relies on trust, but trust could be broken!
- These all have cyber analogues!
- Identification by location
- If you are at a top secret military base and you only you have access to it.
- Because you got there, you are clearly authenticated, so we don't have to re-authenticate you.
- Physical constraint
- Identification by knowledge
- Challenge question
- Identification by recommendation
- Key exchange with a trusted source

Authentication Mechanisms

- Something you know
- Passwords
- Something you have
- Smart cards
- Tokens
- Something you are
- Biometrics
- Somewhere you are
- Military base
- Apple headquarters

Passwords and Single Sign-On

- MyUCLA uses single sign-on
- Shibboleth system
- If others use the same authenticated machine, how do you address this?
- Logout when you are done!

Handling Passwords

- Encrypt the password and then add salt (a random string)
- This makes it harder for dictionary attacks to crack the password
- The one-way hash will always be the same thing

One Way Functions

- If you use the salt, it will be a breach and these data breaches are worrisome because it is hard to keep track of passwords.

Q. Does logging off not clear cookies?
A. It depends on how the server is set up; once you log off, it should destroy the session on the server side.

- A lot of computers, especially banks, keep fingerprints.
- If you go to a new browser, there will be various forms of authentication i.e. messages like we don't recognize this computer.

Q. Why is it okay to keep salt in plaintext?
A. It just tells you it isn't going to be the same as other users who have the same password.

- If I use a password 1234, the value is hashed to the same result, but we add the salt, which adds only a small amount of information

Q. Can the salt help you in anyway?
A. It is only helpful if the salt is the same between two users and the hash is the same between users, meaning it is the same password, but this is **highly unlikely**.

- If the salt is the same, you can do one attack, which is obviously something we want to avoid

Password Management

- Forgotten passwords
- Don't send it directly to them unencrypted because the hacker can exploit this.
- Generating new passwords

Limit Login Attempts

- Passcode slows down on iPhones
- Watch more closely
- Watch the user's actions more carefully and make sure they aren't doing anything suspicious or fishy.

Q. How did the FBI get into the iPhone?
A. FBI was trying to do brute force but there was actually a backdoor to it.

Wireless Networks and Passwords

- You can use Google Translate to translate a page.
- You can still see stuff
- When you go to a website, it does a DNS request and downloads a corresponding certificate.
- We go into some sort of bank website i.e. Chase or Bank of America.
- Is there an attack that a rogue network can do?
- Man in the middle attack.
- Sniff an entire transaction and see passwords and all your info.
- Use HTTPS and you could generate a certificate that will be trusted and have the green lock
- Lots of certificates for common misspelling
- Literally replay and run your messages.
- Because it has your certificate, it can decrypt all your traffic.

Characterizing Biometric Accuracy

- False negative: If they do match, they are from the same person but you reject
- False positive: If they do NOT match, but you accept it
- Face scanning is a good example of this

Transitive Trust in TPM

- You trust the app because the OS says to trust it
- Today, we have mobile phones, laptops, and the cloud

TPM and the OS Itself

- OS can request TPM to verify applications it runs
- Make sure your data or program is NOT tampered with
- Provider is going to have some software or agreement with that customer to share the source

- Source and executable will be signed and the user can use a TPM module.
- You as a remote user can request some verification that the software we say they are running is actually running.

Remote User Authentication
- Passwords fall under “What you know”
- Public key crypto falls under “What you have”
- Generally considered stronger and also easier
- Use SSH keys to get into Deter and generate that as well

Process Authentication
- Let’s say I am running as PID 5001, what would I change it to and how would that be advantageous?
- Change PID as root and the program could run as root in order to switch your password.

Reusing Pages
- Theoretically, they could share a space and one process can potentially read secret data.
- The keys can be read into memory and stored there.
- Use this to decrypt a lot of sensitive data

Strategies for Cleaning Pages
- Linux uses a “Don’t Bother” strategy
- Windows uses a “Clean pages in the background” strategy

W 5 T Lec 4-26-16
- Midterm
- Covers reading material and lecture material that ends on Thursday

Authentication in Operating Systems
- If we build on an insecure basis (OS), then it adds up at the higher levels
- In all security measures, we care about the person asking us to do something
- We have to authenticate each system call, otherwise, we cannot perform proper access control
- Done typically through logins or 2-factor authentication
- User ID + password
- This is the users who we expect to be
- Given a primal process that we work with that is instantiated to a command line shell or GUI
- Process runs on your behalf
- That process then takes on your identity and receives your full set of privileges
- Requests resources of various kinds

- Send a message across the network.
- These will be tagged with your identity

In-Person User Authentication

- Use passwords or maybe biometrics
- Get primal authentication!
- This user just started using the system and it has to be tied to your identity

Remote User Authentication

- Done via password
- Once in a while we use public key crypto
- ssh to a machine: provide info to the public key from the machine you are coming from (you)
- This is how the remote machine authenticates you
- Sometimes it is just a process i.e. logging into a web server
- Log into your FB account and authenticated by your process-level security
- They could set up on every computer a user account known to every OS
- Of my 6 million users, this is the particular person who has this permissions
- Too much of a pain in the ass to maintain things at the OS-level
- Instead, we try to have accounts at the process-level (FB level)
- At the level, is this the appropriate FB user? Is it the right software entity among the several FB users that we work with?
- Suggests two different sets of users at this site.
- What the application level knows about
- What the OS level knows about

Process Authentication

- secure shell into a remote machine or when we have a login screen for our user
- some kind of process tied to our identity
- Usually kept as simply numbers
- More or less arbitrary number
- What is your username? What is your real name? Where do we store your home directory?
- At any rate, the process gets created for you and the OS creates a process control block with a particular kind of structure
- Every process running Windows, Linux, Mac OS has a field that says this process is associated with this user.
- The user cannot just arbitrarily change this field, since it can only be changed by very special circumstances.
- Nobody else can change it except the OS
- When we are running the process represented by the GUI, we will be running other processes

- He wants to start up a new process with the program associated with that icon
- Start a new process based on whatever the 1st parameter is
- He creates a new process and new process control block
- Has a field for who owns this process
- What it normally does is for whoever owned that process, it gets copied into the ownership field in the new process control block.
- All the child processes share the identity with their old parents
- System calls require privilege
- Root users in Linux systems
- Allows you to say a process belongs to user X and change it to user Y
- Go into the process control block and change the identity
- This instantly changes the access permissions of that process, and whatever is used to access goes to the new identities

For Example,

- Process X wants to read file Y
- The OS says when we make the open call, we are going to pass the info that this is associated with user Bill
- Send a parameter to the call that is made further down into the stack for further procedure calls
- File system gets way down there and somewhere, we have info on the disk that we need to compare ownership and access control for this
- See if they match and let it happen
- All the other systems i.e. rpc, requests to memory, or more or less treated in the same way.
- Alternately, read the data into the process control block and slap it into a parameter

Protecting Memory

- Generally speaking, OS does NOT let you reach down into memory i.e. stacks, heaps and fiddle with data as an ordinary user
- In kernel mode, you cannot make things change
- Only the OS can change it
- Relatively safe from that perspective.
- You can be pretty sure that everything that gets done is tied to the original requestor.
- Make sure that the right thing happens
- Other resources the OS deals with

Q. How does kernel extension work?
A. Gets added to the system only by a privileged user. He can do anything he wants in the system. Gets hooked into a place that is expecting to have extensions. If the kernel extension is a bad piece of code, you are kind of screwed.

- When you get an update or extension to Windows, it is signed by a public/private key pair from Microsoft
- More open source OS may not be checking as carefully (Linux)

- One of the most important resources is memory
- RAM memory in this case

What Is In Memory?

- Executable code
- If you can improperly change the executable code, you can make them do anything.
- Copies of permanently stored data
- Any secrecy and exclusivity associated with data wants to be ensured that not anyone can go look at it.
- Integrity issues too
- Temporary process data
- Stack - keeps track of what procedure we are in and what are the locally allocated variables in that procedure
- Make sure we have integrity of that information

Mechanisms for Memory Protection

- We logically divide what is going on in our computer into processes
- We want it so that the OS looks at every set of processes like it has its own set of local resources.
- If we can keep this separation strong, we can protect our data from other processes that shouldn't be able to access or data.
- Virtual memory techniques were NOT built for security purposes.
- Built to allow more flexible systems.
- NOT designed specifically for security, but it turns out to be really great for computer security.

Paging and Security

- Memory is divided into page frames
- Specific to the OS we are working with
- Every process has an address space in logical pages
- These pages are logical, meaning you can map any logical page into any page frame
- Processes cannot use logical pages because that is just a concept
- If a process makes use of logical pages, we can use it in a page frame.
- With a whole bunch of processes running, what happens in a modern system is that the logical pages are scattered throughout the page frames.
- The expectation si that every page frame is its own entity that can be protected
- Just because you have access to Page X doesn't mean you have access to Page X+1 or Page X-1

Protection of Pages

- Translate to page frames to store the data
- All addressing that a process can actually say through any mechanism has to go through the page table

- Always uses a translation process in the page table
- Gives us a point of control
- We can ensure page table controls translation
- Unavoidable at the hardware level
- Built into the hardware since the wires lead here
- To protect page frames, we have to set the page tables properly!
- Process X's page table only contains translations to pages that process X can see

Page Tables and Physical Pages

- Each of these represents 4K of RAM
- Take the logical pages and find somewhere in the physical memory to store the data
- Take the page tables and choose more or less and random
- Take a bunch of these and see where it maps to.
- Note: they need not be contiguous, but they only need to be somewhere we can translate to.
- You can store some of these elsewhere; typically, on a disk drive
- There can be some logical page frames we aren't using.
- These would just say we wouldn't bother until he uses it.
- Do the same thing for process B, and A and B have various pages are mapped to page frames scattered across memory
- Any instruction of Process A gets checked by the hardware and gets only the translated page frame
- Don't fiddle around with the translation
- Process B does the same thing
- **You can't even name each other's pages**
- Impossible to do this, and makes it very difficult to break this security

Q. Page table, only the OS can access?
A. Yes, the page table is kept in a place where the OS can access. The OS has to be very careful and this relies on the fact that you cannot fool the OS.

- The OS says it is either a page frame you got, or you are asking for a new page.
- The OS can then choose a free, unused page frame and use that one that doesn't belong to someone else.

Security Issues of Page Frame Reuses

- Same RAM, same page frames
- May be reusing the same page frame in the next second
- At any given process, the process is going to relinquish that page frame
- It then becomes free and can be assigned to another OS for that process
- When the old process has that page frame, it then says I am done and we give it to someone else.
- What happens to that old data?

- When we give the page frame to the new process, perhaps it can read al the old data!

<Bookmark>

Process B

- Deallocates a page
- That page is no longer allocated and what we used to have allocated is listed as free.
- Another process can use that page.
- Process A says I need that page and we can get a free page.
- This page frame is NOT used by him anymore and whatever data we are using is still there.
- If process A reads that data, it can be used for process B

Strategies for Cleaning Pages

- Don't bother!
- This is the Linux strategy
- Read whatever data is lying around in that page.
- When process B says we are done with that page, we can zero out every byte of data in that page.
- Put 0's everywhere
- Takes time!
- Whenever you deallocate a page, you write a bunch of 0's into that page.
- That is the moment at which I really don't care.
- If I have a free page frame, big deal!
- The only time I have a problem is when I reallocate the page.
- This means that when Process A wants a new page, he gets the reallocated page of Process B and do that 4K worth of copy 0's in
- Another paradigm is that people often don't use those pages, so why don't we wait and see when we run those instructions
- Give it to Process A, but put a little mark on the page tables saying this is a dirty page.
- If he ever issues an address on that page, stop, zero out all the addresses on that page, then let him use it.
- Pay a penalty for zeroing it out when he decides to use it.
- In the moment between zero on deallocation and reallocation, we find we have spare time when there is NOTHING to do!
- Wait until we get to a point where this is nothing else to do.
- Wait for a message to come in and why don't we come to our page frames and see what we have got.
- Let's user our spare time to zero out these pages.
- We can then give him one of these pages and this gets done in windows.

Special Interfaces to Memory

- This sounds great!

- It sounds like we have perfectly provided RAM
- This is true assuming our guesses our correct
- Modern processors go through the paging hardware.
- There are other ways to get to memory without using the processor
- The BUS that hooks things up can put instructions onto the bus and go to a physical RAM address and tell me what is in it.
- If you have this kind of interface, you bypass these kinds of page tables.
- If an attacker can get to this interface, they can read the physical memory and write physical memory
- Lose guarantees you got through memory mechanisms
- Bad ideas built into computers
- Known as the **firewire interface**
- On smartphones as well!
- This is a terrible security mechanism and why isn't there any access control mechanism at all.
- This leads to an **evil maid attack!**
- Still powered up where no one can log in and unless they can log in, no one can authenticate themselves.
- Evil made comes in with a special device that goes through the BUS and it can read everything out onto your memory
- There is nothing you can do it.

Buffer Overflows

- Assuming virtual memory protection, we have pretty good protection of our memory accesses
- This sounds good because we can't change process ID and the process cannot get to data it shouldn't have access to.
- If it is my process, it can do anything I can do
- Copy all contents of directory A into directory B
- Expect process to make a byte-for-byte copy into directory B
- Go from A -> B, and avoid copying to C
- We don't expect anything should happen to directory C because we didn't specify that in our instructions
- How does the OS determine what should or should NOT be done.
- Peter can access directory C, and this process can do whatever I am capable of doing, and it is for what I could do.
- If someone can get into that process and change into those instructions, wipe out everything in C and we can have permissions to go to directory C
- It would be very bad if someone was capable of changing the code of our running process.
- It is a very common cause for compromises in OS
- Due to a flaw in how OS handle process inputs
- We have stupid programming languages like C that lets you do things you shouldn't
- We have stupid programmers who don't check their inputs

- If they only checked their inputs properly, we would never have buffer flaws
- We have a problem regardless of how you spin it!

What Is a Buffer Overflow?

- Requests input from a user running the process remotely.
- I am getting a bunch of info coming in from messages and it should be from whatever website I render from you
- You are going to be reading a message buffer and you will be looking at a device and we want to translate the data
- Build a buffer and see if we get some data in and we are going to copy it into the buffer
- Read all the data and if we didn't do things right, we need to see how big the buffer was that we built.
- We will allocate 100 bytes if asked, but what if he gives you 200 bytes?

For Example,

- Lecture 8, Page 34
- If the user entered > 32 characters, you are rekt!

Well, What If the User Does?

- gets doesn't specify string length
- First 32 bytes go into the buffer you allocated
- Dynamic variable so it is on the stack
- The stack, in addition to keeping dynamic variables, also keeps track of calls you make
- Because this particular buffer is the first thing you allocated, it is quite close to other records.
- Very close to info about what you should do when you are finished with a particular routine.
- After filling 32-bytes, I am going to keep copying bytes
- Page in memory belongs to my process and I can access that page of memory.
- Therefore, I must be able to write that page of memory
- That page frame belongs to me and as far as hardware and software is concerned, it writes it!
- After writing 32 bytes, it goes to the rest of the bytes.
- They are going to go into the next few adjacent locations in the page frame.
- You can change your return value and you go to a different code you thought you were.
- If you are really clever with the buffer overflow, you can specify the piece of code you want to execute.

Why Is This a Security Problem?

- The attacker can cause the function to "return" to any arbitrary address.

- The attacker cannot do anything the user of this process wasn't allowed to do.
- Anything the user is capable of doing can be accessed with a clever buffer overflow attack.
- The attacker can get here and fiddle with privileges he has gotten.

Is That So Bad?

- Yes
- A media player can write configuration and data files
- In most OS, unless something special is done, except for a tablet or laptop, there is really only one user that can do anything and practically anything is available to that user.
- Anything on that computer can belong to that attacker.

The Core Buffer Overflow Security Issue

- These programs run under your identity and maybe it is okay for you to access the data, but you don't want to give access to that data for everyone.
- Give this with downloaded programs or playing around with webpages.

Using Buffer Overflows to Compromise Security

- Fiddle around with code that is really nasty.
- Could have done nasty things without dealing with buffer overflow.
- Just because you have a piece of code from the web server doesn't mean we can limit what the code is supposed to do.
- Jump out and do whatever the code wants us to do.
- Buffer overflow lets you get around other pieces of security on code you have downloaded for someone else.

Effects of Buffer Overflows

- The attacker is looking for a buffer overflow in a root program
- He can then do anything he wants to in a system
- Can have buffer overflows in stack overflows and heap overflows
- Gains a foothold on your machine
- Before they can do anything else, they have to be able to run code on your machine
- Until then, they cannot get very far.
- Merely running code on your machine is NOT everything they need to do, but it is an excellent starting point.
- Gives them an ability to run an arbitrary piece of code on your machine.

Stack Overflows

- Most common kind
- You would like to jump to another place instead and do what the attacker wants to do!
- NOT always the case! Sometimes, the parameter will fiddle with parameter values or return values.

- The buffer overflow generally says to get me to somewhere else.

Heap Overflows
- Keeps track of bulk data
- There are buffers there
- When it overflows, it usually prevents modification of return addresses

What Can You Do With Heap Overflows?
- Alter variable values
- Edit linked lists or data structures
- If you have indirect function pointers, if you run into the following condition, we can force people to jump to a particular list.
- Harder to exploit than stack overflows.

Some Recent Buffer Overflows
- Cisco Cable Modem software and Cisco Adaptive Security Appliance
- Pro-face GP Pro-EX industrial control system
- Xen Hypervisor
- A heap overflow
- glib
- A bad place for a buffer overflow
- This is used EVERYWHERE!
- A vast amount of software was compromised

<Bookmark>
Fixing Buffer Overflows
- Write better code (check input lengths, etc.)
- Programmers need to check input and this should always be checked
- One way to ensure this is to always check your inputs.
- Use programming languages that prevent this
- Don't use C or C++!
- Structure i.e. length goes with a data structure, and we can have the programming language automatically check this
- Java does this for you
- Java is a virtual machine itself and some Java implementations have had buffer overflows in the implementation of Java
- By using a flaw in the way that the Java is implemented, it will have problems
- Add OS control to prevent overwriting the stack
- Put things on different places on the stack
- The attacker should NOT be able to figure out the return pointer easily
- Randomize where you put things on the stack
- Microsoft's ASLR does this
- Don't allow code to be executed form places where buffer overflows occur

- The attacker wants to jump to a new place so we need to protect against this.
- First, we write the piece of code we want executed and jump to that piece of code.
- Why am I going to execute that piece of the stack?
- The only time I execute this is when we perform a buffer overflow.
- We should say this is NOT executable, and it has been determined that self-modifying code is VERY BAD!
- Nowadays, we don't have the ability to write self-modifying code.
- The compiler says we take a piece of data and we write an executable somewhere.
- An ordinary program doesn't create a new piece of data and executes it.
- Windows DEP prevents people from having both execute AND write permissions!
- Not all OS have these futures, and when Microsoft added these features, people were complaining and they shouldn't have been doing this, so Microsoft had to put some controls into their systems.
- We could run it without DEP and now we would have programs that were vulnerable.
- Attackers get to become more and more sophisticated.
- Attackers cannot introduce new code that he will run.
- The attacker can still do a buffer overflow and cause you to jump to a new place with executable code.
- We have a program we are running and what would be wrong about jumping to that piece of code?
- Return-oriented programming
- Typical program has a library associated with it i.e. glibc
- What if we wrote many return pointers?
- Each one would jump to some little piece of code in glibc, so why don't we jump to an arbitrary instruction in the middle of the routine.
- It is a whole lot of work by hand to run through C libraries and do what you want to do.
- We can build a compiler to do this and the compiler should understand all the code available in the C library and translate that into the instructions you want and run through the glib in order to build your program.
- This sounds like sci-fi but it really has been done effectively!
- Some actual attacks make use of this weakness.

Protecting Interprocess Communications

- Special privileges where processes can have different levels of privileges
- Messages
- Semaphores

IPC Protection Issues

- Certain things to worry about that are theoretically impossible to deal with.

- What we are worried about is if one process is using IPC to steal data from another
- How could process A steal from Process B

Message Security

- Lecture 8, Page 48
- How could B get the secret from A by just asking

How Can B Get the Secret?

- If the OS is careful about who makes system calls, B cannot pretend he is A.
- He could break into A's memory but this would be handled by page tables
- A message will be going through memory and it will be set up at a buffer.
- Cannot just create a message out of thin air.
- The OS controls the entire process
- Can we eavesdrop on it?
- The message will be copied from process buffer and there will be optimization
- If we cannot get into the message, page reuse is an area where we have to be a bit careful.

Can an Attacker Really Eavesdrop on IPC Message?

- This is for messages, what about sockets and other mechanisms?
- They rely on processes wanted to send messages and the OS will fiddle around and copy it from one place to another.

So When Is It Hard?

- If there's a bug in the OS, it can be used.
- Periods when data is NOT in the OS you trust.
- In this case, it will be compromised at that point.
- What if Process A has a secret, process B wants to know the secret, and Process A actually wants to reveal the secret

Distributed System Issues

- What if RPC is really remote?
- Make it look like a local procedure call.
- The hard part is authentication and it has got some information in its messages.
- There is something in those messages that indicate who made that procedure call.
- Why do you believe it?
- This procedure call should come from the owner of Process B rather than someone else.

The Other Hard Case

- Problem we have to solve and set it aside from how to authenticate messages
- Process A has a secret and Process B does NOT have this secret
- Process A wants to tell secret to process B
- The OS has been instructed to prevent hat
- Part of Bell-La Padula
- Do this in any mandatory access control policy
- Mandatory policy and the OS must make it happen.

OS Control of Interactions

- OS can “understand” the security policy
- Process A should package up the info and put it into the message.
- Based on this label, we need to prevent this from happening so we can look at the label and say “NO! That shouldn’t happen!”
- You can regard any of these as a communication mechanism.

Covert Channels

- A tricky way to pass information
- Does NOT require mechanisms for passing info, we are going to pass it in using some other channel
- Requires cooperation and prearrangement between sender and receiver
- Party in between must NOT know what you are up to.
- You can have processes actively preventing you from running that security policy

Lecture 8, Page 56

- Secure building that contains secrets inside.
- No network connections whatsoever.
- What if you get a spy into the building?
- He cannot just remember the secret; he has to communicate it to someone else WITHOUT flash drives, network connections, telephone
- He has a friend with binoculars
- He turns off some lights and runs around to all the floors and turns on and off more lights
- Gradually, the secret filters out the info but the secret has been communicated!
- This is a covert channel!

Covert Channels in Computers

- One process “sends” a covert message to another
- Sometimes between computers
- Disk activity
- I know the secret via 0’s and 1’s, and I tell the other guy that I am going to make the disk active for some period of time
- Watch disk activity bit by bit
- Page swapping

- Ensure that a whole lot of pages get swapped by my process
- Time slice behavior
- Communicate bits with time slicing procedures
- Peripheral device
- Turn on microphone and see if it is in use!
- Limited only by one's imagination!

Handling Covert Channels

- If you know the covert channel's details, it is trivial to prevent it.
- If they are using disk activity, you will get a constant amount of disk activity whether or not process A sends a 0
- If you don't know the channel, it goes between hard and impossible
- Even if you become a security problem, you probably don't have to worry about a covert channel
- Uncommon problem!

Stored Data Protection

- We often save data for a long period of time on a file system.
- If the OS supports multiple users, typically, they will share the device.
- You can address the access control by using read/write access control bits
- In addition, you have to worry about the fire-wire issue.
- Way of getting to a device without protections that the hardware protected.
- Perhaps, we can access things in a way that does NOT go through the OS

Encrypted File Systems

- Often use cryptography!
- Data on the disk can be accessed on many different ways and we need to avoid using the raw disk without data getting in the way.

An Example of an Encrypted File System

- Lecture 8, Page 61
- Caesar cipher
- User is supposed to use the key to decrypt the data and this is what we are talking about.
- Issues for encrypted file systems:
- When does the cryptography occur?
- Where do we get the key?
- What is the granularity of cryptography?

When Does Cryptography Occur?

- Transparently, when the user has to ask, lets decrypt the file.
- The hardware of the disk drive can do decryption for us.
- In the OS, we can figure out if it is encrypted and then decrypt it.

- File system can say as we go up through the file system, it is built tup
- Explicit user command
- Or implicitly!
- Where does it exist in decrypted form?
- There is often transportation between disk and the file

Where Does the Key Come From?

- From the user's keyboard?
- Not the best idea.
- Somewhere in the file system?
- Don't want them to get the key but we are concerned that people can access the disk drive holding the file system.
- Stored on smart card?
- System has to provide the key.
- In the disk hardware?
- This leads to the question of if someone is accessing the disk directly, can they get that key?
- Sooner or later, we need the key available to decrypt the data block.
- Where are we going to put the key?
- If we put it somewhere, how long are we going to keep it there.
- If it gets swapped off the disk, are we going to have a copy of the key sitting somewhere on the disk drive.
- We can find the plaintext version and that is NOT good.

What Is the Granularity of Cryptography?

- Are we using different keys?
- Can we choose different keys for different files?

What Are You Trying to Protect Against With Crypto File Systems?

- Unauthorized access by improper users?
- Why don't we just use access control?
- We could have some serious problems with data we care about.
- We might as well simply rely on access control
- The OS itself?
- If you don't trust your OS, you are bit in trouble.
- What protection are you really getting?
- Sooner or later, he will be dealing with data, and who deals with RAM?
- What have you gained? Nothing!
- This isn't going to help you one little bit!
- Unless you're just storing data on the machine, then you're getting some protection by having the data encrypted.
- You can encrypt the data before you give it to the cloud provider.
- Never gets decrypted when you are in the cloud.
- When it comes back, then it is decrypted.
- Data transfers across a network?

- Compromised then, why don't we encrypt it before we send it across the network.
- Someone who accesses the device without the OS.
- Is this a realistic threat in my environment?
- If I am running my data center, it may be unlikely that an unauthorized party can get to your disk drive.
- Your laptop computer sits around all the time and someone may very well be able to access that drive.
- **This is when you want to use file system encryption**

Full Disk Encryption

- "Encrypt the whole goddamn disk" - Peter Reiher
- Gets encrypted/decrypted a it enters/leaves the disk
- The primary purposes is if the disk is stolen or improperly accessed, then we will get a disk full of encrypted data with the key.
- Most important for portable machines.

HW vs. SW Full Disk Encryption

- HW advantages:
- If done in hardware, it is fast and it is totally transparent and we can siphon the data.
- Hardware is set up very easily and it is easy to setup.
- HW disadvantages:
- More expensive than software (not that much, though)
- 10% cost on top for he normal cost
- If you have a serious problem, this is still reasonable
- Only available on a small set of disk drives
- Issues of backwards compatibility
- Move forward in various hardware areas
- SW Full Disk encryption is much more common

Example of Software Full Disk Encryption

- Microsoft Bit Locker
- Unencrypted partition that holds your bootstrap
- The full disk encryption isn't encrypting it.
- Bootstrap loaders aren't secrets. If anyone wants to know, integrity is a big issue.
- If you have secure boot in place, integrity isn't too often a problem.
- Key can be stored in special hardware or we can use TPM or a USB drive
- This is done in software, so it brings blocks of hardware and we run AES to decrypt it
- Performance penalty
- Microsoft claims "single digit percentage" overhead
- Found to be 12%
- If you keep moving on to and off of disk, you may see a penalty in software unless the caching works properly?

Q. Why does it make more sense for laptops and tablets to use hardware upgrades?
A. Hardware is built for upgrades in the market.
Q. Can you yourself improve this?
A. Unless you have a disk drive by the original manufacturer, you wouldn't be able to do it, so you probably have to buy a laptop that has it built-in.

- Limits to what you can get.

Lecture 9: Network Security

Some Important Network Characteristics for Security

- Degree of locality: LAN or the Internet
- Media used: wired vs wireless
- Protocols used: IP, below IP, above IP?

Degree of Locality

- Some networks are very local i.e. Ethernet
- If we are talking about protect LAN, we know how far it goes because we know how far it stretches
- We also have a small # of users and a small # of machines
- Maybe a couple of hundred users at most, and often, not always, users have a common goal.
- If we have a small business, we would expect them to do business type activities and they would be relatively cooperative.
- They are your employees, so if YOU go out of business, they lose their jobs.
- There are other networks that are not-local like the Internet!
- Anybody can get on the Internet, all the people are on the Internet, so not a whole lot of guarantees on the Internet.
- The Internet backbone is shared by everyone!

Network Media

- Sometimes, they are actual physical wires, cables, telephone lines
- You can physically protect them i.e. using cables buried deep in the Earth.
- Local Ethernet is running through your ceiling and walls
- If it is very important to secure the wired network, you can make it hard for attackers!
- Nobody can come and put a pair of banana clips and eavesdrop.
- Other networks are satellite or radio links
- Signals traveling through the air are NOT capable of having the absolute limit of where the signal is.
- If we have 200m, it is an average rough guess and there have been people who have made 802.11 networks work over miles!
- Putting access points into my building doesn't mean it is the only place I could get that signal.
- Cannot protect that signal by saying my building's walls are secure.

Protocol Types

- TCP/IP is most used
- Covers only some layers of the networks
- Linked layers and physical layers that have nothing to do with IP
- Further, there are layers of the network stack above.
- Anything happening at a higher level is not available to TCP
- Lots of security protocols i.e. routing protocols like DNS or LDAP
- Network management protocols where we should set up our switch.

Implications of Protocol Type

- The protocol defines a set of rules where communication should occur
- Try to do the same things we assume we have ordinary, uninterrupted, save communications with partners
- Set of rules defined where people who are running the protocol
- The rules have not been set up to be totally complete
- If people are trying to screw each other, implementations can also NOT be perfect.
- Sometimes we must do this and this doesn't mean the implementation you have gotten is a perfect implementation of those protocol rules.
- It may not follow all the rules to perfection

W 5 R Lec 4-28-16
Sample Midterm

- Multiple-choice
- Short Answer
- Everything lectured on up to today and anything in the readings
- Potentially on the test, so check out all the reading materials
- Anything that appeared in labs will NOT appear on the test
- Closed-book, closed-notes

Threats To Networks

- A lot of problems we see in computer systems today is closely related to the fact that they are in a network
- Wiretapping
- Impersonation
- Attacks on message
- Confidentiality

Wiretapping

- Info is moving across the network and an attacker can listen to the electronic network and hear what is going on or influence it.
- Some wiretapping is **passive wiretapping**
- Listening to what is going on
- **Active wiretapping**
- Change traffic material

- **Packet sniffers**
- Work at high semantic level and understand formatting of packets in the particular network.
- Provide info about packets moving across the network.
- Used on a broadcast medium, most commonly a wireless medium
- Ethernet
- 802.11 (wireless medium)
- Most messages are not intended for you and you can see if they represent packets or not.
- Wiretapping on a wireless network is very simple!
- Go to the physical place and put up an antenna
- Normally set up in a way to get signals intended for your machine

Impersonation

- Used to connect up different machines and take some action based on coming from a different remote machine.
- Attacker who is NOT that user can achieve his goals by pretending to be someone who does have those privileges.
- Only authentication that is possible is info carried in packets
- No keyboard
- No biometric reader
- Send them across the medium
- This is a problem we have to deal with.

How Do We Determine Packet “Identity”?

- Packets have an address in it.
- IP packet except when referring to a very low level.
- A bunch of bits and anybody could put any source address that they want to.
- Linux machines do this and you should NOT regard the fact a particular IP address occurs in a particular packet.
- We are more likely to use cryptographic techniques like public key cryptography
- These could only have been created by the legitimate user and we tend to bootstrap this whole process using public key cryptography
- Using his private key, this message couldn’t have been created by anyone else.
- Typically try to set up a symmetric key and the guys sending and receiving packets should only be able to read it.
- We cannot encrypt every bit of a packet
- Internet routers cannot forward encrypted headers
- They need to know what the destination address is in order to deal with the packet properly
- All packets have on the outside: an unencrypted header
- This means that at some level, we are going to have untrusted information

- Dealt with by Internet routers and our own equipment.
- We can then check the checksum and see if it is okay.
- Under certain circumstances, it will now be too late.
- The way we determine a packets identity is we just know.
- We don't ever 100% know
- If there is a wire in between two machines, a packet comes in across the wire, where it could have come from
- Probably the other machine!
- Unless there is wiretapping though!
- It had to go through this machine if it was connected via wire
- What is purported to be in this packet is really that.
- Assumptions are usually good (most of the time)

Violations of Message Confidentiality

- Misdelivery: if message goes to the wrong place, then the person can read the contents of that message.
- If message is going to multiple hops, there is a possibility that whoever controls that router can read the message.
- Maybe they aren't and the NSA might.
- As packets pass through this router, we can grab portions and read that information and use it later.
- Anytime it goes to a particular router, it might be able to read it.
- Force the packet to go where you want it to.
- Routing protocols tell you where to send your packets.
- Convince the right set of routers to send information to IP addresses and read all your packets.
- Play a man-in-the-middle game.
- Not always going to be possible, but it has happened with sufficiently strong opponents.
- There are things one can learn by looking at portions of the packets that are unencrypted.
- **Traffic analysis**
- Learn how many packets went from A to B
- What the timing of the packets is
- A lot you can learn for info traveling across the network.
- Example:
- Voiceover IP call
- If you look at an encrypted voiceover IP communication, without breaking cryptography, you could get 30-40% of the content
- Very sophisticated stuff about phone names and voiceover IP's and how they convert speech into packets.
- A bit of a disturbing result

Message Integrity

- NSA doesn't care if you use strong cryptography or not.
- It may mean that they can learn everything by traffic analysis

- Attacker might be altering a packet between the point it is sent and the point it is received.
- Requires good access to the message path
- If the router is compromised, the attacker can seize control of where the packets are sent.

Denial of Service

- Another interesting network phenomenon
- Computer is supposed to do something for legit users
- Attacker's goal is to prevent service to users.
- Prevent providing services to users.
- Networks turn out to be easy targets, especially with typical networks like the Internet.
- Done simply by flooding the network.
- You throw more packets into the network than its capacity can manage.
- Throw enough stuff in and nothing legitimate will get through.
- You can corrupt the routing tables and we can avoid routing loops.
- A -> B -> C -> A -> B -> C …
- Certainly packets won't get delivered because internet technologies are worried about this condition.
- Sent to a black hole and gets dropped.
- You can flood routers or destroy some key packets!
- When you work at higher levels like TCP instead of IP, it makes assumptions that packets will get through
- If packets don't go through, TCP assumes congestion!
- If attackers wait for a packet and corrupt it so it gets dropped, TCP treats it as congested.
- If well-thought out, goes from stream of data to little drops of data that is relatively useless

How Do Denial of Service Attacks Occur?

