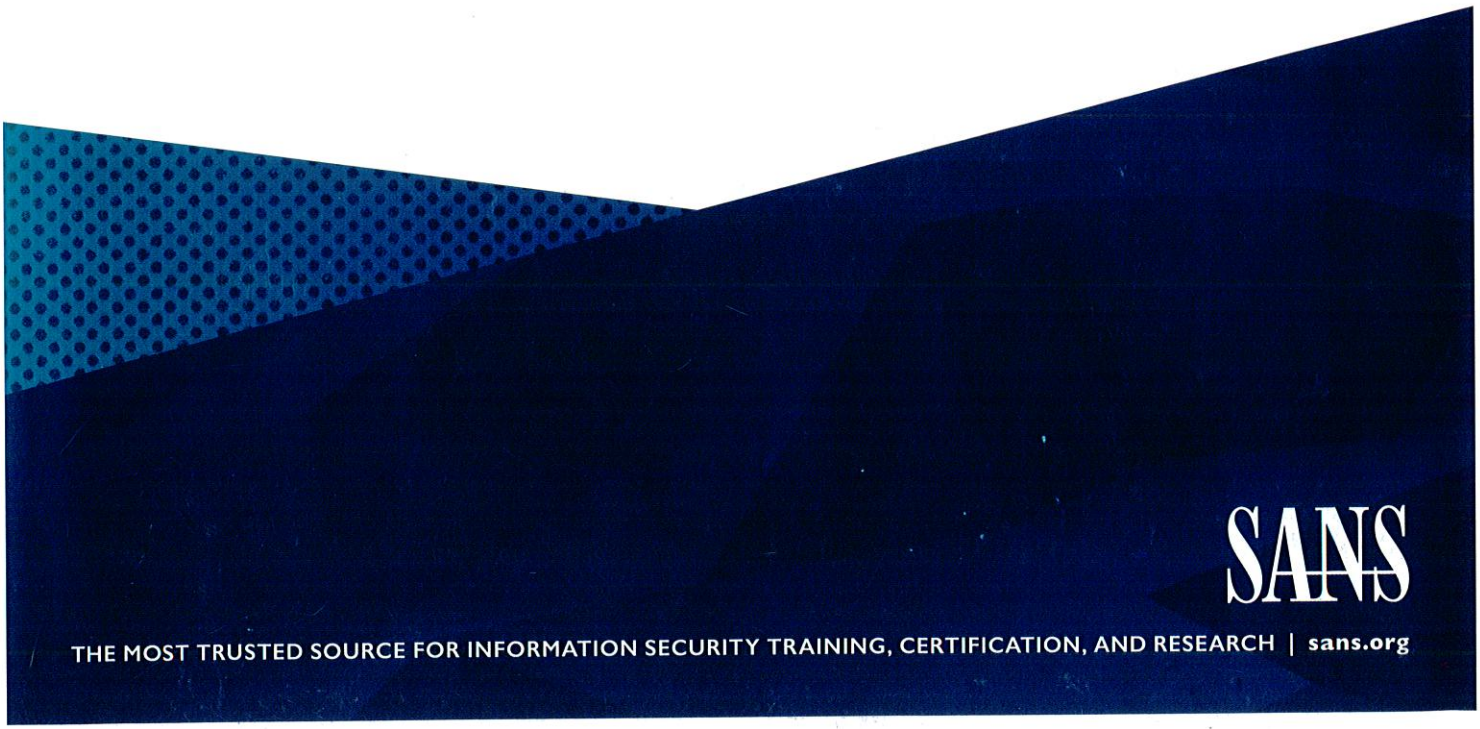
SEC401 | SECURITY ESSENTIALS BOOTCAMP STYLE

401.2
Defense In-Depth and Attacks



SANS

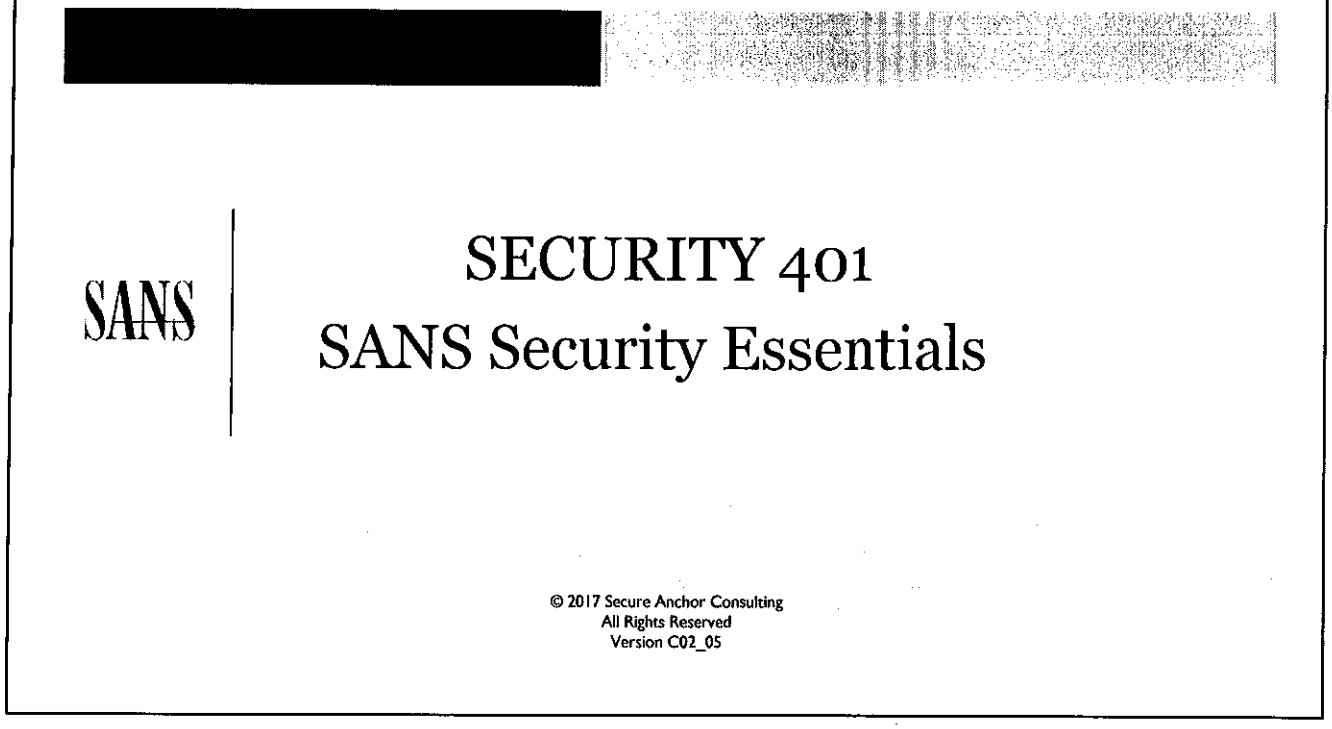
THE MOST TRUSTED SOURCE FOR INFORMATION SECURITY TRAINING, CERTIFICATION, AND RESEARCH | sans.org

SEC401 | SECURITY ESSENTIALS BOOTCAMP STYLE

401.2
Defense In-Depth and Attacks

SANS

THE MOST TRUSTED SOURCE FOR INFORMATION SECURITY TRAINING, CERTIFICATION, AND RESEARCH | sans.org



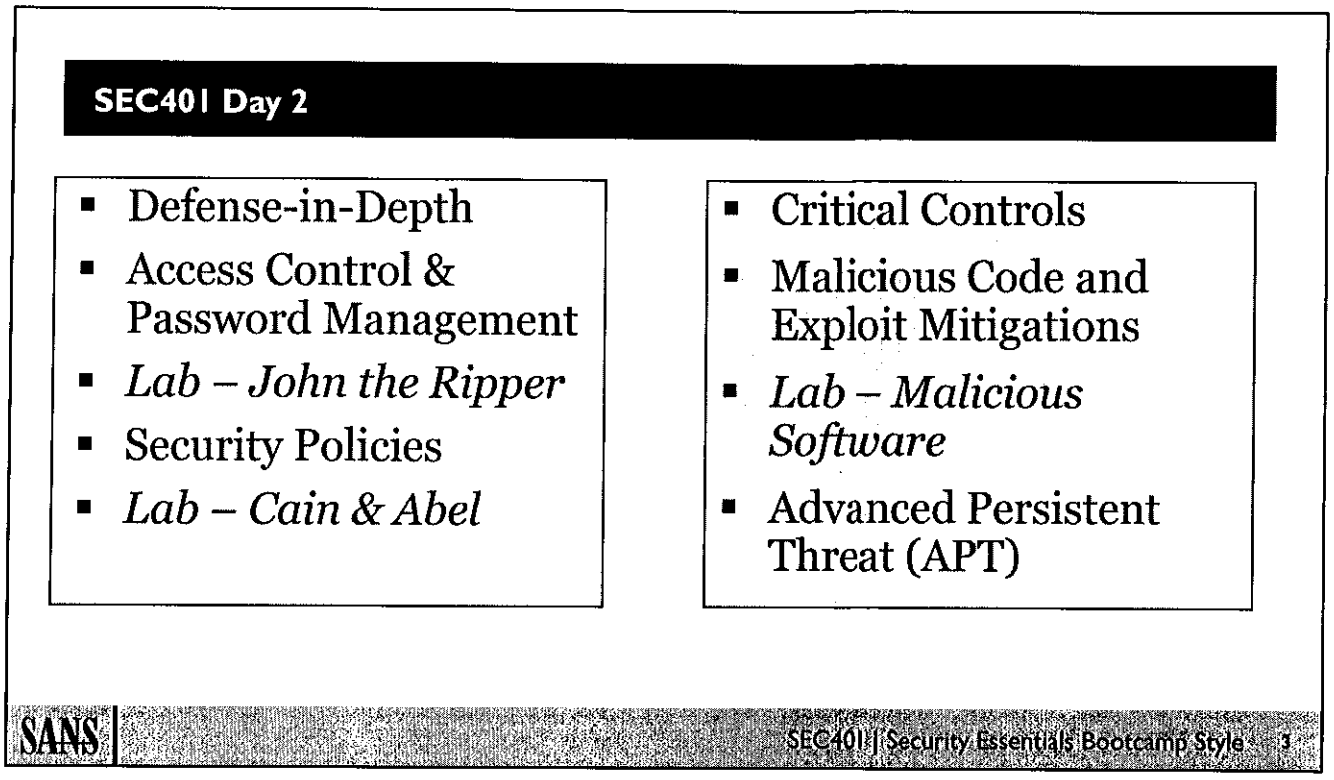
This page intentionally left blank.

Security 401: Day 2

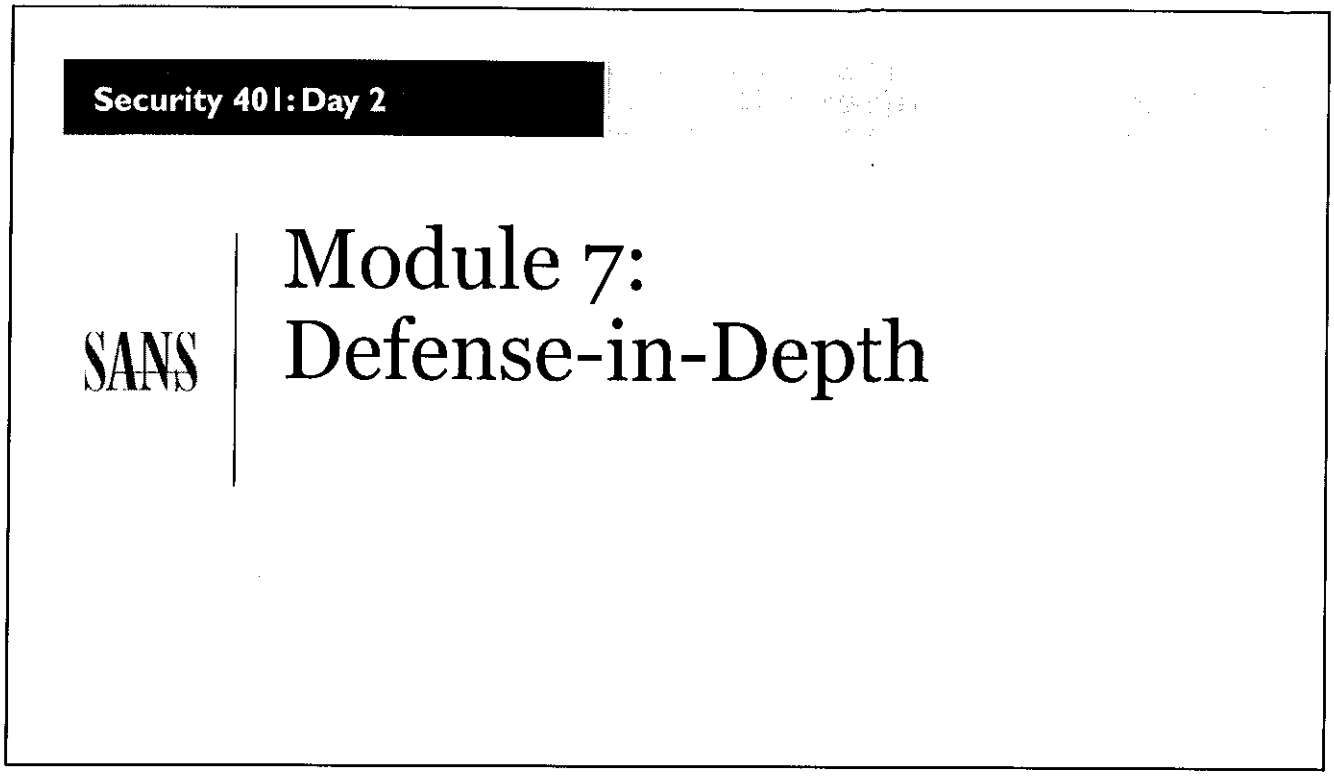
SANS

Day 2
Defense-in-Depth and Attacks

This page intentionally left blank.