- A lot of bad packets thrown into the network.
- A lot of people play the appropriate Internet protocol game
- Conditions where it is easy to overwhelm portions of the network.
- Internet lets you inject packets to be sent to anyone else.
- Relatively easy to create this kind of flooding

An Example: SYN Flood

- Denial of service attack can overwhelm other parts of your system.
- Much of the communication is based on TCP
- You can attack the TCP protocol and the implementations in real computers
- SYN floods do this based on a vulnerability in implementations of TCP (particularly OS)
- Set up in-order, reliable delivery and the way you' do that is to make an agreement between sender and receiver.

- In an ongoing sense, send me more data and so on and so forth.
- Arrange for both sides to agree we have a TCP connection between us.
- We have an understanding of how many packets should slow so we don't overwhelm the receiver.
- TCP has a handshake for this
- SYN message (stands for synchronized)
- In order to have a connection, both parties need to know about the connection to know how many packets are received.
- Which ones have been acknowledged and which haven't?
- Need a data structure describing the current state of our instruction
- I am going to send a SYN and this tells my machine to reserve space about this connection
- Table in most OS that says we don't want to spend all our time on TCP connections and allow it at any different moment.
- SYN flood is that attackers sends a SYN message and we won't do anything else, except send another SYN message.
- Purpose is to fill up that table.
- If he fills up the table, no one else can use it!
- Reject real TCP session!
- To deal with this, we use SYN cookies or firewalls with huge tables
- This tells the guy on the other side to drop incomplete connections.

Normal SYN Behavior

- Server provides HTTP service to some clients.
- Open TCP connections with one entry for every TCP connection
- The entry in the table will be kept for the connection.
- Three connections already set up for this particular server.
- Under normal behavior, this normal machine that just wants to behave properly would send the SYN message.
- The server then says to reserve space for the guy and this table entry would describe a new connection.
- Send a SYN/ACK, assuming all is going well, send a message in ACK.
- We are now ready to actually move data and we move from client to server.

A SYN Flood

- Open last connection and we have an attacker who sends the SYN message.
- Send back a SYN/ACK and send another SYN message.
- Send another SYN message and we can ignore SYN/ACKs repeatedly since the attacker doesn't care about the response.
- Fill up last open spot!
- Actual user says "God, I cannot use this service!"
- Not an overloaded server but the attacker is NOT going to bother sending new messages.

SYN Cookies

- Wait a certain amount of time because we don't know how long it will take for a certain user to do this.
- We are going to throw away entries and this is all well and good unless the attacker keeps sending SYNs.
- Send SYNs a lot faster than legitimate users are and we are more likely to get that table entry.
- What can we do about this?
- Make a big enough table.
- Dynamic data structure that uses up a lot of memory.
- Here, we are in a situation where our table is almost full and we are NOT going to give up that entry because someone sends us a SYN.
- In comes a SYN, and in this scenario, this SYN is from a legitimate user.
- We are NOT going to give him the last slot in the table, so we send back a SYN cookie instead.
- We are NOT going to save any room in the table
- SYN/ACK goes back and part of that SYN/ACK is a cookie
- One of the things you set up is sequence numbers (each is unique)
- We choose one number to be a sequence
- SYN Cookies lets us choose a unique number for this particular SYN that we can determine through some cryptographic mechanism.
- Get a bunch of info for various kinds and put it in a hash function
- Throw that sequence number into a SYN/ACK
- Don't add anything new to the protocol and use it in a very careful way.
- Put that number into a packet and don't remember it in the server side.
- Don't maintain any new information and see why this works in a moment.
- Use this guy's IP address and port as well as a local timer.
- Combine all this information through a cryptographic hash function so the person doesn't know.
- Protocol doesn't care which number you use as long as it is an integer.
- All that matters is that it be an integer who fits in this field.
- Back will come an ACK message assuming this is a legitimate guy.
- The ACK message says this is a SYN/ACK but what happens is for every packet, you will increase it by 1.
- Back will come an ACK packet containing a sequence number with 1 added to it.
- Server says it is in an overload state and it is using SYN cookies
- The question I am asking is if this ACK is associated with the SYN.
- We probably want to set up a connection and see if this is a proper response.
- It takes a look at the sequence number and some information.
- Same IP address as if he is sending the SYN
- He is setting it to my port and he is NOT supposed to be changing ports in the middle.
- Take everything that is purely local and we can run it through the cryptographic hash.

- Is the number that I get equal to that?
- If the answer is yes, this is a proper session.
- Set up a TCP connection and I should accept it.
- The only issue is the timer
- An easy way to deal with it is using a low resolution timer.
- Under normal circumstances, it will come back before the timer ticks.
- It can be timer+1, timer-1, etc.

Q. How much overhead does the cookie take?
A. No overhead in the header, but overhead in the server side for the content. If we have a quick cryptographic hash function, then the overhead is relatively low.

Q. Wouldn't be easy to forge a valid ACK?
A. Easy to form an ACK but a valid one is harder. If you send a SYN/ACK, it will occur, but the attacker has to go through the process of slowing down and if you do that, you cannot forge your IP address. It will link to another machine.

- Set up an open connection and use it properly.
- At which point, when you go to an open connection, we need to accept it and say we have given you enough open connections and we aren't going to give you more.
- Server has some secret in its cryptographic hash function; it just has to be hard to guess.
- Cannot use sequence # in SYN/ACK to just deduce what the input should be.
- Good cryptographic hash functions make it hard to connect things back to inputs.

- If attacker has 10,000 machines, he can have each set up 2 to 3 connections just as a legitimate user could be.
- Needs 10,000 machines actually doing that.
- Server doesn't have to store the cookie value (or anything!)
- If I have two slots, I am getting in 10,000 SYN requests and I don't have to save anything.
- Just check if it is a legitimate ACK or not.
- Big savings of the server.
- Another important point is that SYN cookies do NOT change the TCP protocol at all.
- No change on this basis with the possible exception that the server opens the connection or not on a different basis.
- In particular, the clients do not have to have any change on the TCP implementation
- It should work out just fine and this should be taken as an indication that they are behaving badly.

Q. Can we give the last few slots to legit users but what about the other slots?
A. We will time those out, and there always have to be timeouts for various reasons like my client failed.

- Don't want to maintain open connections former.

- If you are a site always being attacked or if you are occasionally attacked, you want to avoid overhead.
- If you have a lot of gray slots, you leave SYN cookies on until life gets better.

Q. How widely implemented is this?
A. Quite widely. They have the ability to use SYN cookies if you turn them on.

Q. Why can't the attacker forge his own IP address?
A. In order to get a slot, you need to send back an ACK with a proper sequence number.
- The SYN/ACK must be delivered to his machine and if he forged a random IP address
- This is useful it turns out!
- Used to determine DDoS attacks.
- Listen to SYN/ACKs you didn't send and you can deduce the amount of attacks that has happened.

General Network Denial of Service Attacks
- Particular type of denial of service attacks
- Still very popular because NOT everyone runs SYN cookies
- NOT everyone has run it so it is still widely used.
- Other DDoS attacks that are quite common.
- Forget about all these table in your router
- If you cannot handle the data transmission packet rate, you will get DDoS'ed
- If more packets sent that can be handled, service will be denied.
- Hard problem to solve and no general solution that can be deployed today.
- Sites where you can keep track of what kinds of attack are being sent on the network.
- DDoS attacks based on volume and someone is always being attacked.

Distributed Denial of Service Attacks
- Prevent a site from doing business.
- Send a bunch of junk at you and if it is junk, I discard it.
- None of my packets will cause anything, but they will get discarded if received.
- I will send too much bandwidth to you
- Moreover, if you cannot tell the difference between my bad packets and good packets, some legit packets will get dropped.
- If you randomly choose which get dropped, probably, all the legitimate packets will get dropped.
- If you are using TCP, what happens when a packet gets dropped?
- There is congestion here, but it backs off.
- Do I backoff?

- No way!
- The good people back off and bad people don't.
- After packets get dropped, the bad stuff takes over and the good stuff gets reduced down.

Q. If it is congestion control, if you are trying to DDoS, does it send these packets too?
A. You, the attacker, are under control of what you send, so you don't have to obey protocols.

- You are going to push ahead with the attack.

The Problem

- The attacker has through some mechanism compromised machines across the Internet.
- Effectively, he owns them because there are various kinds of vulnerabilities and he can send whatever packets and machines will be compromised.
- Attacker will take down the server to prevent it from giving service.
- Nothing suspicious and only a bit of traffic.
- As it goes through the Internet, goes to more and more traffic and by the time it gets to the target, there is too much.
- Essentially what happens in distributed denial of service attack.
- Generally trying to annoy you.
- Serious annoyance and they will make life difficult for you.

Why Are These Attacks Made?

- Done for extortion
- If you make money by having your site, you aren't making money.
- They are then contacting you and then we stop the attack.
- Sometimes, we stop the adversary from doing something important.
- One thing that happens a lot is before they make the attack, they launch a DDoS attack on the bank.
- Oh gosh, a DDoS attack on the bank and they all start running to do something for the DDoS attack.
- Attackers then break in and steal the money.
- Sometimes, this can be a bit worse if you direct it at some guy at the edge of the Internet.
- If you do this, and are successful in attack, you can take down the Internet.

Attack Methods

- Pure flooding
- Of network connection
- Of upstream network.
- Overwhelm some other resource
- CPU resources
- Check a cryptographic checksum and send some garbage values and spend time checking checksums.
- Maybe you can attack memory resources

- Application level resources.
- If you understand characteristics of hashing algorithm, you can get pathological behavior on the hashing algorithm.
- Can be directed attacks or reflected.

Why “Distributed”?

- The really serious ones are you want to attack some e-commerce sites.
- You are going to be getting a whole lot of traffic and they have a fairly good amount of bandwidth.
- If you want to overwhelm them, you want something pretty heavily provisioned.
- Single laptop or smartphone and it is hard to generate enough traffic to overwhelm a serious server.
- How do I overwhelm a serious server?
- Use 10,000 machines (or a huge # of machines)
- Not that hard to compromise machines for this purpose.
- If you do it with a single machine, all a defender has to do is use various measures to prevent an attack on you
- If you have 10,000 machines, you have to take measures to handle things scattered in the Internet.
- Different paths through the network and defend against all of them.

Q. How expensive is it to do a DDoS attack?
A. Many machines have been compromised and they are wondering what to do with compromised machines.

- Launch DDoS attacks, and rent out compromised machines to do DDoS attack.
- You can rent out compromised machines to launch a DDoS attack.
- This is happening every single day and there is an underground economy based on using these compromised machines to make money.

How to Defend?

- Many times, when people first hear this, they say let’s drop a bunch of traffic.
- Not enough to stop a flood
- ENSURE SERVICE TO LEGITIMATE CLIENTS!!!!!
- If there is still no service offered, you haven’t done shit!
- You need to offer legit services, otherwise the attacker has won.
- It does NOT help to deliver a manageable amount of garbage.
- You need a defense to help stop you from being overwhelmed by traffic.

Complicating Factors

- The Internet was designed to deliver traffic and it is very well-designed and it does NOT care if it is a packet you want to get or not.
- Address that every packet gets to you
- IP spoofing lets you create fake IP addresses.

- Set up filtering at some point and how do we know if it came from bad machine.
- How are we going to set up filtering to catch all these packets?
- Legally, why can't we arrest the attackers?
- The machines belong to someone in a far-off land so it isn't feasible.
- We will negotiate with each of them and arrest the person that is there.
- Attacker is able to choose each bit and he doesn't care what you do with it, so it gets difficult to set up any filtering rule.
- If you send packets that fit like this, it will look like that and get you through your filter.

Basic Defense Approaches

- Over provisioning
- We aren't going to have a 10 GB connection and if we can handle 10 TB of traffic, you will have a hard time!
- Works well for Google, Amazon, or Microsoft!
- Dynamic increases in provision
- Need only 10 MB per second and we need an agreement on our ISP.
- If we are getting more traffic, we are temporarily sending something to you.
- This means as long as you increase your provisioning, we can give service to our users.
- If we are talking about things like running on the cloud, it becomes a feasible thing to do and we can give one virtue machine due to your web server.
- Give you a 3rd one, 4th one, 10th one, etc.
- This becomes a more feasible approach.
- Hiding
- If they don't know where you are, they cannot attack you.
- The problem here is that how do your legitimate clients know how to get to you.
- Set up a huge overlay network with 50,000 sites.
- If you want to talk to me, talk to anyone of those 50,000 sites.
- Forward messages to someone else and it will get to that site.
- One of the first things people did in the early 2000s was to figure out the source of the attack!
- Built mechanisms that would track where they attackers were common from.
- WORTHLESS!
- Now what the hell are we gonna do?
- Reducing volume of attack
- Get rid of a bunch of the attacks which is difficult because it is hard to distinguish good packets from the bad packets.
- We somehow want to figure a whole lot of bad stuff is coming from over there.
- This is going to get filtered (collateral damage)

- If we get rid of enough of the bad stuff, at least you are offering services to somebody.
- Practical approaches from a provider that do something along these lines.
- None of these approaches are totally effective.

Reflection Attacks

- I want to overwhelm you and in a reflector attack, he is NOT going to send them to your machine.
- There are these services where you send a request, they send you a response.
- If you send a tiny request, you get a big response.
- Works beautifully for DDoS attacks!
- Spoof IP address in the source field and go into the DNS table.
- Back comes a MB of information and it goes to the spoofed address.
- This is called **amplification**
- Take a small request, get a big response.
- The big response will go to target.
- Quite difficult to deal with because people sending this garbage data are NOT compromised sites.
- Legitimate servers on the Internet used for legitimate purposes.

Q. Wouldn't a good thing to do in a case of heavy traffic is to recognize this and accept only requests from a whitelist of groups?
A. You can do this kind of defense and depending on your site, it could be good or bad.

- Bad for Amazon because it wants more clients to buy stuff and go away.
- You cannot setup a filter with 100 million entries
- What machine is it sitting at?
- It might be at a firewall, which has to look up each incoming packets and look up good people.
- The attacker can spoof IP addresses and if they are on the list, it gets harder.
- What you have to do is filter ranges of IP prefixes.
- Everything in this /24 will get dropped.
- If there are one or two good sites, you have cut out those sites as well.
- You end up doing more collateral damage.
- Except stuff that is really close to the destination, you have to work on these prefixes.
- Reiner has research projects that do precisely this kind of thing.

Traffic Control Mechanisms

- Try to do things to control amount of traffic
- Internet does NOT control traffic at all.
- We sometimes would like to be able to control this.
- Filtering based on characteristics of packets
- Source address
- Other things

- Rate limits
- Not going to drop everything but no more than X traffic gets through
- Traffic analysis
- Watch the attacker and see who is communicating with whom and at what volumes.
- What do we do about this kind of thing.
- Padding and routing control

Source Address Filtering

- Drop things because it is the wrong address
- Most commonly you do this because it is a spoofed address.
- If you can only tell that a packet's address was spoofed, you could tell if it should be dropped.
- Want to tell us when a packet comes in to see if it is the right source address or not.
- Some of these technologies can be used in particular places.
- Ingress filtering (aka egress filtering)

Source Address Filtering for Address Assurance

- Traffic gets into the router and if they are good, they should be sent onto the destination
- If you happen to know, everything over here has a particular set of IP addresses
- Therefore, we know we can drop it.
- Can be done most effectively at the edge of the network.
- At that point, the router knows a whole lot about things being sent to the Internet.
- This will say that your users cannot spoof IP addresses that go to anyone else.
- This does NOT help you with incoming addresses being unspoofed addresses.

Source Address Filtering Example

- Lecture 9, Page 33
- We have a routing machine and everything else is out there.
- In particular, the range of IP addresses is a range like 128.171.192.*
- If the source address is NOT in the range, you can drop it.
- All of my machines cannot spoof IP addresses even if taken over by an attacker

Source Address Filtering in the Other Direction

- Doesn't actually defend my machines, but prevents me from attacking other people.
- Often called egress filtering (aka ingress filtering)
- Stuff comes into the network from your router.
- What addresses shouldn't be coming into your network?

Filtering Incoming Packets

- Lecture 9, Page 35
- The packets are coming in the opposite direction and your router should decide to drop things or not.
- What is the source address?
- It is in my range! WTF!
- I would never see it and we wouldn't see it on that link.
- It has got to be spoofed!
- With this source address, they should be going out, NOT coming in.
- So we drop the packet.
- Limited to addresses within your own range.

Other Forms of Filtering

- Things you should do, and moreover can do.
- They have told everyone to turn things on and every router you can buy has this capability.
- A lot of people do this and the outgoing addresses cannot be spoofed.
- You can filter on the basis of other things like worm signatures, unknown protocol identifiers.
- Set of unallocated IP addresses in IPv4 space
- Intended to be use for LANs only
- If you see this packet anywhere, this is a spoofed address.
- Someone performed a DDoS attack on South Korea and they performed the DDoS attack of the whole country.
- South Korea had about 8 different points where you could connect outside, and these areas got smashed!
- Anybody fortunate to work for them couldn't get in and out of the country!

Realistic Limits on Filtering

- In the middle of the Internet, it is hard to figure if an address was spoofed or not.
- Partially, it is a very hard problem and in the Internet core, they are moving them as fast as they possibly can.
- They have very little time to make a decision about anything and they have to make lightning fast decisions.
- There business is moving packets and nobody pays them to drop packets.
- In particular, they are very unhappy about dropping good traffic and we don't want to do that.
- Unless you tell them not a single good packet will get dropped, they will NOT want it.
- If you screwed it up, and they install it, the Internet stops working!
- They don't like the idea
- It is more worth it to filter near the edges.

- Typically, unless you are talking about your very own piece of equipment, they only have a limited amount of stuff for us.

Rate Limits

- We want a certain amount of traffic to go through to ensure we aren't overloaded.
- If we don't do this in a discriminatory manner, it doesn't help very much.
- This kind of stuff is of more use for non-security purposes.
- Guaranteeing quality of services to particular data flows.

Padding

- Sometimes, attackers are watching what you are doing based on traffic characteristics
- Who did you send packets to?
- Goes back to the early days of radio (WW II)
- Armies would use radios to communicate to each other on the field
- People tried to listen to radios on the other side.
- If you saw a lot of radio traffic where there didn't used to be a lot of radio traffic, possibly the enemy would be setting up an attack
- If you wanted to launch a big scale attack, you had to send a lot of messages.
- Watch for activities.
- Reporters in Washington D.C. were pretty good at figuring out when other military units would launch a strike secretly.
- Observe how many pizzas were being sent to the Pentagon after hours.
- Watch what is happening carefully!
- Make sure a certain # of pizzas is sent without changing the value.
- Make an unpredictable, narrow range of traffic that ALWAYS happens.
- Use it for important busy stuff.
- Vary the content of the traffic, but the AMOUNT should remain the same.
- Throw in a bunch of extra traffic and make sure the padded traffic looks like the real traffic.
- Good stuff and bad stuff will look random.
- People typically encrypt everything and we have to do this carefully.
- There will be clues that let clever attackers to separate padding from real traffic.

Routing Control

- Use IP address X to talk to IP address Y
- Figure out who owns X and who owns Y
- See who is running to the ISIS recruiting websites and see who is joining
- People don't like their traffic traced this way.
- Now, the people looking at this traffic notice patterns between alternate routes
- *Onion routing* - TOR (The Onion Router)

- You have to use something VPN like and toys can conceal things with a VPN
- If you have 500 people working on the West Coast, and 500 people working on the East Coast
- You have to say encrypt everything and at this endpoint, slap on the IP address of another endpoint.
- Same destination IP address and decrypt it.
- Hide what you are doing using encryption
- With routing control, you do it by shuffling messages around.
- If the only thing from A->B->C->F is shifting the message around, it isn't hard to figure out what is going on.
- Some of it comes out from F and we are shuffling things around and trying to mix up all the traffic and make it hard to tie packets going in to packets going out.
- You can use TOR today and if you think about this, serious performance overheads and you will have to encrypt all this tuff
- Build up packets in layers like an Onion
- At X, you do all the encryptions layer-by-layer
- Peel of layers of the onion and we find it at the last layer.
- Each encryption costs time and overhead.
- Doing all kinds of deliveries and the overheads will be high.
- Conceal what you are doing to some extent.
- Eventually, there are other algorithms and X is talking to Y and they keep changing it again and again.
- People can just keep doing this forever.

Q. Onion routing, is this an open source project?
A. U.S. military funded it. Reiher funded it! Restrictive of what citizens could or could NOT look at.

- Could have dissonance and it would have been great for illegal drug sales.
- Practically all the traffic is related to illegal activities.

Firewalls

- Machine that is intended to protect a local network from bad stuff on the outside.
- Sits between your LAN and the Internet
- Everything goes through the firewall and creates protection.
- Runs special software that is good at regulating special traffic.

Typical Use of Firewall

- Drop the firewall in place and in theory, the network doesn't blow up.

Firewalls and Perimeter Defense

- Has millennia long history
- Build a castle with big strong walls
- He cannot attack you (in theory)

- Has a terrible history
- The walls ALWAYS get broken
- You protect the inside of something by preventing people from getting inside.
- The firewall host is called a *bastion host*
- Essentially, you control the entry and exit points.
- Good stuff inside, bad stuff outside
- Why is this bad?
- If you are wrong and the perimeters gets breached, you are screwed.

Weaknesses of Perimeter Defense Models

- Windows passwords are an example
- Crack the password, you can do anything
- They are NOT bad ideas, but you cannot rely entirely on a perimeter defense to defend your system.
- Lecture 9, page 45
- Attacker tries to come in and bounces off
- Sometimes, the attacker blows a hole in your firewall and compromises your machine.
- Eventually, he will have your entire network!

Defense in Depth

- Same people who created castles also created multiple defenses.
- Don't use one single defensive mechanism
- Make sure they aren't susceptible to the same attack.
- They have NOT necessarily defeated your whole system.

So What Should Happen?

- Lecture 9, Page 47
- If he tries to take over a node, that node should reject him even if it comes from a trusted buddy.

Or, Even Better

- Lecture 9, Page 49
- Patch the walls and figure out our weaknesses

So Are Firewalls Any Use?

- Yes, you need a firewall as part of your defensive artillery.
- There is background radiation, which are automated attacks happening all the time for everyone
- Programs that try to compromise any node on the Internet and every possible IP address gets probed.
- They try to use old attacks and we don't like having old attacks to get through.
- All the background radiation should bounce off a firewall and we need to be careful so they are NOT susceptible to old attacks.

- Reiner doesn't have a favorite firewall and they aren't all that different.

The Brass Tacks of Firewalls

- What are they really doing?
- Getting in the way of every packet delivered.
- Looking at every single packet coming in and makes a decision
- Makes a binary decision and drops it.
- How they make the decision can be complicated or simple.
- It maybe that for the defense, instead of dropping a bad packet, we will try to figure out what is going on.

Types of Firewalls

- Elements of sophistication possible.
- Filtering gateways (screening routers)

Filtering Gateways

- All I know about is packet headers
- Look at IP addresses, port #'s, protocol #'s
- Make decisions based on these
- Typically, let the packet through or reject it.
- Don't maintain any state
- Do NOT remember that we just dropped the IP address and let the next packet come through.
- This is beneficial in many ways because this is a cheaper firewall to create and run.
- Set of rules and get down to the rule and particular fields to do that.

Example Use of Filtering Gateways

- Allow remote access from particular external machines i.e. telnet
- Could allow full access to external machines
- Don't care what kind of protocol it is but it gets in.
- Unless you are on my good list, I don't let anybody come in.
- Only accept things for HTTP
- NOT running DNS server and drop it if necessary.
- Only looks at header!

A Fundamental Problem

- IP addresses can be spoofed
- You can improve this using IPSec
- Hasn't improved very much in terms of firewalls.
- You can perform ingress/egress filtering.

Filtering Based on Ports

- Ports are an identifier for this information
- You then say that you will allow things that go to the following port and you would have a DNS server you want people to use.

- If you have uncommon ports, ports that are used by uncommon applications, you don't worry about these weaknesses.
- You can prevent people from getting in at all and it won't matter anymore.

Pros and Cons of Filtering Gateways

- + Fast
- It is a quick decision to say yes or no.
- + Cheap
- Not doing anything particularly sophisticated
- + Transparent
- Doesn't look like firewall is there or not.
- - Limited capabilities
- - Dependent on header authentication
- - Generally poor logging
- - Rely on router security
- Assumes router does the right thing.

W 5 Dis 4-29-16

Scripts executed on login

- System/bashrc
- user/profile
- bash
- Put permissions for root
- If you edit the sudoers file and run NOPASSWD, you modify their shell files and you need to give yourself sudo access, how do you do it?
- You can put a command in those files that gives you sudo access, which will usually require the sudo command

0. NOPASSWD is convenience: There will always be vulnerabilities though
0. Lack of Permissions setting means other people on your system can set things up

Buffer Overflows:

W 5 O.H. Reiher

- It is usually hard to pinpoint the attacker
- Law enforcement organizations like the FBI can catch some hackers most commonly because they leave information lying around.
- Less common to be traced to these people.
- Usually, if they are trying to make a profit, they somehow have to make money out of something and convert it into usable form.
- If the law enforcement agencies put the money in the bank, then they can get him.
- Banks are relatively careful and the way it has happened is that some other bank that got compromised and they did not take money out of the bank in Bangladesh.

- Instead they tried to use the credentials and ran away with it and that worked to some extent.
- Millions and millions of dollars and an active investigation.
- Everyone stinks at cybersecurity but the U.S. is relatively better than other nations.
- They usually all get D's and F's and this is given by companies who are set up to do this.
- They require you to do the audit but don't require you to fix that problem.
- Salts are related to passwords
- Dictionary attacks (people run them through the hash function) - there are canonical hash functions and there are a very small # of them ever used.
- A given version of Linux will have one hash function. They have your password file and all they need to do is encrypt your dictionary by passing it into the hash function and comparing it to the password file.
- They would need to take a big dictionary and they need to run through the hash function once, and this makes it too easy.
- Then, they would take your password file and anything in your encrypted password is what it claims to be.
- We can make life harder by saying don't just encrypt the password. You also need to throw something else in like a 32-bit random #
- Attacker can no longer take the dictionary of a million words because everything will be different for every different user.
- This means the attacker would have to take the salt and the million word dictionary with the salt
- Keep going into the next entry and different salt would have to do it all over again.
- The next guy would have to check it a 3rd time and it is of no use.
- Always good to try to make more difficult for an attacker and this effectively
- Salt is a plaintext that is random each time, so he has to check the full million words.
- It wouldn't do much to put it encrypted and this means you have to decrypt the salt!
- The machine that compromised would be able to figure out the key to decrypt the salts when you need it.

TPM

- Hardware - hardware security technology
- In Macs and TPM has been in most machines for quite some time.
- What TPM does is to secure boot.
- It offers you a guarantee that the OS you are booting is the OS you think you are booting.
- It checks if you are using the right bootstrap loader.
- You could be using the right bootstrap loader.
- It has to be careful about what the bootstrap loader is and you can have a high level of assurance

- Attackers tried to corrupt your OS
- You end up booting a compromised OS that the attacker can make use of and it will check your OS before it boots it.
- Compromised the bootstrap loader as a strategy
- If you are worried that the bootstrap loader, it will have the bootstrap loader check the OS.
- We want to be 100% certain you are running this with hardware.
- Hardware goes to a particular place to get the bootstrap loader.
- Using my own hardware, I am going to check that code, and I am going to do a hash on it and check it against a hash I have saved.
- If they have matched, then say we are running the bootstrap loader correctly.
- Kernel hash, check in TPM hardware and save a hash and see if it is or it isn't.
- TPM stores keys in a secure fashion.

Q. Why did Microsoft build SecureBoot?
A. They felt they could do it at a booting level and they are booting the right thing and it is also hardware.

- A way to get appropriate signature to allow booting on this hardware, but generally speaking, we don't know if the Mac OS has the TPM system.
- Modern commodity computers are built-in.
- Chromebook OS does NOT even have a BIOS built-in, so you have to install the OS separately.
- Doesn't want a general-purpose booted.

- Attackers can only steal an encrypted version of the private keys.
- We would have to look around to see where the private keys are, but your public keys are located in known_hosts
- We can set up our firewall in particular ways to catch things.
- While there are protected measures, we still get attacks up these characters that work.
- Many websites are susceptible and doing the full careful analysis is really rather difficult.
- Not a trivial thing to do, and we are going to get the input in a certain way and we don't want it interpreted as SQL and if the program was smart enough, you would be safe and for general purposes, it is quite difficult to look at all possible ways.
- For URL's, there are so many different options.

- HTTP is unencrypted, going in plaintext!
- HTTPS - interactions are encrypted and they use bootstrapping technique and they use the public key of the website and from that point onward, everything is going to be using that symmetric key.
- It is possible for a website to stop using HTTP
- Cost (cryptography costs)

- In order to use HTTPS, you have to have certificates and they always cost a certain amount, and nowadays we have “Let’s Encrypt”
- They do very little effort

Bootstrapping: I have to start from somewhere.

- My machine is currently off, and we need to be running an OS.
- How do we go with a machine with currently nothing to something with an OS.
- This was the first bootstrapping step.
- We are starting with a clean slate and we have to get into something up and running
- Clean slate and up and running and working.

Reflection Attacks

- IP message with a from address and a to address
- The applications that get this message will almost always send the response to those who sent the message.
- Send him a packet so it goes to him with the apparent address being your target address.
- Put in Amazon.com’s IP address
- It appears to come from Amazon so I will create a response and send it to Amazon.com
- Instead of saying I am the attacker.
- Reflects off another source.
- Advantages: reveals to the IP address and it seems it is from a perfectly legitimate site!
- There are certain services where we get a lot of traffic to the destination.
- If there is a 40 byte request, we get 2000 bytes at the target and you get magnified.
- Usually attack someone who has more capacity than me.
- I send a request and a response goes to the target, if the request I send is smaller than the response, and I had to spend 40 bytes and I got 2000 bytes.
- The server who did the response resulted in the increase.

Padding

- Used to prevent patterns in the traffic.
- You would encrypt everything and you have to encrypt padding traffic that looks like real traffic.

What if I only have 1 MB/s of traffic but I need 10 MB/s of traffic

- Create 9 MB of junk and the guy who receives this traffic has to understand that what he gets is junk.
- Assuming something I care about, the attacker who is listening in always sees things.

Technically you can write your own firewall and chances are, the one built into your OS is better than the one you write.

- Linux has IP Tabs
- Mac has firewall systems
- Windows comes bundled with Anti-Virus.
- Everything in the reading is covered, and this was NOT gotten to in the lecture.
- Readings that are NOT part of the textbook.

Q. A lot of big enterprise companies use Windows as part of their legacy. Because of this, attackers often create viruses that specifically target Windows. Why don't they switch to use Unix/Linux in this case?

A. Windows built up a huge market and it is very expensive and then it is expensive. All the people have to learn a lot of new stuff. Microsoft won the war of who will run the OS on most systems.

- Microsoft mostly aren't running the Windows OS.

Mobile phones attacks

- Attacks based on malicious apps
- Apple tightly restricts on the App Store
- Show us your App and they test your App and they put it in the market.
- Pretty much the same security model as running an application model on the computer
- Google Play Store lets almost anything in
- They thought they had a better security model
- When you look at the access permissions, this app should have those and never unless we uninstall or reinstall will it get a different set of permissions.
- People would very carefully look at permissions and why does this want to gain access to my Contacts?
- Nobody ever looked and why shouldn't I have Internet access for my app.
- Android has to go closer to the Apple model.
- They didn't follow the "least privilege" idea
- Apple and Linux have security flaws more or less on a monthly basis
- This is another place where Android is having more problems than Apple.
- Apple controls the OS and this is the version that is getting installed.
- With Android, Google says this is the old version and the manufacturer decides what we want.
- If you want to accept a patch, we decide to accept the patch, so the individual manufacturer of the smartphone will choose the old version of the operating system will be this.

- Android model said we aren't going to be smartphone manufacturer; rather, we are going to be the software developer and as long as you have this processor, we can run this processor.

- Apple builds both the hardware and software and Apple lost even though people thought they have better hardware and better OS.
- Relatively expensive compared to commodity hardware, and it was so much cheaper to buy
- People are thinking of mobile phones and there is a certain cache to the Apple brand
- Some of that back in the 1980s
- Groups of people who would only buy Macs back then.

- Microsoft started making their own hardware with the Surface Pro
- Microsoft has had trouble late in the hardwares.
- Zunes are the knockoff the iPod
- Two years after the iPod came out.
- Had a few features that allowed
- They essentially more or less failed with Bing, but Xbox was good!
- Gained a sufficient advantage in that ways.
- Office won because it was out there to run on the Office machines that they created long before anyone provided an integrated suite.
- You bought your OS from Microsoft and it comes more or less bundled.

Hardware Exploits

- Much more carefully designed than software
- As a result, it is relatively rare to find serious bugs in a hardware chip
- Intel had some arithmetic issues in some of those chips that cost them hundreds of millions of dollars
- While there were not obvious bugs, there are intentional backdoors built into the hardware and the NSA has been able to do that.
- Once we go down from processors to other types of chips, it becomes less safe and some routers had some vulnerabilities in this case.
- Memory chips don't tend to have this problem so there isn't a whole lot of room to muck around.
- Cram as many memory cells in your chip.
- You cannot fit as many memory cells and you won't be commercially competitive.
- A chip that does 802.11 has more room for possible bad behavior and simple bugs.
- Generally speaking, there are thousands to millions of times of software to hardware, so it is easier to look for problems in software.
- For hardware, you have to physically be there.
- Possible to build things into router chips.

The Florentine Deception

- President of Symantec so he knows a great deal about this field.

Bluetooth has NOT been compromised in hardware

- There can be vulnerabilities in the applications and the bits you use can cause problems, but it isn't Bluetooth's fault more than IP.
- There is isn't any reason for Bluetooth to have any particular securities.

- Monday O.H.
- 12 PM

W 6 M O.H.
- Close to done with Lecture 9, but we did NOT cover Lecture 10.

W 6 R Lec 5-5-16
- In the general area of firewalls, we were talking about two types of firewalls
- 1) Treat each packet as a totally separate entity and look at its header
- 2) Let's have a deeper understanding of what is going on. Let's look at what will happen if we deliver this packet to the machine and this requires us to consider flows of packets.

Application Level Gateways
- Proxy gateways
- What applications are you actually running on those hundred machines
- Easy to filter things out based on header information
- The firewall is supposed to help protect against this
- Proxies stand in for the actual application and says it will know what will happen because it understands that application
- Not a perfect understanding because we don't have perfect knowledge of what is going on
- HTTP packet that is going to Skype.
- Don't know much about the internal state of the machine
- Do the same thing with the packets as a basic firewall
- See a packet, make ad decision on whether to let it through or not.
- If you try to do this game, the firewall must remember things and cannot treat individual packet as a separate entity
- In any system with a state, you have a more complicated state

How Application Level Gateways Work
- Regard everything as part of a flow
- Alternately, this is something I have heard about and I have to identify which flow it is.
- Either, I will have a firewall hard walled configured to TCP, HTTP, etc., or you expect to see proper TCP behavior from this flow and I will do something
- You can have an empty framework to plug something in and we can have a proxy that deals with the HTTP flow and it can represent an online multiplayer game.
- Each proxy would specialize in the appropriate application it is supposed to protect.