This is the outline for Day 2 of SEC401.



Module 7: Defense-in-Depth
This page intentionally left blank.

Objectives

- Defense-in-Depth Overview
  - Risk = Threats x Vulnerabilities
  - CIA Triad
- Strategies for Defense in Depth
- Core Security Strategies

In this module, we look at threats to our systems and take a "big picture" look at how to defend against them. You will learn that protections need to be layered: a principle called defense-in-depth. We explain some principles that will serve you well in protecting your systems. The following are the key areas that will be covered:

- Defense-in-Depth Overview
  - Risk = Threats x Vulnerabilities
  - CIA Triad
- Strategies for Defense in Depth
- Core Security Strategies

Defense in Depth Overview

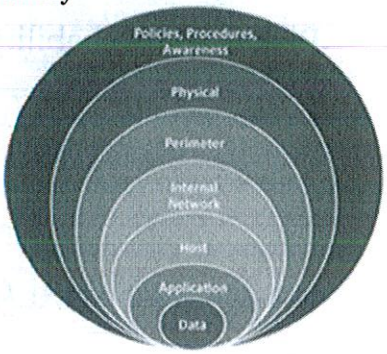
The student will understand what defense in depth is and an overview of the key areas of security

This page intentionally left blank.

What Is Defense-in-Depth?

- There is no magic solution when it comes to network security
- Any layer of protection might fail
- Multiple levels of protection must be deployed
- Measures must be across a wide range of controls
- Integrate defense-in-depth

Prevention is ideal, but detection is a must; however, detection without response has minimal value



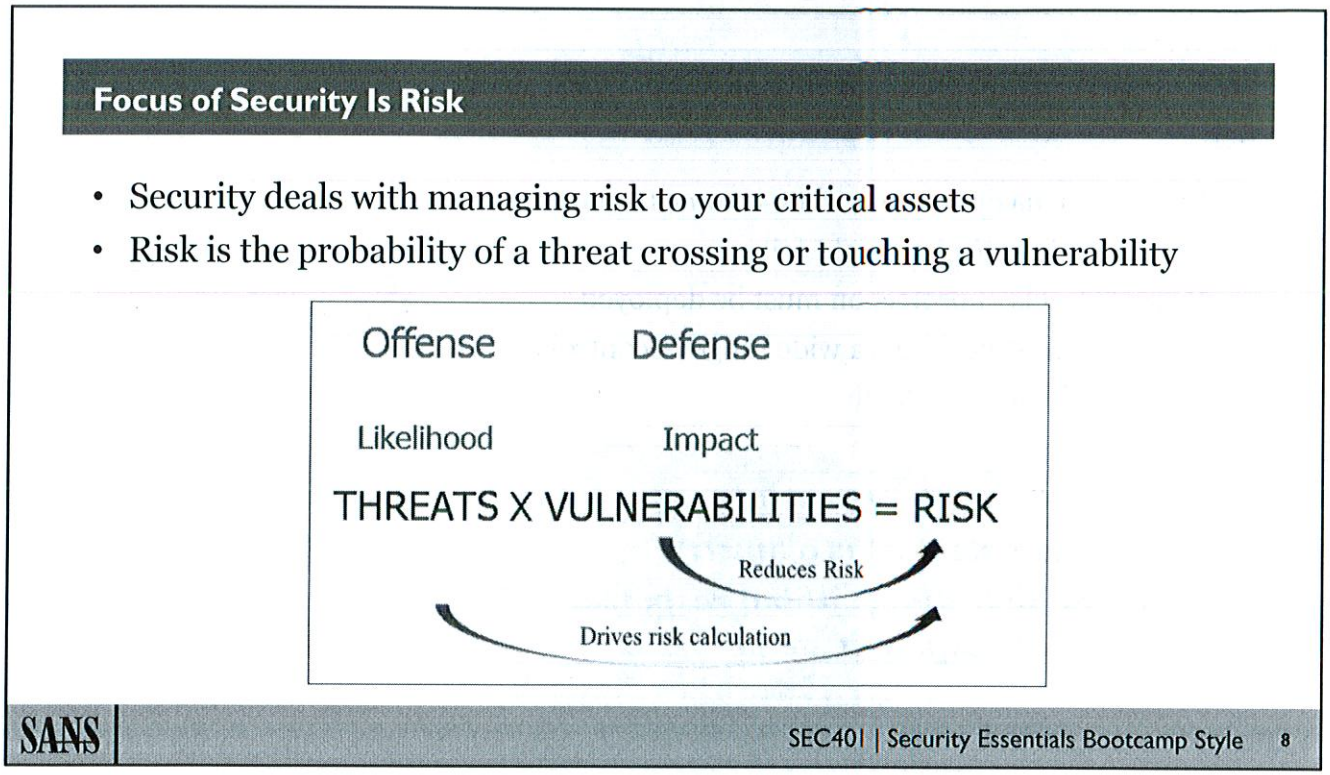
Layers of Defense in Depth

Network security is a comprehensive, integrated approach in which multiple solutions are tiered together to accomplish a goal. There is no single security solution that will make an organization secure, because any single measure could be bypassed (and miss an attack all together) or compromised. When protecting any entity, take a key person like a President of a country, for example; there are many people, measures, and systems put in place to keep him secure. The same robust approach needs to be applied to your network or any critical asset at your organization.

When it comes to network security, there is no silver bullet. Multiple measures that complement each other must be put in place across a variety of control options. For example, you would deploy a preventive measure such as a firewall, a detective measure such as an IDS, and a deterrent measure such as a guard at your front gate just to name a few. Even if one of the measures failed, the other measures would be able to detect the attack before there was a problem—or catch an attack in action—to minimize the amount of damage caused.

References

1. http://www.bizforum.org/whitepapers/microsoft-5.htm.
2. Information Security Basics - http://cloudn1n3.blogspot.com/2014/11/information-security-basics-part-2.html



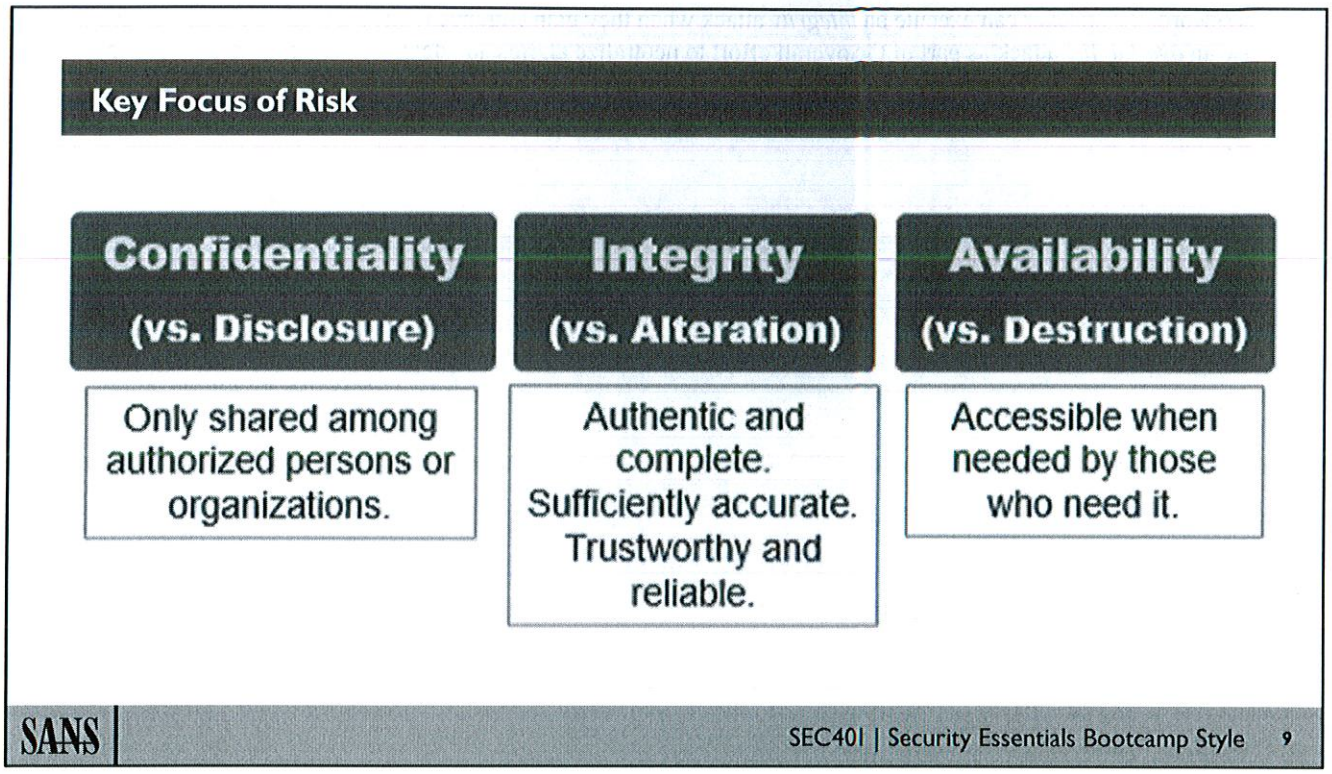
Risks, threats, and vulnerabilities are highly interrelated. Their relationship can be expressed by this simple formula:

Risk (due to a threat) = Threat x Vulnerability (to that threat)

This formula shows that risk is directly related to the level of threat and vulnerability you, your systems, or your networks face. Here is how the formula works:

- If you have a very high threat, but a very low vulnerability to that threat, your resulting risk will be only moderate. For example, if you live in a high-crime neighborhood (thus, high threat) but you keep your doors and windows locked (so you have a low vulnerability to that threat), your overall risk is moderate.
- If you have a high vulnerability to a threat (by keeping your doors and windows unlocked), but the threat itself is minor (by living in a safe neighborhood), once again you have only a moderate risk factor.
- If, however, you have a high level of threat potential (a high crime area) and your vulnerability to that threat is very high (no locks), you have a very high-risk factor.

We must understand that it is impossible to completely eliminate all risk. Therefore, the job of the security professional is to constantly track, manage, and mitigate risk to an organization's critical assets.



Security is all about understanding and controlling risk to our critical assets. More specifically, we are trying to minimize the risk and protect the *confidentiality, integrity,* and *availability* of our critical systems. In deploying measures to protect our systems, we need to deploy a defense-in-depth approach to security.

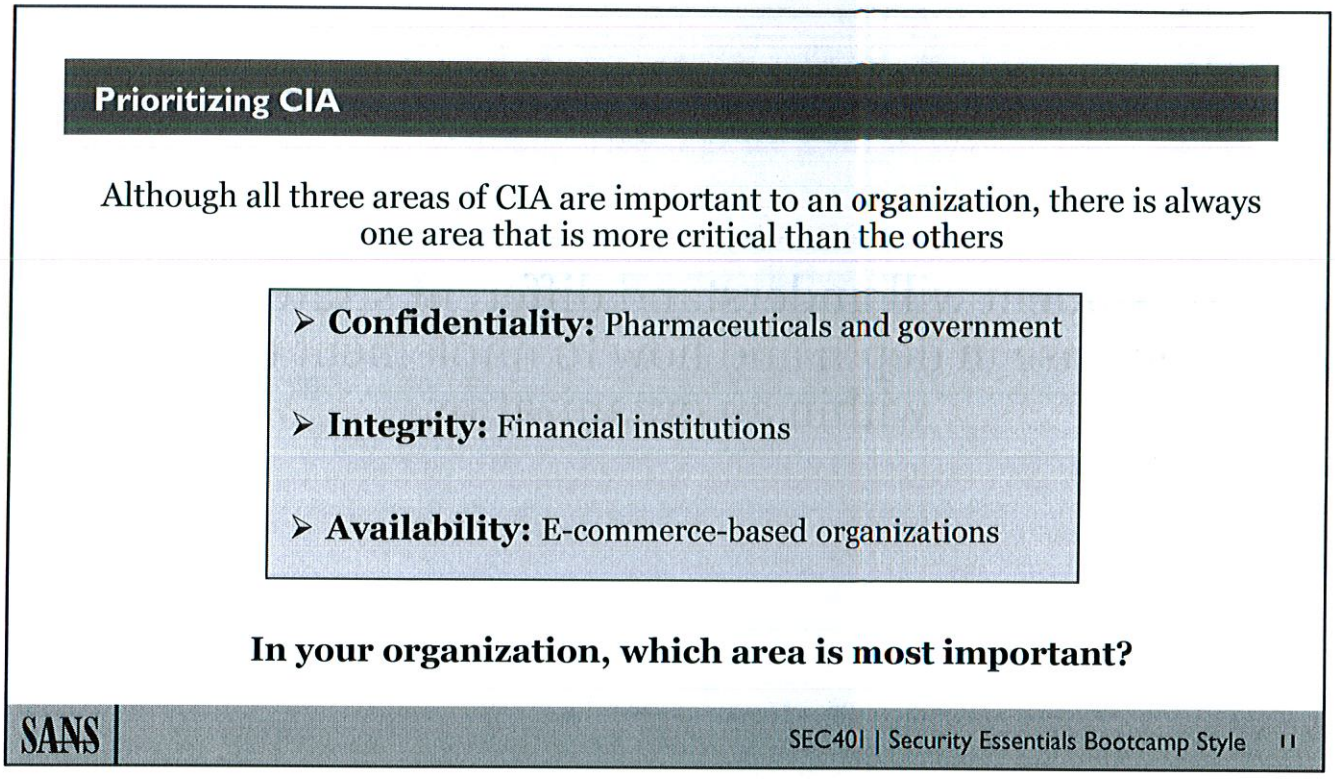
Traditionally, information security professionals focus on ensuring confidentiality, integrity, and availability often called "CIA". These are three bedrock principles about which we will be concerned. When first exploring any new business application or system, it is a good habit to begin by thinking about confidentiality, integrity, and availability—and countermeasures for protecting these, or the lack thereof. Attacks can come against any or all of these.

For example, you have been assigned to oversee the security of your employer's new e-commerce site—its first attempt at conducting business directly on the Internet. How do you approach this? What should you consider? What could go wrong?

Think C-I-A: confidentiality, integrity, and availability. Customers will expect that the privacy of their credit card numbers, their addresses and phone numbers, and other information shared during the transaction be ensured. These are examples of confidentiality. They will expect quoted prices and product availability to be accurate; the quantities they ordered and the prices to which they agreed not to be changed; and anything downloaded to be authentic and complete. These are examples of integrity. Customers will expect to be able to place orders when convenient for them, and the employer will want the revenue stream to continue without disruption. These are examples of availability.

Keep in mind that the dimensions we have been discussing can be interrelated. An attacker can exploit an unintended function on a web server and use an exploit to list the password file. Now, this would breach the *confidentiality* of this sensitive information (the password file). Then, in the privacy of their own computer system, the attacker can use brute-force or dictionary-driven password attacks to decrypt the passwords. With a stolen

password, the attacker can execute an *integrity* attack when they gain entrance to the system. And they can even use an *availability* attack as part of the overall effort to neutralize alarms and defensive systems so they cannot report his existence. When this is completed, the attacker can fully access the target system—and all three dimensions (confidentiality, integrity, and availability) would be in jeopardy. Always think C-I-A.



Which pillar of the CIA triad is most important to your organization? For example, an e-commerce vendor would rely on our online resources for registration. Without online resources, they would be unable to provide services to their customers. Because an organization cannot operate without customers, the priority is availability.

After availability, the next most important dimension of CIA is integrity. Often with banks and financial institutions, integrity is the most important to ensure the accuracy of accounts.

Different organizations will have different priorities in the CIA triad. Confidentiality is usually very important to government organizations or entities with very sensitive information, and integrity is important to financial institutions. Understanding what the priorities are for your organization is a tremendous help in prioritizing security plans for your organization, from design to incident response.

Strategies for Defense in Depth (DiD)

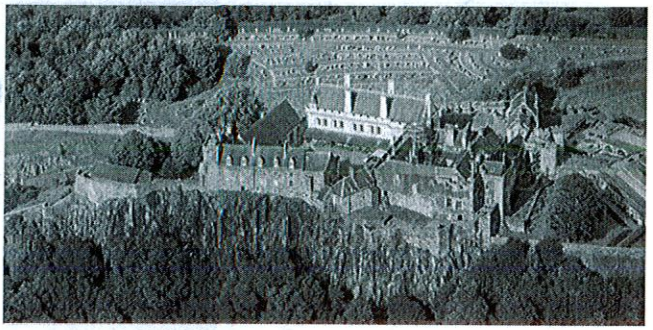
The student will understand different strategies for defense in depth and how to implement them within an organization

This page intentionally left blank.

Approaches to DiD

Deploy measures to reduce, accept, or transfer risk

Four basic approaches
- Uniform protection
- Protected enclaves
- Information centric
- Threat vector analysis



SANS | SEC401 | Security Essentials Bootcamp Style 13

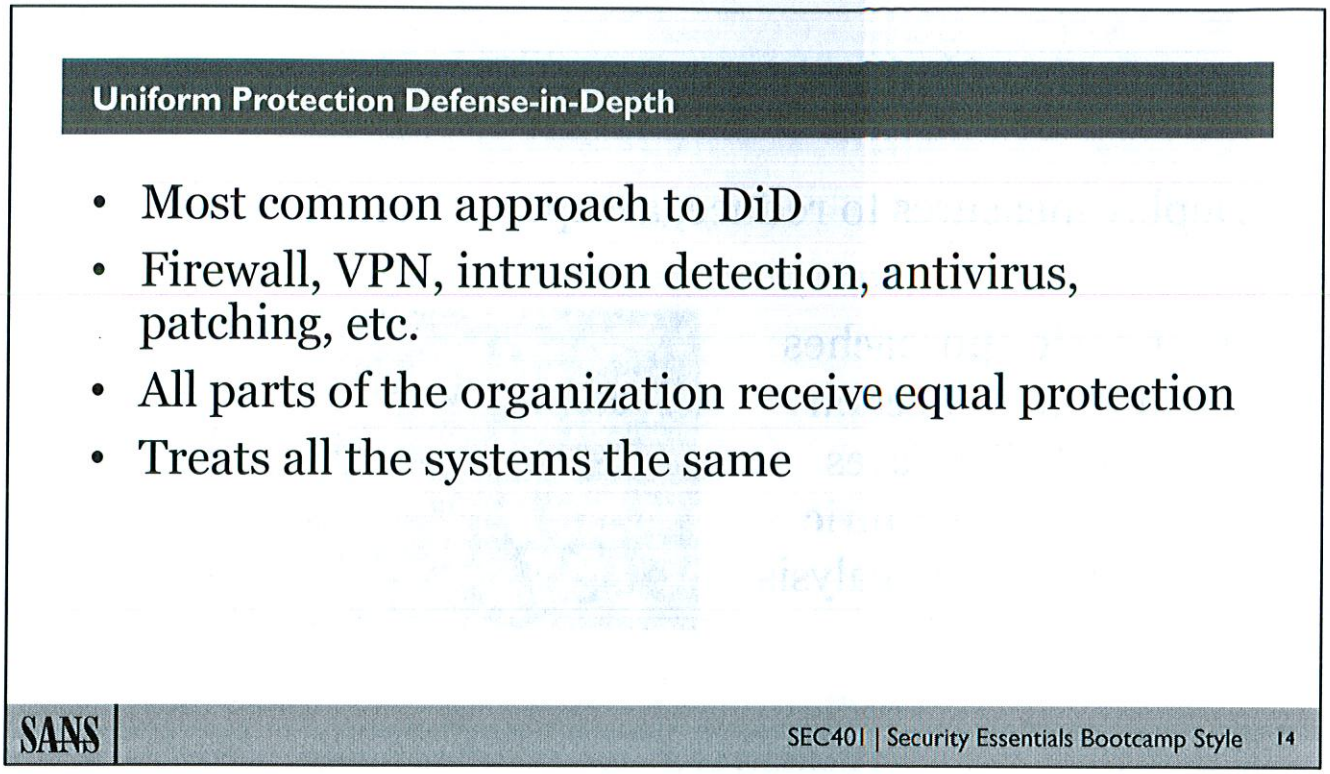
The concept behind defense-in-depth (DiD) is simple. The picture we have painted so far is that a good security architecture, one that can withstand an attack, has many aspects and dimensions. We need to be certain that if one countermeasure fails, there are more behind it. If they all fail, we need to be ready to detect that something has occurred, clean up the mess expeditiously and completely, and then tune our defenses to keep it from happening to us again.

We examine four approaches to defense-in-depth:

- Uniform protection
- Protected enclaves
- Information centric
- Threat vector analysis

Reference

1. Stirling Castle - https://en.wikipedia.org/wiki/Stirling_Castle



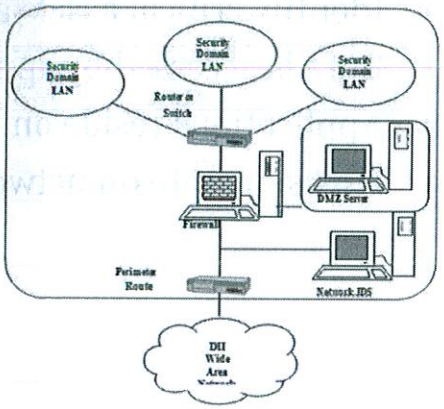
Uniform protection treats all systems as equally important. No special consideration, or protection, is given to the critical intellectual property of an organization. As a result, this approach can be more vulnerable to malicious insiders, because the systems are not separated or categorized within the network.

The majority of attacks succeed because they take advantage of well-publicized vulnerabilities for which exploits have been created. The best answer is to patch the systems, but this takes time. Of all the approaches to defense-in-depth, this one can be the weakest, unless you have a good uniform protection design.

This is also by far the most common approach and usually the starting point for most organizations.

Protected Enclaves Defense-in-Depth

- Work groups that require additional protection are segmented from the rest of the internal organization
- Restrict access to critical segments
- Internal firewalls
- VLANs and ACLs



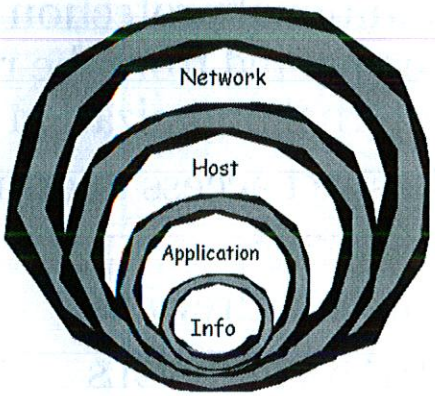
Protected enclaves involve segmenting your network. This can be done by implementing many VLANs across a single network, VLAN segmentation of switches or firewalls to separate out the network. This is a simple, yet effective, technique. Reducing the exposure or visibility of a system can greatly reduce the impact malicious code can have. For example, if you have 5,000 systems on a network and a system gets compromised, it could spread to all systems. However, if you create separate segments with 100 systems per segment, the virus would impact only a small percent of your systems, and cleanup and damage would be minimal.

References

1. Network Enclave - https://en.wikipedia.org/wiki/Network_enclave

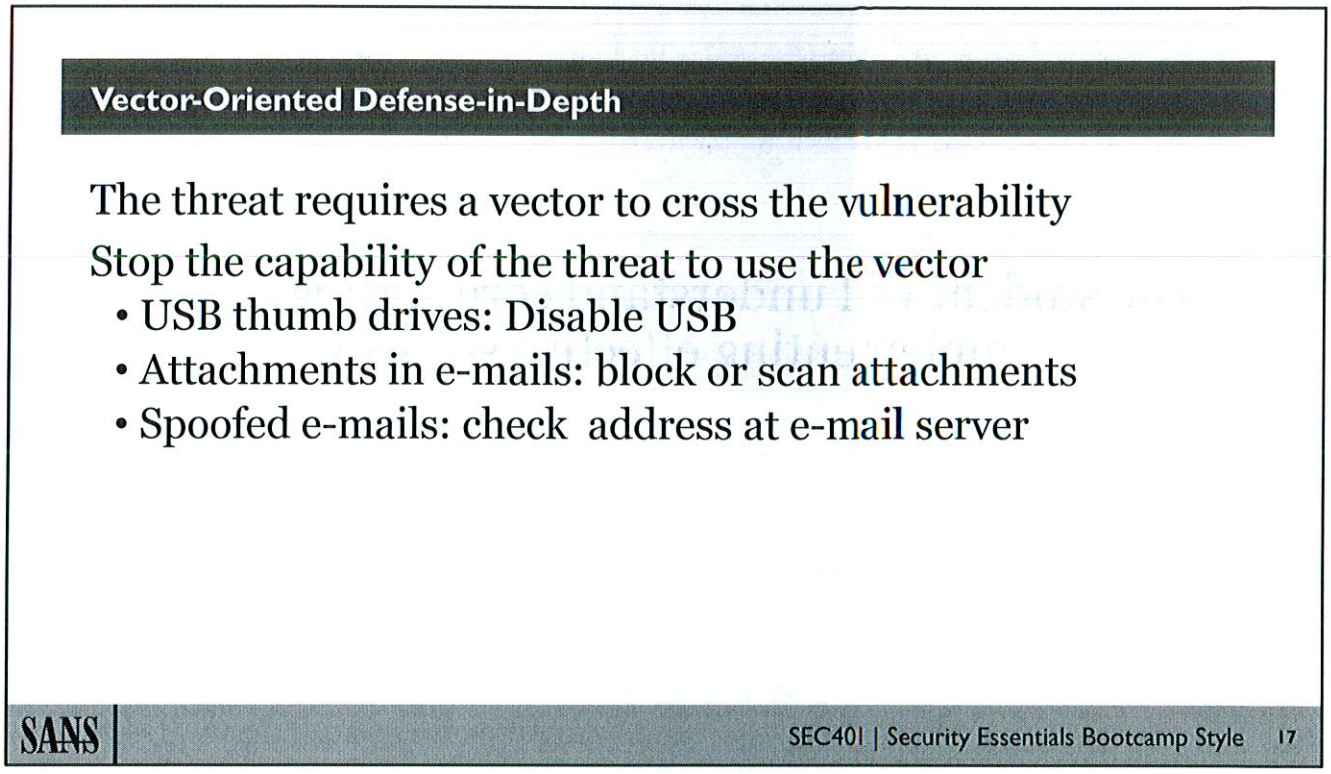
Information-Centric Defense-in-Depth

- Identify critical assets and provide layered protection
- Data is accessed by applications
- Applications reside on hosts
- Hosts operate on networks