- You can do things based on header information and you will look beyond the IP header
- Things are based on ports and can be specific to the TCP header.
- This can give you some notion if this is a TCP flow or an HTTP port.
- Therefore, this goes to the HTTP proxy.
- Ultimately, while the process by which you analyze the flows can be a lot more complicated than the filtering firewall.

Deep Packet Inspection

- Don't just look at the IP header
- Look beyond the headers and we could be talking about part of the HTTP requests
- Seeing beyond the end of a GET request and I will remember that it will look like the second part of a GET request.
- If it is, then I return to the state and wait for either a brand new GET request or an outgoing flow.
- Do something that is very specific to the protocol and also what is going on from the point of view of the proxy.
- Sometimes, we will have a very sophisticated understanding of what is going on.
- A guy could get a page and understand what links are on that page.
- It could be some random link NOT associated with that page and there could be something weird going on.

Firewall Proxies

- You are going to have a framework and the ability to plug in proxies into that framework.
- Work on its own when plugged into this framework and it can be attached to this proxy.
- We can have a video conference and we can have something coming into this application
- If it isn"t from one of the IP addresses, we will have a problem
- Proxies need to be specialized, but there are limits to what you can do.
- You don't see the endpoints, sender, or receiver.
- This is all we got to work with, so we need some degree of complexity to guess what is going on/
- Packets are coming in and we want that packet to be delivered as soon as possible.
- Just look at a couple of fields in IP headers and we could even build the in hardware.
- If I want to figure out if this particular request fits into the type of the web browser, this can be really complex to do.

Pros and Cons of Application Level Gateways

- + Highly flexible
- Drop packets that appear to have a mysterious origin

- + Content-based filtering
- Virus detection
- + Potentially transparent
- When things are going well, it should look like there is no firewall in the first place.
- - More complex
- More bugs are likely
- - Highly dependent on sophistication
- Anyone of us can probably build a firewall based on IP headers and that is really easy to do.
- Proxies are potentially complex, and is it actually better than filtering firewalls?
- Only better if the proxies are more sophisticated and better at catching attacks.

Q. Is filtering more static than other types of firewalls?
A. Filtering is more static and is not as flexible. Based on limited set of information from the header fields of the IP packet and at most will go down to the header fields of the TCP packet.

- Application level firewalls have a greater understanding of history and it will be part of the existing flow and it is an application firewall rather than a filtering firewall.

Reverse Firewalls

- We need to keep the bad stuff out!
- Presumed we only needed to only look at incoming packets.
- Bad stuff has gotten inside for whatever reason.
- Even if something is inside your network and is doing bad things, does this mean you cannot do anything about it?
- A typical bad thing is stealing information from you.
- How are they doing that?
- They are sending packets out from your network to the outside world.
- They are taking an entire database and this is going to their site.
- Under a normal unsophisticated configuration, they are going through their firewall machine.
- Why don't we also watch what's going out?
- This is a reverse firewall!
- Keeps bad stuff from sending data out of your network.

Possible Uses of Reverse Firewalls

- Network printer and he can compromise that.
- He can see what you are printing but he cannot get to other machines.
- He can start pinging all the other machines in your network and figure out what operating system you are running
- Figure out things based on your machine and do basic stuff on your network.
- Based on this info, this is how I will attack the guy.

- Don’t let them do this! This shouldn’t happen.
- If you have a network printer, it should talk to other machines but it is never supposed to send a packet out to the Internet and there will be a difficulty in the network.
- For a botnet, we need to watch for that kind of traffic.
- All kinds of things with a reverse firewall if set up properly.

Stateful Firewalls

- Voiceover IP, it is commonly carried by UDP
- Running a phone call and there is a lot of stuff going on with connection status.
- It has state and you cannot really properly understand what is going on unless you understand the state.
- If you think about this, you will realize that it gets a lot more complicated and they will be very busy doing business with the outside world.
- Not worried about load but what does this mean in terms of what the firewall has to do.
- You will have 100,000 entries for the state of connection of this machine
- Creates a data management problem with performance issues

Firewalls and Transparency

- You don’t want the firewall to be there if there aren’t any problems
- Unless you make a particular type of protocol, the network game has to have some connection and we won’t let you do this unless we have a firewall on his side.
- Ideally, for an acceptable application, you want the firewall to not be there at all.
- Be careful when setting up a firewall’s filtering rule.
- Users will become unhappy in this situation

Firewalls and Authentication

- Allow things for people who we trust and prevent people who we don’t trust from entering
- We have to have some degree of authentication i.e. looking at the source address of the IP packet
- Not very strong because IP can be spoofed
- We can do cryptographic authentication, which means it needs to know about the cryptographic rules!
- If we need to do something at the destination machine, we need it at two different places.
- Key security means we don’t want the key at multiple places because that is less secure.
- Authentication is NOT really possible in these kinds of firewalls

Firewalls and Encryption

- Firewalls cannot do anything with encrypted data unless it knows the key.

- There must be an unencrypted IP header
- Everything else might be encrypted, so you need to look at the IP header and do some deep packet inspection.
- Cannot do deep packet inspection if it is encrypted.
- Decrypt it then and decide if the packet is okay or not.
- What if the destination is doing end-to-end encryption, and no one else in the middle can read it.
- You have to share keys and it gets kind of complicated if you need to do that.

Lecture 10
Firewall Configuration and Administration

- The firewall is your first line of defense against and attacker
- You can be quite sure that the attacker will be quite sure they can overwhelm any other problems you want.
- Bad things are hitting the firewall and you will be getting bad stuff coming in.

Firewall Location

- I have my machines I want to protect and we have the big, bad Internet out there
- Put the firewall in between!
- Maintain all of our internal data that is vital for running our data.
- Figure out accounts receivable, and sensitive info not intended for the public!
- This is very private info but we might want to maintain a web presence to buy things.
- This should be available to literally anybody and this should provide information through one of the users on the Internet.
- Someone would want to buy one of the widgets and say do we have any widgets in the warehouse?
- Figure out the price and send it to whoever he wants it for.
- All these interactions between public web server and private database/ network
- The right way to do this is by dividing up the network into segments and each will have its own security characteristics
- Relatively safe, inside stuff that is reserved internally for the company.
- We will divide the network into segments that handle the locally protected stuff and also the outward facing stuff.
- We can use multiple firewalls

Firewalls and DMZs

- DMZ is a demilitarized zone: common way to build a network for a typical organization
- What you will do is use multiple firewalls to divide into multiple segments.

A Typical DMZ Organization

- Lecture 10, Page 5
- Web server that provides information to your company from the outside world.
- Internet is good and bad so we have a network connection that goes into the local office.
- Connect up the web server and connect it up to the local machines and employees need to be able to connect.
- Anything coming in has to come from the firewall
- Needs to require authentication from the outside user and this requires some form of cryptographic authentication
- At any rate, it will be quite restrictive and we need to be smart about where we position our DMZ.
- We do need to prevent the web server from being compromised so we need a 2nd firewall.
- All the information will come on the Internet and it will go through that firewall.
- If OTOH, if it goes to one of the machines, it has to go through two firewalls.
- Demilitarized zone - not quite so safe as between our private network but is safer than taking raw stuff off the Internet

Advantages of DMZ Approach

- If we watch what is going on in the network, there are no web servers where we can sit outside.
- If we sit outside and see HTTP requests coming in, our firewall will prevent those web requests from coming in.
- This tends to keep inherently less safe traffic away from the resources.

Dangers of a DMZ

- By the very nature of the fact that you have an outer firewall, bad stuff can get through th firewall.
- More likely to be compromised and what if he gains control of your web server?
- He can have a foothold and the ability to start looking around in your network.
- Once he has compromised one of them, he can probably get all the other networks as well.
- Machines in the DMZ can communicate with others without going through any firewalls
- Compromise one machine in the DMZ, compromise all of them!
- We must not regard the machines int he DMZ being trusted!
- They are inherently less trust-worthy to any other machines that are more protected.
- If you can go around the side, there won't be any sort of protection
- Software gets updated and configurations get changed.

- System administrator probably has his machine sitting in the production network and he is a very privileged user.
- There should NOT be a lot of interference from machines that have firewalls

Q. Would it hep to have several connections?
A. This would work out okay but it is not really necessary. It does require at the software level to configure different connections and things cannot possible get into another.

- Renting a second line gives you a strong assurance

Q. External firewall is less restrictive?
A. The one on the inside is more restrictive. If the outside one is more restrictive, you have no need for an inside one. They need to liberal enough to let anything through.

- If your firewall is compromised, then of course, you are getting no protection whatsoever.
- If outside one is compromised, you still have your inside one.
- Inside one must be inherently more protective, and therefore, it is a lot safer to have two separate machines.
- We are talking about appliances that are pretty cheap to purchase.
- To save a little bit of money, we need both firewalls in one physical machine.
- Use the strongest virtual machine technology to support them.
- I am running a good VM technology and I have one VM that represents firewall A and another VM that represents firewall B.
- This VM technology is itself secure and people keep attacking virtual machine technologies.

Firewall Hardening

- Hardening makes it harder to compromise the machine
- People compromise the machine by finding a flaw or misconfiguration
- Something like that and they make use of it.
- The less software you are running on any machine, the fewer flaws you will be running that machine.
- The more software, the more likely that the machine will be compromised
- This is the firewall machine period and it doesn't do anything else.
- Make sure that the minimum amount of software is running to allow the firewall to run.
- When you install Windows, now we are going to disable all these wonderful Windows features and Windows is a general OS.
- Turn off all these background tasks just to keep things going in a reasonable fashion.
- Firewall is NOT doing that kind of stuff and it is just software that is compromised
- Close known vulnerabilities
- Every piece of software in the world has known vulnerabilities
- For all machines in general, we want to patch the vulnerabilities ASAP

- We also want to strictly limit access to that machine
- This is a machine where only the system and network administrators should be able to log in.
- People can only go to a hardware console and configure it using a keyboard and mouse.
- The remote login must come from the system administrator machine
- This is the only thing that logs in here and he can only log in from that machine.
- This ensures that it is from the right machine and it is from a system administrator user.

Keep Your Firewall Current

- Software needs to be patched every so often.
- Keep out attack packets and attackers build new attacks
- Sometimes, they can express that characteristic and see how it looks like.
- You want to have somebody who is in your organization that is keeping track of your kind of thing and need to know what kinds of defenses are suggested.
- Change rules to make sure you aren't vulnerable
- Much of this can be automated but things will turn out to NOT be right!
- Here at UCLA in the CS department, we have Vint Cerf
- Badass in the computer network world.
- He is like Jesus. Sometimes he comes to visit Kleinrock
- We would do things for Vint Cerf that we wouldn't normally provide
- Open the following port and he was one of the guys who build the original IP protocol.
- The problem is that Vint Cerf would be here for a day or two and then he would go away.
- This meant we have to close the port!
- We have a bunch of grad students doing a bunch of research in computer networks that normal people don't do.
- When one of Dr. Gerla's grad students needs to use a protocol, what are we going to do in the department?
- Open the following ports and the grad student will spend his next two years to work on his Ph.D dissertation.
- He will go on his to brand new job and no one will use his ports in this case.
- Important issue in practical, real-world situations that can cause problems
- You have to think of things for a little while because your business demands it.

Closing the Back Doors

- Capture every incoming packet and determine which are safe and which are not.

- This assumes they all go through the firewall and it allows us to deliver things from the outside.
- Wireless connections are really important for this purpose.
- Hard to force through the firewall because it travels through the wireless.
- If you are doing one of these things to allow devices like smartphones, then this works via wireless.
- Sometimes, it may be using the cellular network, so you have to be careful about such things.
- Your portable computer needs to connect up via a wired connection.
- That portable computer was out in the big, bad world and it is bringing a whole load of stuff.
- Sneakernet mechanisms involves putting it on some medium like USB flash drives where people can plug it in and send malware.
- There may be other entry points i.e. a modem via the telephone line
- This meant you could have a server somewhere and nobody uses this anymore!
- In a lot of cases, they are still sitting there and if you call that info, you will be in the network.
- Put a firewall at every entry into your network.
- If you have multiple firewalls, you have to make sure that all of them are kept up to date.
- Attackers will attack you where you are weak!
- You need to be aware of your weak points and shore them up to make them stronger.
- If he can get in the backdoor, you are screwed.

What About Portable Computers?

- Lecture 10, Page 11
- Bob and Carol are perfectly normal and legit users
- Xavier is coming in and his machine loaded with malware
- He connects to the Internet cafe and then compromises everyone else's machine
- Alice comes later and she gets compromised to.
- Bob's office has a wonderful firewall and there is a wireless network internally.
- Bob gets through the firewall and infects everyone because they never checked his machine.

How to Handle This Problem?

- Quarantine the portable computers and do not give them instant access.
- Do not permit connection to the wireless access point
- Put them in a constrained network and don't allow them to browse the outside network.
- This doesn't allow you to compromise everyone onsite.

- Network access control -> when you connect a machine up to the network, we want to ensure that machines are in good condition and will not cause security problems.

Single Machine Firewalls

- We talked about having a firewall that is a separate machine to protect your network.
- When your firewall gets breached, you will have no protection whatsoever so we need a 2nd line of defense.
- It basically says my machine will hope that it's firewall is indeed effective.
- We also want a software firewall built into my machine and it has to pass through my software firewall.
- If a machine tries to infect me, it has to get through my single machine firewall.
- Fully under control of the machine
- The network administrator can set up a configuration that locks things down and you have to be careful about this.
- It is under the control of the machine itself
- Its only purpose is to protect the machine and this is available on most modern operating systems.
- It comes bundled with the OS and it is merely a matter of finding it and turning it on.
- It may prevent things from happening that the user doesn't want.
- This is something we need to turn on in order to get the protection.
- The Mac OS allows you to do system configurations and lets you get to the firewall.

Pros of Individual Firewalls

- + Customizable
- You know your firewall can let certain web traffic through.
- If you happen to run Skype, you open the ports for Skype and it is under your control typically.
- + Can use in-machine knowledge
- Keep track of what application you are running
- Do things dynamically and open Skype ports when necessary
- + Provides defense in depth
- Attacker blows a hole in the firewall and this allows you to have attacks bounce off the other machines.

Cons of Personal Firewalls

- - Only protects that machine
- - Less likely to be properly configured
- People usually use the defaults which aren't as secure
- Overall, the weaknesses are acceptable

Encryption and Network Security

- One of the core techniques we are going to use is cryptography
- Encrypt a lot of the traffic that goes across the network
- You have made a very high-level statement and the actual security depends on details
- When you say you will protect the network traffic, a lot more has to be said.
- Multiple protocols to move data across the network
- ATNLP
- Where are we going to encrypt?

Link Level Encryption

- Lecture 10, Page 18
- Under ordinary circumstances, if we did nothing with crypto, it would be delivered directly.
- Protect messages with encryption
- Gets temporarily delivered and it gets decrypted and reencrypted with a different key and possibly a different cipher
- At that link, it gets decrypted and reencrypted.
- It keeps going off to the next chain and the process continues.
- Destination now consumes the plaintext and we have different keys and potentially different ciphers at every hop.
- Every router saw the plaintext

End-to-End Encryption

- Take plaintext and encrypt it with a key
- Goes without being decrypted to the destination and decrypts it.
- Normal way network cryptography is done.
- You can use link encryption and end to end encryption on the same network communications

Where Are the Endpoints, Anyway?

- Where is the start end and where is the destination end?
- Drop down into the IP layer and see where to put things
- Do it through a lot of the network via VPNs
- Which decision do you make and this leads to implications about who knows the keys and who do I trust?

IPSec

- Stands for “IP security”
- More or less secures IP
- Carefully designed Internet standard and widely accepted.
- Not the only way it is done but it is very well-understood
- People who build OS, routers, switches, etc. are supposed to understand IPSec to interoperate with their equipment.
- Usually, it is intended on an end-to-end basis.

- Reasonably general and it is supposed to sit below the transport level but on top of the IP layer.
- Slipped a layer between IP and TCP
- If you have taken 118 with Reiher, there was a lot of generalities in layers and it is NOT the full truth about what happens
- IP layer and TCP layer
- OSI model says that is what you have.
- Shoved a layer in between

What IPSec Covers

- Message integrity
- Message Authentication
- Sent by party who you think sent it
- Message confidentiality
- No one can read it except the intended recipients

What Isn't Covered

- Security associations
- Some of these topics are covered in related standards
- If you have crappy key distribution, you need to set it up in the right way.
- The key exchange is NOT part of IPSec standards

Some Important Terms for IPSec

- Security Association (SA): IPSec doesn't tell you how to handle this
- Relies on this and this is a key distribution issue
- Without proper key distribution, you are screwed.
- If you don't properly handle this you are screwed!
- Security association has a one way security channel with particular properties
- IPSec doesn't tell you how to handle how the receiver should handle this
- This is someone else's problem and that is what we wait to use
- SPI (Security Parameters Index) - take the security parameter index and uniquely identify what security association this works with
- Destination IP has to be with the packet and the IPSec protocol goes with it.
- SPI allows people who need to know how to match it up!
- Figure out which packet to run AES and key Z.
- Figure out which security association to use and which key to use as well.
- We will have some kind of data structure that keeps track of security associations that are ongoing.

General Structure of IPSec

- Designed to work properly with either IPv4 or IPv6
- Meant to work with a wide variety of different ciphers
- Distribute keys in any suitable way and it has sub-protocols
- Question of not being able to reach agreement

- IETF (Internet Engineering Task Force) - anyone can show up at a meeting and start arguing about a standard
- Companies argue about the standards to come out the way they want to.
- It can take years to reach agreement on critical points
- Sometimes, you get standards with a whole lot of options
- The better options rise above and the weaker options fall out of behavior

Encapsulating Security Payload (ESP) Protocol

- Even within this subprotocol, you can say that the user wants to have the following packets sent from X to Y.
- It has a payload and we will encrypt the payload and take whatever IP headers and send it off.
- Only the payload we want will get encrypted.
- Here is an IP packet with possibly multiple headers
- Take the whole damn thing and encrypt it
- No router in the Internet can deal with a full encrypted packet

ESP Modes

- Transport modes
- Take transport level protocol and encrypt its data
- Leave the IP headers alone
- Tunnel mode
- Take the entire IP datagram and encrypt the whole thing
- Slap some new IP headers on the whole thing and we can have unencrypted headers wrapped around

ESP in Transport Mode

- You can put it into one of these ESPs and encrypt these entire payloads.
- We can encrypt the entire payload and we will have characteristics of it like length.
- Add a few headers so people can know what the hell you are doing.

ESP Transport Mode

- Lecture 10, Page 29
- Data sent is encrypted and also authenticated.

Q. Why is ESP HDR not authenticated?
A. When this packet arrives, this IP header will only have generic info. It won't have any information at higher levels.

- Protocols we use are IPSec and ESP transfer mode.
- Since we need to know that, this has to be unencrypted.
- ESP Auth is unencrypted to figure out if it is from the proper person and we need to do our authentication
- If it does authenticate, we don't care if we did it on encrypted stuff vs unencrypted stuff

Using ESP in Tunnel Mode

- Send an IP datagram with a plethora of information
- If we are working in tunnel mode, we encrypt every bit of it
- We need to put something on the outside like a cleartext IP datagram
- All a router ares about is an IP header and at the receiver we reverse the process.

ESP Tunnel Mode

- Lecture 10, Page 31
- All this info gets encrypted and we can throw in a brand new IP header and this info is sitting at the end.
- He can deal with packets and IPSec people will look at this and say "Aha!", there should be some transmission information and we can check based on the security parameters
- I better deliver the IP packet since it has a full IP header.
- Deliver it to the machine that did all this info.
- After all, we have an IP packet.

Uses and Implications of Tunnel Mode

- Security gateway on each side of the communication
- Sending machine either do not know how to use it or will not pay the cryptographic costs.
- Configured to say we don't send unencrypted packets.
- I, the security gateway, will perform the cryptography on this packet and put it into tunnel mode.
- Take original packet and we machine X will do the tunnel mode encryption and put a header on it to deliver it to machine Y (which is the security gateway on the other side).
- Has some advantages i.e. sender and receiver don't have to even know how to use cryptography
- This is one way to say embedded systems still get encryption over the Internet
- We can hide the identity of these guys and we need to send on the guys on the Y side
- They will be able to look at the headers and see!
- Every packet on the sender side will have the entire packet encrypted and they will be able to send from X to Y.
- Conceal traffic patterns

Q. Can you use this to send to private addresses?
A. Yes but the person on the other side needs to understand what you are doing.

What IPSec Requires

- Protocol needs to be handled properly and do the right thing.
- Have some database on your machine with 18 current security associations
- Which protocol do I use and this is kept in the database.

- You have to have some kind of property that plugs in and there is the ability to plug in a cipher.

The Protocol Components

- Presumption that we have legacy equipment and did not speak anything else.
- We had to have IPSec work properly with those pieces of equipment.
- As long as sender and receiver send IPSec, the stuff in the middle doesn't have to understand.
- IP header and everything about IPSec sits after the IP header
- IP is a datagram protocol and treats each packet as a totally separate packet with no relation to other packets
- We sit below the transport layer but above the network layer, so we don't have any relation to the actual packets
- From the protocol perspective, no relationship between packets 1, 2, and 3
- Aren't we going to be using a cryptography mode to find feedback.
- We need to do it external to the protocol
- Give it a packet, it does what it is supposed to do, and then it forgets it.

The Supporting Mechanisms

- Keeps parameters for security associations
- It has to look it up and use cipher block chaining mode and the following key.

Plug-In Mechanisms

- We can do many different things and have these complexities built-in.
- There have been a bunch defined by standards and it doesn't really matter if the thing we have plugged in is a standard or not.
- Given a copy of the code, you can plug it into the IPSec and choose an unused number and work from there.

Status of IPSec

- Accepted Internet standard
- Widely implemented and used
- Windows 2000 and onward
- Linux 2.6 and onward
- Mac OS 0.6 an downward
- Architecture does NOT require everyone to use it.
- AES is the cipher is the de facto standard for bulk transport
- Eventually, it is expected that AES will become the default

SSL and TLS

- SSL - Secure Socket Layer
- TLS - Transport Layer Security
- IPSec was doing cryptography at the network level.

- Common standard for securing web browsing
- Lets you use crypto between two people without any prior communication
- What cryptographic mode are we going to use and how are we going to get a key.

The Basics of SSL

- Built on assumption of client-server operation
- Client, server, or both says we need protection
- You have to come up with agreements
- Negotiation can fail if the cryptographic standard is NOT up to par.
- After negotiation, here is how we will exchange our keys and then from that point on, our key will be encrypted and this is done at the application level.
- Stuff gets decrypted when it is delivered to the other end

Common Use

- Server provides an X.509 certificate meant to deliver a public key belonging to this party.
- SSL can be used so the client can provide the X.509
- Generally, the client doesn't have to do anything though.
- Client provides material to set up a session key
- Client and server derive the same session key and agree on
- All we know is he is communicating with somebody and no one else has that key
- This turns out to be a perfectly reasonable thing for web technology
- How can anyone do business across the web?
- The server does not give a shit who the client is.
- He can tie the client identity to the protected session he is working with.
- Set up SSL section and if the server says I have to know who you are, you provide the login identity.
- Nobody else can read the password except the server you are sending it to.

Crypto in TLS/SSL

- A few options are supported but it usually requires authentication
- Use RSA or elliptic curves for PK part
- Don't use public key cryptography for bulk encryption
- Symmetric ciphers i.e. AES, DES, 3DES (apply DES once with key 1, encrypted results with key 2, and double encrypted results with key 3)
- 3DES is the equivalent of a 112 bits number of key.
- Triple DES doubles your protection at the cost of doing multiple encryptions which have some room for error
- You don't want to use all of them because they are not all secure.

Use of SSL/TLS

- Core crypto of the Internet

- Used in all the major browser and every part of the world uses this cryptography

Security Status of SSL/TLS

- Don't use SSL because it sucks
- There is a lesson to NOT build your own cryptography
- You will fuck up for sure, so don't do it mother fucker!
- Early versions of TLS is not so secure and it is now fairly secure.
- If you use the wrong cipher, it is NOT secure.
- There are certain chosen plaintext attacks using all ciphers
- Not a problem in all uses of it.

W 6 Dis 5-6-16
Lecture 10

- Potential problems with DMZ
- Difficult to separate the two networks so it is hard to let people from the DMZ into the LAN
- It is difficult to correctly set up the firewall and people can get where they are supposed to be without getting into the internal network.
- Link Level Encryption
- Encrypts and decrypts after every hop
- Use symmetric keys because the efficiency of the algorithm actually matters here
- **You would use this to defend against traffic analysis**
- Infer something based on the source and destination on how long they are talking to each other.
- You don't ever put your IP header in your source and destination
- The only people who know where you are trying to get it to are people who are decrypting it.
- You have some metadata i.e. source, destination, and timing of the message.
- You can follow the packet and observe the timing and the Linked Layer defends against this
- TOR is a good example of this!
- No one knows both the src and dest in TOR
- You have a packet and you encrypt it with multiple layers
- End-to-End Encryption
- Uses asymmetric cryptography to set up a session key because they are only doing at the ends.
- If you use Diffie-Hellman, you could see the messages but you wouldn't be able to see the plaintext.
- Assuming the security model works, the nodes in the middle cannot reach the data.
- End-to-end would be used when you have the possibility that someone is untrusted in the network.
- IPSec

- Occurs in the network layer of the IP stack
- End-to-end encryption and the ends are the gateways where they create new sessions.
- What is the difference between link and multiple versions of end-to-end?
- Theoretically, the payload is never decrypted but the ESP headers are there to help out with the routing
- Covers message integrity, message authentication, and message confidentiality
- What Isn't Covered
- Non-repudiation: symmetric key example and people share a symmetric key.
- If you get taken to court for sending a message, you cannot say you didn't send the message. You CANNOT deny what you did!
- If you want to lie and deny things, you want repudiation. If you want strong guarantees to send a message and/or you are a legal signer, you want NON-repudiation!
- Signing with the private key is an example of non-repudiation
- They don't handle key distribution so it is difficult and sending it between routers tries to hide what is going on and you want it guaranteed that it is coming from this person.
- If you use symmetric keys, you cannot have digital signatures because you have a trusted third party to check.
- How to use the UCLA VPN?
- You have to get the Cisco AnyConnect and it has a Cisco client and it gets the keys.
- Another problem with some kind of security and anyone can download these keys.
- They authenticate you with a username and a password.
- A bit confusing to see this and we need to understand the intention with Diffie-Hellman
- Security Association
- What kind of services to you use to access the VPN?
- SEASNet and we want to know the data coming back is from someone who they say they are.
- Difference between TLS and IPSec?
- Can TLS/SSL do everything IPSec can do?
- IP layer is between the network and transport layer.
- TCP/UDP are the two types of transport layers.
- TLS only works for TCP and it doesn't work for UDP!
- IPSec should work with UDP because it is at a lower level!
- This is a huge difference!
- TLS would be faster because it is at a higher level and there is less encryption and decryption
- The IPSec encrypts the actual IP datagram and there will be retransmissions with constant encryption and decryption so this will slow IPSec down.
- Encapsulating Security Payload (ESP) Protocol

- The mode they add is tunnel mode and it can do tunneling and set up proxies for this.
- Netflix bought traffic outside the U.S. so they can use VPNs to watch movies.
- ESP Modes
- Transport mode
- You don't care where you want to hide things because you want it to go quicker.
- Tunnel mode
- Used if you want to hide where you were sending data.
- For secure communications like TOR would use this.
- ESP in Transport Mode
- You have to do the routing here and extract the frames i.e. TCP, UDP
- ESP Transport Mode Diagram (Lecture 10, Page 29)
- If anyone happens to be able to explain the packet format, we have an original IP header and ESP Hdr and the ESP Auth
- ESP Auth is probably some cryptographic checksum.
- ESP Hdr is used because the server doesn't actually know it is communicating with ESP
- If we send this packet out, anyone can participate and this requires some infrastructure and everyone needs to upgrade.
- Using ESP in Tunnel Mode
- Encapsulates a lot of details so you don't need the Internet infrastructure to handle IPSec.
- You can achieve IPSec security just by wrapping it in the original IP packet.
- The payload of the packet is the IPSec.
- ESP Tunnel Mode
- Same idea of not updating and this manifests itself
- Swaps the packet and puts it in a new IP header
- Big advantage
- Uses and Implications of Tunnel Mode
- Used when there are security gateways
- Has trusted handlers
- Plug-In Mechanisms
- People don't see the value or don't want to set up crypto security.
- Has to be incredibly easy to use
- Algorithms can get swapped out and it is a scheme for defining this protocol and this is used for plug in and play.
- The Basics of SSL
- Uses Diffie-Hellman
- Common Use
- X.509 certificate
- Actual Use of Encryption in VPNs
- Encrypted in the tunnel mode

- Used a Cisco AnyConnect to connect to the UCLA gateway and the IP would be the gateway.

Q. Is there any advantage to using the Cisco AnyConnect vs. anything else?
A. The one built on Mac is pretty good. There is more proprietary software with other things i.e. features and enhancements that Cisco has.

- Don't install a 3rd party app for this, VPN technology is not advancing at a breakneck pace.
- VPN standard is pretty well established.
- The General Solution For Wireless Security
- Uses link encryption rather than end-to-end encryption
- If you have your packet and you go to Google.com, what is the destination IP of that packet?
- Wherever Google is!
- What is the point of encrypting the link?
- You want to prevent sniffing.
- Think of the three acronyms
- Masquerading can occur.
- In order to get onto the Internet, you connect to a network which provides a service.
- People often use some sort of DHCP
- All these services are provided and if you do this unencrypted, someone can tamper with this information

Josh's Research

- End-to-End Encryption for What'sApp
- The source and destination in this messaging example would be user to user.
- If you use What'sApp, you can send info and support end to end encryption for it.
- What are the attacks and vulnerabilities of end-to-end encryption?
- What is the most vulnerable parts?
- Key distribution
- People have skepticism of this and sometimes we can use brute force.
- If you want a backdoor to end-to-end encryption, you can manage the key distribution
- Josh works on end-to-end privacy

W 7 T Lec 5-10-16
Virtual Private Networks

- Protect data on networks that we don't completely trust.
- One thing proven to be very common is that we have a bunch of trusted people.
- We have a bunch of people on the other side of the world and we are happy with both of the areas of the network that I am in direct control over
- I don't trust anything in between though!

- If you have a trusted network, you would't have to worry anymore, but the problem is that you usually have the Internet (which you usually cannot trust)
- You can rent a line and this gives you pretty good guarantees that no one else shares that line
- Pretty expensive
- You can use a VPN instead!

Encryption and Virtual Private Networks

- Not super sophisticated; based on cryptography and tunneling
- IPSec's tunneling mode
- You will get a more trustworthy network by encrypting everything
- You cannot quite encrypt everything because the network cannot deal with encrypted headers
- Everything else can be encrypted, so we want our enclave over here and let's set up a firewall at the edge of that enclave.
- Anything coming out will go through that firewall and we can apply cryptography to it.
- Encrypt everything coming in, decrypt everything coming out
- Instead of doing end-to-end encryption, the work stations don't have to worry about encryption at all.
- At any rate, we can tunnel the packets and instead of encrypting the payload, we can wrap the packet.
- There is a whole lot of packets going back and forth between those firewalls.
- It appears that these packets are going from one firewall to another firewall.
- There will be a lot of packets, but you cannot tell which machine is taking to which because you only see addresses

Actual Use of Encryption in VPNs

- A lot of VPNs rely on IPSec
- They have a high degree of assurance that it will work just fine.
- Because we are doing cryptography, this means we will have keys.
- The point at which they get encrypted will mean they use a key.
- We are removing them out of a tunnel and splitting up all the destination machines.
- The other firewall must know the necessary key to perform the encryption.
- Use primarily symmetric cryptography
- This leads to a key exchange issue; we should not do too much cryptography with the same key
- We have a very high overhead, and security protocols mean there will be multiple messages.
- We will use the same key for a lot of packets.
- How do we get the keys back and forth between the VPN?
- Did it manually in the past.

- Now, modern VPNs say we don't want to have all these by hand configurations
- Instead, they have automated mechanisms like IKE or proprietary key servers
- They change the keys every so often
- The issue is primarily overhead
- For a fairly strong cipher like AES, we can use the same key for a pretty long time as long as we are careful about key management

VPNs and Firewalls

- Often, the VPN is bundled into a firewall product.
- You want a firewall machine and you probably need more than just the VPN.
- While you mostly trust the guys here and there, what if there is a problem?
- You would still like some degree of protection from a firewall.
- You want the firewall to do filtering based on certain rules.
- VPN encrypts everything, so first you decrypt and then you apply those firewall rules.

VPNs and Portable Computing

- This isn't the only way VPNs are used nowadays, so in particular, if you are working with a laptop from off-campus and you want to use special UCLA resources, you need to use the UCLA VPN
- Run a VPN on a portable device, while the other end is across a firewall
- There is a big collection of UCLA machines and a bunch of other machines by people who work at or attend UCLA.
- This is a somewhat different situation regarding key distribution, and there wouldn't be a VPN between those two points.
- You have to have software running on your portable device to give you your VPN requirements.
- The firewall has to be built into the computer itself and this is where we use the VPN technology on your end
- We need the key on our end, so this requires key distribution
- All the UCLA affiliated people using a VPN implies we need approximately 5,000 keys
- If it was public, everyone can read everyone's VPN.
- Instead, we need a separate key for every single user, and this means our VPN machine is going to need to be more sophisticated in its key management.
- You need a robust key generation and key exchange protocol.
- There needs to be a piece of software that sits on your laptop to perform encryption and decryption
- Software has to be secure because it sees everything.
- When unencrypted packets pass through, we see things that pass through to other people.
- To get trust, you want it preconfigured on your machine.

- Put it onto your laptop or smartphone and since it was from a trusted authority, you believe it is legit.