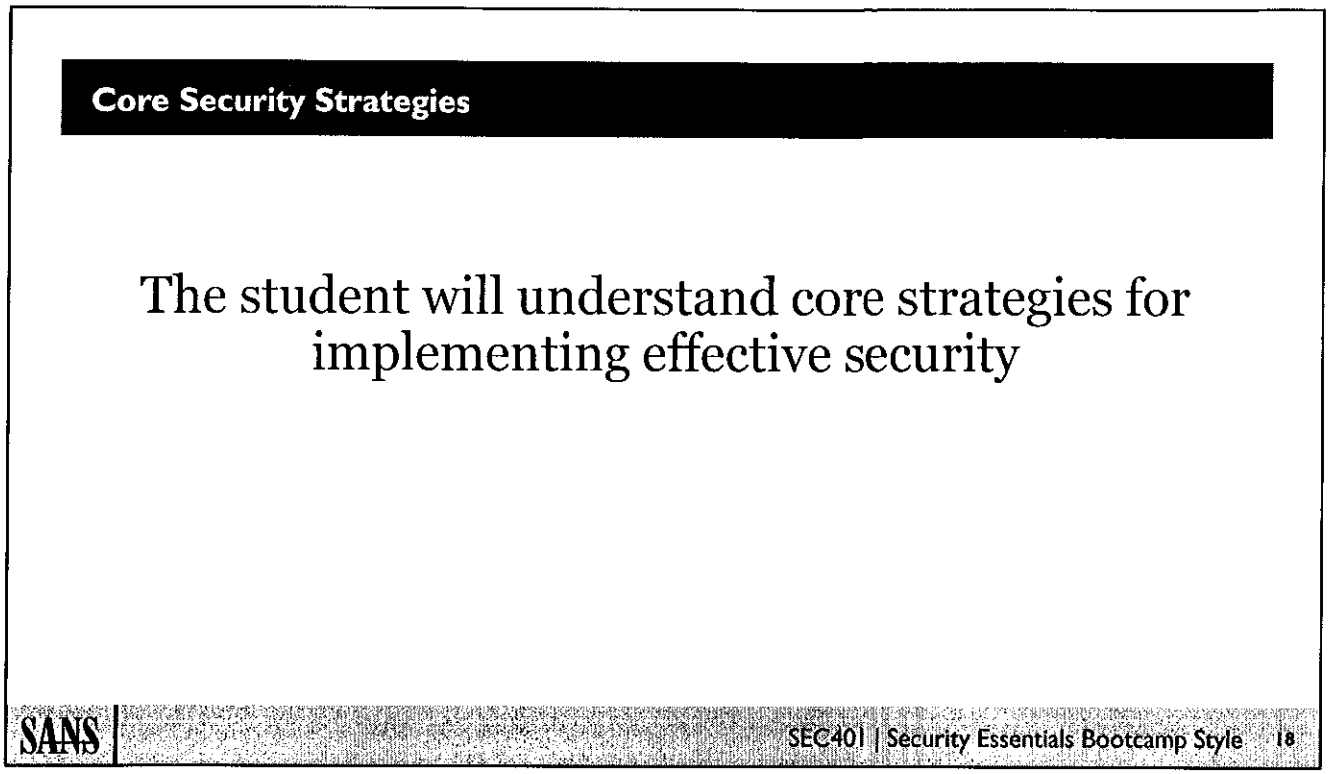


This slide shows another way to think of the defense-in-depth concept. At the center of the diagram is your information. However, the center can be anything you value, or the answer to the question, "What are you trying to protect?" Around that center, you build successive layers of protection. In the diagram, the protection layers are shown as blue rings. In this example, your information is protected by your application. The application is protected by the security of the host it resides on, and so on. To successfully get your information, an attacker would have to penetrate through your network, your host, your application, and finally your information protection layers.

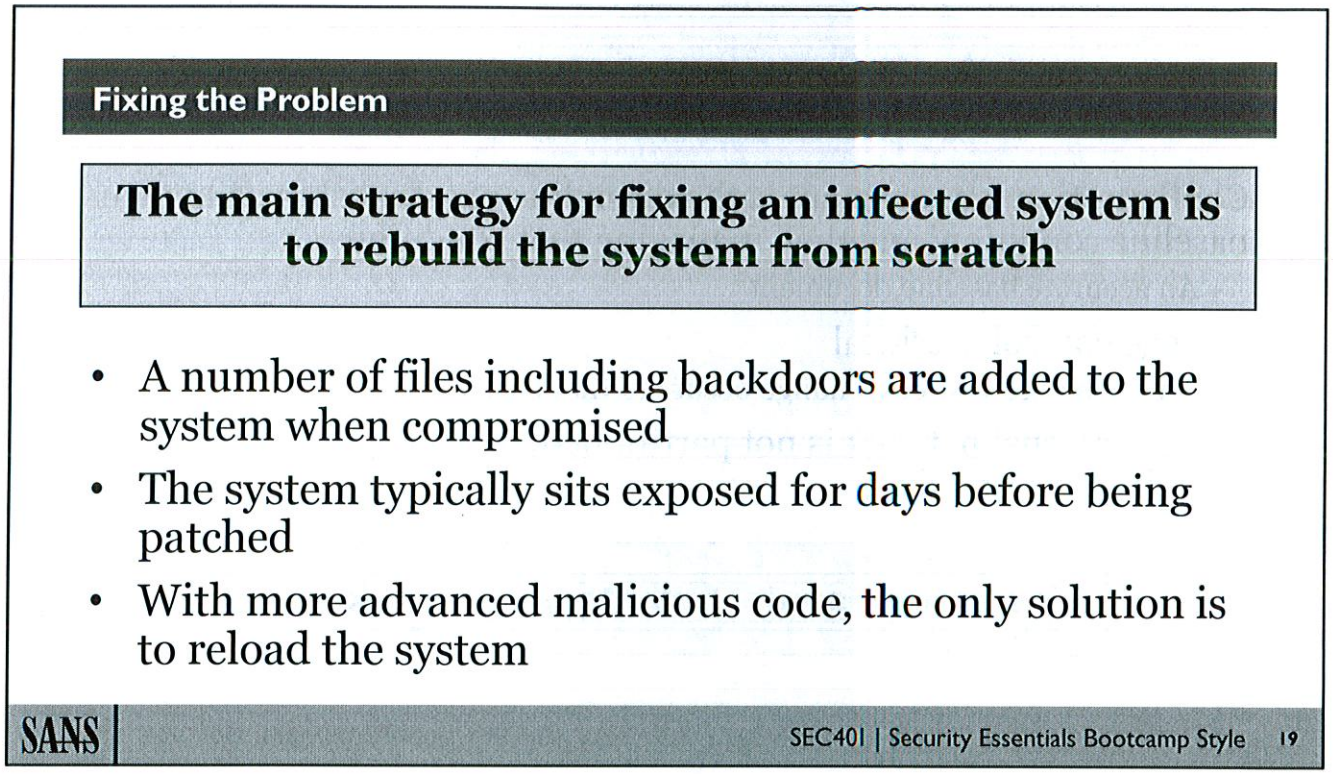
Information-centric defense starts with an awareness of the value of information within an organization. Identify the most valuable information and implement controls to prevent non-authorized employees from accessing it. A good starting point is to identify your organization's intellectual property, restrict it to a single section of the network, assign a single group of system administrators to it, mark the data, and thoroughly check for this level of data leaving your network.



Vector-oriented defense-in-depth involves identifying various vectors by which threats can become manifested and providing security mechanisms to shut down the vector—for example, disabling USB thumb drives. In an ideal case, you want to remove the vulnerability so the attack has no chance of success. However, in many cases, the vulnerability cannot be removed and there is an active threat. When you have a high risk and you cannot remove the vulnerability, you would mitigate it by removing the vector or avenue the threat would have to use to compromise your system.



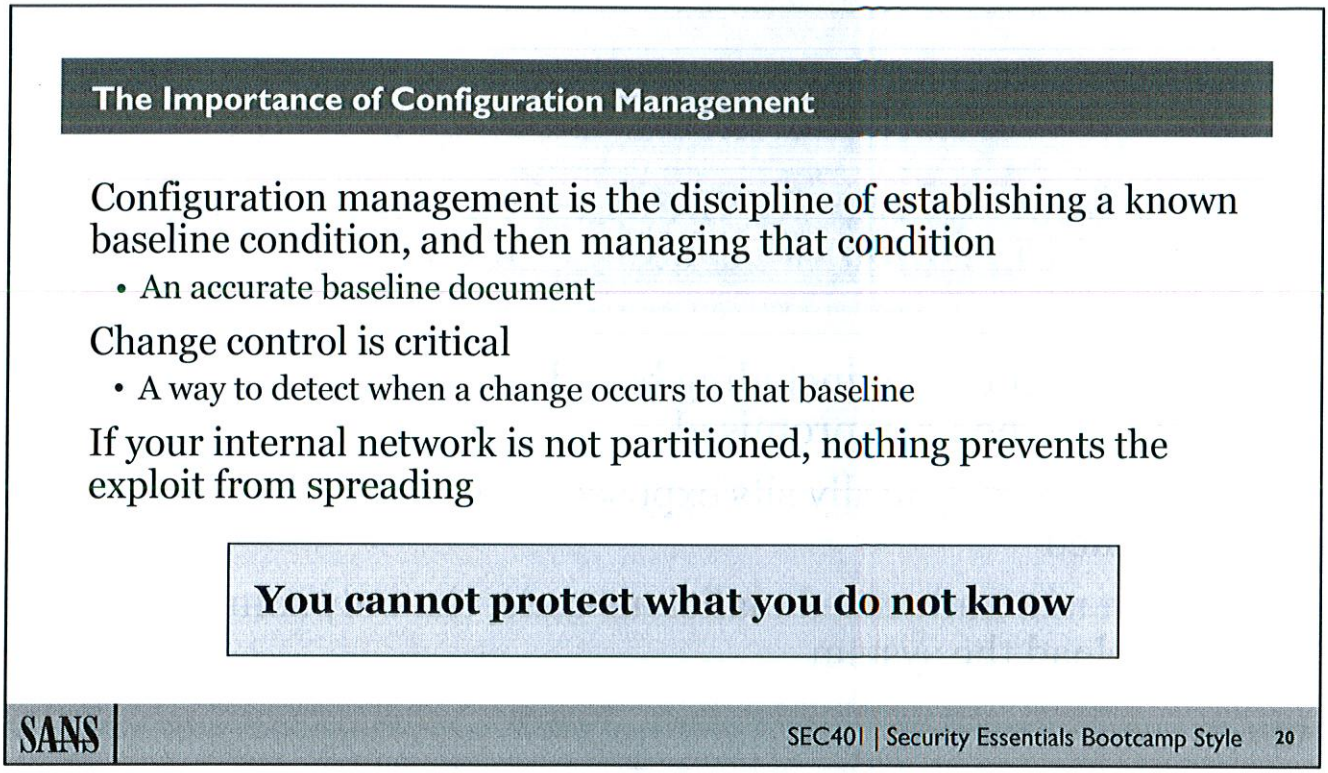
This page intentionally left blank.



After a system has been infected, we are left with a *giant integrity problem*. A number of files, including backdoors, are added to the system. The system typically sits exposed for days before being patched.

Most attacks are so successful because of the prevalence of undefended perimeters, operating systems left unchanged and certainly unpatched since their default installation, and one application automatically installing another. These were failures to practice defense-in-depth against known "vulnerabilities."

The point isn't how the exploit worked, but why it was successful; and, more importantly, what measures could have lessened its impact or stopped it altogether. How might this have been different if defense-in-depth were being practiced? Systems could have been properly configured and kept patched, and services could have been separated onto different systems. A baseline could have been established so as to detect any alterations, instead of being blind to what might have transpired through the backdoor. Are you taking these steps today?



The primary attacker strategy is to scan, looking for a vulnerable system, and then establish a beachhead or foothold by compromising that system. Then the vulnerable host that got compromised can be used to attack other systems, either in the same facility or in other organizations. This is one reason for saying, "risk assumed by one is shared by all."

In the early stages of protecting a site, a perimeter defense, such as a firewall, is about the only reasonable thing you can do. Although chokepoint defenses such as firewalls can yield some protection to internal systems, they can be circumvented in a number of ways, so the organization turns its focus to identifying and fixing vulnerabilities—what we call *hardening* systems. It takes a lot of energy to get to a known, reasonable configuration.

Configuration management is the discipline of establishing a known baseline condition, and then managing that condition. Now, of course, change is inevitable, and change generally is thought of in two major categories: repairs and improvements. Although vulnerabilities might occur while fixing something, they are far, far more likely to occur when deploying something new. We can label adding software, upgrades, new features, and new systems as "new construction."

Before you can do new construction, you need a building permit, and part of the building permit process is a design review and an inspection. The building permit process gives the organization an opportunity to ensure that the new construction introduces no new vulnerabilities into an organization. Of course, it's not foolproof, but it sure is better than not doing anything at all. And it has a lot of benefits! Perhaps the most significant is that the earlier in the development lifecycle you identify a problem, the cheaper it is to fix it. The main question is this: Can you detect that new construction is taking place (regardless of whether or not a building permit has been issued)? Said differently, can you detect a change to the network (or computer) infrastructure if no one has indicated that the change is (or will be) taking place? If you can't detect the change, how can you inspect it? If you can't inspect it, how can you assess and manage the risk? You can't. To manage your configuration, you need two things:

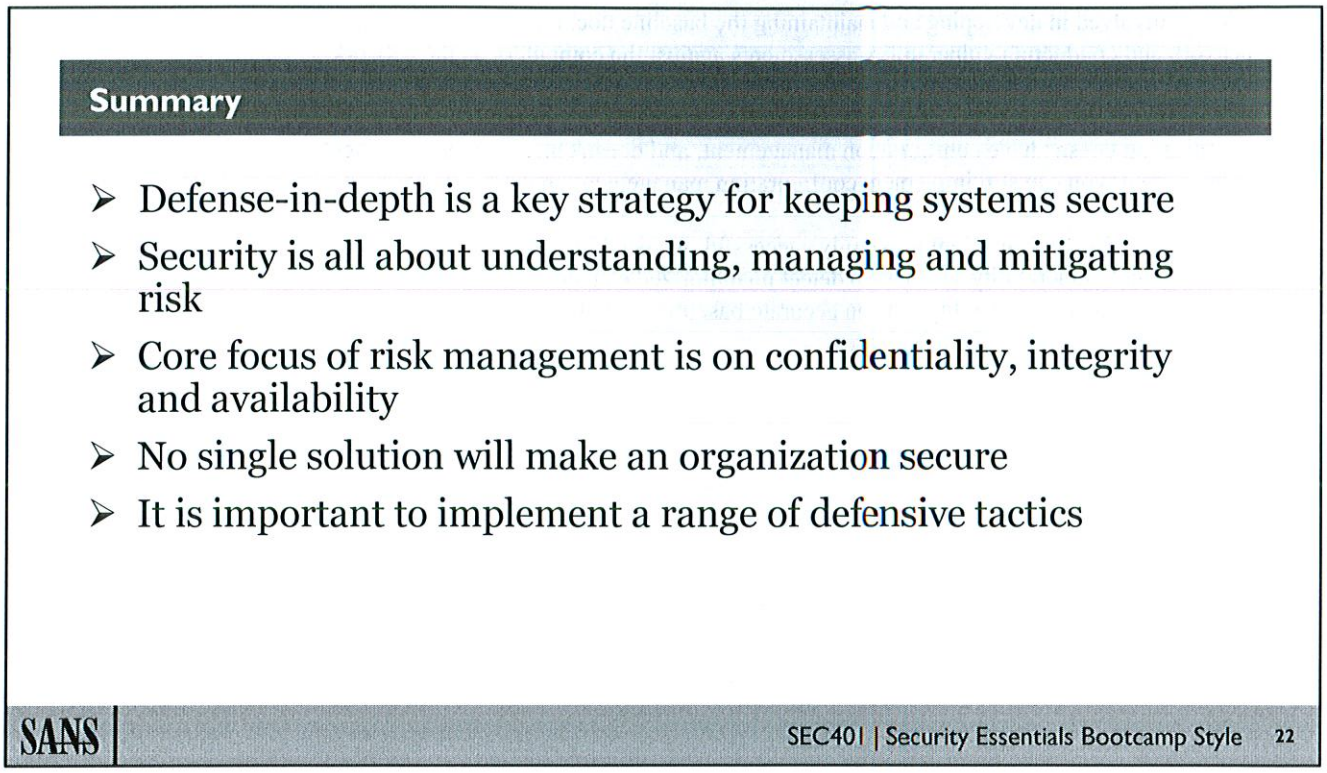
- An accurate baseline document
- A way to detect when a change occurs to that baseline

If either is missing, you might have a tough time managing the configuration.

What is involved in developing and maintaining the baseline document? Typical components include mapping the network and conducting vulnerability assessments against the computers on the network.

All improvement starts with one person willing to exert the energy needed to make a difference. If your organization doesn't have configuration management, and doesn't plan to ever implement configuration management, you can still implement configuration management on the things for which you are responsible.

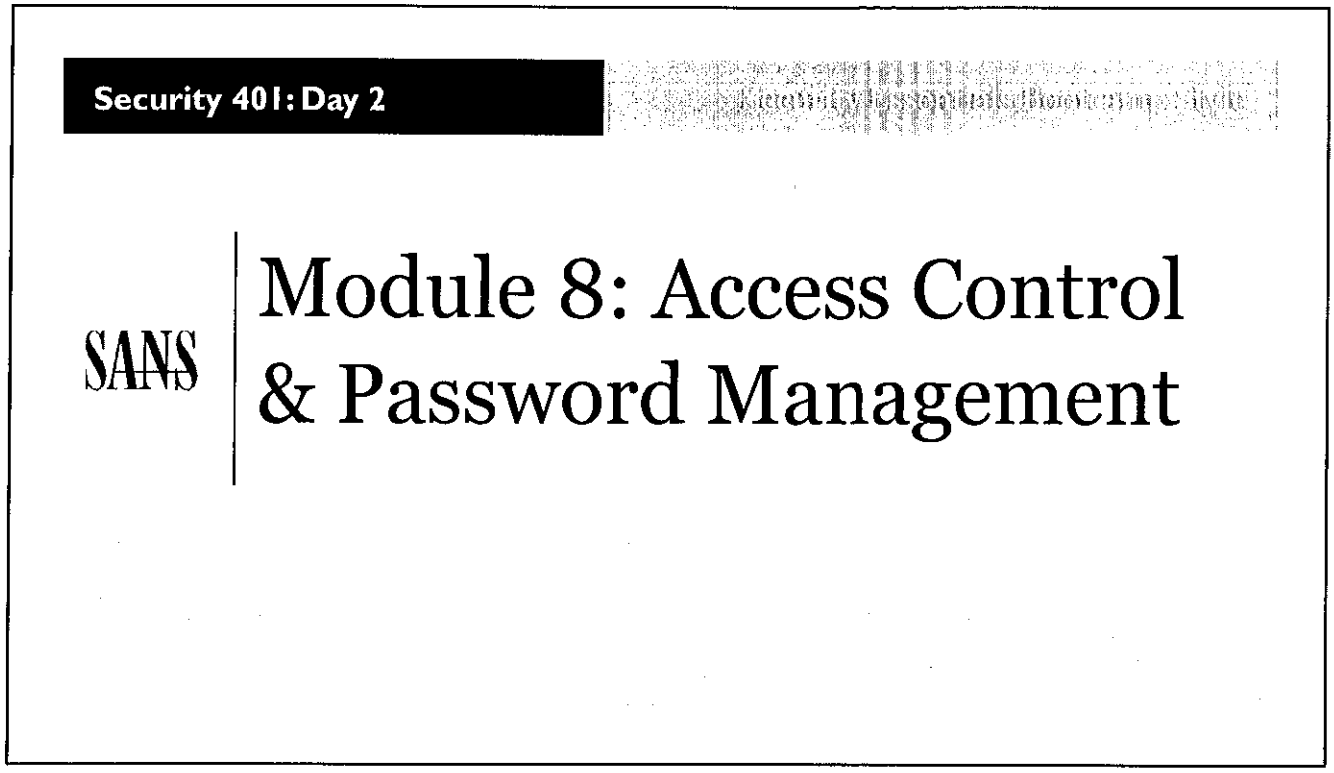
For configuration management to be truly successful, we need instrumentation, such as system scanners, network mapping, and vulnerability scanners to detect unauthorized change—and we need to use those tools on a scheduled basis. Only a facility with an accurate baseline is likely to practice anomaly detection to find attacks for which there are no known signatures.



When it comes to security, it is all about having an integrated approach—no single measure will keep you secure. Defense-in-depth is based on the premise that any single method of defense can be bypassed so it is critical to have several different components working in synergy. A key motto to always remember is that prevention is ideal but detection is a must.

While defense-in-depth is important, ultimately, security comes down to managing and mitigating risk. Before you do anything in the name of security, you should always ask 3 questions:

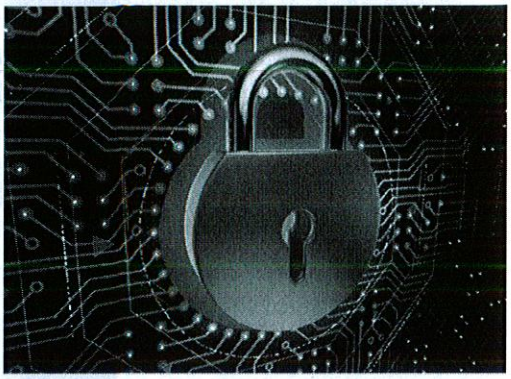
1. What is the risk?
2. Is this the highest priority risk?
3. Is this the most cost-effective way of reducing the risk?



This page intentionally left blank.

Objectives

- Access control
  - Data classification
  - Managing access
  - Separation of duties
- Password management
  - Password management technologies
  - How password assessment works



This section discusses the principles of access control. Access control models vary in their approaches to security, and we explore the underlying principles, strengths, and weaknesses of some. A brief discussion regarding authentication and authorization protocols and control will be included.

We also spend considerable time discussing the most common type of access control: the password. We delve into password files, storage, and protection.

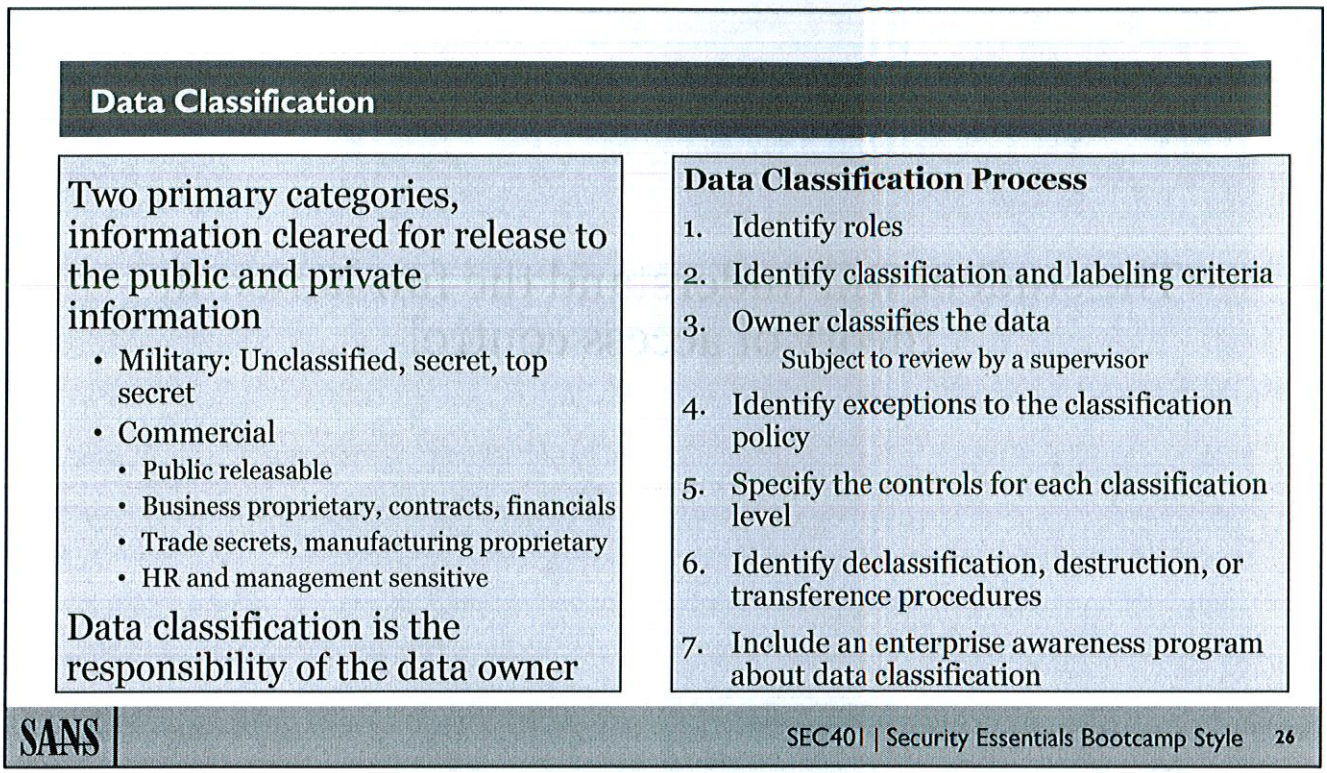
References

1. Role Based Access Control Docker Tutorial - http://www.infosiftr.com/2016/03/20/role-based-access-control-docker-tutorial/

Access Control Theory

The student will understand the fundamental theory of access control

This page intentionally left blank.