VPN Deployment Issues

- 30,000 students and ~10,000 - 20,000 faculty
- Potentially, all of them could be using our VPN
- We have to download that software and we will have some downloaded code and if we use the UCLA VPN, we have to go to a website and download the VPN code.
- Download and install the applet
- Use the VPN and get the applet via web browsers
- This is going to be downloaded using a secure protocol i.e. SSL or TLS
- They will have a copy of the code sitting on the machine
- Q. Can't just about anybody install this?
- A. Yeah they can, but we only want to let legitimate users in
- We have another form of authentication and we can do this using user ID and password
- It is relatively easy to do that, and a portable computer might have been compromised and people put in bad code.
- This is a whole lot better than nothing from the POV of you (you are safe) and POV of UCLA (no random guy gets their resources)

Q. How does Cisco's 3rd party tool verifies this?
A. Cisco wrote and licensed the code to UCLA. At that point, it is our software but the signature of the code says this is Cisco code and this can be done by checking the certificates into our web browsers

- UCLA has been given the Cisco software and this is a protocol. There are just messages going back and forth between the two machines. We want the true copy of the Cisco code and we can try to verify that since it is next to impossible.
- We can determine if this is a legitimate user or not and figure out how to get software compromised or not.
- If user ID and password are being sent, we should be using a VPN so how do we configure a VPN
- We need to communicate patterns across the network, and it will be a variant of Diffie-Hellman key exchange
- If you are worried about the NSA, Difffie-Hellman is a bad idea.
- It relies on people who want to exchange the keys to agree on a prime #.
- They have to agree that this is the prime # they want, but essentially, it works by relying on the hard problem of factoring very large prime numbers.
- If you are running a Linux box, there is a particular number for the Diffiie-Hellman exchange but it is possible to solve it.
- It is NOT cheap, but people figure if you want to break Diffie-Hellman, it will cost about a few hundred million dollars and take about a year.
- That sounds infeasible for most of us, but the NSA has a black budget of a few billion dollars for this kind of thing.
- NSA wants to crack key exchange on every Linux box in the world.

- This means anyone using that prime is screwed since they figured out anything can be changed.
- There is a handful of parties motivated and wealthy enough to do this.
- All the security of our web browser is based on Diffie-Hellman key exchange

Q. Couldn't we say to use different prime number generators?
A. You need a very large prime #, and this isn't cheap. What is really going on is that you are using a library and we can do an elliptic curve version of Diffie-Hellman.

Wireless network Security

- Looks like part of the Internet and is typically a broadcast network.
- 200 meters or less
- Wireless networking is often done through mobile computing
- Pops up in another Wi-Fi access point, so this can be done with changing sets of people who work with them.
- Fairly open networks and lock down those networks.
- Essentially anyone can use them.

Types of Wireless Networks

- 802.11 networks
- Bluetooth networks
- Point to point type networks
- Still sending radio waves though.
- Cell telephone networks
- More beneficial to setup one of the networks on the top of buildings and send the signals across buildings
- A tight directional beam is much harder to eavesdrop on.
- This needs to be done way up in the air.
- Base this on difficult, physical characteristics
- Do this when you are moving things >= half a mile

The General Solution For Wireless Security

- They are inherently less secure than wired networks
- Put something on the wire and put something on via magnetic induction
- For optical, you cannot do that unless you are the point exactly where it ends
- Wireless is much less secure
- We use link encryption rather than end-to-end encryption
- Most things we learned previously used end-to-end including VPNs
- Wireless network encryption uses link encryption
- Cryptography is applied at the point we are sending things and this is removed at the other end of the access point.
- Not used on any further travels of that packet.
- Applied on that wireless link and my laptop is probably going to use link encryptions

- This gets decrypted as it hits the access point and the process repeats over and over as it is sent across the network.

Why Not End-to-End Encryption

- We need to do link encryptions because some non-wireless destinations aren't ready to deal with crypto
- Just because the guy at the other end isn't prepared doesn't mean you shouldn't protect yours
- Link encryption authenticates yourself to the wireless network and you have to use it in encrypted form and thus, we will get a stage at which we can ensure whoever is using my network is an authorized user.

802.11 Security

- A series of protocols set up by the IEEE over short ranges.
- Portion of the electromagnetic spectrum
- The first few versions had nothing whatsoever for security or cryptography
- It became quickly known that this was a mistake afterwards
- It is hard to change the standard, so people using the old version of the standard can use the new version of the standard.
- Once the standard is out there, we will have millions of units in the file and all of them have been designed to work with the standard.
- When we look at the 802.11 protocols, we realized we couldn't change it!

WEP

- Theory is that this is just as good from a security perspective to the access point that was very hard to tap.
- Change something without changing the protocol, and this is perfectly backwards compatible.

What Did WEP Do?

- Uses a stream cipher and we want a pretty big key.
- Use a key that is stored on the computer using a wireless network
- Have some kind of customization for those who want to use our cryptography
- Use an initialization vector
- Go through a negotiation and provide an IV to generate a key stream for you.
- Checksum for integrity purposes

What Was the Problem With WEP?

- Each access point should get a new session key
- You need to generate a new session key each time
- This requires an IV each time and this is to be combined each time so we have a method of exchanging with our partner.

- Because initialization vector is required each time, it makes key deduction attacks hard each time.
- 24 bit IV is too short and it wasn't carefully used.
- This meant if someone was listening in and they could inject messages, they could send whatever they want in terms of IV
- This allows us to figure out what that permanent key was, and we can figure out the session key.
- Done back in 2001!
- If you were in the vicinity of the wireless network running WEP
- Took less than 1 minute to get the key.
- WEP sucks ass!
- You might as well NOT have bothered using WEP!

WPA and WPA2

- Generate a new key for each session
- TKIP has vulnerabilities so we shouldn't be using it.
- AES mode has not been cracked yet.
- WPA2 is desirable because it does have AES in it!
- This is what we should be doing!
- Nobody has cracked WPA2 using AES
- If you have no cryptography running, switch to WPA2 and go to the configuration manual and click a box.

Honeypots and Honeynets

- Honeypot is a machine that is supposed to be attack
- Supposed to be attacked to be set up
- You want to learn more about what an attacker is doing
- Gauge potential threats and weaknesses about your own system
- If honeypot machine is kind of worthless, the only purpose is to attract attackers, get them into their machine, and learn what they are doing.
- I want to know what the bad guys are doing, and let's make them think they have one of our machines.
  - You have to make sure you are careful to look like they are good machines.
  - Most attacks nowadays are completely automated via scripts.
  - If they are NOT sophisticated enough, they might not be able to figure out they are in a honeypot

Setting Up A Honeypot

- Assuming everything else is okay, this machine will get compromised
- Set it up in such a way that it is relatively easy to get compromised!
- You probably for example want this in your DMZ and NOT in a more secure area.
- The purpose of doing it is to figure out what the attacker is doing.
- If you would like to know what software he has installed, you need software running on the honeypot machine.

- Secretly run software on this machine that the attackers thinks has been compromised
- The virtual machine has a lot of ability to observe what is happening in the physical machine
- People who built VMs try to hide the fact that you are in a virtual machine.
- If you are under attack, you already know that, but presumably, you might figure out you are under a particular kind of attack that wasn't supposed to be comprisable.

What Have Honeypots Been Used For?

- These traces can indicate what attackers do when they compromise a machine.

Honey nets

- A collection of honeypots on a singe network.
- Why don't I set up a whole network of machines!
- This is typically a set of machines with the same IP prefix.
- This sets up a whole lot of machines and we want it to look like there is a whole attraction subnetwork.
- It is very expensive to say we have a honey net and buy 100 boxes.
- Done with virtualization and set up a bunch of physical machines each with its own IP address and this appears to be a full network.
- Typically if you do this, keep everything you care about outside this network.
- Segregate this network from the rest of your real network with protected measures.

Q. This is like at trap, so if it is on a separate network, wouldn't they be suspicious?
A. It is not necessarily advertised that this is really a separate network.

What Can You Do With Honeynets?

- Good to figure out the spread of worms and the usage of botnets.
- Honey nets are one of the major tools to figure out botnets.
- Some packets get delivered to the target and we end up receiving a reply.
- A response will go back and we have to figure out where this will go.
- Let's say we have 100 machines:
- They are going to be completely passive and sitting around.
- If we have responses, somebody else must have spoofed the packets because they are making a DDoS attack.
- We will have the source address of whoever sent it, and we can figure out who is being attacked by a DDoS attack.
- If you are clever, you can deduce a lot of other things and we can figure out what percentage belongs to the honey net.
- Assuming random spoofing, you can figure out you got your share of responses.
- Multiply how many responses you got, and this is information on how the attack stops and starts.

- Figure out if it is a SYN flood or ping.
- Good work on deducing DDoS attacks on the Internet.

Honeynets and Botnets

- People know there is a capture of a bot through their honey nets
- They can use this information and observe its behavior to figure out other characteristics about that bonnet.
- Security firms need to capture bots form their honey nets.

Issues With Honeynet Research

- Bots try to spread to other machines, and this is an unfriendly process.
- You want to be careful about what you want the bot to try to do.
- You have to make sure it is clear that the bot is not working, so the attacker won't try to use a different attack

What To Do With a Bot?

- Look at machine language and try to figure out what is going on.
- If you develop the skill, you can figure out and read machine language.
- Figure out what this bot is doing and the bot manufacturer's doing like that, so they try to write code that is hard to figure out what is going on.
- People who are defending try to overcome obfuscation in the bot.
- Somewhat effective in tracking down the bot.
- It turns out that very few people write brand new bots
- Take an old bot and fiddle with it.
- Descendant of what I saw two months ago on the other machines

Do You Need a Honeypot?

- Probably not; only useful if you have a lot of time to spend watching the honeypot.
- All the good it does for you is provides information
- Large enterprises like IBM do want honeypots and you want people spending 10-20 hours a week figuring out what is happening to the botnet.
- The exception to small businesses is if your job is observing hacker activity

Q. Are there any legal issues if you use a honey net and it gets used to attack others?
A. Yes, if your machine attacks you, then there is potential liability.

- Nobody gets sued and they watched as they attack your system.
- It is morally difficult and you might have some legal ramifications

Lecture 11 Intrusion Detection
Introduction

- We have things like ACLs and they are different in many ways but they don't always works
- Inevitably happens: no perfect mechanism for defense

Intrusion Detection

- We want to be able to assume however good of a job we do won't be perfect.
- Sooner or later, someone s going to break into my system and wouldn't it be good to detect the fact that someone has done so.
- Having detected that, we can use countermeasures.
- Stop him from doing further damages and close off avenues to our system.
- In order to do this, we have to be observing our system and breaking through our defenses.
- We cannot normally have people who are watching all the time.
- There isn't hope that we can catch something at that speed.
- Some automated thing will detect if we have been attacked successfully.
- Give him the best information about what has happened to remediate the problem.

Why Intrusion Detection?

- Bad thing could have been achieved by separate operations.
- It may involve things we couldn't have foreseen.

For Example,

- Set UID root on your programs and people will have privileges to do whatever on that computer.
- You can have things on a UNIX system and never let this happen.
- If you can see 50 programs, you can report that we think we have a problem here.

Intrusions

- Detect intrusions in a systematic, automatic way.
- Any bad thing that can happen to your machine.

External Intrusions

- Evil hacker has broken through your firewall and someone outside has gained privileges to use them in some way.

Internal Intrusions

- Allow users to have some set of things and don't want them to have full privileges to do anything at all for these.
- Have these suitable to the job we are doing.
- people with limited privileges try to escalate their privilege.
- Can be quite harmful intrusion.
- Intrusion where someone with limited access gains less limited access.
- Internal intruder has some sort of foothold in your system.
- Many of the things you are done mean you can signal a problem
- He is a user of your system and he can probably talk to other people and gain more knowledge of your system.

Information From 2010 Verizon Report

- Worked with US Secret Service and indicated that external breaches were the most common
- 48% of all cases entailed something fairly serious happened to your system.
- In almost half the cases, back in 2010, there was often a component that was partially internal.

Basics of Intrusion Detection

- We watch what's going on in the system
- Observe behaviors we think characterizes the intruders and we want to detect legitimate access to the attack.
- We don't want to signal that there is a problem and we can devote some resources to watching what is going on.
- What cost are we willing to pay to check if we have detection at some level.

Intrusion Detection and Logging

- Logging is a very common thing that is done on the system
- Logs tend to get very, very big.
- We can use a whole lot of disk space and it will be very useful.

On-Line vs. Off-Line Intrusion Detection

- Examining logs and going on.
- Off-line analysis needs something.
- All of my logs and we need to detect there are bad things in the logs.
- We could be doing heavyweight analysis and doing big costs computationally.
- These resources might NOT be used during ordinary business hours
- Figure out what i going on after when the problem occurs.

Failures In Intrusion Detection

- The failures take on one of these characteristics
- False positives
- Everything is just fine and the system says "Oh my god we were attacked"
- False negatives
- Looks good to me but there are problems
- Subversion errors
- Programs can be poorly written and this applies to the intrusion detection system
- There have been cases where the intrusion detection system was the exploit!

Desired Characteristics in Intrusion Detection

- We want a continuously running system

- Fault tolerant: the attacker should not be able to make it crash
- Observe deviation: Figure out band stuff is happening
- Easily tailorable: Easily complicated systems so there will be a few systems and we have to detect these.
- Each person needs something slightly different from everyone else and take a small piece of software and tailor it to our own environment.
- We want to change the behavior of our system over time.
- We will have a change in the accounting system and as our system changes, what is good or bad will have to evolve to match our system.
- We want it difficult to fool, so we want to stay under the radar and we need to alter the intrusion detection system.

Host Intrusion Detection System

- One process running on your system and it is a background process of some sort sitting there.
- Going to be good at detecting problems at that computer.
- Examines logs of that computer

Advantages of the Host Approach

- Lots of info to work with
- A whole lot of info to it.
- Further, the host intrusion system only needs to concern itself with the host.
- Usually, information is available in a readily understandable form
- The other kinds of info are outputs of programs i.e. ps to get all the processes running in human-readable form

Network Intrusion Detection

- Something bad came in over the network and a lot of bad things happen on the network.
- Why don't we watch the network?
- The Ethernet we are running will watch all the information running back and forth across the network.
- Listen to all the packets going back and forth by sniffing the network.
- Run a little program and run updates on what we have seen.
- We can then generate the overall network picture.

Advantages of Network Approach

- Don't need to use up any resources on the users' machines
- This is a process competing with processes on another machine
- Host-based systems and we need to configure these.
- Network machines you have to worry about.
- A lot easier to observe things that are happening on multiple machines.
- This is something that needs to be relatively easier to see

Network Intrusion Detection and Data Volume

- If you have 100 machines, each has to normally handle 1/100th of a traffic, and this keeps these machines fairly busy.
- It has to be able to deal with each of these packets quite quickly.
- Vast amount of time to look at every packet in detail.

Network Intrusion Detection and Sensors

- Use programs called sensors to get only random data.
- If you design our sensors right, we have reduced our data volume very heavily and this problem becomes manageable.
- Throw away 99.9% of the traffic and this isn't relevant to network intrusion detection

Wireless IDS

- Watch the traffic that goes by and this is done for 802.11 networks.
- There are specific problems that occur in this kind of network and since I am running 802.11, I am running them.
- Break WEP in < 1 min by doing weird things.
- Intrusion detection system listening and you can see if someone is trying to break WEP
- Some guy with an antenna up and he is trying to break our WEP protocol.
- This particular kind of attack only understands lower level things

Application-Specific IDS

- You can have an IDS designed to deal with a specific protocol i.e. SQL
- Generally speaking, this is used if you have a machine that does something very specialized and very important.
- Detect if someone is trying to do something to your database server.
- Only look at application-level stuff and throw away other stuff.

Styles of Intrusion Detection

- Misuse intrusion detection
- Certain things are bad
- Anomaly intrusion detection
- Not sure what all the bad things are and we are going to watch the behavior on my system to see if it deviates on normal behavior
- Specification intrusion detection
- Try to detect deviations from "good states"
- If you are in a "good state", you are happy

Misses Detection

- I know these things are bad, I know what it looks like
- Watch for signs of these bad things occurring
- Signature detection: series of characteristics specific to that bad thing

Level of Misuse Detection

- Detect known attacks at a higher level

- Look for a bunch of popped running setuid protocols and run these
- One guy in the course of about a minute runs all of them and he is trying to become root
- File permissions changing quickly

How Is Misues Detected

- Examining logs but it is too late to detect
- Monitoring system activities
- You may not be in a good position to detect this
- Intrusion detection programs are processes running at the user level.
- Don't have full information about everything going on in the system.
- Scanning the state of the system
- Don't leave traces behind
- Sniff the network
- Works for network intrusion detection systems
- Only works for the network

Pluses and Minuses of Misuse Detection

+ Few false positives
  - If bad things occur, you know you have a problem.

+ Simple technology
  - Essentially, we have a bunch of patterns and behavior. Compare behavior and patterns for matches. Computer scientists are damn good at this.

+ Hard to fool
  - Hard for attackers to perform actions without signaling the system
  - Only true for things it knows about because it doesn't have any pattern to compare to.
  - If you don't know the problem ahead of time, you cannot do anything.

- **Only detects known problems**
  - **This is why this is a major weakness and not the recommended solution!**

- Sometimes the signatures are hard to generate

Misuse Detection and Commercial Systems

- Does misuse detection and many of these systems are very similar. Not all that different!
- Comparing patterns and how different is this from another program that does that?
- Not too different in high level architecture.
- What determines if you got a good one or bad one is your signature library
- Anyone doing this commercially allows you to do comparisons.
- Anyone who wants to stay in business will figure this out.

- Terrible flaw of only detecting known problem

Anomaly Detection

- We are going to look for things that are potentially not normal
- If it is bad, we should signal a problem.
- Build a model of valid behavior and try to observe deviations of the model.

Methods of Anomaly Detection
- Statistical models
- Follow the pattern of links
- Program behavior
- Normal way people use this behavior is this exits.
- Overall system/network behavior
- This system is used from 9-5 and we see which IP addresses it goes to.
- Suddenly in the middle of the night, we see TB of data being transferred.
- Expert systems can encode things automatically
- Various kinds of pattern matching
- Other methods that people use where there is a model and you compare the model to behavior
- Pattern matching sounds like misuse detection
- Things can blur together and we need to figure out what a particular system is.

Pluses and Minuses of Anomaly Detection
+ Can detect previously unknown attacks
+ Not deceived by trivial changes in attacks
- When looking for signatures, it can figure out things like changing variable names when plagiarizing code off GitHub
- Hard to figure out nature of attacks
- One of the nice things is that given what you know, you can figure out what the attack is going on.
- All you know is something is different from what it used to be.
- Shitload of false positives
- Can be expensive and complex
- Techniques can be quite expensive from a computational point of view.

Anomaly Detection and Academic Systems
- Research will one on anomaly detection
- The only thing we don't know is the misuse we are looking for.
- The only research to be done is to look for signatures of that attack and this doesn't excite us
- Misuse detection isn't cutting it, so we also need anomaly detection
- Truthfully, there aren't many systems currently using this.

Specification Detection
- Certain states that are good states and we don't know about the other states.
- As long as we go from good state to good state, we are happy as can be.

- We detect when the system is in a non-good state and we figure out the problem

How Does This Differ From Misuse and Anomaly Detection?

- Specification detection says this is good and everything else is bad.

Some Challenges

- What is this state we are thinking of?
- How can we get information about this state?
- Look for transitions between states and figure out if it is every hour, minute, or second.
- If it is possible to god rom good state or bad state, we can have a problem we don't detect.

Protocol Anomaly Detection

- Good way of detection problems because network protocols are very carefully defined.
- States in the protocol that are defined in terms of states.
- This means that we are already somewhere keeping information about the state of this protocol.
- Set of states that are acceptable to get into.
- We actually do build this into some systems i.e. Snort and Checkpoint

Pluses and Minuses of Specification Detection

+ Formalization of what we ar looking for
+ Limits where we need to look
  - Don't worry about it if it is not part of the state
+ Can detect unknown attacks
  - New vulnerabilities can be a problem
+ If you have 12 good states, you can have characteristics that don't match the 12.
  - You need to be able to specify the correct state
  - Few elements are defined as precisely as a protocol
  - You need to locate there might states
  - Attackers can do what they want without changing from a "good" state

Customizing and Evolving Intrusion Detection

- You cannot build one static intrusion detection system and make it work.
- Different load levels and other parties they interact with.
- Everyone's environment is different.
- Customize this!
- As things change, our intrusion detection systems must change with them.

How Do Intrusion Detection Systems Evolve?

- Manually
- Install things or build a new model of ordinary behaviors

- Figure out a good new state I didn't know about before and this recognizes good when it really is good.
- Somebody smart fixes this
- Semi-automatically
- Guy at Symantec is going to deliver stuff to you.
- Automatically
- Makes sense for anomaly detection systems
- Ordinary behavior changes over time
- Needs to change too
- System admin needs to run a new admin and have the system run something new.

A Problem With Manually Evolving Systems

- We cannot change things in this way.
- They usually change a lot more slowly and it needs to be from a software update schedule.
- Virus detection program and we need to update a virus database.
- That's how often these things change.

A Problem With Evolving Intrusion Detection Systems

- Attackers know you are doing this and want you to accept some evil behavior
- Move ordinary behavior and change things in a little tiny bit.
- A slight change in things from your system.
- Your system can have a new normal and sooner or later, he can nudge it into a realm of what's normal.
- Also possible with manually detecting system, but it is harder because he has to convince you to make the change

Intrusion Detection Timing

- You can get false positives in your system and false negatives
- Tune to increase one at the cost of another
- Usually, you cannot decrease both of them.
- Which choice should you make?

Practicalities of Operation

- Generally speaking, intrusion detection systems are program you install on your computer and run and it is essential y normal application.
- Logs that can sniff packets and make system calls

Practicalities of Audit Logs for IDS

- Operating systems can only log what they figure is important.
- Intrusion detection system was set up to diagnose bottlenecks or bugs in your system.
- Since the attacker knows you have an intrusion detection system running, he will try to edit your log.

- Very common for an attacker to do.

What Does an IDS Do When It Detects an Attack?

- Automated response
- Shut down the “attacker”
- Look for SQL injection and make it run in a more careful way.
- Set an alarm and the machine over there is under attack.
- Don’t take any actions
- Write it in a log and hope someone looks at it one day.

Consequences of the Choices

- Automated
- Too many false positives
- Automated response may not be effective
- Alarm
- How often can you send those alarms
- Another false positive and you are NOT going to take any action
- Stuff happening at computer speeds
- Can he take actions fast enough to save you?
- Logging
- Doesn’t really do anything since it isn’t an active method of finding a solution

False Positives and IDS Systems

- For an automated résponse, things shut down.

Consider a Case for Manual Response

- Your website gets 10 million packets per day.
- Your IDS has n FPR of .1% on packets
- Say each one takes 1 min to handle
- That is 166 man hours per day
- You need 20+ full time experts just to weed out false positives
- NOT a feasible thing to do!

W 7 R Lec 5-12-16

What Are Your Choices?

- Tune to a lower FPR
- It causes more false negatives, so more bad stuff will get through
- If FPR is too low, what is the point of running the system?
- It lets everything through
- Military and big companies have a triage system that runs signals
- Runs through an automated system to filter out signals in the hopes of getting rid of false positives
- The actual analysis that can distinguish between a false positive and a true positive is computationally complex.

- Can be very expensive and now you can reduce the problem to 1/10th of 1%
- Spend a few resources pulled out that way.
- Some human being is going to look at this.
- You might run a system where it is people's jobs to look at these.
- If it looks suspicious, send it to a human being
- Happens at big companies like Google
- Spend one to two minutes to look at signals coming into this thing.
- The ones I am not confident on will be sent to another level with an analyst checking it.
- You need extra resources to do all this automated work.
- You can start using human beings who are expensive to use the signal.
- Systems that are running 24/7 and attackers are working 24/7
- You need somebody watching AT ALL TIMES, so this is a very expensive thing.
- You need probably 3 people per day, who are willing to spend 8 hour workdays.
- If you do that, there is not much point in having this system.

Intrusion Prevention Systems

- IPS: People say this is quite different from IDS system
- Takes an automated action
- You don't just signal to a human user; you are going to automatically take an action and this is much more rapid.
- There will be a lot of time when the attacker will be doing stuff and you need to take remedial action in an automated way.
- This could be very bad!
- You need to be sure that either you have a very low false positive rate or the actions taken will NOT be crippling to your system.
- Run web sales and if your IPS system gets a false positive and turns off all access to your web server, that is really bad.
- Almost as bad as an attack.
- What people do can be very sophisticated!
- However, you often have a firewall and it isn't often done at all.
- It is possible to have an intrusion prevention system that is more careful than this and you can reduce the privilege of your web server if you are afraid your database is getting attacked.

Sample Intrusion Detection Systems

- Snort
- Bro
- RealSecure ISS
- A lot of commercial entities who are in this industry
- If you buy a bundled security solution, part of the bundle is an IDS system

Snort

- Network intrusion detection system
- Meant to look at packets coming in from a machine
- Public domain
- Open source
- Designed for Linux, but also runs on Windows and Mac
- Intended to be very extensible
- Framework with a bunch of rules and enforce these rules to observe what is going on
- Basically have rules where you plug things in and have rule-based description of traffic
- Very widely used

Bro

- Developed at Lawrence Berkeley Labs
- More sophisticated methods than Snort
- More anomaly-detection oriented
- More general and extensible and NOT as widely used

RealSecure ISS

- Bundled into IBM security products
- You are running your big office environment with 2000 users and all these servers
- Have a client-server architecture and run this on all your machines and you can analyze your output from many sources and figure out what is going on.
- Network detection and host detection - components will try to detect problems that have arisen on that host.
- Hosts can then reply to the server and see which machines have been compromised.
- Dedicated machine that runs the server and it will be fairly well configured and safe.

Is Intrusion Detection Useful?

- A few years ago, 2/3 of people in a particular surveys said they did use one.
- The other half used intrusion prevention
- Gardner group reported that IDS is failed but they were wrong!
- Signature-based IDS is especially criticized
- Similar to virus detection, which has some serious shortcomings

Which Type of Intrusion Detection System Should I Use?

- Here are some advice on what I should do:
- Run several IDS
- Many types like ones of different host and networks
- Each detects different things and we can catch a wider variety of attacks.
- Having multiple systems is good defense in depth

The Future of Intrusion Detection?

- Less clear what is going on in the future.
- Thought up in the 1970s and people have been working on them ever since.
- What has been claimed to be possible in IDS systems has never been realized.
- If prevention fails, what will you do then?
- You need a system administrator watching what is happening to catch your technical problems.
- A lot of attacks are NOT detected for a very long time.
- Some are NEVER detected
- Bad because bad guys can do whatever they want.
- We want to be better at developing these types of intrusions.
- If you are serious about security, you will use them.
- They aren't perfect, but they are better than nothing, no?

Conclusions

- The ones helpful in security are not terribly sophisticated
- They aren't all that complicated than firewalls for example
- Research improves them with more sophisticated techniques, but we might not achieve what we wanted originally.

Lecture 12: Malicious Software

Introduction

- Automation on your computer is really helpful
- Tools to build programs, debug things, etc.
- Attackers are fundamentally like you with some background in working with computers
- Attackers have built software that have done much of the dirty work for them
- Previously, in the 1980s, kids in basements with modems would try to break passwords by hand.
- Nowadays, almost everything is automated.
- Special types of software to perform these tasks.
- Speed
- Build malicious software faster.
- Mutability
- Changing attack measures
- Anonymity
- Instead of tracing commands back to you, a computer can do it for you while the attack is going on.
- Makes it hard to trace attack back to the attacker

Where Does Malicious Code Come From?

- Somehow, the attacker persuades you to download and run a malicious piece of software.

- You often download executables when you go to websites, and these can be run automatically and we can have some way to convince you of running them.
- Even if you buy commercial software from a website you know they are reputable, there could be malicious code in the software that got in there without their knowledge.
- You can give them access to the machine and there could be software flaws to exploit.
- Insiders can have the privilege to install software when he wants to and he can introduce malicious code.

Magnitude of the Problem

- In 1994, there were 1,000,000 annual infections
- In emails across the network in November 2003, approximately 1 email out of 93 contained a virus
- You can get a virus in your email everyday.
- Spam contains attachments with malicious code.
- 2008, 50% of respondents had malware
- 20% had bots
- 2009, when people discovered they had problems, it had been compromised for almost a year.

Q. What are bots?
A. Botnet types of malicious software.

Viruses

- Thought up more or less in the 1980s
- Reiner was in a meeting as a grad student and a USC student created a computer virus.
- In their CS department, he could introduce one malicious piece of code, and suddenly it would be everywhere.
- Self-replicating program that is going to spread into other programs
- These programs would try to copy their code into the other programs
- Analogous to human viruses which spread through a host.
- Not a standalone program; instead, malicious code is added to a program that looks like you have a good program.
- In addition to whatever the program si supposed to be doing, it spread the virus from machine to machine.
- Not all malicious code is a virus!
- Term is abused and many people who claim they have a virus might NOT have a virus.

How Do Viruses Work?

- Somer or later, the program runs and it runs via cronjob or whatever.
- In most cases, it has the full privileges of a user and it can do certain things on the system.
- Includes the ability to write to other programs

- It may well-be that the person running the infected program has write permissions
- Okay, I am running, let me look for other executables.
- Is this executable already infected?
- No! So let's change the 2nd executable to have virus code.

Infecting the Other Program

- Lecture 12, Page 11

Macro and Attachment Virus

- Written into machine code
- Binary executables
- Hard to take a running load module and change the code and make it do something else
- Easier to write in a higher level language
- People attacking with malicious software would require executable code and it would be a whole lot easier to work with than machine language
- A lot of programs have had the ability to be extended, not only in the program itself, but also the data
- Powerpoint, Excel, Word docs, etc.
- In addition to having data it can describe, it could also contain a macro that is executable like any other program
- Email attachments
- Web pages
- In many formats, you can change the contents of these executable elements
- Put new macro in word document as long as you have write permissions on that word document
- Saying the machine language code has to fit in any file is very hard; fitting in a macro is much easier!

Virus Toolkits

- Developing any code from script ch is tedious.
- There is NOT need to rewrite repetitive code, so let's use software tools to do this!
- Attackers built virus toolkits to help those without machine language expertise
- Insert code in executables and make it run every single time
- Create a brand new virus each time
- The good thing for defenders like us is that each virus looks relatively similar
- Inputs you are getting from the high school kid are very unlimited.
- How much uniqueness can come out from this limited input where you can create code?
- Look for particular things that are characteristic of coming from toolkits.

How To Find Viruses

- Scan for signatures and look for multilevel generic detection

Precautions to Avoid Viruses

- Don't import untrusted programs
- Who do you trust though?
- Do you trust that code or not?
- Websites have a whole lot of things that are not created by the website.
- Advertisements like Taboola who didn't write it, so you go a couple of steps back to find where the code came from.
- Viruses have been found in commercial software
- A bunch of hackers who created Back Orifice were embarrassed to get infected themselves
- If you say to trust someone, you NOT ONLY trust their honesty, BUT ALSO their caution
- Try to maintain trust and hope it is well-warranted.
- You could still get in trouble because you trusted them.
- Are they someone who is careful with what they do?

Other Precautionary Measures

- Scan incoming programs for viruses
- You need to be smarter than they are because they are hiding and you are seeking
- Limit the targets that viruses can reach via privileges
- Code executed out of the web browser would NOT have the right to write most executables on your file.
- You can bring in malicious code that wouldn't have privileges because they moved away from this model.
- People hated this because it broke a lot of legitimate features of Windows 7.
- Monitor updates to executables carefully
- If the guy with the privilege to access executable can change permissions, this doesn't help!
- Anytime an executable changes, I want that to be logged.
- You need a broad definition of "executable"
- Word documents are written all the time, and they are meant to be mutable.
- If he has changed your word documents, you don't want to be alerted too many times.

Containment

- Program that is captured is NOT harmful until it runs.
- Once it runs, it has full autonomy and privileges.
- If you don't trust it, it cannot do malicious things and we wouldn't be in trouble.

- Requires you to set up virtual machines in which you run the executable with a limited set of privileges
- This does require that you need a virtual machine and give the right privileges to the virtual machine.
- If the VM has the same privileges as everything else, you have NOT helped
- We need just enough permissions but not too much
- Requires judgment and most people don't understand the consequences of giving a particular privilege to a normal person
- We cannot expect normal users to foresee the consequences of adding privileges to libc

Viruses and File Sizes

- Take a remediate action once we find the virus in 5 of our executables.
- After virus came in, the virus cannot do anything new besides what was already there.
- Keep track of the size of executable files
- Not going to work for macro virus very well because the document changes in size if the owner edits it.
- A typical program's machine language is often NOT very tightly packed.
- There can be No-ops in various places.
- Build viruses in such a way to look for empty space and replace what is in empty space.
- If you can find enough empty space, the size will NOT grow!

Signature Scanning

- If I only had a hash, when he replaces no-ops with his own instructions, it will come up with a different hash.
- Save things if you are sure you have a good executable.
- This is something we can do as well.
- Let's not bother with all of that; there might be a particular reason for corruption
- If we knew for a fact that a virus always contains a bit pattern, to determine if we have an infection, we can use that to detect viruses.
- Virus detection scans signatures to detect problems!
- What if you are downloading an executable you have never seen before?

How to Scan For Signatures

- Create a database of known virus signatures
- Look for matches against the data pattern.
- A few other places besides the file system and this can help check boot sectors.
- Virus detection from Symantec look primarily for other types of malware.

Weaknesses of Scanning for Signatures

- What if the virus takes active measures to prevent you from finding the signature?
- You can only scan for known virus signatures
- If you cannot find it in your database, you don't know what the virus is.

Polymorphic Virus

- Creates various versions of itself
- Challenging to do because if complexity of copying itself

Polymorphism By Hand

- Malware writers have made a living selling malicious code and breaking into shit
- They know defensive approaches that people are currently using.
- A good hacker company maintains an active subscription to every anti-virus program in the world.
- If Symantec can detect our malware, we will change it until it is NO Longer detectable.

Stealth Viruses

- I have my database and a whole bunch of files that may or may NOT be infected.
- Read all of the bits out and provide it to my virus detection program to do a match to it.
- Stealth viruses try to undermine the number of bits
- They are aware of this content information and cautious in changing it.
- Open a different version of the file that isn't infection
- The Brain virus does this
- You need a resident virus: the malware needs to be continuously running on the computer, so it needs some presence running in the background
- How do you know which processes are running your machine?
- Why don't I just change the PS command?
- A. There are a lot of clever things that hackers do.

Combating Stealth Viruses

- If I have a clean source of the system. If it has NOT been infected, I can reboot off the instead of rebooting is on the disk.
- Concerns that the malware can be elsewhere like the boot sector.
- People have put viruses and malware instead of the memory of peripheral devices.

Other Detection Methods

- Intelligent checks analysis
- Files like word docs will change.
- We care if the piece of malware is malicious and look for attack invariants
- Identify and handle "clusters" of similar malware.

- Anything trying to hide in a particular way can be looked for and used as a sign

Preventing Virus Infections

- Run a virus detection program
- Unless you disable it, a virus detection program si going to keep asking you about it.
- Disable program features that run executables without users asking
- Automatically running code
- People have started putting malicious software and putting it in flash drives

How To Deal With Virus Infections

- Get to version of OS that you are positive that cannot have been infected
- Mac OS X - DVD that comes from the system and you have a pretty good intuition that it is right
- Similar to Windows

Q. Reboot, secure boot on Windows or no?
A. Not reinstalling OS because a problem might NOT be present there. Work with a version that is correct until you understand what the problem is.