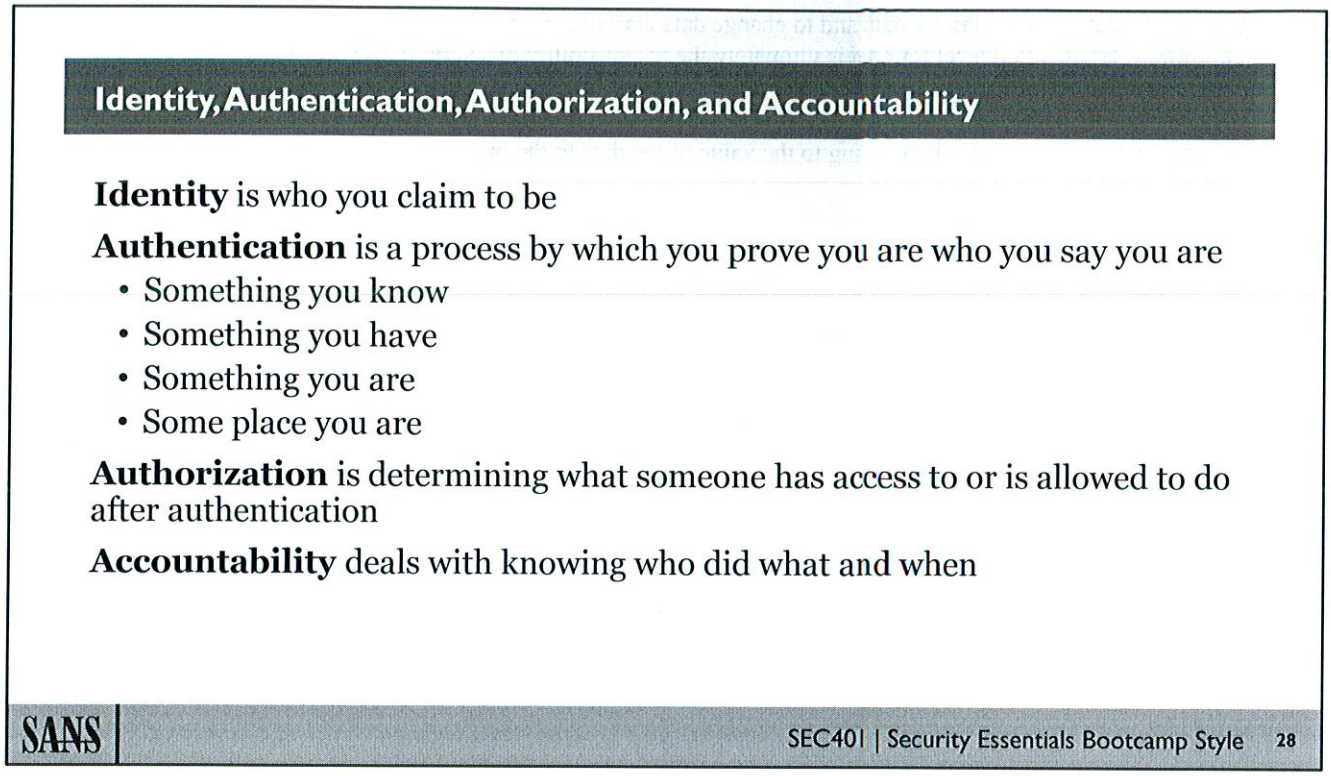
There is information that needs little protection from a confidentiality perspective—information that is considered cleared for public release. All other information needs to be protected, but at what level? The reality is that no organization has sufficient resources to protect all information with the rigor that the most sensitive information requires. Consequently, organizations often classify their data into differing levels so that appropriate protections can be applied based on the sensitivity of the information and on the potential impact of loss. The loss might be in terms of confidentiality (what we usually think of regarding government or corporate secrets), but also could be in terms of integrity or availability.

Governments, and their militaries, started the phenomenon of labeling data in order to apply higher levels of protection to sensitive data that could harm their country's national security if it were indiscriminately communicated. Subsequently, this classification practice has become commonplace in the corporate world, as well. A quick listing of common government classification levels follows:

- **Top Secret:** The highest levels of protection are given to this data; it is critical to protect.
- **Secret:** This data is important and its release could harm national security.
- **Confidential:** This data is important and could be detrimental to national security if released.
- **Unclassified:** Data owners prefer to keep this information from being released, but the nation would not be harmed if it were.

Generally, the best strategy for classifying data is to use a few clearly delineated categories and train your personnel in distinctive category use. The basic categories of sensitive information are business proprietary, the information about the cost of procuring, profit margins, contact lists, and others. A second category is the information about how products are created. This is often the result of research and development and includes know-how, trade secrets, and an understanding of what will not work and why. A third category is Human Resource and Management proprietary information, which is sensitive because so much personal information is contained in this category. You only need a different category when you have a significant quantity of information that requires significantly different protection.

Who has the authority to classify data and to change data classifications in your organization? Setting the appropriate classification level for data is ultimately the responsibility of senior management. The IT security professional can assist management in making these decisions by using risk assessment techniques to quantify the value of the data and the impact of threats to the confidentiality, integrity, and availability of the data. Access to the data can then be controlled according to the value of the data to the organization.



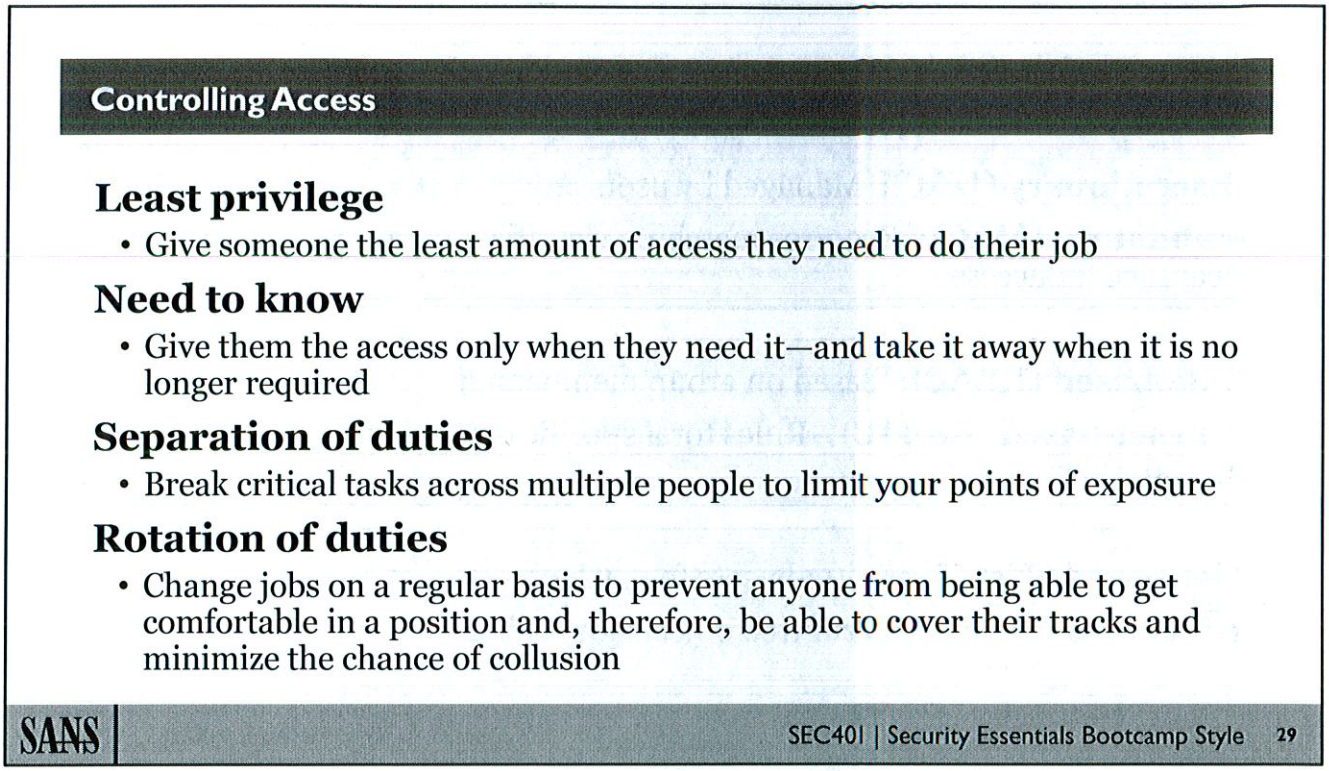
Let's briefly look at access controls, emphasizing the importance of defense-in-depth. To protect critical assets, you have to be able to identify, verify, approve, and track who has access to a given piece of intellectual property (IP).

Identification is the process of claiming to be a certain person. Typing in a userID is a form of identification. The problem is anyone could claim they are a given entity, so how do you know that they are who they say they are. This is accomplished through authentication. *Authentication* is proving that you are who you say you are and is done in one of four ways:

- **Something you know:** By remembering a piece of information and presenting it, you can prove that you are who you say you are. The best example of something you know is a password.
- **Something you have:** By possessing something, you can prove that you are a given entity. Token-based schemes in which you carry a token that generates a new password is an example of something you have. If you have the token and can type in the number on the token screen, you can authenticate; otherwise, you cannot.
- **Something you are:** An alternative way to authenticate is by presenting a unique attribute tied to your physical make-up. This is often called *biometrics*. Hand scan, thumbprints, and retina scans are all examples of biometrics.
- **Some place you are:** Global positioning systems (GPSs) can also be used to authenticate that you are in a given geographic area. With sensitive information, you might want to only allow someone to open a document if he is within the walls of a certain building.

After you have been properly authenticated, you then have to determine what you are allowed or authorized to do on the system. *Authorization* should be based on a principle of least privilege, where an entity is given only the minimal access needed to do the job. After access is granted using the principle of least privilege, you want to make sure individuals are held accountable for their actions and you can trace back what occurred on a system through detailed auditing; this process is called *accountability*.

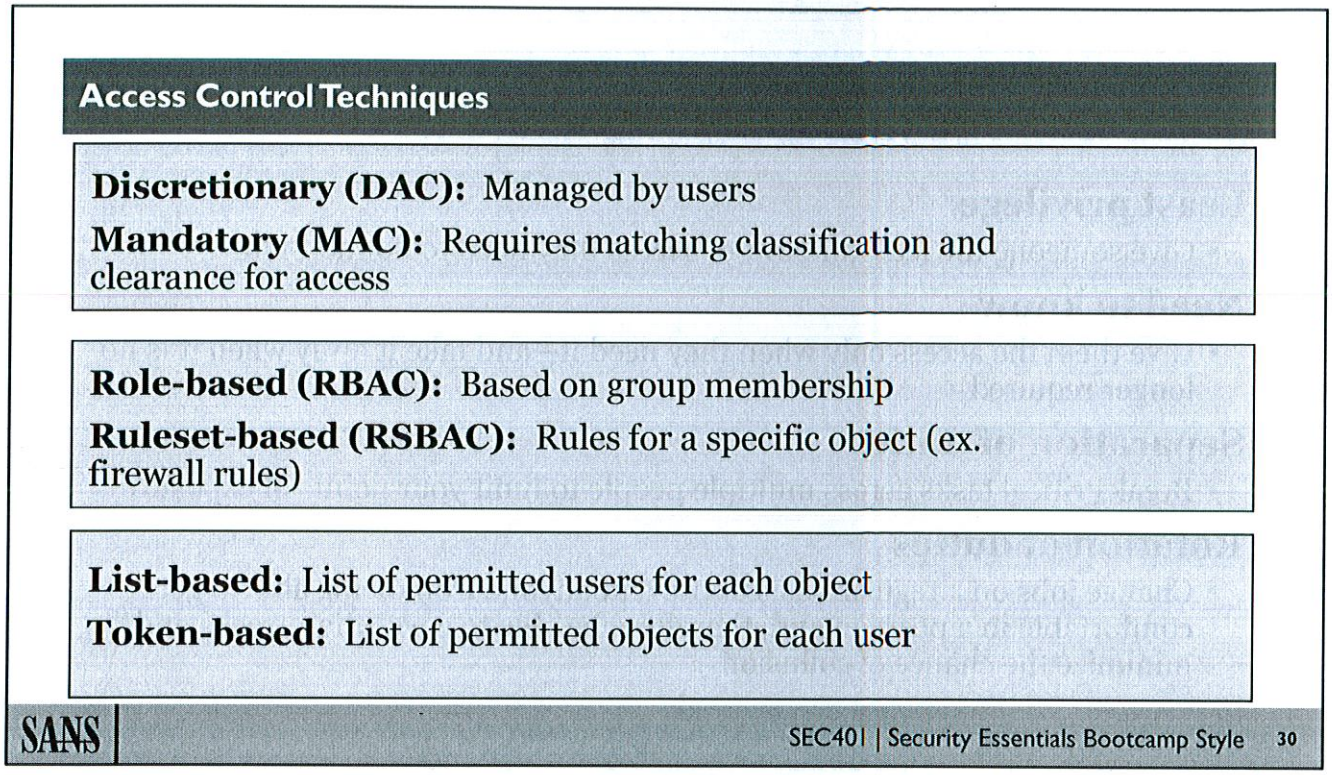
As you can see, all of these measures work together in synergy to properly protect critical assets.



Now that we have looked at the role that identification, authentication, authorization, and accountability play, we look at some principles associated with access control that you should utilize to make sure your security is as robust as it can possibly be.

In assigning access, you should give someone the least amount of access they need to do their job. However, this access should not be given all of the time; the access should be granted only when it is needed to perform a job function. For example, if I am the director of HR, the principle of least privilege would say that I need access to every employee's personnel file. On the other hand, the need to know principle would say that you should only give me that access when I have to review a file during a performance assessment—and not all of the time.

With least privilege, we are allowing people to do their job; however, we are only giving them the minimal access needed and no more. In some situations, this works. But, what happens in the case where the minimal access granted is still too great a risk and cannot be taken? In those cases, separation of duties needs to be implemented, where a given task is split between two individuals so no single individual by himself can make a decision. Separation of duties works; but the more people work together, the greater the chance they will collude in order to accomplish a crime. The more people work together, the more the power of separation of duties erodes away because people build trust. To minimize the chance of this occurring, rotation of duties needs to be performed. This is where people are rotated out of certain jobs at set intervals so the chance of two people colluding is minimized.



Access Control Techniques
Let's start out by briefly examining six common types of access control:

- Discretionary Access Control
- Mandatory Access Control

- Role-Based Access Control
- Ruleset-Based Access Control

- List-Based Access Control
- Token-Based Access Control

Discretionary Access Control (DAC) consists of something the user can manage, such as a username or password. For example, a user might choose to give a document password to someone without notifying the administrator. Windows peer-to-peer networks or standard Linux file permissions are good examples of DAC.

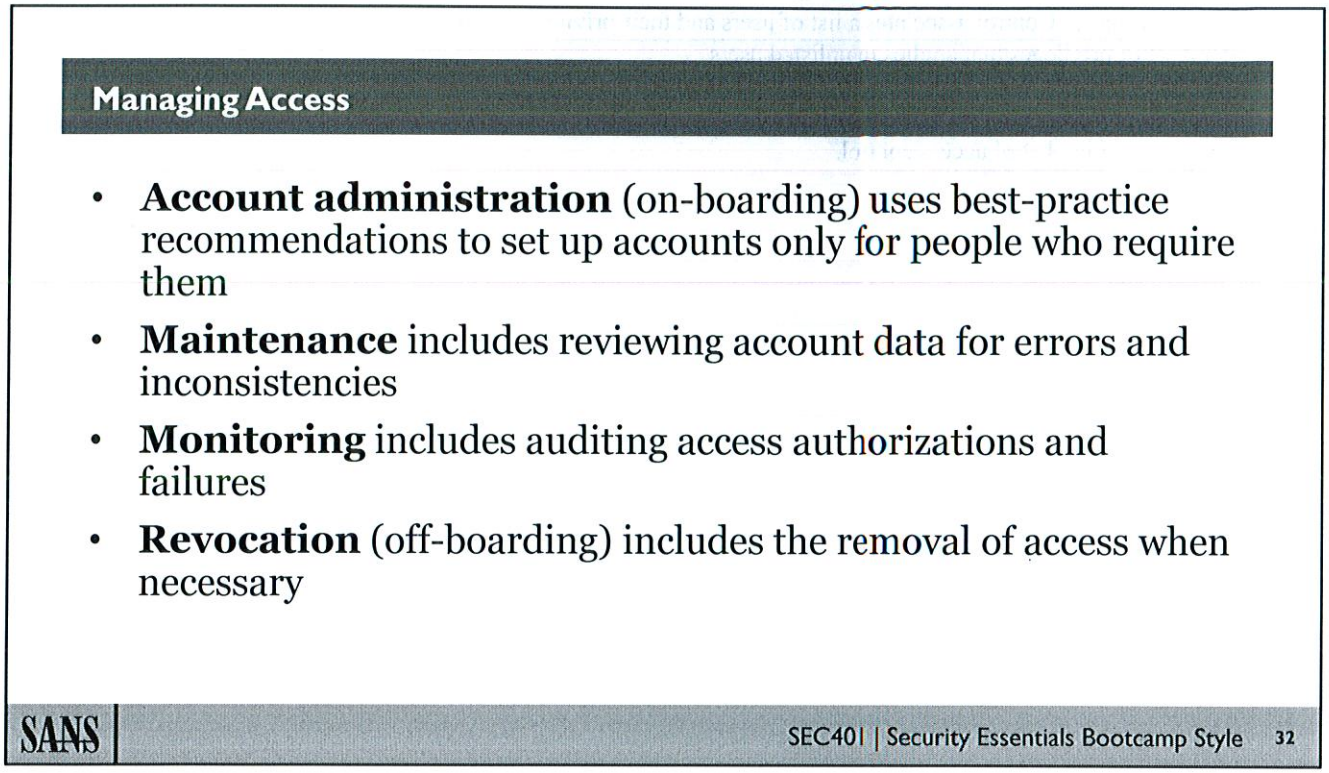
Mandatory Access Control (MAC) controls all access. Controls are set by the system and cannot be overwritten by the administrator. MAC requires a lot of work to maintain because all data has a classification and all users have a clearance. Users must have the appropriate clearance to access data classified a certain way. Users cannot give their clearance to another person.

Role-Based Access Control (RBAC) assigns users to roles or groups based on their organizational functions. Groups are assigned authorization to perform functions on certain data.

Ruleset-Based Access Control (RSBAC) targets actions based on rules for subjects (entities) operating on objects (data or other resources). RSBAC is implemented in a variety of software programs and operating systems.

List-Based Access Control associates a list of users and their privileges with each object. Each object has a default set of privileges that applies to unlisted users.

Token-Based Access Control associates a list of objects and their privileges (called capabilities) with each user, the opposite of list-based access control.



User accounts, data, and their relationships must be actively maintained, perhaps by an entire team of employees. This process, called "access management," consists of four tasks: account administration, maintenance, monitoring, and revocation.

Account administration is a set of best management practices. The administrator verifies the individual before providing access—this is the most important step in the process. This is also an opportunity to teach users not to distribute any access privilege they have (tokens, passwords, and so on).

Maintenance is the process of reviewing account data and spot-checking for inconsistencies or errors. Periodically, account management staff should review and update lists of users and authorizations. Review should take place automatically when employees transfer departments or locations or are assigned new or different duties.

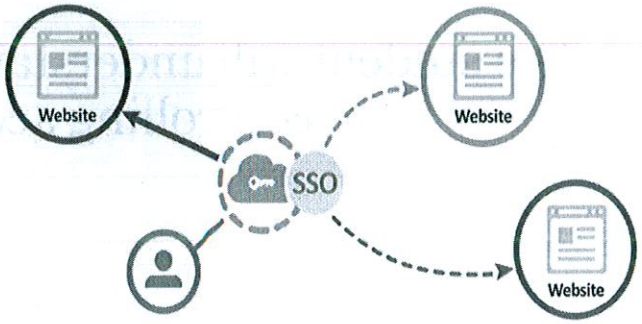
For accountability, authentications and authorizations should be *monitored.* System administrators should log both successful and failed attempts to log on to the system. Logging of the use of systems resources (files, programs, printers, and so on) should be enabled based on risk assessment of the value of those resources.

For instance, successful and unsuccessful access to the payroll database should be logged, but access to a public document store would not necessarily need to be logged.

Almost as important as account administration is *revocation.* Account management staff and system administrators should promptly revoke privileges when they are no longer needed, especially for users who have been fired.

Single Sign-On (SSO)

- Have to log on only once
- Credentials are carried with the user
- Simplifies user management
- Allows for centralized management
- Have to remember only one set of credentials
- Should be used with multi-factor authentication



The SSO technology allows a user to log on once with one set of credentials and use those credentials to access different resources.

There are different ways that Single Sign-On can be implemented. One of the older, and still common, ways is the use of scripts that will mimic the login process between different servers. It is easy to implement but has some very serious security implications, because you often have credentials stored in plain text files.

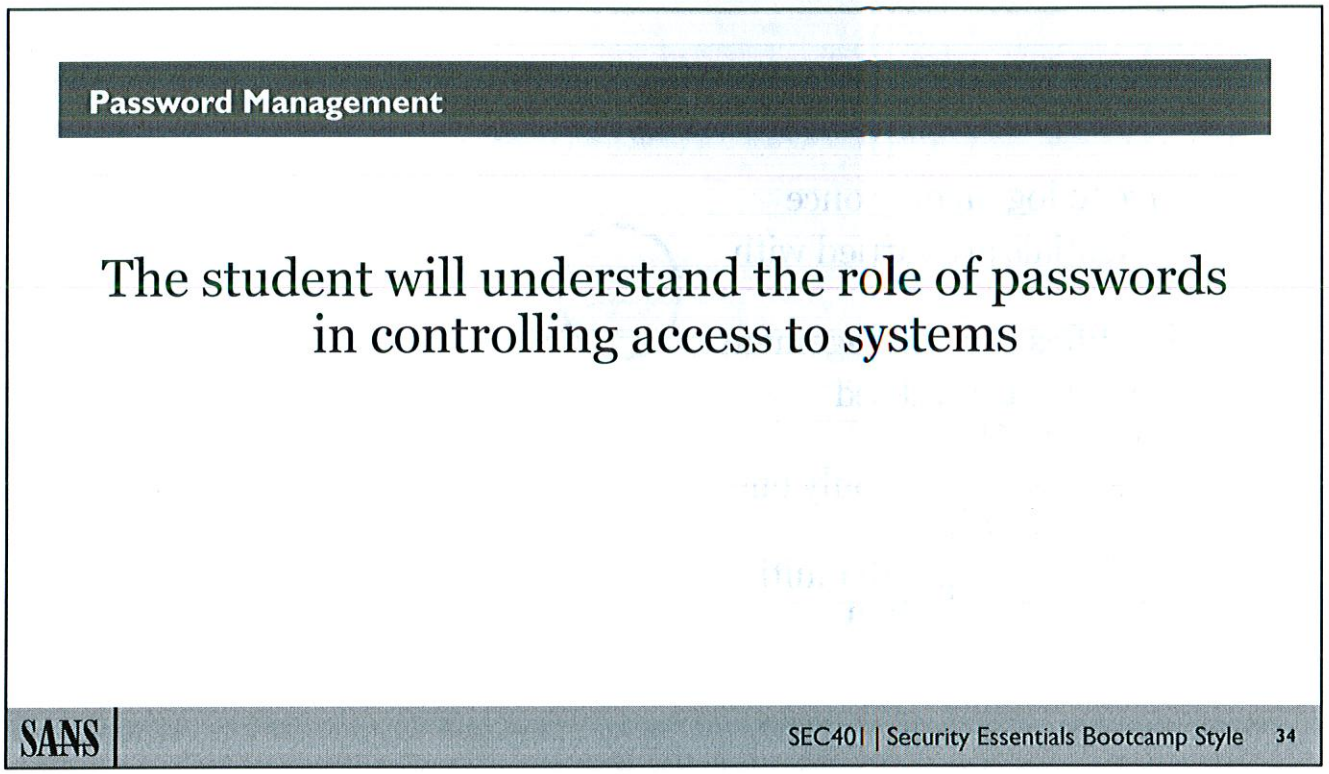
Another way that SSO is implemented is through the use of a central directory service, such as LDAP or Microsoft Active Directory. This allows the creation of a user account once on a single platform. From that single account, the user will be granted access to different platforms or services.

Kerberos can also offer the SSO through the use of tickets where credentials are stored. Not all operating systems are kerberized or can make use of Kerberos. It is often necessary to install a third-party software package, and there are some compatibility issues between the different versions of Kerberos.