- Some of the files on the system including legitimate, important files may have been infected with malware.
- If you have a clean backup, you can bring that in case.
- Proof-of-concept code that showed infection of firmware in peripherals
- Make sure your OS hasn't been corrupted!
- Fortunately, no one is actually using this attack in the present day.

Disinfecting Programs

- We aren't going to say to throw away all this stuff; we want to remove infected code.
- It is quite hazardous since you could get it wrong if you cannot get rid of the malware.
- Make sure to take other stuff with it like a program with a lot of data.
- Not worth the trouble to clean out an infected file.
- You better have it saved somewhere in backups.

Trojan Horses

- Seemingly useful program that contains code that does bad things
- Comes from the ancient story of the Greeks and their war against Troy (where Istanbul is today)
- They had NOT succeeded into breaking through the walls
- Pretend to give up and had enough of this war and because we felt so sorry, we built a huge wooden horse and they will let them in
- They will take the huge wooden horse and celebrate.
- The huge wooden horse was filled with Greek soldiers and the Greeks slaughtered all the Trojans after they were partying.

Basic Trojan Horses

- Program you pick up somewhere that you might have even heard of
- It does a little something else as well that is pretty bad.
- Could be subtle or flagrant, but it is common to put these in games or applets.
- Bogus security products (download this shit and we will make your computer secure again)
- Don't accept these kinds of offers for security products.
- Flash drives are another hardware vector and we want to plug it into the machine.

Recent Trends in Trojan Horses

- GozNym Trojan steals money from infected customers' back accounts
- AceDeceiver Trojan targets iOS devices
- USBThief Trojan targets non-Internet connected devices
- Infected via USB
- Marcher Trojan
- Pretends to be an Adobe Flash installer
- Very popular way and if you want to do a wonderful thing, you don't have the right version of Flash and I will show you the wonderful stuff on my website
- Adobe has terrible software flaws
- Triada Trojan
- Alters SMS messages from Android devices

Trapdoors

- AKA backdoors
- The attacker would like to get into a system and you may get rid of that malicious piece of code, so while I am here, can you allow me to get into your system without knowing it.
- Compromise one of the network facing applications to let you get into the system.
- Often found in login programs or other system utilities.

Trapdoors and Other Malware

- Malware can allow an attacker to get back in
- You should work on the assumption that malicious software installed trapdoors even if you get rid of that malicious software

Logic Bombs

- Typically things that occur in a legitimate program, often very important
- Explodes under certain conditions (does something really terrible such as deleting all your data)
- People often are NOT very nice an they will get back at you some way, somehow.
- Often used by disgruntled employees

- Former TSA employee got two years in prison
- If you do write malware, don't write a logic bomb in a program you are actually developing for someone, so don't do this shit!
- People don't go to prison if you put logic bombs in nation states i.e. North Korea
- Caused problems or South Korean banks

Extortionware and Ransomware

- Attacker breaks in and does something undoable
- He won't do it unless you pay him money.
- Break-in is often done through another mechanism like "social engineering"
- Encrypt everything because that ensures you get the vital information
- Hospitals are good target because they often have patient data on treatment and conditions
- Hospitals cannot have this information unavailable.
- When they break in, they try to take over your backups and encrypt it for some period of time so they can ensure you don't have any other version available.
- One thing you should do with backups is to make sure they are actually okay and that you can restore from backups.
- These aren't timed or triggered, unlike logic bombs
- These are something where there is a human attacker and he causes it to happen.

Worms

- Virus tries to move from executable to executable; worms try to move from computer to computer
- If you release it onto the Internet, it will spread to many computers
- The interesting thing is you then move on.
- Infecting other machines is the primary goal
- The Internet worm is the most famous example

The Internet Worm

- Created by a graduate student at Cornell in 1988
- Released accidentally on the Internet in Nov. 2, 1988
- Internet was NOT popular at this time, but many universities and companies did have this.
- People who had it started to rely heavily on it.
- It spread really rapidly and hit 6,000 machines
- Significant fraction of the Internet

How Did the Internet Worm Work?

- These variants allowed you to execute improperly on processes

The Worm's Actions

- Said "Let's infect these systems that were unaffected"

- Didn't do it quite right since it re-infected already infected systems
- Created a new process each time
- You could end up with hundreds of processes on these machines and this would cause tens of thousands of processes running on the Internet
- This meant that systems were going to wedge (stop working)
- Process after process had no room and they would NOT be able to do anything
- What happens if you go in and kill a bunch of processes
- They will do it faster than the system administrator can kill them.
- People didn't know what was going on because they had never seen anything like this before.
- As far as everyone knew, anything on the Internet could be taken down
- Reiner worked at JPL and it had bad things
- Cornell student was prosecuted and got some probation and got a professorship at MIT

Stopping the Worm

- Internet type problem and then kill all those processes
- Get rid of all machines that had the worm's infection
- Take all this machines offline and get rid of all the processes infected until the worm was cleared out.
- We had to patch the flaws so we didn't get infected again if it was put out again.
- Firewalls didn't exist in 1988!
- If you are the Internet, you wanted to get any interesting packets at the time.
- Bad idea!

Effects of the Worm

- Around 6000 machines were infected, which took substantial disinfection
- Many machines were brought down or pulled off the net as a precaution
- Smart people got together and used email to communicate on uninfected machines, so it would have been tough to figure out what happened.

What Did the Worm Teach Us?

- The existence of some particular vulnerabilities
- Showed the weaknesses of connecting to the Internet
- It was NOT clear at the time
- Showed the dangers of trusting others
- Trust could be misused since we could use trust that other people had to spread the infection
- People trusted the infected machines and this led to them being infected.
- Denial of service is easy
- Security of hosts is key
- We had to fix hosts in the end to get rid of network level threats like botnets

- Logging is important
- Look at the logs but this is how they found the problem
- Observing the logged information about what had been occurring on the machines

Code Red

- This was NOT some graduate student's experiment out of control
- Attacked Windows machines
- Caused a lot of problems

How Code Red Worked

- Didn't know who would be susceptible so attempted to connect to TCP port 80
- If successful, sent HTTP GET request to deface the web pages on your web server

More Code Red Actions

- Triggered a DDoS attack on a fixed IP address at a particular time

Code Red Stupidity

- Chose a really bad method to choose another host
- Used a random number generator
- Requires a seed and put it into the executable
- When put into another machine, the seed came along with it.
- Started using the same set of random addresses to look at
- First guy got knocked on by every single Code Red infected machines
- All the Code Red machines chose the same machines!
- DDoS attack on a particular fixed IP address, so we could just change the IP address

Code Red II

- Used smarter random selection of targets and didn't reinfect machines
- Added a Trojan Horse version of Internet Explorer
- Left a backdoor on some machines
- Doesn't defuse web pages but you just wanted control of those machines
- Didn't turn on periodically

Impact of Code Red and Code Red II

- Code Red infected over 250,000 machines
- Code II is essentially dead
- But someone might revive it
- Code Red is still there but it is still idle and only infects very old versions of Windows
- There are still machines that run Code Red but it has already visited every machine that has that vulnerability

Student

- Scary worm that showed up in 2010
- Targeted at SCADA systems
- Attacks industrial equipment that controlled other physical machines
- Targeted at a particular type of machine that controlled centrifuges that enriched uranium.
- Attacked Natanz, Iran
- Why use a worm to attack one place?
- Very hard to attack, but if we want to build something that is fairly aggressive, maybe there will be some path we could use to get there.
- There are two air gaps (computers that aren't connected to a network)
- How do we get across an air gap?
- Computers can temporarily connect to download updates OR use USB drives
- We don't know what path to take or have a pay to carry in things, so instead, let's see if we can get the code to move in there on its own power.
- Destroyed centrifuges and made them useless.
- This was very specifically targeted

Where Did Student Come From?

- Stuxnet came from the United States and there is some speculation it came from Israel
- There is no solid, reliable evidence that this is the source.
- This could have been created by the U.S. military or intelligence agencies
- People argued that this came from the NSA that this came from SCADA attacks
- Non-expert NSS Labs researcher easily broke into Siemans system (which is a branch of SCADA)
- Someone got hold of Stuxnet and used it to steal certificates

Worm, Virus, or Trojan Horse?

- Term often used interchangeably
- Trojan horse refers to a seemingly good program that contains evil code
- It does something bad
- Viruses seek to infect other programs
- Try to infect every program on the computer
- Worms try to move from machine to machine
- "How to own the Internet in your spare time"
- You could infect the entire Internet in 10 minutes.
- More targeted about where it wants to get to.
- Don't obsess about classifications

Botnets

- Attacker has compromised a lot of machines
- He broke into those machines and can run whatever he wants on those machines

- Organize his collection of machines to work with them as a unit.
- Uses basic distributed system techniques
- Typically used it to perform an attack
- You often want to use a lot of horsepower i.e. DDoS attacks

What Are Botnets Used For?

- Spam (90% of all email is spam)
- If you send enough spam, you can eventually make money
- Sooner or later, people who don't like spam figure out those machine are bad, so these machines get blacklisted
- Hosting pirated content
- Better than putting it on your own machine because someone might sue you
- If you put the pirated content onto a botnet, they will sue a poor old grandmother.
- Hosting of phishing sites
- Download malicious software and there will be people looking for phishing sites
- Keep finding new sites to host your phishing.
- You can look through those hundreds of thousands of machines and look for things that are valuable.
- Build bigger botnets and these will eventually be ineffective.
- Bonnet requires increasingly fresh set of machines that aren't blacklisted.

Botnet Software

- Software should control the machine and participate in distributed system activities
- Send SYN's and end at this hour
- Get everyone together and make sure everyone agrees
- Consult with other bots and get people coordinated to perform the attack
- This allows downloading of new code
- You want to be able to update you botnet and have a better version of your bot control software.
- Guy controlling the botnet has to be able to talk to each other.

Botnet Communications

- Shut down the IRC channel and then the bot cannot communicate anymore
- Set up a peer network
- Peers, super peers, multiple paths to get to any other point in the network.
- Conficker's botnet uses peer techniques
- Security botnet tends to have security built-in
- Messages coming in across the network needs security because anyone can send those messages.

- I am going to stop those messages from being sent and in some sense, another attacker who wasn't your buddy can figure out your communication mechanism and take over your botnet.
- Built-in security and use passwords and all kinds of stuff.

Botnet Spreading

- If you were successful, you would dump botnet code
- Nowadays, used through phishing and Trojan Horses
- Human beings can install bot code and attackers can be religious.
- No one true way to spread the botnet.
- Any attack you can think of could be used to achieve your goals.
- Anyone running a peer file sharing network could be used to copy to your machine and run it.
- Regardless of the details, your mechanism will be automated nowadays.

Characterizing Bonnets

- Most commonly based on size
- Conficker had over 5 million

Why Are Bonnets Hard to Handle?

- Scale
- Anonymity
- Lego and international issues
- You have a ton of machines in many different countries
- You cannot legally go to a machine remotely and get rid of a configure botnet
- You cannot force people to clean up your nodes.
- You will try to interfere with the bot operations
- If you are setting up a peer network, we will attack the super peers.

Approaches to Handling Bonnets

- If you are Amazon, you won't shun all those who are on the Conficker botnet because you will lose 5 million potential customers.

W 8 T Lec 5-17-16

Why Are Bonnets Hard to Handle?

- They are dangerous and do a lot of harm
- Why are we having trouble getting rid of the,?
- They are big so it is hard to get rid of them
- Anonymity because it is hard to link it to someone
- With such large collections of machines, they can be scattered across different countries which have different laws regarding cybercrime.
- Go to whatever efforts you need to know the nodes in the botnet.
- What are you going to do about those 100,000 nodes?

Approaches to Handling Bonnets

- Clean up the nodes
- Most people who own machines with bonnets don't know about it and don't care.
- If it isn't causing the owner to have any particular problems, then the owner likely doesn't care!
- People infected often have a hard time getting rid of the bonnets because they aren't very technical in the first place.
- In particular, we cannot force people to do this.
- No legal requirement to remove malware form your computer.
- Bonnets are distributed systems and they have to have communication between various elements of the system
- Let's interfere with the operations of the botnet.
- Get rid of some of the messages and filter them out.
- If you are trying to interfere with a bot on a 3rd party machine, you may NOT have any legal right to do anything.
- Bonnets are often built as peer-oriented network
- This ensures connectivity in the botnet among super peers.
- This won't involve millions of nodes, but rather than hundreds of nodes
- Target the super peers to weaken the botnet and destroy them.
- Super peers can be put into one ISP, so you need the cooperation of one organization.
- Shun bot nets
- They are infected machines but usually what they are doing is legitimate
- Companies aren't interested in dropping traffic as long as it is NOT problematic
- Most of the code running out of the computer is a result of user actions and not the bot's actions

Spyware

- Software installed on a computer that is meant to gather information
- Put onto your computer without your knowledge
- Once the software is on your computer, it has a great deal of privilege regarding what you do.
- Very likely, you gave it the same privileges that you have.
- I would like to know what is going on in this computer, these records can be relatively innocuous like knowing every website you are looking at.
- This is often done to give better targeted advertising.
- Get the ads you really want to see because you will get paid more if it is targeted to someone who might really buy the thing that is being advertised.
- Sometimes, it is NOT truly evil stuff, but other times it is i.e. stealing your password, SSN, credit card number
- Ideally, spyware is something that you don't know about on your computer
- Violates privacy of the machine's owner
- Gathers info about you and reports it to a 3rd party.
- Stealthy behavior and it is hard to remove

- People who create the spyware doesn't want you to remove it!
- They put trap doors and alternate sources or other methods of watching you.

What Is Done With Spyware?

- You could have a piece of spyware by someone who burglarizes houses and you can watch what happens on your computer and steal all your stuff.

Q. Targeted advertisements?
A. Different parties who have internet commerce and you are looking for a good advertiser who will put it in the right place.

- Special methods of telling who will be interested in your product.
- I will pay you to target my ads!
- This could be perfectly legitimate.
- Fiddling with web browser cookies. Trace all the websites you visited and this gives you information regarding what you are interested in and how to target his advertisements.
- Use techniques to get information i.e. deceptive things

Where Does Spyware Come From?

- Usually installed accidentally by computer owner
- Certainly without knowledge of the full impact
- Uses various techniques to break in i.e phishing schemes or zero day vulnerabilities
- Sometimes it is the payload of a worm
- Other times it is from botnet nodes
- The ability to upgrade software and download new programs and functionality
- When the distributed system in question is a botnet, it is not so hot.

Malware Components

- Malware is a business industry for people who build it
- People make a living selling malicious software to attack people's machines.
- Particular components that are commonly reusable that people want in their malware.
- It doesn't want to visible to the owner of the machine
- Various types of things that are common requirements for malware.
- A very natural approach is to say okay, I will NOT build it 18 times. I will build one good component that is reusable
- Insert into malware without having to rebuild everything from scratch
- Get installed on someone else's computer
- Somehow, you have the ability to run some piece of code and if you are on someone else's computer, you retain control.
- You want it to be really hard to get rid of you

Dropper

- Simple piece of code with limited functionality
- Get a small piece of code running out someone else's machine.
- This piece of code in a sophisticated malware is going to be a dropper.
- The dropper can be a very general type of thing and this would be a very generic type of thing.
- You like your dropper to be very small and you want it installed and you don't want it downloaded.
- Fiddle a bit with configuration of the dropper and you can handle both of these things

Rootkits

- Usually, an attacker wants to be able to keep attacking your machine
- He understands that once you recognize you have a piece of code on your machine, you want to get rid of it.
- Two things to do if you are an attacker:
- Hide!
- If he notices you are there, you don't want him to figure out where the bad code is living.
- Even if he does figure it out, it will be hard for you to get rid of it.
- Find someway to get hooks deep into your system.

Use of Rootlets

- Worms or viruses will use a rootlet to gain access to your machine
- Sony made themselves unpopular by installing rootlet to prevent piracy among its music tracks.
- They installed the Sony software and this generated a rootlet when they used the CD!
- It was removed eventually but this generated terrible publicity for Sony
- Not enough to say there is only a few executables
- We need to prevent the rootlet form getting removed
- Replaces system components with compromised versions
- We have a compromised driver that is lying to you.
- Usually rootlets are rather elaborate and we need to find just one to solve your problem.

Ongoing Rootkit Behavior

- Base functionality is the ability to get on your machine and run an arbitrary piece of code.
- If they have to be running some process, when you try to do a listing of all processes on your machine, there can be a compromised component.
- Can conceal the fact you have network connections, files, registry entries, etc.
- If I have a compromised machine, the guy is NOT careful with his security
- Maybe someone else compromised this machine too!
- If they have, as I install my rootlet, it is getting rid of theirs (like Cuckoo hashing)

Q. Can we use the rootlet removal strategy by placing your own rootlet?
A. Not the best idea, the best practice is to reinstall the software from an uncompromised source.

Lecture 13: Secure Programming
Introduction

- We should all care about writing secure software
- Large companies that create a whole lot of software i.e. Microsoft or Google
- Define your security goals
- You have to understand a lot about what the program is trying to do and see if it is working the way we want it to work.
- We have to understand our goals.
- Use techniques that are likely to achieve those goals.
- We want it to go through the entire lifetime of the software process.
- Think about security as one of the goals you try to achieve.
- It is supposed to NOT be susceptible to various kinds of attacks

Designing for Security

- If they even think about security, they worry about it later
- Priority is getting the program working and then fixing security later.
- Really bad ideas because people have been trying this for 50 years and it never works!
- They end up with Adobe Flash or the version of Windows before Bill Gates ordered a redesign
- Security retrofits offer too many opportunities to be attacked

For Example,

- Lecture 13, Page 5
- Windows 95 and its descendants
- Not designed with security in mind
- The result was that security professionals assumed if you turned on a machine, the Windows 95 machine was compromised
- People had already gotten it
- A whole lot of effort in improving security of Windows 95 had some weaknesses in it.

Defining Security Goals

- Think about which security properties are relevant to your software
- Think about it in a more detailed sense:
- What bad things will happen if you are NOT careful with your program?
- It should NOT be the case that anyone can see the stuff I am holding for them.
- Will there be privacy issues?
- Am I collecting people's credit card info?
- If so, I need to be careful how they are stored and transmitted.

- Is availability important?
- Is there anything I can do to stop attackers from running DDoS attacks
- How does what you are doing interact with what is below?

Security and Other Goals

- Security is never the only goal
- Easy program to build if you only care about security
- But this is NEVER the case.
- Let's say you are building a web presence for a company, they need a good experience from a company you want to get and you cannot say that you will always make a decision for greater security at the cost of everything else.
- Engineering is about tradeoffs
- A lot of desirable characteristics and you cannot achieve all of them
- Let's say you want to build a circuit that lasts 50 years
- It then becomes hard to fit into a device that makes use of it.
- This is just another tradeoff to think about.
- Similar to how we deal with other catastrophes
- Building a bridge: we need to handle an earthquake to withstand a magnitude of 7.5 or whatever.
- Then, we design with that in mind and if we get an 8.0 earthquake, we accept the risk because it is NOT worth the extra cost.

Managing Software Security Risk

- Handle certain types of data and certain processes will be controlled by this data.
- Given too much risk, this could completely blow up in my face and we need to minimize risk.
- How?
- Various compromises we can make i.e. web commerce systems and ordering particular items.
- Attackers can take advantage of a lack of authentication, so we need a user to be more careful about logging in and this will be less convenient for him.
- We have limited to risk that they will lose money and we don't want to risk tradeoffs
- We need to make compromises and consider the tradeoffs you are making from an engineering perspective.
- Is the amount of risk sufficient to make the cost of whatever I am going to do worthwhile.
- I can either have unencrypted sessions or I can encrypt end to end.
- If I have all my sessions encrypted end to end, I will have spent a lot of computing resources encrypting and decrypting my data.
- We may need 50 extra machines to handle this vast amount of traffic.
- Ensuring that people aren't willing to eavesdrop and don't want to pay a cost.

Risk Management and Software Development

- Security is something that should be considered within this model.
- Start doing a design and through several cycles
- In spiral model, add security risk analysis phase when you consider alternatives

Incorporating Security into Spiral Model of SW Development

- Consider security as part of the risks and you keep going through the spiral tighter and tighter.
- Figure out what the mitigations are

But How Do I Determine Risk?

- Do the best you can and use the knowledge and tools you have got to try to do the best you can to identify risks.
- When you are in this phase, ask yourself what are they and what is valuable/important.
- Discuss a few principles that will help you later on.
- If you do this, you will be better than 95% of all developers.
- Big step forward if this is all you do and you cannot get it 100% right.
- You will overlook things and there will be stuff that attackers find and they will probably compromise your stuff.
- Happens a lot less frequently if you have taken a few steps up front.

Design and Security Experts

- Someone on a software development team should understand security
- This is a person with a specific responsibility of caring about the software's security
- Explain to other people the risks we are talking about.
- Here are some mitigations we could take and understand if we should use cryptography or not.
- Smart cards or passwords?
- Having an expert can help determine the implications and make it a lot better.
- The expert should be involved from the very beginning

Principles for Secure Software

- These are good ideas designed by security experts
- Following these guidelines can still lead to crappy security, but chances are, you will improve the security of your software

0. Secure the Weakest Link
   - People have heard about specific weaknesses and only worry about things they know
   - May or may not be a useful strategy
   - People who are making these attacks are not religious about types of attacks

- They do NOT give a shit if they use buffer overflow attacks or SQL injections, they just want to get into your damned system
- If you have gone ahead and secured something that is really hard for you to attack, you won't do yourself any good and concentrate on your most vulnerable elements.

For Example,

- Lecture 13, Page 15
- Those attacking your web site are NOT likely to break transmission cryptography
- People have NOT had success switching from DES to AES
- They are just going to take over your site with a buffer overflow or phishing or password guessing.
- Prioritize things according to the urgency.

0. Practice Defense in Depth

- Don't have one defense where if it is overcome, renders your system totally vulnerable to the attacker
- Compromising in one piece of software won't result in the total compromise of everything.
- Try to protect your applications from bad things in other pieces of software.

For Example,

- Lecture 13, Page 17
- Validating your input
- Ensure it is the kind of input you are expecting to see
- Make sure its s NOT too big to put into this buffer
- Write a little routine to check everything regardless of the source.
- If it has come through this one routine, it must be good, and from now on, I can trust it.
- This is probably NOT the best way to write your code and then check if there are particularly risky things as I thought it was.
- Write an important piece of code and it should check everything already.
- There could be a bug in your code and bad input can get through.

0. Fail Securely

- Programs should crash gracefully
- This can happen if a program failure is NOT totally fatal
- Sometimes, what happens is if you fail into modes that aren't secure
- In order to do this operation, you need to escalate your privileges just for a few minutes and then drop the administrator privileges and move on to ordinary activities
- Attacks on programs by feeding it crap, and seeing what happens if that program fails because they haven't thought of failure conditions

- Make sure failures do NOT leave you in a state that is potentially comprisable.

For Example,

- Lecture 13, Page 19
- All kinds of security protocols and NOT all of them are known to everybody.
- Let's negotiate the protocol for Java
- Eventually, they agree to something.
- Server will give the client the security protocol and they can agree to use it.
- Allow them to falsely authenticate himself to you.
- Client thought he was running in a nice security fashion and fix it.

0. Use Principle of Least Privilege

- Give minimum access necessary and you cannot do a whole lot of other stuff
- Another thing to keep in mind is that you give it to them for the minimum amount of time required
- You can say that for most systems, for the following period of time, you can give him more privileges and take it away later.
- We often are talking about you writing a piece of standard code (could be buggy!)
- If the bug is in a place where he doesn't have privilege, he cannot misuse the privilege

For Example,

- There is only certain information in your database
- You are selling stuff to your customer and they have not yet delivered things to you.
- At any rate, we want to give access control to determine if you have a web server, you will get to the database for some purposes.
- If you have some program running on a web server, you should NOT give your program access to that database at all.
- Only the web server can get to the data to access webpages that need it.
- Do NOT give full access to everything on the database.

0. Compartmentalize

- If you are careful, not only about dividing things but also how pieces interact, you will NOT compromise the whole system.
- If each component has some separate protections, then each component can protect itself.
- You should be careful about how things should interact.
- Any one of the 12 other components should only be allowed to do things they are supposed to do.

For Example,

- Lecture 13, Page 23
- Traditional Unix had terrible compartmentalization
- Everything had root privileges back then
- You probably cannot remove people's print jobs from the print queue nowadays!
- Web servers are intended to provide service for large amounts of clients simultaneously.
- Typically, we want to maintain compartmentalization between those people.
- Your interactions should be disjoint and you don't want to change the data that goes out to you.
- Design the program so we cannot carefully compartmentalize everything else.
- Systems in the real world tend to NOT do things in such a good way but at least they try to!
- Asbestos allows you to create compartments.
- Don't provide pointers to your stuff for his code.

0. Value Simplicity

- You love simple systems and hate complex systems
- Usually a good idea to choose the simple alternative over the complex alternative.
- Complex systems have behaviors that are harder to predict
- You don't know what is "proper" and underlying systems will be complex.
- If you have a complex system, it is hard to understand what exactly is going on.

For Example,

- Lecture 13, Page 25
- Reuse software when possible
- If the software is secure, just use that! Don't build a new one
- Cryptography should have one implementation and one place where cryptography gets done with one set of routines.
- It is a bad idea to write cryptographic code.
- Very smart people with an extreme amount of experience have written crappy cryptographic code.
- Since it is hard and people have already done it, use what they have before and don't try to reinvent the wheel.
- You are more likely to have bad things happening in complex pieces of code.
- A simple algorithm is better than a complex algorithm
- Presumably, there might be advantages to the simpler algorithm, which is why we should choose it.
- Why don't you instead pick the simple algorithm whose performance is good enough?

- Another thing from a security perspective is that they just love new ideas and tools (Arko and angular.js)
- Not really a great idea because things that have been around longer tend to be safer than things that are brand new.
- Is the advantage I am getting really significant, or is it just new and shiny.

Especially Important When Human Users Involved

- Users will NOT read documentation
- People who aren't computer science professionals won't read it either
- Users are lazy
- They ignore pop-ups and warnings
- No need for complex user decision for user security
- If user makes the wrong decision, we will tend to go to a secure state.
- Make it easy to figure out the results of the bad situation

0. Promote Privacy

- Don't compromise the user's privacy
- Don't publish passwords or SSN on websites
- But, a lot of things can happen to a computer system that may or may not be your fault.
- If your system had to have that data, it should be a risk you have to take.
- If you didn't have a need to store the credit card number, why store it?
- Sooner or later, all the data you are keeping will be stolen by someone.
- You don't want to ask for data you don't need, and just keep it for the period when you need it.
- If you have to keep it, keep it encrypted.
- Strong legal issues related to this in the healthcare industry or in Europe in general.

For Example,

- Lecture 13, Page 28
- Google Maps
- Integrate it with Wi-Fi hotspots around the world.
- How do we do that?
- We drive cars by and gather the data by putting up an antenna and putting out the info.
- In addition to listening to Wi-Fi hotspots and the types of protocols, they also heard a lot of data packets, so they essentially stole a lot of people's packets
- Europeans were very upset with this and fined Google

Q. If it is public, is it okay?
A. This is okay in the U.S. but the problem is they kept copies of the packets, which is what pissed off the Europeans

0. Remember That Hiding Secrets is Hard

- It is very hard to keep secrets and in particular, some people believe they can give the secret to someone else, and a secret will be a secret.

- Amit Sahai does this via sophisticated cryptographic techniques
- If you are trying to put a secret into something you give, you must assume they will learn the secret no matter how hard they try to hide it.
- Eventually, the secret will be found.
- Dividing things up into different parts of the program and they will figure out.
- Just because you aren't smart enough to do it, doesn't mean some hacker who is smarter than you cannot do it.
- There may be someone out there who can do it.
- People out there who look at object code and can figure out what it is doing.

For Example,

- Passwords often "hidden" in executables
- Give a piece of code to a customer and they need a secure communication back and have one password and put it somewhere into the software.
- Zhuhai had a surveillance system and they put in a hard coded password and sold them.
- Of course somebody found the password eventually.
- Android apps containing private keys are in use (and compromised)
- They have handed the private key to everybody in the world who downloaded the app.
- Digital rights managements always do this
- New ways of sending movies or music to users and this prevents it from getting pirated.
- Often has a hidden key either in software (mostly) or hardware (sometimes), that eventually shows up to the public.

0. Be Reluctant to Trust

- Don't automatically trust things!
- Gain some reason to believe you can trust it.
- If you want to trust a remote user, check that before you trust him.
- If there is another component in your system, you need cryptography to check first.
- Client did something reliant on trust first and this caused problems.
- When you say you are trusting someone, indeed, the party is perfectly honest and wouldn't intentionally try to screw you over.
- If they haven't been careful, they will have gotten some security problems on their system.

For Example,

- If you buy a piece of software or an open source library, why do you trust it?
- There have been security problems in these pieces of software that have been intentionally put there by attackers

- Will there be bad things for making use of a potentially compromised library?
- It would be nice if you had to use someone else's components to use defense in depth here.
- If it isn't as good, I will allow it to have this effect on my program but prevent it from having further effects.

0. Use Your Community Resources
- Use respected security stuff over untested stuff
- Don't build your own stuff!
- A lot of people who are trying to keep an eye on things happening on the big bad Internet
- All kinds of services that will tell you about various sorts of bugs and security flaws
- Good idea to keep up to date about things like this.
- Keep up to date on trends and flaws in security you use.

Choosing Technologies
- When you get a choice, it is nice to choose wisely to have the security features you want.

Choices and Practicalities
- You get choices in principle, but NOT choices in practice.
- Chances are, you will be working for a company (unless you start your own company)
- They have made some choices already and you have to live with their choices
- If you are building commercial software, chances are good you have to build for Windows for sure.
- That is where most of the commercial market (80-90%) is, so that is a given

Operating System Choices
- Rarely an option since Windows is so ubiquitous
- 15 years ago, I think I will go with Mac OS X or Linux was a big win over Windows, but nowadays, not so much.
- Microsoft got their act together, and Linux and Mac OS X weren't as good as we thought they were.
- Solaris is the 4th OS and it is fairly unusual.
- BSD is still out there.
- For practical purposes, you will be using one of the main 4.
- Each has stream of security bugs coming out more or less on a monthly basis.
- There are a lot of problems in all of them and they are trying to fix their bugs.

- It is fairly unusual for people to attack the security of an operating system and it is harder to exploit this than in the application.
- People often try to go for the apps rather than the OS
- Exception is if there is an environment where high degrees of trust are necessary
- If you care about this choice i.e. the U.S. government, you want to use SE Linux or Trusted Solaris
- Not updated nearly as often!
- May not interoperate properly with things that do up-to-date operations
- Typically not a compromise you will make.

Language Choices

- Hard to switch legacy code.

C and C++

- Worst security choice from a security perspective
- Buffer overflows are possible in C and C++
- People often choose C or C++ for efficiency
- Other languages have extra stuff that won't run as fast.
- How important is efficiency for your application?
- For a lot of code, you don't need that last percentage of performance.
- You will see the same visible performance from the user's perspective and NOTHING would happen better.

Java

- Less susceptible to buffer overflows
- Better at error handling
- Special built-in security features that people don't take advantage of.
- Weaknesses
- Exception handling issues
- Some problems in inheritance
- More complex language than C or C++ so there is more problems in the compilers
- Security problems in C but NOT the C compiler.
- Oracle had 9 remotely exploitable security flaws in the 2nd quarter of 2016 Oracle patches
- Happens on a quarterly basis that fixes some # of security flaws.
- Multiple serious security problems in recent years with Java

Scripting Languages

- Depends on language
- Many are type safe, limiting buffer overflow attacks

Scripting Language Security Issues

- Interpreters might have security flaws
- More likely than compilers

- Scripts are usually delivered to the users
- Relatively easy for the attacker to examine
- A whole lot more people can read scripts than can read binary
- Scripting languages make heavy use of system calls
- dangerous for things like eval()
- Evaluate this and have this data and run it from off-site.
- If the call libraries, there can be overflows there
- Python buffer overflow in 2014
- Many script programmers don't think about security at all.
- People often go into scripting languages without knowledge about more sophisticated things in the scripting language.

Open Source vs. Closed Source

- Shall I use open source software or closed source?
- Closed source - you haven't shown them the source code
- At most, you have shown them the compiled version.
- If you find the bugs, you need to increase security.
- The other bugs will be found when they occur.

Is the "Many Eyes" Argument Correct?

- Linux has security bugs similar to Windows
- The mere fact that the code is out there doesn't mean they will read it.
- You aren't going to find any bugs if NOBODY reads it.

The Flip Side Assignment

- Hackers will try to exploit bugs and it can be found via a black-box approach
- Observe system behavior
- Buffer overflows are typically found this way.

The Upshot?

- No solid evidence that it makes a difference from open source or closed source.
- Building an AES encryption from the Linux library and cryptographers actually do look at this stuff.
- TLS had a common security flaw and this was open sourced.
- By having big companies pay people to look at it, they were able to help determine critical weaknesses on open source software.
- Common libraries with obvious security limitations
- Not a scalable approach

One More Consideration

- Edward Snowden published information from the NSA
- NSA loves to get people's keys
- When it is closed source, nobody else can check the keys

- The government has given an order to tell nobody that they have put a trap door there.
- If it is a open source repository, groups can put something in there and maybe somebody will see it.

Major Problem Areas for Secure Programming

- If you are in very serious security terrain like missile development, you have to avoid falling into traps
- You cannot care about everything, so pick and choose what will affect your entire system.

Example Problem Areas

- Buffer overflows and other input verification issues
- …

Buffer Overflows

- The main thing we think of for insecure program
- People have complained about these for years
- Often related to social engineering
- For a more technical approach to fooling the user.

Preventing Buffer Overflows

- Use a language with bounds checking
- Modern languages other than C/C++ or assembler
- Not always a possibility
- Deal with the issue more a fundamental level.
- Check if they did it right by checking the bounds yourself.
- Avoid certain things that people do that are prone to buffer overflows.

Problematic Constructs for Buffer Overflows

- A lot of ways that allow you to copy data into a buffer

Why Are These Calls Risky?

- Dangerous if the attacker can affect what is getting copied into the buffer.
- If the attacker can control things, it is not safe.
- If you are NOT 100% sure that the attacker can control it, you should just check and don't use things like strcpy() but rather strncpy() where you can specify a maximum size

What Do You Do Instead?

- You could have a string with specifiers and this describes what we want to do and there is a precision of the characters we want to copy.

W 8 R Lec 5-19-16
Is That All I Have To Do?

- C constructs are inherently dangerous and shouldn't be used.

- Merely avoiding these constructs wasn't enough.
- You could easily do so without knowing you have done so.
- Be aware of Integer overflow problems

An Example:

- Lecture 13, Page 56
- packet_get_int() comes from the user
- If the number of packets > 0, then you have to set up the buffer
- You have one pointer to a string because we have to go through one by one and copy the pointer to the string into the area.
- Ask for that area and only copied enough area to fill it no more

Why Is This a Problem?

- nresp is provided by the user
- We shoulda locate a buffer of nresp entries
- Problem is integer overflows!

How Does That Work?

- The argument to malloc() is an unsigned int
- The maximum value is 2^32 - 1
- 4,294,967,295

What Is the Result?

- We have allocated 128 entries in response[]
- The loop iterates hundreds of millions of times
- Copies data into the "right place"
- Will cause a massive buffer overflow, and the attacker can copy exactly the right number of bytes.

Other Programming Tools for Buffer Overflow Protection

- At first glance, that program looks pretty good.
- Instead of simply saying to NOT use a program with buffer overflows, we can use software scanning tools to look for problems
- Some are quite sophisticated, others aren't sophisticated at all.
- If you have bounds checking, you need to use that option manually.

Canary Values

- Analogy to the "canary in the mine"
- A problem we have in mining coal and other things is that there are invisible gases that replace oxygen and you die if there are too much around.
- Bring a canary in a cage and hang it in the mine.
- If these gases escaped, the canary would die before people would, and if the canary keels over, we have a problem and we need to get the hell out.
- Place a random value at the end of the data structure
- If the value is not there later, buffer overflow might have occurred.

Data Execution Prevention (DEP)

- Developed by Microsoft
- Most, but not all buffer overflows cause the execution of brand new code.
- Why do I have an area of memory where I could both write data and also execute out of.
- An ordinary program would never do such a thing, and we would never approve of self-modifying code.
- Since we never want to do that properly, why should we allow that at all?
- You never have a writable page that is also executable.
- Wherever it could write, there is no possibility of writing out of there.
- Helpful, but NOT sufficient because you can have other types of sophisticated attacks.
- Either it doesn't require new code at all, or they say we make use of code already there (return-oriented programming)

Randomizing Address Space (ASLR)

- Address Space Layout Randomization
- Knowledge of where things are and they need to overwrite the useful things.
- Need to know the return address with the appropriate place to jump to with his attack.
- Where in the heap, libraries, stacks, etc. gets randomly changed, so this makes it hard for the attacker to write working overflow code.
- Merely makes it hard; the attacker could still possibly find it.
- A lot of programs are written so it doesn't meet the standard specification of the OS
- Programmers usually use deeper knowledge than they are supposed to use.
- This is to optimize efficiency and the problem is if you turn on ASLR in Windows is that they break!
- Program can turn of ASLR if it needs to with a flag in its creation.
- This is fine because the program works but it is NOT protected from buffer overflows.

Lecture 14:

Error Handling

- The attacker knows you are going to have a lot of possibilities of having errors in your code.
- If you did write some code, there is code there and it is probably NOT executed very often.
- Probably NOT heavily tested.
- If you did test it, you probably just tested if it went to the error routine.
- The reason this is problematic is because there might be an undetected error that you have not found
- Attackers often try to compromise systems by forcing them into error conditions

- If something happens that they like, they can figure out what the problems are

A Typical Error Handling Problem

- Probably whatever you are trying to do cannot be done because there is an error
- You might not clean everything set because you haven't tried to do everything.
- Old values lying around that have NOT been changed
- One of the most common ones is that you have gotten extra privileges and you hit an error condition

Some Examples

- Apache HTTP server had a condition where it had a weakness against remote denial of service attack
- ntpd had an error handling flaw
- Clock-based attacks on some of the security protocols against why we want different machines to have similar clock values
- This attack lets the attacker set the target's clock

Checking Return Codes

- You make a lot of method calls and procedure calls
- Always check your return values from these calls
- Microsoft's OS manager said the new grads do NOT check return calls
- Someone didn't check their return codes
- This is why Reiher wants us to check return codes
- Microsoft makes you do this stuff.
- Dangerous to go ahead if you call the routine and gives you back the return code that does partially what you wanted.
- If you don't check that, who knows what will happen when your program goes ahead.

Privilege Escalation

- Normally, a program runs with a typical set of privileges
- Available to the user when they had the program
- If you need extra privileges, it will be constrained by the UID model of your system

An Example Problem

- If you have a program running setuid and allows a shell to be forked
- It gives the caller root privileges
- It can do anything!
- What privileges will that have?
- Root privileges!
- This allows a caller to be given privileges to things he never should have had.

- Using flaws in the program to gain greater privileges.

A Real World Example

- Lenovo System Update Service
- Users aren't supposed to futz around with the user settings
- Running on these machines and we don't want them to install updates.
- Provide them with a program with legitimate updates.
- We can be sure to have an updated system without problems
- Difficulty here is we need to create a temporary account
- Run it with the privileges of that account
- Account name and password were predictable and all someone had to do is to run update service and this will have a particular password.
- The moment after I have this account's info, I Have logged in.
- Attacker can seize this opportunity

What To Do About This?

- Avoid running programs setuid
- Minimize the use of these calls
- Sometimes you have to do it, but it is dangerous
- If you have to give them privileges for sure, follow least privilege
- Don't make things to convenient and new users should NOT have to set up everything so the right files are associated with the right people.
- If there is a problem, they could literally do whatever they want.
- If you have to increase the privileges temporarily, change back and get rid o those privileges as soon as you can in the program
- Make sure there is NO path where the privileges get executed and continue expanding
- You could create a virtual machine to run these dangerous programs
- Set up all the resource settings to simulate the environment, but also limit what is possible on the virtual machine
- Maximizes the damage control on the machine

Virtualization Approaches

- Theory of a virtual machine is that it is really a separate machine
- Every single thing is virtualized and you should do the appropriate thing int he machine.
- Virtualization technologies have flaws very often
- Allow you to escape from the virtual machine
- Hard to create virtual machine that does that
- If attacker get s piece of code from the virtual machine, he will try to use it to crack the real machine.

Race Conditions

- Moderately common cause of security bugs
- Happens when w can run multiple program simultaneously or a program with multiple processes or threads

- From the point of view of people writing these programs, the order in which things happen is very unpredictable
- Hard to estimate what will run at which time unless you have done specific things in the program to make sure it happens
- When writing these programs, you typically aren't thinking about these.
- You assume that statement A will happen, then the next stage and the next state….
- You know exactly what order it is happening in your program, but you don't know what is happening outside.

What Is a Race Condition?

- One of the results is unthought of and the user doesn't know how to prepare for it.

Security Implications of Race Conditions

- You thought the next thing that would happen is that your lines of code would run next
- Multiprogramming issues would allow things to happen in between instead.

The TOCTOU Issue

- Time of Check to Time of Use
- Some delay between checking and when you actually do it
- What looked good when you performed the check may NOT look the same as when you actually do it.

A Short Detour

- Unix has two associated user IDs
- Effective ID: Supposed to be used at any given moment to check the privileges of a process
- Real ID: ID of user who actually ran it
- System calls also lets you change your effective ID.

Effective UID and Access Permissions

- Your privileges are based on your effective ID and it happens until you change your effective ID
- A root user has universal root access privileges

An Example

- Lecture 14, Page 19
- Create a temporary file
- Runs setuid root
- Lets check to see if we have access for read purposes
- It is a root program and if it wants to read that file, it can read that file
- I know I am a setuid root program, but I am working on behalf of an actual user, so I have to adhere to the needs of the user

- See if the real ID and tell me if it is okay or not.
- I was run by some other user
- 0 means “yes”
- If it is NOT 0, then you are trying to access something you aren’t allowed to
- We have said, “Don’t just open the file. Check to see if it is okay”

What’s (Supposed to Be) Going on Here?

- User in another program can use links to control what the actual file is
- access() can check whatever is linked to this file

What’s Really Going On Here?

- OS might schedule something in between

How the Attack Works

- Attacker puts innocuous file in /tmp/userfile
- Program that the user can read
- For an innocuous file, you can quickly change this file and it could link to the sensitive file

The Dynamics of the Attack

- Lecture 14, Page 23
- Let’s run this program and see if it is okay to run
- He didn’t get what he wanted so he didn’t get to read the evil file.

Q. How is this temp user file point to another file?
A. Using hard links. A file is a file, or it can be a pointer to another file.

- Done through the command line i.e. ln or in a separate program.
- This make not work if /etc and /tmp are NOT in the same file system
- Run the program and see if you can access the tmp user file
- Are you allowed to access that?
- He can manage quickly to change the name and now tmp user file is a link to /etc/secretfile
- We can check if it is okay for you to run the file and you need the effective user ID to be root.
- Root can open this file for sure, and the attacker succeeds.
- He gets to see the contents of the secret file even though he should never have had permissions to do that.
- Timing issues that is difficult to get your mind around at first.

How Likely Was That?

- Timing had to be exactly right!
- He could try it as many times as he wants
- If he is lucky, he can influence the system to make it more likely to get right.
- He can run a low-priority program to interfere with ah higher-priority program

- Timing attacks here are very good
- Impossible to check when you have permission or not.

Some Types of Race Conditions

- File races
- Changed what file you get to see
- Permissions races
- Check when permissions change but you have been told you get it.

Q. Difference between file races and permissions races?
A. File race is changing the file, permissions changes the type of permissions on the same file.

- Depends on how things are running
- The fact you had the file open was NOT a race condition.
- Change contents and also permissions (both a file race and an permission race).
- Check when the user still has the permissions
- Some other program can check if you have sensitive information and this program has permissions to do anything.
- Opens the file for you
- Ownership races
- Who owns the files
- Directory races
- Shuffling the directories

A Real Example

- Linux System Tap utility
- Go inside and get information for you to be provided
- Tell you more information since you are root and this was intended to be a general utility
- When you do this, it checks your privileges to see if this is a module that is already for you to load
- At the time it checks the privileges, there will be a delay.
- Check to see if he wants to load module “foo”
- He needs to make sure he slips in a different module with high privileges
- A real TOCTOU issue that is exploitable

Preventing Race Conditions

- Minimize time between security checks and when action is taken
- Be careful with alterable files
- Contents, pointers, permissions, etc.
- If you have to do something with system checks, use locks to make sure you aren’t blocked at important pieces of the code
- Race conditions are hard to think about.

Randomness and Determinism

- How are we going to get random numbers?

- As we discussed in key generation, we need a seed

Pseudorandom Number Generators

- PRNG
- Obey certain properties from truly random functions
- Actually deterministic
- If you plug a function back into the function, you get back the original
- Share many common proprieties with truly random streams of numbers

Attacks on PRNGS

- Based on a string of numbers, I can guess what your PRNG is.
- Knowledge of how your program behaves is very key.
- State attacks
- How did you start up your pseudorandom number generators
- If they know what your PRNG is, they know your input AND your output
- State attacks will let you guess what is being put into your program

An Example

- Lecture 14, Page 31
- Poker deals with random cards
- If you know what card everyone else has, poker is an easy game.
- How do we determine what cards that people get
- Flaw in PRNG
- Badly seeded number that people could find, then they would know everyone else's cards.
- This is a state attack because we knew the state of the program as well as the seed value

Another Example

- Lecture 14, Page 32
- Netscape's SSL implementation was the precursor to Firefox.
- Based on knowing time of day, process ID, and parent process ID
- Generate the same pseudorandom # and this could be gotten by on any other process.
- Just do what you do for any user, and this was able to break in keys in about 30 seconds

A Recent Case

- The chip-and-pin system was used to secure a credit card.
- Needed PRNG to encrypt cards
- Cambridge found weaknesses in their PRNG which allowed attackers to withdraw cash without your card
- There have been attacks in Europe that have done this!

How to Do Better?

- Use hardware randomness whenever available

- Use high quality PRNGs
- Make sure it is good quality!
- People should not be able to do cryptographic attacks out of it.
- Even with a great PRNG, if they know the initial input, they could generate all the other numbers.
- You have to find something that is much more random, and you want some truly random number into your input.
- Never use seed values from outside the program because this causes all kinds of trouble later on.
- Allows state attacks on your pseudorandom number generator

Proper Use of Cryptography

- Encrypt, decrypt data
- Often used to achieve security effects
- It is easy to use it improperly
- Do NOT write your own crypto functions if it has been written before
- Remember the Microsoft guy who said to check your return codes?
- He also doesn't want you to write your own crypto, so don't fucking do it.
- Your ego is too big; don't design your own encryption algorithm unless that's your specialty
- Professional cryptographers will break it, so you need tremendous experience and background in order to create a strong crypto algorithm.
- People have ended up creating cryptographic algorithms that are NOT secure.

Proper Use of Crypto

- Even with good crypto algorithms, problems are possible
- Bugs are possible in:
- Keys you choose
- Key management
- Application of cryptography

An Example

- Lecture 14, Page 37
- Use public key cryptography to distribute one of these symmetric keys
- Send them the encrypted version and now he knows the key and no one else does.
- Use triple-DES for the symmetric key cryptography
- They were distributing 168 bits of keys.
- Everything seemed to work just fine
- One of the symptoms of perfectly terrible cryptography use
- Everyone says it is all great when it actually isn't
- Someone noticed the protocol and part of the RSA key exchange was always the same each time they ran it.
- After all, they were supposed to be distributing a new key overtime.

What Was Happening?

- RSA encryption code was given bad parameters
- It failed and returned an error
- They didn't check for the error!
- Why bother checking if it did fail?
- The result is they never applied RSA encryption at all.
- They were distributing a brand new key everytime but this was in plaintext.

Another Example

- From an Android app
- Derived and encryption key from the user's password
- They ran it through their algorithm and ran it each time.

Why Did That Cause Problems?

- Android's default character set is UTF-8
- If it gets a bit pattern that isn' a proper character, it translates to the hex string "EFBFB"
- A lot of these bit patterns were not matched by UTF-8
- All of these got changed to that, and this resulted in a key that you have chosen and distributed that was far from random
- Send this across the wire in string form.
- That looks like a perfectly good random password which was cast to another password that was problematic.
- Be very careful about key generation!

Trust Management

- You often have nowadays multiple different components that are running with each other
- You need to trust other components to do things for you
- Every time you trust some other components, you have to ask yourself:
- Do I need to trust that machine or user?
- Don't trust other programs
- Don't trust other components of your program even if you wrote it.
- Take the paranoid approach -> nothing is as it seems!

Trust

- You need to determine how much trust you should give
- Accept a data value from 1 and 10, but you should at least check if it is in the range
- Compartmentalization helps to limit the scope of your trust
- They will compromise one portion of your program and this will not result in the compromise of the entire program

Two Important Lessons

- A lot of security problems arise because of unverified assumptions

- Trusting someone doesn't just meant to just trust their honesty
- We also have to trust their caution and see if they have suitable precautions themselves
- Take whatever you have accepted as being good and do NOT drop down to their level.
- You need to determine that we don't have a SQL injection in our code.

Input Verification

- Do NOT assume that users have followed the rules on giving input he is supposed to give you.
- Users can provide you with almost anything!
- If you don't check, you can perform your operations on almost anything!
- Just because they gave you good input in the past doesn't mean they will give you good input in the future.

Treat Input as Hostile

- You may be getting exactly what he sent you, but why believe that?
- How do you know he isn't trying to use a perfectly secure channel to screw you?
- Assume the other programmers working on your team give you good stuff?
- Assume it is dangerous!
- Even if it comes from your own area of trust, it is better to assume it is junk.
- Attackers can use code paths from a place you don't trust.
- Successful programs get updated and things change.
- Inputs may NOT be as careful

For Example

- Shopping cart exploits on the World Wide Web
- Just like in a supermarket, you can put an item in the cart and take the entire set of items and check out
- Our shopping cart is a data structure stored somewhere
- The most common way is as a cookie
- You want to buy that and a cookie is delivered to your system saying you want to buy the following item
- All the cookies go back to the server and now I know what to charge and what to send them.
- Some of these cookies weren't encrypted, and this meant users can alter them.
- No integrity protections here.
- Shopping cart cookie was to be customized and there are multiple prizes for the same items
- He wants to buy the following piece of electronic equipment and that is the prize he wanted to pay.

What Was the Problem?

- The system trusted the shopping cart cookie when it was returned
- There was no reason to trust it
- You can encrypt the cookie
- Makes the input more trusted
- Sure, I got a cookie back on an item, let's look at it
- If it is a suspicious price, reject it
- Do sanity checks on what you get back
- Even if it is encrypted
- Make sure it is a sensible thing to do.

Variable Synchronization

- Often the case that you will have a couple of different variables that are related to each other
- Supposed to have some constant relationship between two values of those variables
- You probably have a variable that describes the length of that buffer
- This is the next thing I want to look at in that buffer
- Pointer should never go beyond the length of the buffer
- If you change the length, you may have to change the pointer as well
- What happens if you change one of the variables without changing the other?
- You can run into a security problem

An Example

- One of the things they had was a place where they had a buffer
- cdata was a pointer to the buffer
- length is the integer length of the buffer
- Provided by the user with potential leading space and trailing white space
- Let's get rid of this crap!
- Deal with the meaningful characters in the string

The Problematic Code

- The problem is that when we get rid of leading whitespace, we did NOT decrement len
- This meant that we could overwrite the end of the buffer with NULLs
- It would possibly be the case where you overwrote this in such a way to cause various problems for the program

Variable Initialization

- Some languages let you declare variables without specifying an initial value i.e. C or C++
- Let's you use them without initializing them
- Big problem!

A Little Example

- Reiner did NOT initialize aa, bb, cc
- How the hell did they get the values 11, 12, 13 respectively?
- This is because the OS reuses the stack frame in the same way, and the same integer used to hold a holds aa, b holds bb, c holds cc

Why Is This Dangerous?

- Allows values from one function to “leak” into another function
- Maybe he can use the influence from the first function to influence the behavior of the 2nd function

Variable Cleanup

- Common to reuse a buffer or other memory area
- Get a bunch of messages into your system
- If I am dealing with it one at a time, let’s keep reusing it!
- Reuse memory area under your control
- You might NOT have properly cleaned it up from another memory area.
- Happened to Microsoft’s TCP/IP stack
- Old packet data treated as a function pointer
- If it was carefully crafted, they could force your TCP/IP stack to jump to that routine
- Cause more or less arbitrary code to execute

Use-After-Free Bugs

- A lot of memory structures allocated in the heap, and let’s use it for various purposes
- You could get explicitly or implicitly freed
- In some cases, the pointers can be used to access freed memory
- C or C++

An Example Use-After-Free Bug

- fragments are pieces of messages that come in and can be used for each particular fragment
- Let us free that fragment and under certain circumstances, let us return the length of the fragment
- We just performed that operation and perform it so that we can use that freed pointer frag to see what we have there.

What Was the Effect?

- This usually crashed the program.
- Combined with other vulnerabilities, it could be worse i.e. arbitrary code execution

Recent Examples of Use-After-Free Bugs

- Attackers tend to look for “fashionable” bugs
- A few years ago, Use-After Free became fashionable
- Products that were vulnerable

- IE
- Adobe Flash
- Mozilla
- Google Chrome

Some Other Problem Areas

- There are many other areas with problems.
- Arithmetic issues
- Integer overflow in Adobe Flash
- Signedness error in XnView
- Off-by-one errors
- Clam AV had a denial of service flaw

Yet More Problem Areas

- Null pointer dereferencing
- FreeBSD had a null pointer dereference in 2016
- Punctuation errors
- Comma in the wrong place
- Typos and cut-and-paste errors
- iOS vulnerability based on inadvertent duplication of a goto statement
- This fucked up Safari!
- People could compromise iOS systems!

Why Should You Care?

- Many of these flaws were never exploited
- A genuine fact that has been demonstrated by research that the vast majority of security flaws are never exploited!
- Many are not!
- It could be exploited but it probably won't be.
- How on Earth would anyone figure out I have a weird problem?
- They have to study the source code
- There are 500,000 lines of code, who the hell would look through that to find an error in some obscure place of the code.
- It is NOT something that would be easy to exploit?
- You could have made the bug occur, but it is hard to change configuration settings for the bug to be exercised
- Not really possible for an attacker to make it occur.
- Not a security problem that will lead to a bad situation

So…?

- Some of these problems do lead to bad situations
- Sooner or later, people get screwed.
- It is hard to find!
- People think it is hard to exploit and people sometimes think someone else will take care of this.
- "It will get taken care of by someone else"

- Code auditors
- Testers
- Firewalls
- We don't have to worry since the firewall will take care of it.

That's What They Always Say

- There are some diligent attackers who put a lot of effort into breaking into the system
- Attackers can be very clever
- Crash my program since there is a reason he wants to.

But How to Balance Things?

- We are always working in an environment to build and test programs.
- We cannot expect perfection since we simply don't have time.
- How are we going to design and build code that does NOT fit into these security traps
- We would have to forget about this secure coding stuff.
- If you develop good coding practices, you tend to make less mistakes
- Not because you have taken extra effort, but because the way you code leads to less mistakes!
- Never have a use-after-free error because you are careful about that.
- Always initialize your variables after creation.
- These will tend to avoid these problems and won't cause extra time.

Some Good Coding Practices

- Validate input
- When you get input from outside your program, you need to check that it is supposed to be what it is.
- If you are only supposed to accept certain values, it should meet certain characteristics
- Be careful with failure conditions and check your return codes
- Check to see what happens and always check the values
- Make sure it is sensible and that it cleans up
- It needs to be undone and this will not only help against vulnerabilities but will prevent data from getting mucked up.
- Avoid dangerous constructs and learn about particular C system calls that are dangerous.
- Try to keep it as simple as possible
- Complexity is the enemy of security.

W 9 T Lec 5-24-16
Lecture 15: Web Security
Web Security

- I do something on the Internet, I use the World Wide Web as one of the components
- Underlying web components there

- Much of this stuff is financial in nature
- Companies make a lot of money through the Internet
- A lot of private information flow
- Taxes, medical info, keeping track of all kinds of things in our personal lives
- A great deal of valuable personal information is flowing around across the web
- A very obvious target, so it is attacked all the time

The Web Security Problem

- We have a lot of users
- A large % of the traffic is web traffic
- A lot of servers
- Vast #'s of websites (millions)
- A whole lot of people on one side interacting with entities on the other side
- They have no relationship with the entity on the other website.
- They are the citizens of a particular country working on someone else's website.
- We don't know a lot of what is going on in the other side.
- We probably didn't worry about if we worked with that site before.
- We care more if it has what we are looking for.
- Essentially, we moved what amounted to text in HTTP protocol requests and responses to render fairly simplee web pages.
- This is really quite complex.
- No central authority in charge
- There are people who have islands of authority
- DNS names, people can give those out
- There is no single authority to give out DNS names
- How does the web work?
- Industry groups set a standard and want people to follow a standard
- They don't have to follow it, and if the standard proves to be inconvenient, they don't follow this standard.
- You cannot go and say if there is a bad website, but you can go to the law enforcement agency in that locale.
- This could be a website in Uzbekistan (what would you do?)
- Little to no security experience and people may not care
- You just have to keep coming out with new stuff all the time and produce new contents to show your users
- Otherwise, you fall behind and someone else gets the business
- They don't know what the hell they are doing from a security perspective
- Many critical elements were not designed with security in mind
- They said security was someone else's problem so there were insecure bases
- Microcosm of the overall security problem

Aspects of the Web Problem

- Lecture 15, Page 4
- We have multiple clients, so the server has to interact with each of them
- Who are we protecting?
- The server presumably has something valuable for the people who run it.
- We want to protect the server from the clients and some of these people aren't nice.
- Most of the clients are perfectly good people.
- Some of the servers have not so nice sites.
- Let's protect clients from the server, too.
- Most clients are good, but some are NOT
- They need to attack the other client, but they need to use the intermediate to perform the attack on another client.

Who Are We Protecting?

- You have a client who is sitting here and some servers may be okay, and others may not be.
- We need to attack the other server, and clients need to use the servers to attack each other.
- Protect everyone from the network because we are moving everything across the Internet
- He could be sitting here listening to the traffic
- He can cause bad things to happen even if he is not HTTP web traffic at all.

What Are We Protecting?

- Server has private data
- Business
- Making money on the web based on special information that only they know about.
- Clients are doing certain transactions and we want those transactions to maintain their integrity and we don't want it to be altered.
- In some cases, people attacking these systems don't care about the web stuff and they don't care about information you have stored on the Amazon cloud.
- They only care about your machine and once they get your machine, they get everything on there as well.
- Perhaps, they want a million records on the server and use any vulnerabilities that they can find to compromise the machines.
- We aren't talking about compromising a particular transaction, but rather taking over machines.
- Sometimes, we care about server availability
- Chinese government has disagreements over who should control certain parties
- Other governments have similar feelings about certain things.
- Locally or globally, they need to servers to make content unavailable
- They do want particular clients not being able to get to particular servers.

Some Real Threats

- We get standard stuff of people attacking software
- Buffer overflows and compromises of the machines.
- Client attacks servers
- Social engineering
- Attacker doesn't want to try to find technical vulnerabilities; he tells you to follow certain links
- SQL injection
- SQL is a database language
- Particular attack that works on sites that perform database transactions
- Client attacking the server
- Malicious downloaded code
- Server attacking the client

More Threats

- Cross-site scripting
- Became popular when clients started uploading content to the web.
- Editing Wikipedia
- Putting some/thing on their FB page
- Clients attack each other
- Threats based on non-transactional nature
- HTTP keeps no state
- Each HTTP message that comes in is self-contained
- There need to be no earlier or later HTTP message
- Very frequently, you don't just hit one link, but rather, you navigate and perform complex interactions
- When you think about that, this suggests we need state
- It is sort of encouraged for people NOT to work too hard on keeping state
- This can lead to problems
- This is a client attacking the server
- Denial of service attacks
- Somebody trying to make sure a server is NOT available for other clients

Yet More Threats

- Browser security
- Single process running on the OS that is the web browser process
- Processes share memory and typically share privileges
- All of the code typically has the same set of privileges
- Interacting with a dozen website in one process
- If one of the servers is NOT fully trustworthy, can he use the fact that all of this memory is going to do something bad?
- Server attacks a client's interaction with another server
- Data transport issues
- Go through common network elements like links, gateways, firewalls, etc.
- Anybody who can get access can get things from the World Wide Web

- Certificates and trust
- We do a lot of things based on trusting another party.
- We will now show you your bank balance because you are who you are trusted to be.
- It is okay for you to have my credit card # because I believe you will send me the item I bought
- If you are NOT careful, you can get into fairly serious problems!
- The trust issue is the server attacks the client.
- We need to verify who the client is and make sure we don't give the client privileges it shouldn't have.

Compromise Threats

- We are running a bunch of software here
- Web server is a complex piece of software
- Browser is a complex piece of software
- There are a lot of lines of code and lots of stuff going on
- There could be security flaws everywhere in the web.
- These are pretty much the same as for any other network application
- For most places, the web server is the most important application on the machine and if you can take over the web server process, you can do a lot of bad things on the system
- There could be more than just buffer overflows
- That sends data across the network and this is the same problem
- Well designed code that isn't exploitable

Solution Approaches

- Regardless of how good a job we have done, there will be some security flaws in it
- Patching
- Preferable to have a good code base so we don't have to patch as much
- Be careful regarding access permissions
- If we only run it to support the web server, we have to be careful about that.
- Perform a lot of testing and evaluation to see things that slip through

Compromising the Browser

- People compromise the web browser
- Sort of like an OS
- Can essentially do all kinds of things for you
- You can do a lot of things without exiting your web browser
- People run almost nothing except things that come through the web browser
- You can do almost anything i.e. a voiceover IP telephone call.
- You can have your bank account updated and all of this gets done through the web browser
- Share a bunch of things and it isn't actually generating an OS process

- Windows, Linux, or Mac OS X generates the process abstraction
- This is where you get OS code that gives you protection
- It doesn't have the security features of most OS
- OTOH, the most dangerous things you can do are built-in right away
- Arbitrary extensibility
- You can add any code at all times
- Another dangerous thing is running multiple threads of control simultaneously
- This is why we have processes in OS
- Browser is running one process so it is NOT getting a whole lot of nice compartmentalization

But My Browser Must Be OK….

- Lock icon ensures nothing bad can happen
- The lock icon does NOT mean your browser is safe

The Lock Icon

- The window/tab you are looking at shows a digital certificate that worked
- This is somewhere and we have a public key that is used to sign that certificate
- I trust that public key (signing authority)
- We got a certificate that worked
- Check the certificate and it is signed by someone the browser trusts.
- Someone you trusted said this is a good public key
- What are the implications here?
- Somebody gave him a certificate and you hope he wasn't compromised
- Just because the website is who you claimed to be isn't who you think it is.

What Are the Implications?

- Somebody gave him a public key.
- Go to Amazon.com because he hit a typo and everything is fine.
- Would you actually notice the difference?
- People frequently don't.
- The person who signed the certificate is still trustworthy and they weren't careless about their signing certificate
- People have NOT behaved poorly through malice but rather through carelessness.

Another Browser Security Issue

- Tabs
- Go to your bank balance and there you are in that tab
- I would also like to see some cute cat videos and download it.
- While I am interacting with my bank, I will alternate between the cat videos and the bank balance.
- What if there is an evil script in one of those cat videos?

- That evil script can potentially steal things.

Same Origin Policy

- Well, why don't we build something into the browser?
- We cannot rely on the underlying OS security, and we need to put something into the browser to help deal with this problem.
- Meant to foil the evil cat video from stealing info
- Built into all modern browsers
- Don't run a 20 year old version of FF and it will be in there.
- Flash is NOT a browser, but it is commonly used in the World Wide Web.
- If I am a browser, pages from one website can only access things from that one website.
- This means that the browser has to be properly tagged and checked overtime.
- They may NOT always succeed; it depend s on how well he code is written
- Pages from a different origin cannot access each other's stuff

Web Cookies

- When the client interacts with the website, it includes the cookie and the browser stores cookies to give them back to websites
- Overcome the fact that HTTP is a stateless protocol
- People try to build those protocols to be stateless as well.
- You are buying something from Amazon.com but you haven't yet actually purchased it.
- When you are sending another request, he was going to buy the following piece of music.
- This info is in the cookie, and they are very widely used!
- Thousands of cookies on your machine.
- This is how you maintain a session
- HTTP is a stateless protocol
- How does i know you logged in?
- Kept in a cookie, so we know who this guy is!
- All of this stuff is kept in the cookie, and they are in the context of a single cookie representing the browser.
- The evil cat site can feed them back to the evil cat website.
- The same origin policy is thus applied to the cookies.
- I want the cookies that I can see, so I won't be given cookies that belong to some other site.

Same Origin Policy and Cookies

- Script from one domain cannot get the cookies form another domain
- Typically defined by DNS domain name and the application protocol i.e. HTTPS, TCP, UDP, etc.
- Occasionally asks for the port i.e. port 80 for web.

Q. Doesn't Chrome have separate processes for each tab?

A. They have decided this is safer, but you have higher overhead for the process and it is much more difficult for the processes to interact.

SQL Injection Attacks

- These can occur in other kinds of network interactions but in principle, it can happen even where someone types a command into a computer
- Usually, it is through the web.
- Over the course of time, as we have gotten more and more complex applications, it becomes necessary to have a whole lot more data to do the correct thing.
- A good way to handle this is to store them in a database.
- The most popular query language is SQL-based
- Database representation can be anything arbitrarily different
- Somebody can write an interface to the database, and as long as it is SQL, it can be easily accessed.
- A language in which you can express “Please get” or “Please put” information
- Web pages are NOT static things nowadays!
- Rather, they are dynamically created as needed
- Based on all the context, I am going to build a brand new webpage and will get it out of the database.
- Build a page that is customized for a particular client
- The client provides this information, so you are going to send a query to the database to build a custom webpage.
- Information you get from the client could be unpleasant!

SQL Injection Mechanism

- if you have a backing database, it can say you need to generate a SQL query to build a webpage.
- How do I know I need to build a webpage for Peter Reiher?
- He filled out a form and pull out that POST request and slap that into the query and get his information.
- We need to build him the right webpage from this information.
- In a SQL injection attack, I didn’t send him a name, I sent in a little piece of SQL code!
- The server could insert that little chunk of SQL and this will change the query and it will become a different query
- Instead, you do something different and this is a simple example

An Example

- Lecture 15, Page 23
- The idea is that the user fills in his ID and password
- Database will run the query and he claims his password is this or something else
- What if we fill in ‘or 1=1; — ‘

What Happens Then?

- Plug in what you gave me in place of the uid
- Time for me to evaluate your SQL statement
- Your username is equal to nothing, but it is okay if 1=1, and that is true!
- Semicolon is the end of a statement
- — is a comment, so ignore the rest!
- Get any record in the database!
- The entire database will be returned to you!
- This is NOT so good and it will automatically log the guy in.

Basis of SQL Injection attacks

- Real problem!
- **Unvalidated input**
- I thought I was going to get a UID, I get a piece of SQL instead.
- Incredibly dangerous!
- Use what he got back and this resulted in an arbitrary SQL query being sent to its database

Some Example Attacks

- 130 million credit card #'s stolen in 2009 with SQL injection attack
- Ruby on Rails had built-in SQL injection vulnerability in 2012
- They fixed that one fortunately.
- You got one for me free for Ruby on Rails back then.

Solution Approaches

- Carefully examine all other input
- Use the database that doesn't use SQL, but there could be an injection attack on the other language

Examined Input for SQL

- Well-defined language
- Unless you are asking someone, you shouldn't be getting SQL
- Filter it out and reject tit
- Provide web interfaces for many different types of languages and there is a whole lot of different input and there are different ways of encoding this
- Unless you are very careful, you may allow something that is NOT properly coded to slip past you.
- Too many backspaces and when you run through the whole thing, it becomes SQL.
- This makes the problem of validating the input very difficult.
- Some SQL control characters are widely used in real data
- Scottish or Irish names like O'Reily

Avoid SQL in Web Interfaces

- You don't even need SQL, but the user input may need to be created without using user input.

- We can have predefined queries and choose between options 1 and 10
- Choose 1-10 predefined queries that have no pieces of code and exactly the SQL I want to run.
- You can do this under certain circumstances.
- May NOT be possible to do everything we really need to do without SQL queries.
- How am I going to build this query?
- That is probably NOT going to work out too well.

Q. For user ID, will it help if they had #'s on the web interface and the user can click on the #'s?
A. You can do your social security # that way and there is a limit based on what you can click on.

- Very tedious way to get your input!
- You ultimately want to make users happy because otherwise, they can be unsuccessful.
- If that is possible, it may be better than asking for input.
- In the case here, if you press a digit, you send in an HTTP request saying your number is what you input
- Value is expected to be between 0-9
- It is much easier to validate ahead of time if you know what the legal values can be.

Use Parametrized Variables

- You can set up your code so that the $ UID is a bound parameter
- When I take the value that goes into that parameter, I will treat it simply as data.
- You will instead say "This is the SQL, just to see $uid is, and take whatever value is there and run it as DATA, not a piece of SQL!"
- There is a particular way of doing this and even if the guy gives it to you, let's see if we have a user in my database where we have 1=1; —.
- Nice solution if you build your SQL interface like this, and you probably won't run into SQL injection problem

Q. If this is so easy, how did it happen?
A. Ordinarily, people wouldn't be willing to take the extra effort.

- Not so useful when it is provided by an untrusted user.

Malicious Downloaded Code

- Download the application and ask if you want to download and run the application
- At least you had the opportunity to not make a mistake
- The modern web browsing experience is very heavily dependent on being able to run customizable code for a webpage you are trying to view
- Otheriwse, you cannot see the active code that is running
- Mostly done through scripts
- Less development time than compiled languages

- Associated with the webpage, there will be one or more scripts in the context of your web browser
- Perhaps under some circumstances, you can fork another process and it won't matter if it is a separate process or not.
- On a typical OS, unless you are doing something special, it will have the same privileges as the generating process.
- This means that if you are the user running on this computer, there will be privileges and we will be downloading the script.

Types of Downloaded Code

- Java
- Full programming language
- Scripting Languages
- JavaScript

Drive-By Downloads

- In order to actually execute this code, you have to push a button or follow a link in order for the code to be executed.
- Sometimes, if you go to a page and hit a carriage return, you will get a script executed with no further action on your part.
- It just runs by itself!
- By merely visiting a webpage, you can compromise your browser because you could be running an evil script.
- If the user needs to take more positive actions, then it really isn't a drive-by download.

Solution Approaches

- All modern browsers are capable of this, so you can be more selective than that.
- There is another side to where the trouble comes from
- We would be running scripts that don't do evil things because it isn't as secure.
- Part of the problem is that it runs with the privileges of our web browser
- User's own process able to do anything with its current privileges
- Let's keep track of all the bad script sand put them on a blacklist
- Check to see if a script is on a blacklist and we will use it on a parametrized value.

Disabling Scripts

- If the script gets downloaded because you visited a website you have never seen before.
- I haven't gotten a particular website on my whitelist so I need to run a script that it gives me.
- You then as the user get a choice to run the script or not!
- Typically, a user will select Yes because it wanted something from that website.

- When this window pops up, it is saying if you wanted this content you wanted.
- If you want it, you have to run this script.
- If you turn off all scripting in your web browser, you do NOT get the experience of the modern World Wide Web at all!
- It is a good deal less and you miss out!
- Whitelisting things
- Usually NOT a feasible solution

Use Secure Scripting languages

- Cannot help against evil scripts
- Against badly written scripts, it could be helpful
- This could be a good scripting languages, so download and run this script so it could be safe.
- Harder to use and they do less.
- People don't want to use them and even if it is secure, there is a lot of bad things you can do with it.
- Not too much space to work with and you cannot force anyone to do this!
- Write good scripts and verify these scripts don't have bugs or problems in them.

Q. What are some secure scripting languages?
A. Perl - figure out an input wherever it goes and do NOT let it go into bad places. JavaScript is bad!

- Do some research into what is felt to be good scripting vs bad scripting languages

Isolation Mechanisms

- Overtime a script gets downloaded and they get run in the context of the user running the browser.
- Sometimes, these are a little less than the full user privileges.
- Let's run the script in a brand new virtual machine and let's limit the harm that can happen.
- To the extent that they are done in the browser, and it is more or less trust.
- You won't know whether they do it right or not.
- If they think they did it right or not, you need to figure it out for yourself.
- The problem we have is that if unless we have a good VM, it is possible to escape it.
- Did they always tag the data properly or not?
- This isn't like the virtual memory process abstraction
- We just hope the guys wrote the code right, or else we're fucked.
- Dump the script into the virtual machine and we could have better guarantees.
- People who do security research have found that there are ways out and this doesn't mean the script will be able to do that but it is hard to make guarantees here

- You create the brand new virtual machine and it can use those resources and not everything else.
- What resources are safe to give to this script?
- The script effectively doesn't run and if we give it too many resources, we may cause it trouble and it is hard to tell what you should give to these virtual machines.

Signatures and Blacklists

- Identify known bad scripts
- Check it against a blacklist
- Same problem against virus protection
- Attacker can write a new script anytime he wants to

Cross-Site Scripting

- There isn't really a solution to it.
- Not a good solution for the big outside world!
- Abbreviated as XSS
- Refers to cross-site scripting
- It used to be the case that the World Wide Web was about downloading content to users
- People began to realize that every using WWW can create their own content
- Take pictures, write text, put comments on things
- Let's share this content with everyone else!
- This is where FB, Twitter, and all these social networking come from!
- GitHub has value and we can share things with this web mechanism!
- Modern WWW depends on this capability
- User is on the website and it will be made available through that web mechanism
- When whoever is allowed to look at that stuff, it can get downloaded using HTTP and gets run.
- XSS attack occurs when we upload an evil script
- When you download my evil script, you will download it and I will take over your machine!

The Effect of XSS

- Have arbitrary malicious code executing on a user's machine
- XSS attacks are very powerful
- Run in the context of a web browser
- When the user doesn't have total privileges, it is still pretty bad.
- Anything really important can probably be compromised in this way.
- Attacker can use this to gain a foothold and escalate his privileges.

Non-Persistent XSS

- Here, if you are an attacker, you create a webpage.
- Put a link on that webpage

- Because of the complexity of HTTP, you can put scripts into links!
- If you set it up properly, the link can point to a legitimate web page.
- What happens if a user goes to that site and clicks on that link
- FB will do whatever it does, and it would echo back the script to the browser, and your browser will run the script
- If FB is careful, they will prevent this from happening and check where the script came from.
- This is typically NOT what is going to happen, and it will get echoed back to the user.
- FB didn't store it; it just temporarily visited FB as part of a link being executed.

Persistent XSS

- Uploading data permanently and they will store all the content and make it available to other users.
- Content provided is the evil script and some user requests the links and it gets downloaded.

Some Examples

- Word Press bug allowed XSS (2016)
- We got an XSS bug for free
- Other XSS vulnerabilities on some huge names including Apple App Store.
- Companies like Symantec had XSS bugs so this is very common!
- D-Link router flaw exploring through XSS

Why Is XSS Common?

- Scripting languages are widespread in the web.
- You don't get the web experience if you don't, and user upload of data is extremely popular
- The question is how to get the user to run your script.
- This doesn't mean anything beneficial will happen for the attacker, and you have to hit on that link to download that script.

Typical Effects of XSS Attack

- Steals information provided to Amazon and this is running in your web browser
- If it is properly implemented, it is running single-origin policy and it should only run things related to FB.
- Looks at browsing history from FB and it shouldn't affect any of the other sites.

Solution Approaches

- Don't allow uploading of anything
- Pretty drastic!

- There are a lot of people whose business is to share information through the web.
- Why on Earth do you want to upload script on your site and give it to other users?
- Not what you usually want to do, so let's NOT do that.
- My browser is where these scripts run so let's have protection in the browser.

Disallowing Data Uploading

- Sometimes, you do have to allow updates.
- Is it actually necessary to show that to users and can we share it to users other than the one who did the uploading.
- If he downloads the script to himself, it is probably NOT going to be a problem.
- Perhaps, if we say we allow you to upload stuff, I won't give it to anyone else and we want to avoid cross site scripting problems.

Q. What kind of business has people upload stuff for themselves?
A. Amazon stores data and you download data when you want it.

- Another thing you can imagine is if you are a bank, and we want our customer to upload a profile.
- We probably don't want to show his profile to anyone anyway!
- He is the only one who could ever download it, big deal!
- If this is how you upload things, FB cannot solve the problem in this way.

Don't Allow Script Uploading

- Why would they?
- I am a site that uploads scripts and see what it does for the machine.
- Otherwise, you don't want it uploaded except for that circumstance.
- Validate the input and see if the guy can use the following stuff.
- Did he give me a script?
- I don't want it! Throw it away.
- Something you probably should do, but you need to look for SQL injection attacks.
- Unless you are very careful, there are tools that look for this sort of thing.
- Try to prevent things from being uploaded.

Protect the User's Web Browser

- Let's say we have a solution that is going to work there
- Same solution as any evil script kind of thing and damaged functionality
- Content security policy and this allows us to upload content from this.
- Steal private content and we don't really want this to happen.
- I am the website, and I should specify where data should or shouldn't go when it came from me.
- Website can come from browser and do the following things with scripts I gave you.

Cross-Site Request Forgery

- Abbreviated as CSRF
- Works the other way around
- Authenticated and trusted user attacks as a web server
- Usually an attacker posing as that user

CSRF in Action

- If he can get a user to visit his website and click on that link, the request will follow to that website.
- If you have gotten an authenticated website, you will try to empty out the contents of that bank account.
- Authentication cookie that proves I am that user and let's prove this.
- This is a dangerous thing that happens
- Bank authenticates the legitimate user and if I have a Bank of America cookie, it goes to BoA along with that request

Issues for CSRF Attacks

- Not easy to make work
- Generally speaking, a website can say if I am getting a request, where did that request come from?
- If it is a link, not only will the link say "Hey this is for BoA", and you need to be careful of where a request came from.
- If you set up a website, then you can be careful and say that it came from some weird place and I need to be careful about this particular request.
- The attack site must allow the webpage to use something useful.
- From an attacker's perspective, it needs to be valuable
- This makes it difficult to use this attack to run arbitrary code on a user's machine.
- Website isn't interested in doing this with a true XSS attack.
- Not trivial!
- If you are a bank site, you are in danger!
- Must NOT require secrets from the user!
- If it requires information from the request, he won't provide the secret unless you are really smart about it.
- Only works if you get the victim to come to your website and click on the link.
- Attacker cannot easily steal information this way.

CSRF In the Wild

- Verizon Mobile App API
- Farming tool kit that attacked wireless router using this approach

Exploiting Statelessness

- HTTP is designed to be stateless
- Various tricks like cookies to remember information

A Simple Example

- Websites are graphs of links (static stuff)
- Maybe he will click on a link on other page and it will take him to another page
- People will navigate a path through this graph going step-by-step
- We are getting a request coming from here and this is something I can show you.
- This means that we have NOT gone through the steps that we thought we were going to for a particular webpage.

Why Is That a Problem?

- What if there are unlinked pages on the server?
- If he can deduce the name, he can generate the HTTP request for that page!

A Concrete Example

- The Apply Yourself system
- Harvard Business School used this
- Harvard would take all the applications and this tells who you should admit and who should not be admitted.
- The theory was that this is done in a transactional way.
- All decisions will be released and no one know anything before that.

What Went Wrong?

- Professors looked at applications and they had to look at that somewhere.
- Look at a particular applicant and tuck that webpage away and we will NOT link that webpage in anywhere.
- When we want to release the webpage, we will put it somewhere else.
- Surely, no one could access them?
- WRONG! They said I bet that my webpage is located here and they generated an HTTP request with that URL.

The Core Problem

- Anyone who does that will NOT be accepted if you got caught doing it.
- Not the technical solution of this problem
- Need to prevent a protocol memory of what came before
- Thus, it will be acceptable to go on to the next step or not.
- Follow the links from page to page and if we hit that link, we should NOT have been able to generate that request.
- This is what should have happened.

Solution Approaches

- I will set this up so it does NOT have read access to the web server.
- Set up a bunch of links and add a bunch of read privileges.

- You could have another system where we just keep track of state and whether or not this request came from a link that was set up.
- Frontend program that observes requests and responses
- Keep track if there was a response to use a link or not.

Q. What does it mean for a backend system to maintain and compare state?
A. You could build a stateful web server

- Cookie-based
- Make sure the cookies are encrypted and we need to be given that web link and hit on that link
- Presumably, he cannot create that out of thin air and we need to allow him to honor it.

Q. If the user clicks on a link that leads to this page, we only want the user to see if they can pass in a variable to the link?
A. It could only if that variable is already in place. Sooner or later, he will get it right!

(Non-) Use of Data Encryption

- Much of web traffic is NOT encrypted or NOT signed
- We can have eavesdropping, MITM attacks, and alteration of data in transit

Why Web Sites Don't Use Encryption

- Primarily for cost reasoning
- There are algorithms we have to perform to encrypt/decrypt data
- Websites that encrypt or decrypt and we need extra processors or machines!
- We need to be willing to pay money for this.
- You, the user, probably don't give a damn since your computer isn't doing shit.
- On the server side, they are doing a lot, so they have the decision on what happens.
- People doing the website perform the cost benefit analysis.

Problems With Not Using Encryption

- Attackers can give information to inject new requests and perform various kinds of attacks
- Attackers can profile traffic and perform this on wireless networks.
- They need at least WEP turned on and see everything that is happening.

Using Encryption on the Web

- HTTPS is the encrypted version of HTTP
- HTTP sitting on top of TLS
- If you are using HTTPS, you are probably in pretty good shape.
- You also have authentication and encryption of the traffic in both directions
- Authentication based on certificates.

- You may notice that if you go to a website and look at the URL, it either says HTTP or HTTPS
- If you are using HTTP, it is NOT getting encryption
- If you are using HTTPS, you are getting encryption
- Sometimes, you are using old browsers or limited systems that don't have the ability to perform cryptography.
- What happens then is you have a negotiation between the browser and the website.
- The browser doesn't know anything about cryptography and typically, the website will say it is your problem.
- It may fall back on HTTP on a much less compelling basis
- Someone may NOT have configured his browser but he should have.
- This has been determined over the course of time NOT to be a good thing.
- HSTS (HTTP Strict Transport Security)
- We insist you need to use HTTPS
- We are using cryptography whether you like it or not.
- Your website has to use HTTPS or else it cannot communicate with the website.

Increased Use of Web Encryption

- Problems of web encryption
- Does NOT help with XSS or SQL injection
- Google announced it would encrypt all search requests in 2014
- FB + Twitter adopted HSTS in 2014
- We are going to institute HSTS a couple of years ago.
- 5% of websites have adopted it as HSTS and 95% have NOT
- There are those who argue that everything on the web should be encrypted.
- All requests and responses should be encrypted.
- Currently NOT the case.

Sometimes Encryption Ins't Enough

- ISP can do man-in-the-middle attacks.
- I will pretend to be the website and I have an ISP that I will accept
- Negotiate with the actual website and create a 2nd website.
- Send it off to the website and it looks like it was encrypted.
- If you sniff your traffic, there is a little moment when it isn't quite encrypted.
- ISP's should do this because it allows web caching and compression.
- Otherwise, it means you trust your ISP
- NSA has compromised key management
- Elliptic key cryptography means they know your key.
- NSA has spied on supposedly private links
- NSA can do this stuff and presumably the Chinese, Russian, and Israeli governments can do this.

- Other people's ISP's cannot do this, but Verizon probably can
- Not in their interest to though since they want to move data through the network as fast as why possibly can.
- Very few people who can do this to you, but if there are bad people it is hard to find out if they are doing this to you.

Conclusions

- Web security problems are NOT generically different, but the ubiquity of the web make it hard to do this.
- If HTTP had been designed with cryptography, we would be better off.
- Legacy kills us here!

W 9 R Lec 5-26-16
Lecture 16: Privacy
Privacy

- Techniques to deal with these.

What Is Privacy?

- We want to be able to keep certain information secret.
- Usually information that we don't want certain groups to get.
- We generally talk about our own personal privacy and we want to control the dissemination of that information
- Point of view of "power"
- There are privacy issues you must look at concerning your customers and other people you are interacting with.
- You may have legal, moral, and business reasons that are in your custody.
- One of the things to be aware of is that it is NOT just a handful of pieces of information like your SSN or credit card #
- All the information about what you are doing is to some extent being tracked by somebody
- They are keeping that information somewhere and doing something with that data
- People can figure out where you were within 10 feet of every moment of every day for the past year.
- They can figure out what you do in your spare time and this is quite a scary prospect!

Privacy and Computers

- There is just information we know is private i.e. healthcare records
- Most information about us is kept on computers
- Not too many people work with paper records anymore.
- Usually on some kind of storage device i.e. hard drive
- Most computer systems are networked on the Internet
- Any computer can send a message to any other computer in the world
- Access any piece of private information
- Often, the information is scattered among your devices.

- However, there are also huge databases in companies that have nicely packaged information in one place.
- They can grab a million passwords and credit card #'s
- These are time bombs waiting to go off.
- Disturbing for older folks like Reiher
- You don't have the vaguest idea who knows what about you
- You cannot have a clue!
- You cannot know who knows what about you!
- Once the information is in someone else's hands, there are no mechanisms to help you figure out what they are doing with them.
- If you give your credit card #, they don't have a legal obligation to not tell that credit card # to everyone else.
- No legal obligation and you cannot know what to do with it
- Who is everyone who knows my SSN?
- Who is everyone who has my passport #?
- You cannot know all of this information?

Privacy and Our Network Operations

- Lots of stuff goes on over the Internet
- Banking, credit cards
- Health care records and information
- Communication with doctors
- Romance and sex
- Family issues
- If you find out your mother has some medical condition, it probably came from email!
- Gmail - Google knows!
- Personal identity information
- People didn't know this kind of stuff, but we changed in the past 30 years for how this element works.

Threat to Computer Privacy

- What actually is causing the problem?
- Cleartext transmission of data
- Anyone can listen to the network and get the data
- Systems on the network that are poorly secured.
- Remote user who does NOT have access can get the data anyway.
- He can do some kind of social engineering attack!
- Wherever we go, the sites can keep track of how we are interacting with them.
- They can keep records of what is happening and this is valuable information
- When you go back to Amazon, they give you a bunch of things they think you are interested in buying.
- They can use that information for other things as well

- Targeted advertisements i.e. Gmail and suddenly, there is an advertisement of stuff coming from that mail.
- Do an analysis on that mail and figure out what the user wants.
- Multiple sites can combine information!
- Governments
- Have a serious interest and watch over its citizens
- More power, sot hey can do watching that other parties would have a hard time doing
- Location privacy
- Providing little pieces of information all the time.
- Connect up to cellular telephony towers.
- Gives a general idea of where you are in.
- As you move, you automatically switch from tower to tower.
- They keep records for that!
- Most smartphones have GPS and you can get updates to other sites on the Internet without you being aware.
- They can get really fine-grained information about your location
- Even if you are keeping information private, people work for that company
- While a company may be honest, not all their employees will be.

Some Specific Privacy Problems

- Poorly secured databases that are remotely accessible
- Stored on a hackable computer
- Data mining
- Modern Internet - getting intelligent suggestions is all provided by data mining
- Used heavily by many companies
- Governments eavesdrop on our network communications
- U.S. government was watching the metadata of all the citizens for the past several years.
- Insiders improperly access information
- Cellular-based tracking

Do Users Care About Privacy?

- Do you mind that Google has given you an advertisement that you want to buy a sailboat?
- Studies have looked into this and people care about privacy, but not in the most obvious way you think.
- Teenagers aren't worried about people hacking into your smartphone and them stealing your information.
- Rather, they are worried about privacy from their parents.
- They don't give a shit about the hackers; just make sure dad and mom don't know what you are up to.
- If you are interested in the privacy solution, see if they are protected from their own government.

- You have to consider what privacy goals your user has when building solutions to protect privacy.
- You have to pay a cost of some kind to protect your user's privacy
- You want it paid back to do something valuable to the users.

Data Privacy Issues

- Can I even know who has got it?
- If I have a bunch of this data, what do I do to protect it?
- Throw it away
- Securely delete it
- If you are not foolish, the only reason you re keeping this private data is because you need it!
- You need it to make a profit and/or provide data that your users want.
- Even if we think we haven't given any particular information out, it may be that through very clever data mining, we learn things we didn't expect to learn.

Privacy of Personal Data

- Who owns the data about you?
- Is this something that is the kind of data we can express ownership over and under what circumstances?
- In a lot of cases, this wasn't thought of philosophically!
- No computers or electronic tracking
- Police could assign officers to follow specific people as long as they didn't violate certain privacy agreements
- If he was walking around in public, he could assume that he had the right to be watched.
- Technologies can look through the wall of your house without going inside or putting anything inside.
- Could the police take this device and track you everywhere you went?
- Is there any ownership of this information of how you moved around in your own home?
- Some data is very personal!
- SSN
- DoB
- DNA record
- Some people have asserted that DNA is patentable by someone other than the owner
- If you have a complete sequencing of your genome, it is very personal to you, but is it yours?
- Every time you go to Google, they keep track and know you entered a specific search term
- They know you asked about "Britney Spears" and they know what time you asked about that.
- Is it something we should be concerned about?
- Not just Google, every site can keep track of this stuff.

- They will get information about where you were before and where you were after.

Protecting Data Sets

- If you re one of the companies, you have gotten a bunch of private data.
- Here's information about where I live and all kinds of other stuff
- Didn't fool anybody or steal it.
- What remedies will I have if my privacy mechanisms fail?
- The LinkedIn database pattern was stolen.
- Peter Reiher's password got divulged on the Internet

Options for Protecting Data

- Careful system design
- Limit access to the database
- Make sure you have access controls and have divided it up in such a way that not everyone on the Internet can access it
- Log your information so it is easier to determine if a bad thing happened
- Auditing
- Go in and take a look at the logs and how to use it.
- Store data only in encrypted form
- If the encrypted database is stolen, the attacker has to deal with if you encrypted things.
- If he hasn't stolen the key, he only has the encrypted data and you may be safe.
- You probably cannot deal with the data in encrypted form.

Data Mining and Privacy

- Allow users to store models from databases
- I want to know something about a whole bunch of users.
- Often, people have a large data set and will allow 3rd parties to perform operations on these databases.
- I know this is private data but I won't allow him to perform a query to figure out everything about Peter Reiher.
- Where do people live in this neighborhood?
- At least in principle, you tell people about aggregates of information, which seems a lot safer.
- Unless we are very careful, attackers can use these data mining capabilities to hone in on a # of different values

An Example of the Problem

- Netflix released a large database of user rankings of films
- They liked to do data mining on all these ratings and then they wanted to make recommendations later.
- The more good recommendations they make, the happier people would be and pay more for Netflix

- Good for the user because he learns about movies he may not have heard about.
- They were humble about the algorithms but there are probably better algorithms that they didn't think of.
- Let's ask everyone to build a better algorithm!
- How on Earth would you run this competition?
- Unless it is a real dataset, how do w know if the algorithm will work on real data.
- User Reiher watched this movie and gave it a 5, what if we don't want people to know that.
- We could use a random identifier instead and then throw this dataset out and go for it.
- Some researchers weren't interested in the prize, but rather seeing if they anonymized this data.
- IMDB provides information about movies
- Who is in the movie
- Who directed the movie
- One of the things they do is the user can rate the movie with stars.
- For IMDB, you need an account so they aren't anonymized
- They took the Netflix information and matched it to IMDB names and we could see who gave it a 5 and who didn't.
- We will get some information about who was who and this worked out pretty well.
- Certain movies have certain types of themes i.e. gay people
- People who are gay are more likely to enjoy gay movies than not-gay people.
- Sometimes, gay people don't want anyone else to know they are gay.
- Researchers were able to correlate those ratings and follow the characteristics of those movies
- They figured out the gay critics were gay because of their positive ratings to gay movies.

Q. Isn't it anonymized? How did they correlate the two?
A. User Smith has a real identity correlated with a public database. Basically, they used a huge search to see what the 53 movies he has rated were.

- This is why few companies will release anonymized datasets nowadays.

Q. Aren't there methods to hide this information?
A. Not done in the Netflix release of the database.

- There are tools meant to combat this problem, but researchers can attack algorithms

Q. What was Netflix's intention?
A. They wanted a better algorithm for recommending movies.

- They got a better algorithm, but they also got a lawsuit.

Insider Threats and Privacy

- Insiders are people who have legitimate access to your data.
- The reason they have legitimate access is because the company does something with your data.
- You are running a hospital with healthcare records.
- Your patients are going to come in and we need to determine if they are allergic to particular drugs.
- People have to look at the data
- Doctors, nurses, etc.
- What if people using this power abuse that access?

Local Example

- UCLA has a huge medical center regarded as like the 3rd-best in the country
- Over 120 UCLA medical employees improperly viewed celebrities' medical records
- Many famous actors, singers, and celebrities have been treated at the UCLA medical center
- They wanted to figure out if their favorite actress was pregnant or not.
- UCLA medical center did NOT maintain good privacy over these records
- People tend to run into similar types of problems and people are always accidentally posting private data.

Encryption and Privacy

- Encrypt the data and store it in encrypted form.
- People cannot read it unless they have the key!
- It is easy to screw up your cryptography and you probably are in a situation where your data is safe as long as your stuff is encrypted.

Problems With Data Encryption for Privacy

- Who has the key?
- This usually means they can use and obtain the key when they don't need it.
- Is it stored in a smart card or a file?
- How well-protected was that key?
- If I am not storing the data, how can I be sure that it was encrypted?
- In many cases, we have had circumstances where private data appeared on the Internet.
- Whoever is storing the data kept it in plaintext form
- Yahoo! has had this problem!
- How do we know they are handling it properly?
- Always the issue of using the data when it is encrypted.
- If I decrypt it for use, is that decrypted copy going to be safely used.

A Recent Case

- Yahoo lost about 450,000 user IDs and passwords in July 2012

- They didn't salt their passwords because they didn't encrypt their passwords!
- Do we know who else is doing this?
- No we do not know, but LinkedIn had this problem too.
- Yahoo

Network Privacy

- Data is NOT always at rest, it gets moved across a network
- We want to protect the data as it flows across the network.
- Start with encryption
- What is the problem?

Traffic Analysis Problems

- Attackers can learn something he wants to learn by observing what data got sent when.
- How can you hide data being sent between these two parties?

A Cautionary Example

- Voiceover IP made across the Internet instead of using the old-fashioned telephone network
- Skype is encrypted with a brand new key and the people who built Skype are sophisticated on how to use cryptography
- Never heard of problems used by Skype's cryptography
- Despite the encryption, people were able to understand what was being said in the telephone cal on Skype

How Did They Do That?

- Speech conversion to data has certain necessary requirements!
- Let's see what kind of information is going from the speaker to the listener
- See what packets got sent and how long these packets have been sent.
- Need to make a few assumptions
- Speaking in English
- Using phonemes and rules of English
- People analyzed the hell out of this data
- At the very low level of individual sound.
- Essentially, they got about half of the conversation right.
- They were able to understand half the conversation and this was the academic research that researchers put into this
- It is entirely possible that sophisticated organizations could figure out what was said.

Location Privacy

- It is NOT obvious that networking has anything to do with physical location
- Typically mobile devices try to maintain the connectivity with the Internet

- Keep sending out data wirelessly
- All of these technologies have a limited range and they may provide information about where you are.
- They can tell anybody who they want to tell, this is who I am!
- Can be used to track people's movements and this is another disturbing problem

Cellphones and Location

- Cellphones communicate to a relatively nearby tower
- In the U.S. and probably most other countries, law enforcement agencies can go to the cellphone company and ask where the cellphone is.
- It is at this particular tower and they just have to ask.
- Anytime a policeman feels like knowing, he can say "Hey! This is where Peter Reiher is at"
- At California, the law enforcement needs a warrant.

Other Electronic

- Location privacy - keeping track where someone is
- Relatively easy to localize a user who has 802.11 on his machine as long as you can capture his signals
- Law enforcement is fond of assigning a policeman to follow suspects
- Issue of manpower
- Obvious you are getting followed
- Police nowadays likes to take a special little device and keeps sending out GPS signals
- Reads GPS signals and sends out that information
- For the moment, in the United States, it is NOT legal for the police to slap one of these on your car unless they have the right to search you.
- Supreme Court ruling: can change if new Justices are chosen!
- Private investigators who are less than perfectly honest and do this right now.

Implications of Location Privacy Problems

- Government surveillance makes this easy!
- Government if they are legally allowed to do so can track you.
- Essentially, you have no location privacy from your government.
- In any urban kind of environment, the government can keep track of where you are.
- Maniac stalkers can figure out where you are.

Another Location Privacy Scenario

- Parents would like to know where their kids are
- They didn't ever really know where we were during this time
- Most of us will be parents one day
- You are NOT comfortable if you don't know where your child is.

- Chances are, when your parents first gave you a cell phone, they were thinking of a way to figure out where the hell they were.
- A cellphone would let you figure out where you are at anytime.
- Is this a good thing or bad thing?

A Bit of Irony

- To a very large extent, communication across the Internet seems to be very anonymous
- Communicate as someone who we will never meet in person and these may or may not be true.
- We don't know if they are who they claim to be.
- "On the Internet, nobody knows you are a dog"
- There is NOT nearly the amount of anonymity in the Internet that you think

Why Isn't the Internet Private?

- You can always map that IP address to a machine and there are rare circumstances when you cannot.
- With IP spoofing, it may NOT be the true IP address
- You may NOT be able to get the response and your IP address should probably be exactly right.
- With enough effort, they can map back to an exact spot.
- It doesn't say who is using the computer, but if it is sitting in your basement, then there is probably a strong presumption that someone is using the computer.
- If they choose to track it back to you, you will have to answer back and say what happened.
- You need to deal with the fact that the law enforcement agent is going to show up at your door.

Web Privacy

- Most work on the Internet is done on WWW
- This allows servers to track us and we may NOT want them to do so
- Over the course of time, people have built a whole bunch of technologies to learn a whole lot about us
- Allow tracking at a more precise, deeper level than you thought was possible.
- Before he came to my site, there was an entire browsing history.

Do Not Track

- If we asked, we don't want websites to track us
- We need a standard to specify that we should NOT be tracked.
- This is a standard that is tracked
- In most web browsers, you can turn on the Do Not Track option

The Problems With Do Not Track

- Voluntary
- If he has turned on the Do Not Track option, who gives a damn?
- Not even a protocol responsibility so people can ignore it.
- Ideally, it is supposed to be honored, but will they?
- Microsoft for sure ain't honoring this shit!
- There is something worse than that
- You might think that this option means they aren't tracking you!

What Do Not Track Really Means

- I will still track you!
- However, I won't provide you services like targeted advertising
- I will keep all your information for my own purposes but I won't help you!
- You don't see targeted ads and you see no benefits, so well shit.

Some Privacy Solutions

- Scott McNeal solution
- "Get over it."
- Accept the fact you have no privacy.
- Steganography
- Interesting but not useful

Data Encryption for Privacy

- Store private data in encrypted form
- Particularly important for devices that are easily stolen such as smartphones, flash drives, or laptops
- You want encryption of that data

A Fundamental Issue

- Whenever your machine is on, it has to get information of that encrypted full disk.
- The only time it is protecting you is if your device is of.
- If somebody is stealing that device, they cannot get at that data.
- It does offer you the ability to protect your data.
- Don't leave it on, turn it off!
- If you leave it on, you are going to have to provide a password on it.
- Firewire can let you get around that without a password.
- Safest thing is to turn the device off.
- Enter your password again in this case.
- Limits the benefits you get from privacy

Full Disk Encryption

- Hashed off a password
- Crummy password == crummy key

Homomorphic Cryptography

- Take a piece of data that represents an integer and add a 1 to it.

- Adding 1 to the data or reencrypting the data means we need to add 1 and convert it to the encrypted form of data.
- You can deal with this data but we need to perform processing on encrypted data
- This is like magic!
- The only difficulty is that perhaps based on a relatively new technology, it is very expensive!
- Often very expensive in terms of overhead for the homomorphically encrypted data.
- Thousands of times as many bytes perhaps.
- Really neat technology that is good for demos.
- Very smart researchers are using this for feasible applications

Steganography

- Hiding data in plain sight
- Cryptography is using this
- I have piece of data A and I am going to alter piece of data B so that it is stored in piece of data A
- Usually means that we are going to take an image and embed some data that isn't the picture
- You can do this with other types of formats like sound

An Example

- Lecture 16, Page 41
- Alhambra floor tile in Spain
- Hidden message and I would like to hide this
- Using outguess, you can hide the message in the image.
- Relatively difficult to tell the difference.
- The one on the right has the message hidden in it.

How It Works

- There is a lot of information in the digital image that you don't need
- Let's you use some low order bits to encode that secret message.
- If you do this carefully enough, the change is NOT perceptible for most human eyes.
- If you are doing this kind of thing, I am afraid you are doing this kind of thing and I want to make sure you aren't hiding something.
- I am using a granularity beyond what the human eye could see.
- Am I seeing things that are NOT consistent with ordinary images.
- If you don't want the medium to be used to transmit the message, that is relatively easy!
- If there was a message there, it is gone.
- People wanted to be robust against this problem
- Prevent you from getting rid of my message so I will do something more sophisticated.

What's Steganography Good For?

- Some laser printer manufacturers wanted to prove that something came from their printers rather than some other manufacturer's printer.
- Shady Rat was a whole campaign against a number of sites
- One of the methods they used involved steganography
- Russian spies were communicating via steganography
- Most useful if your opponent has no clue you are using steganography
- With ordinary cryptography, you don't give a damn.

Steganography and Privacy

- If you have hidden your personal data in your family photos, it might be safe.
- Kind of like crypto, but at least with crypto, you have a mathematical foundation
- For steganography, it is more ad hoc and you hope they don't figure out what you are doing.

Anonymizers

- Network site that is a beneficial man in the middle
- Submits requests under their own or fake identity
- Responses returned to the original requestor
- NAT box is a poor implementation of this
- Everything going out from the NAT box has an IP address associated with it

The Problem With Anonymizers

- Someone is running the anonymized, and they know who is who
- It turns out he is a bad guy doing terrible things to you.
- He may fail for a # of reasons
- Someone may convince him that he is you and there could be code watching everything in the anonymized

An Early Example

- Back in the early days, there was an anonymization service for email
- Create a brand new fake ID and send it to you
- Ran for several years in Finland
- A court order required the owner of this service to provide a real address
- He had to comply with the court order and did not like this, so he shut down this service.
- This has often happened with this kind of service that does anonymization

Q. Why didn't he delete all the records?
A. He couldn't send responses to everybody, and once he had done that, he would get another conflicting court order.

- Happened several times in slightly different forms

Onion Routing

- Researchers came up with this idea to handle the issue of traffic analysis
- If all you cared about was encrypting data, the data would be protected, but we had NOT protected ourselves against traffic analysis
- What can we do about this?
- Onion routing is the solution!
- Conceal the source and destination of information we want to hide privately.
- Each packet is going to be encrypted multiple times.

A Little More Detail

- They are going to have special responsibility and they will each get their own private/public key pair.
- Everyone using the onion router will obtain he keys for this
- Many users are going to use this set of onion routers
- This is in the hopes of providing anonymity
- Set of nodes will mix up all these messages and make it hard to figure out who is talking to whom.

Sending an Onion-Routing Packet

- You don't want anybody except the destination to read the contents of your message.
- You will encrypt that packet with the onion router and you will repeat several times.

In Diagram Form

- Lecture 16, Page 52
- If all they cared about was the data contents, then we want to make sure he cannot tell the contents
- The stuff in the middle are onion routers!
- I am going to wrap it inside another message, and choose another of the onion routers and encrypt it with his whole public key
- Wrap it inside another message and choose another onion router.
- Encrypt the whole thing with his public key

What's Really in the Packet

- Lecture 16, Page 53
- Unencrypted header will go to the outermost onion router

Delivering the Message

- That message gets sent to the first onion router
- Anyone watching this knows that this guy sent a message to that onion router
- Visible if the person is watching the traffic
- Maybe it is for me, maybe it is NOT for me

- When he decrypts it, there is a header that indicates it should go somewhere else.
- The message now goes to him and here is an onion router message
- This looks like it should go to the purple onion router.
- There will be information from the decrypted message that will indicate the sender

What's Been Achieved?

- The only people able to read the contents are the source and destination
- No-one knows who sent the message except the receiver
- You want the receiver to know who sent you the message
- Nobody knows who received the message except the sender
- He does NOT know this is the actual destination
- It is another hop in our onion routing protocol and we don't know what happened
- The message moved on from this next step assuming you got it all right.
- If this were the only message getting sent, you would see it go from one guy to a particular onion router and sending it to another onion router.
- What if there are 5,000 people using the onion router service?
- A bunch of messages will go into each onion router and they will all go to different onion routers.
- Which of the 5,000 going out matches which of the 5,000 coming in.
- It becomes exceptionally difficult to figure out how to match up these messages.

Tor

- Overheads here because instead of using quick Internet routing, we are popping up to the application-level
- Then, we have to pop it back down again so it is very expensive
- Eventually, getting there will take a lot longer
- Onion routing is real, so real that you can use onion routing
- Tor is the most popular of these systems (The Onion Router)
- Who paid for onion routing? The U.S. government!
- The main thing you would do with this is hide from the U.S. government, which is hella ironic
- You can join in and help out and use original onion routing software
- Altered many times
- IETF is investigating standardization for this

Why Hasn't Tor Solved This Privacy Problem?

- Slows the Internet down by orders of magnitude
- Hard to use and Tor wants to make this a simpler thing to use for this.
- Chinese government doesn't like Tor, so you will have difficulties using this in China

Can't I Surreptitiously Run Tor?

- The Tor router knows who to send it to on the next hop?
- This becomes fairly public knowledge and we secretly know there are these Tor routers out there.
- If someone is observing traffic behavior of the node, then Tor does not look like anybody else.
- No connections that are visible here
- Doesn't look anything like what an ordinary computer does.
- They keep trying to improve these problems

Q. Using Tor in Harvard?
A. NSA has done its best to bug Tor routers and they even paid researchers at Carnegie Mellon to get around anonymization problems

- It is the case that compromised Tor routers will NOT provide the anonymization that we seek.
- Surveillance agency can get perfect traffic analysis scenarios

Q. Isn't Tor used for criminal activities?
A. Selling drugs, child pornography.

- Not used by ethical dissidence. Primarily criminals!

Privacy-Preserved Data Mining

- Instead, you cannot ask about individual records, you want to ask queries about records.
- You don't want them to use these queries on aggregates to deduce statistics.

Approaches to Privacy for Data Mining

- Perturbation
- What is your salary?
- Make some changes to the salary and do NOT perturb the aggregate value
- Blocking
- Certain fields in this record to ask aggregate queries about
- Sampling
- I want to know the salary of everyone in this neighborhood so I can get the average salary.
- If I choose 2,000 truly at random, it would be a good representation
- Start asking a bunch of queries and try to get some individual values.
- It will be very hard to deduce anything from an individual.
- Heavily studied area of research

Preserving Location Privacy

- Can we prevent people from knowing where we are?
- Keep sending out signals saying that I am here
- I would like to find out where the nearest gas station is.
- Report current location.

Location-Tracking Services

- Get reports on where my mobile device is.
- For other applications, ask location tracking service for this.
- One thing we might do is turn this on or off.

But…

- What if I am entering a “sensitive area”?
- Turn off traffic while we are in that area.
- We are going to see a series of points saying where you went and then you lost track of him.
- A little while later, what happened?
- He went to the location he was trying to conceal.

Handling Location Interference

- Reduce update rate
- Bundle together areas
- Sensitive area is bundled with other areas that are sensitive.

So Can We Have Location Privacy?

- We have looked at your algorithm and figure out your traffic patterns.
- People who have built Tor will do things so your techniques no longer work.
- This cycle just keeps going and going

The NSA and Privacy

- NSA is the U.S. National Security Agency
- In actuality they are probably the world’s largest, intelligence gathering agency
- In 2013, a guy named in Edward Snowden has been gradually been making secret documents public knowledge
- This really changed our conversations about privacy because organizations were capable of doing a lot more surveillance than we thought.
- Heavily involved in watching things like keeping track of phone calls.
- We don’t know what they are doing with this data.
- There is a major debate going on in these countries.
- These debates are concerning how much privacy we are giving to these governments.
- They need this to protect us from terrorists and we need to achieve that degree of protection.
- Are we happy giving up the amount of privacy we need to achieve this security?

Conclusion

- Privacy is a difficult problem in computer systems
- We will do whatever you want us to do and they haven’t done that.

- Probably an area where there won't be a technological solution
- Legal assistance is probably going to be required here, so this is never a good piece of news.
- Lawyers always say to go for the technological solution
- The worst thing you can do is to get undesirable consequences
- Solving technological problems through legal means is probably not the intention!

W 9 Dis 5-27-16

- Let's say UCLA Health got broken into
- What are some issues related to this health data besides identity theft?
- A lot of famous or powerful people go to UCLA Health, and they want to keep secrets because it can affect their careers.
- Right now, it is made for famous people and there are a couple of hundred celebrities who have had their health data hacked or leaked.
- Disclosure that you weren't really asking for but it caused all sorts of havoc and a disconnect here.
- Automatic disclosure similar to social networks.
- The issue is that today in the physical world, we have had time to define ethics and standards
- For the Internet and tech companies, the law is 10-15 years behind the technology.
- There needs to be precedents established
- Social media networks have some trust factors here.
- The U.K. have a data-sharing agreement with Google and they basically dumped all the health records (millions over the years).
- Google has all these nice, sophisticated algorithms but they need data!
- The upside of doing analysis on this data is so big they are willing to trade privacy risks.
- If you went to Bing, you get points and then after a while, you get free stuff.
- You can write a bot to do this, but Dylan earned enough to get a free, framed picture.
- There are 5 mobile views you can do but you can do Chrome dev tools to hack this.

Privacy of Personal Data

- It just needs to be declared what you are doing with the data before you use the service.
- What if they don't know what to do with the data?
- They need to alert you about this!
- We need to sell this all now and does this apply backwards?
- Can you delete data?

Credit card companies

- Buy a stock and they got prosecuted because they weren't supposed to use this type of data.
- Chipotle has been really crashing and they knew this
- Is there any sort of algorithm that does something in AI?
- If you don't know what you are looking for, it is hard.
- Data has many unintended consequences

Steganography
- Hiding data in plain sight
- Often puts information in pixels of images that do not do much in terms of rendering the image.

What's Steganography Good For?
- Could be hidden on Facebook, Instagram, or Snapchat.

What if Everything was Anonymous?
- A lot more cyberbullying
- Putting your identity online makes people less inclined to post aggressively

When Robin Williams committed suicide, his children were cyberbullies
- Initially, there was this thing if you used real names, this helped to deter against cyberbullying.

Hospital records
- You could get zip code, gender, and a plethora of other information
- Linked it to the voting records and they found out information about the governor, who had a serious heart disease.

K-anonymity
- Inserts more noise and data to handle all the problem of a small data set.
- Number of counts of certain attributes
- Identify a person here

W 10 T Lec 5-31-16
Lecture 17: Evaluating System Security
- Not a great deal of consensus on how to do this.
- Computer science is an engineering discipline and we like to build things and make statements about them.
- We have a hard time making those kinds of statements.

Evaluating Program Security
- What if you aren't going to write a secure program?
- You aren't necessarily going to build a new program.
- Your job is to find out if an existing piece of code is or is not secure.
- How do you do that?

Secure System Standards

- This would be a very easy thing to do if we had some standards
- If we could be sure that existing products or systems met those standards
- Approach used in many other fields
- Look at electrical appliance and make sure it doesn't burst into flames
- Make sure it doesn't cause any problems
- Could we define a standard so that we can make some fairly strong statements about the security of your system
- Purpose was to take a look at two programs that claimed to do similar types of things and get information about cost, speed, security, etc.
- Originally, what was looked at was Operating systems - base level of software in your computer system
- If OS isn't secure, then nothing is secure
- They then went to higher levels as well.
- You built a piece of software and it has or has not met the security standards

Some Security Standards

- U.S. Orange Book - old and archaic
- Common Criteria for Information Technology Security Evaluation - used today!

The U.S. Orange Book

- Defined by U.S. Department of Defense in the late 1970s
- Even back then, they cared about the security of their systems in contrast to commercial computers and there were no PC's back then
- We need to make sure our enemies aren't breaking into our system and causing our military to fail.
- While this was built and it lasted for a while, it is not used anymore.

Purpose of the Orange Book

- The U.S. military buys a lot of stuff
- Purchase secure OS
- How do we know if it is secure?
- They need to come present their OS and explain why their OS was more secure than other people's.
- We want some information demonstrating we have those capabilities and they are properly built.
- Flaws in programming can lead to a completely insecure system.
- Look at "head-to-head" between security of systems
- Example
- Payroll of the U.S. army
- Controlling a fighter aircraft
- Different purposes for these operations

Orange Book Security Divisions

- Subdivisions for some of these groups
- If we have built an OS, we had to get a certification from the government and they would give you a certification saying you have met a standard.
- The exception was the D level, which meant you haven't done squat and my system wasn't secure.
- They are better off with no label than a D

Why Did the Orange Book Fail?

- Very expensive to use.
- Didn't meet everyone's needs.
- Okay for the U.S. Military but not good for commercial enterprise!
- Quite inflexible to meet changes
- In order to get a useful certification like B_1, you had to do all kinds of stuff and this took a good deal of time.
- Take product and documentation and give them to a government agency
- Wait for them to come back and tell you that you have achieved your goal.
- Government agencies do NOT move quickly.
- To get a certified product out to market, it would be extremely slow.
- Time to market is the single most important thing in getting your product successful.
- Even if the quality is inferior to future products.
- Speed is very important.
- It wasn't clear the certifications meant much of anything.
- The whole reason is to get assurance that your product was secure.
- Windows NT was C2 certified, but this doesn't mean that it was actually usable at the C2 level.
- It was C2 certified provided that you didn't attach it to a network.
- A lot of people that weren't U.S. citizens i.e. Russian citizens wouldn't trust this product because it was tied to the U.S. government.

The Common Criteria

- Incidentally, it was called the Orange book because it was published with an orange cover.
- People in the computer industry liked the idea that you could go to a store or shopping location and look up a bunch of products.
- Is it a high security product or a low security product?
- How are we going to do this?
- Let's set up a body that defines some standards and deal with the shortcomings
- We don't want to deal exclusively with OS
- We want to deal with other software like database systems
- We also want to deal with hardware appliances like a firewall

- Designing of the common criteria was based on earlier lessons of security standards
- IETF produces some very long documents associated with the Common Criteria

Common Criteria Approach

- Take your product and have it evaluate to different levels
- You want to have some methodology i.e. Common Evaluation Methodology (CEM) to figure out what EAL level it belongs to.
- It wouldn't make any sense to compare a hardware access point to a software database
- Have something more specific to particular needs that someone has.
- Maintaining database integrity for example
- PP - Protection Profile
- Allows people to specify their criteria for security requirements
- If I want a database that will guarantee your integrity to a high level, my protection profile will specify this.

Another Bowl of Common Criteria Alphabet Soup

- TOE - Target of Evaluation
- Try to give a rating to the product
- TSP - TOE Security Policy
- Here is my TSP and this is the TSP for the product.
- Give you a more detailed notion of what the product claims to do
- TSF - TOE Security Functions
- Enforces TSP
- ST - Security Target
- Shows list of requirements

What's the Common Criteria About?

- What you do with this is that you have a very highly detailed methodology.
- Laid out in thousands of pages of documentation about the Common Criteria

0. What security goals?
0. What environment?
0. What mechanisms?
0. Why should anyone believe you?

How Does It Work?

- I need a secure system, so this doesn't tell us much!
- What do you mean by security in this context?
- Using the CC methodology, you will go through thousands of pages of documentation and figure out a customized profile
- Go and find something that best fits your needs.

- You know that people who are building stuff have built a PP that is already out there.
- Nobody in the world will build a product that meets my PP
- You could hire someone but usually you just buy a piece of software off the shelf.
- None of the PP meet my needs, I will have my people define a new PP and give it to my software developers.

How Do You Know a Product Meets a PP?

- People have doubts whether the ratings are honest.
- People didn't like the fact it was tied to the U.S. government!
- The compromise was to say that every country in the world could decide who they trust.
- If they like, they can set up their own national board or work with some other countries like the European Union (EU)
- This is done on a country by country basis.
- Most countries don't have a committee because this is expensive!
- Usually, there is an independent lab that has a contract and take care of common criteria
- A few protection profiles are commonly used and you can look around and find products that satisfy this EAL

Status of Common Criteria

- In wide usage today.
- People who spend time with evaluation of things under the Common Criteria
- Other people will simply honor another company's certification because they are doing a good job.

Problems With Common Criteria

- Expensive to use
- Not cheap because you will spend extra money
- Slow
- Dump it into the lap of one of these evaluation labs
- Your product is probably behind the market.
- Not clear what it is meant behind these levels.
- Certified to EAL4+ for a particular protection profile.
- Kept requiring security patches so what does it mean?
- It was riddled with security flaws!
- There are some who argued that this involved levels that people achieved.
- Few people get certifications at that level.
- A lot of what is going on is people looking at paperwork and documentation you used when people built their systems.
- This may well be the case that you put in a bunch of effort and you dump it in someone's lap.

- They say a bunch of stuff and out comes an EAL that means very little.
- Especially, the most commonly used EAL would be more looking at the documentation associated with this.
- Not doing anything to examine the code in the product.

Evaluating Existing Systems

- Standard approaches aren't always suitable
- People just don't have better ideas.
- It is about evaluating commercial products, NOT particular systems that some people have built for their enterprise.
- We know from much of what has been discussed in this class that things depend on what you do with their system.
- Very strong cryptographic algorithms could be used in such a way it doesn't take advantage of security
- How you use your overall systems ill determine if you achieve your security goals.
- You need some tool to help you evaluate running systems!
- For particular use of enterprise that needs them.

Two Different Kind of Problems

- Put together custom pieces and we are going to design and implement this system.
- Coding is going to be an important part, but we need an overall secure system.
- My company is running a system and a bunch of my competitors have been attacked with DDoS and their machines have been crashing due to bonnets.
- Is my system better?
- I want to know before word gets out in the newspaper that I have been hacked.

Evaluating System Design Security

- Standards aren't going to work out too well because they won't fit the customized system.
- Our problem is only this customized system, and if they don't assist in our enterprise, we don't care.
- We want to be as right as possible for the particular system we are building.
- Some companies will say if you are secure or NOT secure.
- If that is your business, what will you do?
- Review the process!

How Do You Evaluate a System's Security?

- Do I have a high degree of access to a system
- You won't say if you are going to do a formal security evaluation, but usually, you will have a high degree of access

- You either belong to the enterprise or you have been hired by the enterprise
- You have been hired to evaluate the design for this big web system.
- Is it or is it NOT secure?
- Much of the material here is from a book and Microsoft makes a lot of these approaches available.

Stages of Review

- Where are we in terms of the maturity of the system?
- Are we at the stage where we have just started the design?
- Are we further along and we are trying to get a bigger picture of what is going on?
- We have brought the system up and we have access to different kinds of things in each case.

Design Reviews

- If you are here in early stages, we can have relatively little to work with at this stage.
- We might have a design that is written or visual depiction of something to build.
- Obviously, this kind of review won't find any coding bugs because there isn't code yet
- It could discover fundamental flaws with the overall design architecture
- This could also be useful because you may need to use design review to assign priorities to what to pay attention to.
- You may know where the most critical stuff is in this case.

Purpose of Design Review

- Identify security weaknesses in system
- If I build a system according to a design, what are threats to the system?
- This is where we have to pay attention to the security characteristics in the system.
- Performing threat modeling and think about how the system is going to work.
- Deduce the kinds of attacks that could happen from a security perspective.
- Do this before the system is built.

Attack Surfaces

- Cut down on these attack surfaces.
- If attackers are attacking your software, they need to get in somehow!
- The more ways they can interact, the more potential vulnerabilities you need to be aware of.
- NOT all entry points are equally dangerous.
- There is less possibility of something bad happening and you need to be careful about what you tell them.

- You have to identify what is the most dangerous i.e. lead to escalated privileges.
- This could be run at a high degree of privilege and we need to be careful with that program.
- What is the danger of if they interact improperly.
- The smaller the attack surface, the better
- A system with NO interaction whatsoever is going to have a lower attack surface.
- Anyone who wants to can interact with my system.
- From a security perspective, small attack surfaces are better than big attack surfaces.
- There is more you have to worry about and more places where bad things can happen.
- When trying to design things, minimize your attack surface.

Threat Modeling

- This process is called threat modeling and there are a bunch of ways to do threat modeling
- Many of these steps are common to other ways

0. Information collection
   - Get as much info as possible
0. Application architecture modeling
   - If you just sit down with the guy, you might want to build something a bit more formal
0. Threat ID
   - Looking at that model of the architecture, what could go wrong?
0. Documentation
   - Normally, what you get from this design review is some kind of document and one would hope that people will consult that document
   - The better you have documented, the more useful that will be to the people building the system
0. Prioritizing
   - Hopefully, they look at the built system and do an implementation review and achieve security goals
   - Figure out where to spend your attention later on

0. Information Collection
   - What are the entry points
   - What are the ways to get into the system?
   - Do I have a debugging interface?
   - How would I get in?
   - What am I relying on externally?
   - Does this system rely on buying a database and talking to that database?
   - Am I working with customer websites?
   - How much trust do I put into these external entities?

- Break-ins often happen because they were working with a contractor that did something for them and there was a security flaw with the contractor
- The attacker can get privileged access to Sony's database.
- If you have a guy who comes in and installs the air conditioning equipment that runs off a program, I want to be able to control the AC when necessary
- You say there is an external entity and it is going to be hooked into my system.
- What should I allow it to do?
- Tell it its settings and what it can accept from me.
- What are the major components?
- Needs to stream videos in real time
- Ask how the system will be used and is this the kind of system where we will go away and a customer will get a response.
- Somebody will come and they will do a complex set of things over the next set of 8 hours.
- Are we going to allow people to upload data from our system?

One Approach^1

- Microsoft's approach
- Draw an end-to-end deployment scenario
- There will be a high bandwidth network connection between two computers and all the transmission should go through the firewall and customers on the other side.
- These are the moving pieces we have got in our system and we have a firewall here that needs to be carefully configured.
- Identify several key usage scenarios
- Identify technologies that will be used.
- Apache server
- Oracle database
- Security appliances
- Identify application security mechanisms
- Figure out individual users and the technologies for that particular user.
- Will we ask for a user ID and password.
- Know what you are going to do to achieve this security.
- Transmission security of the data - what encryption algorithms will I use?
- How will I distribute keys?

Sources of Information

- Hopefully you have documentation to describe the system
- Generally you are coming in at an early stage so there won't be exhaustive documentation
- Only works out if you have a high degree of access
- Walk down the hall and ask questions about what you intend to do.
- Standards documentation
- If you have the following version of HTTP, what are you going to do with those requests and responses?

0. Application Architecture Modeling

- Based on this, I want a more organized understanding and I want to have something integrated and telling me about this system.
- Go through some kind of visualization in this system and identify design concerns and prioritize later efforts.
- We want some kind of model we are building.

Modeling Tools for Design Review

- Markup languages like UML
- If you are doing OOP, use classes to describe their interactions
- Dataflow diagrams
- Describes where data goes and what happens to it
- Just use something you are comfortable with and can get the details down!

0. Threat Identification

- We have some models and other information gathered
- Look at the ones that cause us problems!
- If our web server is NOT compromised, the Chinese could have built a design flaw into our processor chip
- Prioritize and think about major threats that are really likely to happen.

Attack Trees

- A way to codify and formalize attacks on a system
- Makes it easier to understand the relative levels of threats
- What happens if this threat occurs and let's take a look at examples

A Sample Attack Tree

- Let's start with one piece of the attack tree and do some documentation review.
- Gather a bunch of information and store the information permanently.
- We need a backend database to hold this information
- One threat we have to deal with is that the attacker can gain access to some personal information from one or more users.
- How could this happen?
- The attacker could just get to the database somehow.
- The attacker could login as a target user and succeeds
- He can then grab personal information because we won't protect a user form himself
- Hijack user's session
- Authenticated user but the attacker can now inject commands that appear to come from that user
- Intercept personal data
- Get the credit card info as things are sent across the network
- How could each of these happen?

- Some of these could access the database but the attacker can exploit a flaw and get personal information out
- Brute force password attack
- Steal the user's credentials via a password
- Try every single password and steal credentials
- Steal a user's cookies and maybe he can grab it, steal it, and reuse it himself.
- Interception of personal data
- Takes user connection
- Sniff the network to hear the information going by.
- We need to do both of these

- Verizon is creating security for IoT!
- Smart lightswitches
- Smart refrigerators
- Smart sprinklers
- Area where there is arguably a little bit more room for general purpose security.
- What software should we install on it?
- Often, we have fairly powerful systems even though their purposes may be generally very simple.
- Have we successfully concealed all the attack surfaces?
- Are those the only things that can be done by those who have the device?

The STRIDE Approach

- One way of doing threat modeling is called STRIDE by Microsoft
- Uses SDL threat modeling process
- Depends on having a good system model diagram
- This diagram needs to show what components you have got and what data flows you have.
- Description of interactions and will it receive feeds of data.
- This diagram also specifies where data and control cross trust boundaries.
- I don't trust that web server!
- Consider the STRIDE threats

STRIDE Threats

- Spoofing
- Tampering
- Repudiation
- Information Disclosure
- Denial of Service
- Escalation of Privilege

How to Apply STRIDE

- Consider each possible STRIDE threat and each of these 6 problems that you see.
- No repudiation issue and pay particular issue to those that occur across the trust boundaries.
- Figure out how these interact with each other.

0. Documentation of Findings

- Summarize findings
- Generally want to prioritize threats
- Be careful about using an appropriate cryptography library
- You have to be careful about this firewall here but it may not be that big of a threat.

DREAD Risk Ratings

- Gives a number to a risk rating
- This threat is more important than that threat.
- Look at 5 different things
- Damage potential
- Reproducibility
- Is this something that can be done every single time?
- Will it always be there or is this something that occurs only on leap days?
- Exploitability
- How hard would it be for an attacker to do this?
- Potential for a heap overflow?
- Hidden away deep in the system and try to exploit it.
- Is this something where we have handed out a device that contains a hardcoded user ID and sitting there in code?
- Affected users
- One user affected or a bunch of users affected.
- Discoverability
- Add up the numbers for an overall rating between 5 and 50
- Small numbers good, big number bad
- Use judgment for this purpose and let me think about all these threats within the realm of these systems.
- It isn't going to be the case where we get perfection
- If you did any reasonable rating, the 45 is probably one where you have to pay attention to.

0. Prioritizing Implementation Review

- The enterprise in question cares a lot about security and has good security properties.
- Their life will be a lot easier if you have your design review in front of them
- It would have been nice if you had notes about the security of the system.
- Something that will be really helpful!
- The extent to which you can figure out how to do it will make it a lot cheaper.

- You need to decide how to focus the limited amount of attention.
- Make a list of major components and identify which component each risk belongs to.

Q. What is a trust boundary?
A. You really trust a machine and give a lot of capabilities and it has a 2nd component that communicates with the 1st component but has a lower level of trust.

Application Review

- Review of the completely written, perhaps running, system.
- Has advantages and disadvantages
- There could be code or a machine that is running and operating.
- Such systems can be very large and they can be larger by any metric.
- There could be more stuff you are looking at.
- If you didn't do the design review, we would like you to do the security evaluation and there might be nothing about the system.
- If you didn't do the design review, you can see what they left you.
- Here you are sitting in front of your system and they could be acting in some different ways.
- They are connected in different ways and you aren't sure what everything is.

Need to Define a Process

- Do NOT just dive directly into the code.
- You need a plan and process
- Limited budget in terms of how much money will be spent, who you have available to help you, and time.
- You need to work within the budget and be pragmatic
- You cannot run formal verification by hand in a program with millions of lines of code
- We need to make choices and have it yield useful results.
- We cannot divide everything up and there is practically nothing here.
- Need to be flexible and able to change things
- Focus on the results - figure out something useful about the system
- Don't just check off the box that the work is done
- Requires code review
- Source code
- Reading code is a different skill from writing code
- If you have never read someone's code, it is different from looking back at code you wrote a year ago.

Review Process Outline

0. Preassessment
- Get a high level picture of what is going on in the system.
- Figure out the purpose of each of your applications.
- Which of the machines does each run on.

0. Application Review

- Spend some time

0. Documentation and analyze
   - Look at results that describe important security characteristics
0. Remediation support
   - Help to fix problems

Reviewing the Application

- Start off knowing very little about the code
- What it's supposed to be like may not match what is written
- You will end up knowing more about the application after reading it.
- Find the deepest problems after you understand how the system works
- What you want to do is get deep quick!
- Get important knowledge quickly so you can start to recognize the problems in the code.
- Know that this database is supposed to send information to an application
- We need to worry about that and if we only know about 3 applications, we need to learn about a fundamental flaw in the implementation of the design.
- Get some results here and we cannot do design review if one isn't present.
- We need to spend a few days doing a design review upfront.
- We also have to expect an iterative process
- You may have to go back to it because you may not really understand what is going on.

Q. Is this usually done in parallel?
A. Usually because it is a very large program. However, you may have to do this on your own.

- Another part of dealing with design reviews and implementation reviews

General Approaches To Design Reviews

- Top-down
- Start with high level information and go down from there
- Bottom-up
- Build a model of the system as you go along
- Hybrid

Code Auditing Strategies

- We aren't too interested in whether the code works or not.
- Don't worry about finding implementation bugs or the code works at a high speed.
- Code comprehension (CC) strategies
- Analyze source code
- Candidate point (CP_ strategies
- Look for issues in key distribution and make sure they do it right.
- Find a bunch of potential issues and go look for those.

- Design generalization (DG) strategies
- Data flows from here to there and make sure that is happening there.
- Find representations of the design and make sure we have uncovered flaws in the designs.

Some Example Strategies

- Trace malicious put (CC)
- Attacker gave us bad input
- Intended to do something evil.
- Where does the data come into the program?
- Let's trace to see what happens to that data.
- Think about what the implications are!
- Analyze a module (CC)
- This is the module that performs cryptography to store things on the disk.
- Simple lexical candidate points (CP)
- Vulnerabilities in these systems like bad system calls
- strcpy()
- grep | strcpy()
- Lexical issues
- Design comformity check (DG)
- Check to see if the code matches what you have got

Guidelines for Auditing Code

- Perform flow analysis in the functions you examine
- Read code more than once and it isn't a checklist
- As you develop a better view of the program, you may have to look again.
- Sooner or later, you will run into some algorithms that are critical.
- Do some desk checks and figure out from a combination of attempts what will happen.
- Use test cases for important algorithms
- Choose inputs carefully
- Look at likely inputs and figure out what if the attacker gives you something really bad!

Useful Auditing Tools

- Source code navigators
- Keep track of things going from function to function
- Allow you to jump around quite easily
- Debuggers
- Binary navigation tools
- Better than not having them but not as easy to use as source code navigation tools
- Live systems have critical places where inputs come in
- Think of some pathological inputs, but maybe there are other inputs.
- There is code called fuzz-testing to see what happens and generate some inputs at random.

- Automates testing within a range of important values
- Usually there aren't computational capabilities to get everything, so get a good representative set.

Evaluating Running Systems

- Somewhere along the lines, we could be running a system critical to our enterprise
- Other people have had ransomware and order a huge ransom to be paid
- People could have a botnet installed and a competitor could have lost a lot of personal data
- We want to know whether we are at risk or not!
- Tell me if I am okay or if I should affect the security of my system.
- If you have to do a full evaluation, it will require a whole lot of work.
- A lot of tools you can use and it will e similar to evaluating a design.
- You need a model of what the system looks like and where is the important data flows.
- Once it gets down to the core, there are very important tools you can use

Logging

- No system has perfect security
- Has the attacker discovered a weak point and has he exploited it?
- Am I in trouble at this moment?
- Most important tool you have for this.
- Keeps track of important system info for later examination
- If you have no reason to believe you are having trouble, keep a log to verify your beliefs.

The Basics of Logging

- OS and complex applications will record messages about activities
- Every time a new piece of email comes in, we have to see if it is completely separate from this piece of data.
- Put into log files and these messages can record events
- Good to log unexpected events
- Attackers will often do things that were not expected.
- If he does things you don't expect, it would be nice to log those unexpected things.
- A lot of attacks leave traces lying around in logs

Access Logs

- Somebody can do a remote login into my system
- Once the guy logs in, which files did he access?
- Basic applications
- Password files?
- It is important to log failures because attackers generally require trial-and-error to exploit a flaw.

- If you log failures, you will know that someone tried to compromise your system

Other Typical Logging Actions

- Logging failed login attempts
- Logging changes in program permissions
- This program used to have a certain level of privileges and there will be a different security effect in the future.
- If you know one of those is lying around, there is no previous awareness.
- Logging scans of ports known to be dangerous
- You know someone is trying to get to you, unfortunately.

Problems With Logging

- A lot of data
- Practically all of it is irrelevant from a security perspective.
- Separate out the good stuff from the bad stuff.
- Doing this by hand is very laborious
- We are going to step aside and write something to a file and this doesn't help you with its ordinary business.
- It gives me information later on but it doesn't help me get a webpage out to the user.
- There will be overhead here and unless you are careful, the log can have a very large value and you will use a lot of disk storage and flash storage in your logs.
- If you never truncate your logs, it just keeps growing and sooner or later, it will take up space.

Log Security

- Attackers aren't stupid, what is he going to try to do?
- Look at log to find traces of you breaking into the system.
- If I have permission to edit the log, you can help get rid of traces in the activity.
- An unsophisticated attacker will erase an entire log and wipe out an entire portion.
- A very sophisticated attacker will be more surgical and he may or may not be successful in removing every trace in the log.
- Append-only access prevents you from writing arbitrary stuff but you can add things at the end/
- You can log to hard copy - not a teletype but a write-once medium (once it is written, you cannot rewrite it)
- You can log to a remote machine - if the attacker wants you to fiddle with the log, you have to break into the machine

Local Logging

- Look at 300 machines and figure out overall what is going on
- Some of these machines will be compromised and you are going to have to share resources with everything else that this machine does.

- Certain delays in order to log messages on the machine.
- Failure of one machine means you have lost 1/300th of your log and this could have information you need.
- You need information over here and this could be much more complicated.

Remote Logging

- What am I going to do?
- Send all of my log messages and this will give me a combined view of all of this installation.
- You can have a specialized machine that does nothing but this, but it could be done specifically for logs.
- 299 machines and nothing gets logged is the logging machine.
- 50 of my machines cannot reach the logging machine, and the attacker can compromise this.
- If there are 300 machines, he can fiddle with the logs and the other 297 will have their logs fiddled with.

Desirable Characteristics of a Logging Machine

- You want it to be highly secure
- Limit who can login to the machine and turn everything else off.
- You won't have users home directories here or other things of that kind.
- You are going to build in a really big log with storage space.

Network Logging

- Don't just log things on your machine
- You can have sniffing on your network and it doesn't have to be on the machine where it happens
- Use it to detect various problems like an attempt to remotely access your machine

Logging and Privacy

- Assuming any privacy requirements
- It can eventually get seen vey someone who isn't; supposed to see it.
- When I create a log message, am I putting private information into the log?
- If that is so, then this is just as sensitive as where the private information normally lives.
- The web server's log will contain credit card numbers, so if the attacker steals the log, he doesn't need to steal from the database.
- Keep logging failed login attempts
- Then, you know this is a password getting attack
- Do NOT log the password that he is trying to guess
- If you log your mistyped password, it will likely be off by a small amount
- If an attacker can get that information, he/she is very close to cracking your real password

An Example

- Network logging doesn't store the payload of the packet, only the header
- You can tell if it is a TCP message vs UDP message
- You can tell a whole lot of what is going on for information.

Auditing

- Practical systems require proper policies and consistent use, but this is very hard to maintain.
- Because you didn't do it, your system didn't achieve the security effect it was supposed to achieve.

Auditing

- You were happy with what you said you were going to do.
- An audit is a formal (or semi-formal) process of following system security.
- If I am looking, you will do it!
- If my system is in good shape, you need to take a look at what is happening.

Auditing Requirements

- You will be tasked to look at policies in your system
- They have to know a good deal about security in general to see if deviation is harmless or extremely dangerous.
- You need to know what security implications are of various things.
- Check and see if the boss screwed up badly.
- Do NOT point fingers at your boss, but you need someone doing the audit that will tell you the truth and won't sugarcoat things.
- You need them to be trustworthy and actually believe what they are telling you.
- If you are a big org i.e. 20,000 employees, you want a group of employees that are your security auditors.

When Should You Audit?

- Periodically
- Switching from Windows to Linux or vice versa
- When problems arise
- User accounts compromised
- Take a good hard look and figure out problems

Auditing and Logs

- Logs are a major audit tool because they keep track of what is going on in the past.
- Auditors will look at your logs and take a look at these
- Here are the logs, so have fun with them.
- Do some automatic examination of these.

- Look at the audit for unusual events and some of these will get audited by hand.
- Get rid of all of this and do this by hand.

What Does an Audit Cover?

- Conformance to policy
- Passwords must have the following characteristics
- No one can bring their own mobile device into the office and turn it on.
- What is the process for this system and adding things to the system.
- What if someone gets fired?
- Immediately get rid of their access.
- Is that actually going to work, or is this ineffective?
- Talk to users and see if the password is compromised?
- You probably shouldn't, and users should not do anything they shouldn't be doing.
- Physical controls
- Software licensing
- Are you using commercial software that you have purchased a license for.
- Do not do things you shouldn't be doing.
- One of your employees fucked up and downloaded Oracle database illegally
- If Oracle finds out, you are going to get sued and lose a shitton of money

Does Auditing Really Occur?

- Yes, of the organizations responding to the survey, it doesn't say much about the quality of audits.

Conclusion

- Don't assume your security is perfect
- Use security evaluation tools to help improve your security
- This isn't something you do once and forget about it
- You do it at all points in the life cycle.

W 10 Dis 6-3-16

1. In symmetric key cryptography