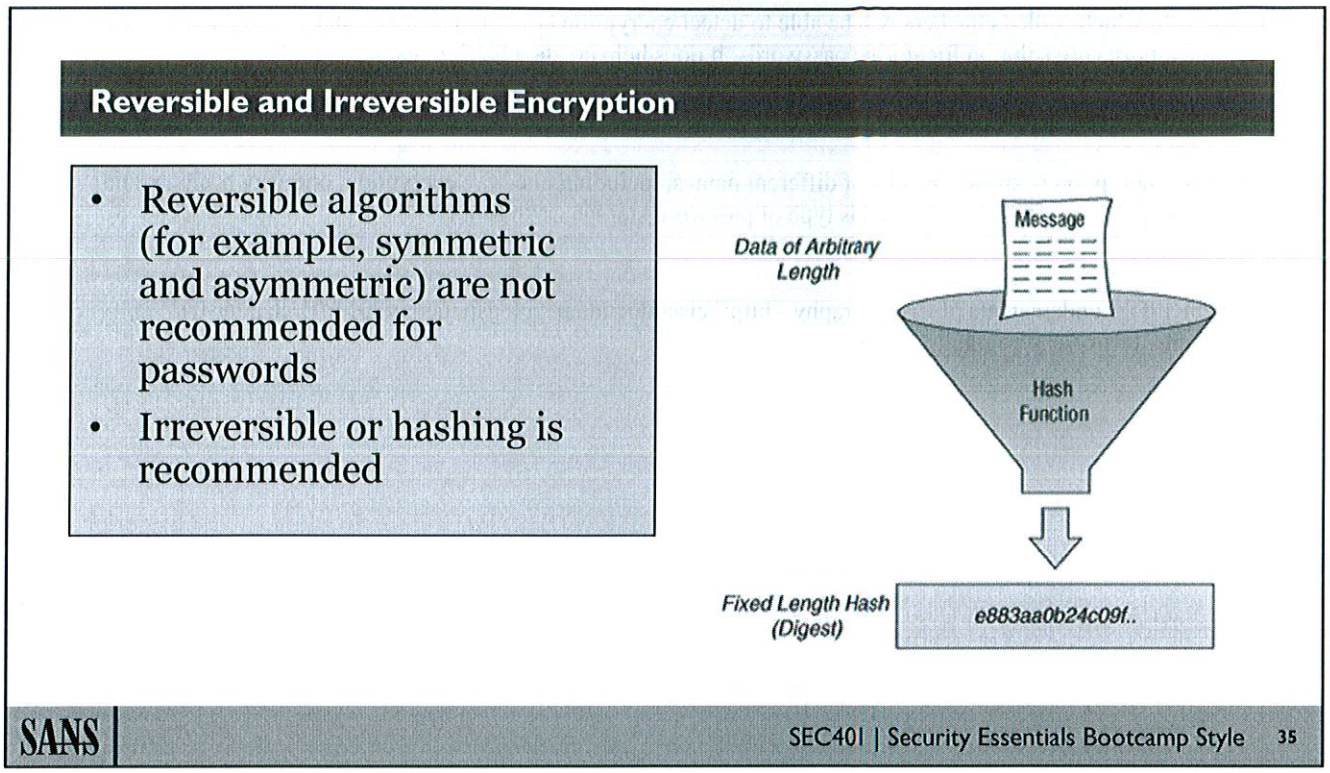
SSO can save you a great amount of administrative time, but will demand some initial investment in money and human resources. Ensure you select the type of technology that will properly support all of your platforms and legacy applications.

References

1. Single Sign On - https://www.oneall.com/services/single-sign-on/



This page intentionally left blank.



Reversible encryption is the kind of encryption you use when you need secure storage for confidential information. Because you want to be able to re-access your data, using irreversible encryption is not an option.

When reversible encryption is used during the authentication process, generally the system will follow these four steps:

1. Request a username and password from the user.
2. Decrypt the password stored with the username in the password database.
3. Match the decrypted password with the password supplied by the user.
4. Allow or deny access.

Reversible encryption can be secure for as long as you are able to keep attackers from discovering the key (or algorithm) that is used to encrypt the password. In complex, closed-source operating systems or applications, it can be very difficult for an attacker to discover where exactly in the system this encryption key has been stored. Therefore, searching through binary files is generally not the approach taken to discover the key.

In step 2 of the authentication process, you can see that the system accesses the encryption key in order to decrypt the password stored in the password database. By attaching special debugging software to the authentication process, an attacker can follow each step this process takes and look at all the variables used to complete the process, including the variable used to store the encryption key.

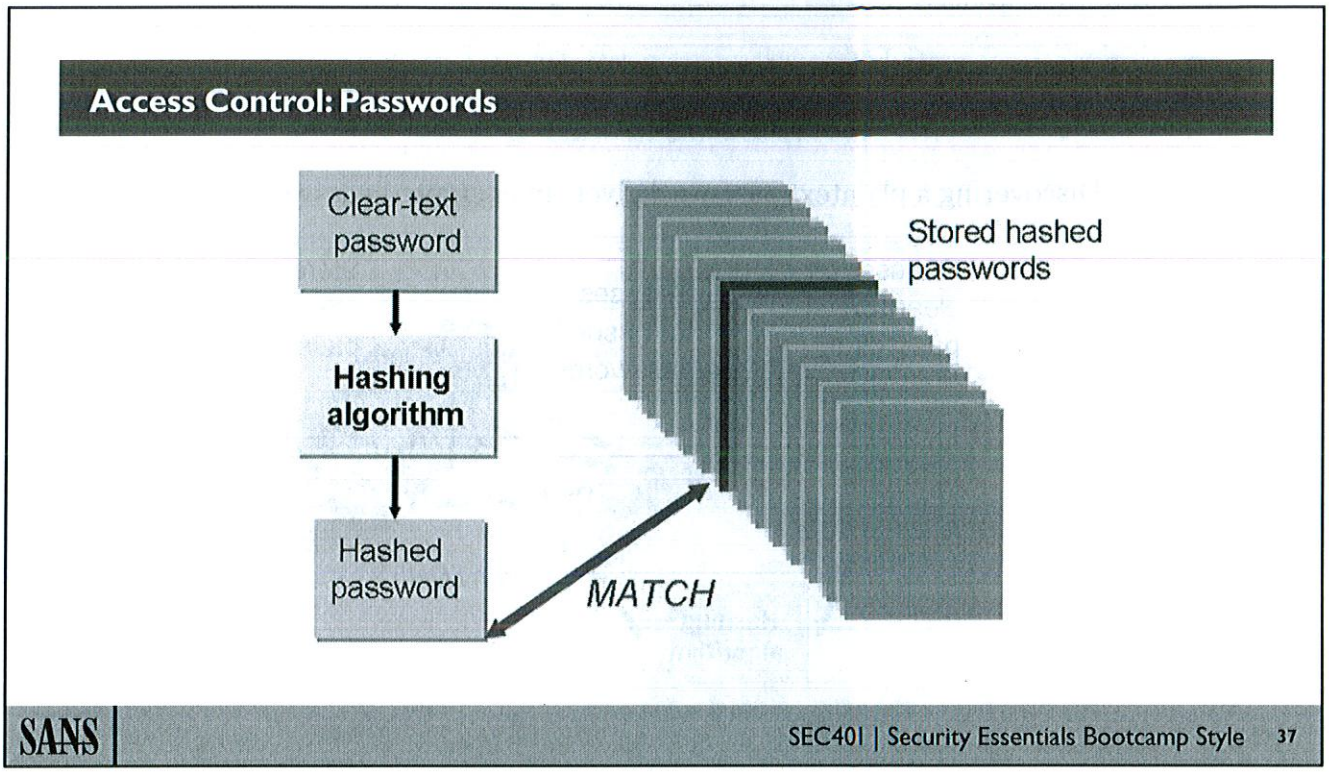
After this key or algorithm has been detected, it can generally be used to decrypt the entire user database. These steps are also referred to as "reverse engineering." They are not only popular for reversing encrypted passwords, but are also commonly used for detecting algorithms used to generate, for instance, software license keys.

The ease with which skilled attackers will be able to detect encryption keys or algorithms makes reversible encryption an insecure solution for storing passwords. It does help create a barrier against shoulder surfing passwords from printed configuration files on the desk of an administrator, but makes a poor solution when compared to irreversible encryption.

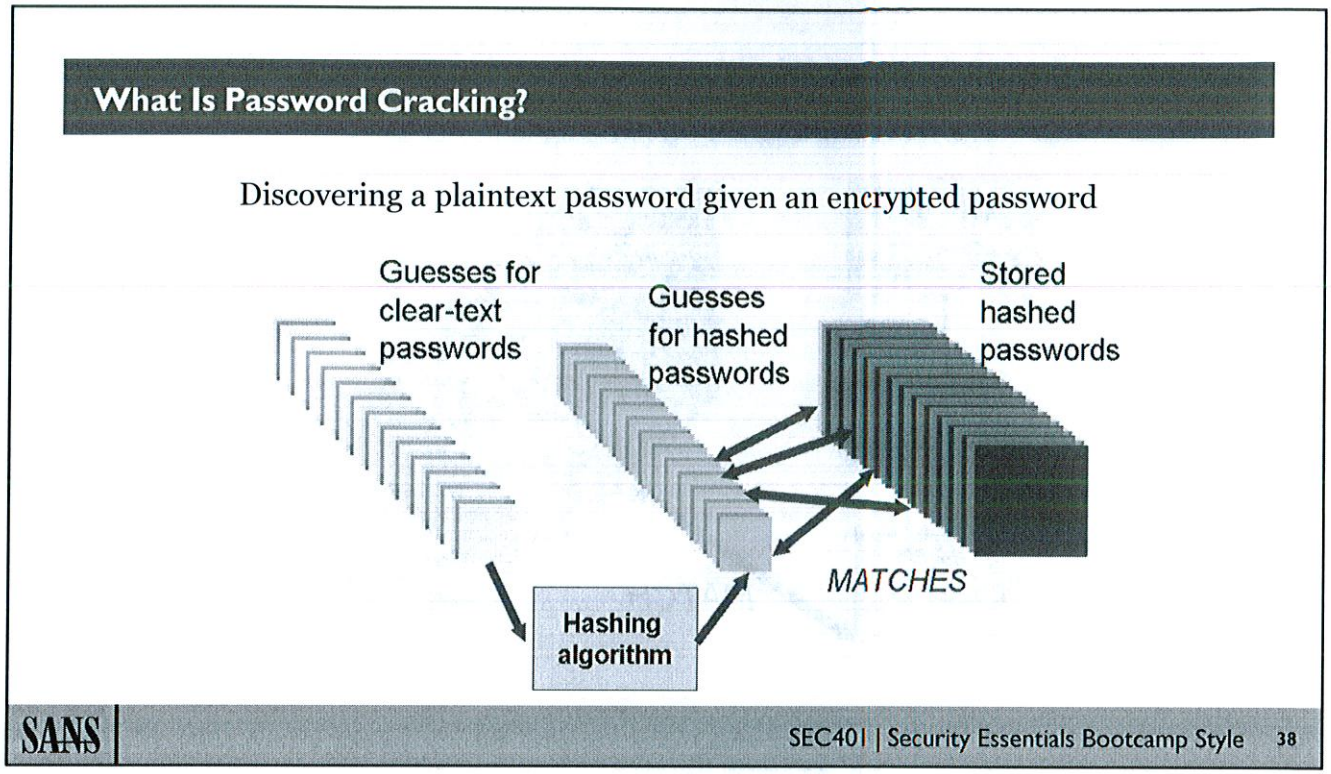
Irreversible encryption is known by a lot of different names, including one-way encryption, one-way hashing, and simple hashing. We continue to refer to this type of password storage as "hashing."

References

1. Chapter 04: Fundamentals of Cryptography - http://ciscodocuments.blogspot.com/2011/05/chapter-04-fundamentals-of-cryptography.html



This slide shows the process is which passwords are stored on the system and how a user is authenticated using hashes.



Password cracking is an offline process of attempting to guess passwords, given password file information. This section begins with a discussion of what password cracking is, why it is important, and what methods are available.

Let's back up for a moment and think about why passwords are so important. Often, passwords are the first line of defense against interactive attacks on a system. Because it is fairly easy for someone to figure out a user ID, the only thing protecting that user's account is her password. If an attacker can gather no helpful information to aid in the attack (such as password file contents or sniffed network traffic), he must resort to either creative or brute-force password guessing.

If an attacker can at least read the password file or obtain a copy, his chances of successfully obtaining an actual password increase significantly. Even if the attacker obtains only a lowly user-level password, it's fair to assume he will log in to the target system as the user and then attempt to break into the root account via local vulnerabilities.

In many companies, passwords are more than just the first line of defense—they're the only security measure protecting servers and internal information. Because most user IDs consist of an employee's first initial and last name (or something similar), it's fairly easy to discover valid user IDs for individuals at a company. Then the only other piece of information needed to gain access is a user password. So passwords must be protected, and they must be very difficult to guess.

Unauthorized disclosure, unauthorized modification, and unauthorized removal are all threats to password integrity. If users disclose their passwords (intentionally or not) by writing them down or sharing them with other people, malicious parties might obtain them. It's even worse if attackers can modify the password data directly because they could change passwords without needing to know the originals. Of course, changing a password is risky for an attacker—users tend to get suspicious when their passwords suddenly stop working.

Operating systems protect passwords by using strong cryptography to hide the original content. Even if the encrypted password is revealed, it is difficult to determine the original.

John the Ripper

Password file format:

maggie:1A2fa6G8h$P/mI.DjciPfud6hnKdm6F.:501:501::/home/maggie
:/bin/bash

John takes no time at all to figure out that Maggie's password and username are the same and continues working on the others

```
$ ./john -format:MD5 test
Loaded 3 passwords with 3 different salts (FreeBSD MD5 [32/32])
maggie           (maggie)
guesses: 1  time: 0:02:31:24 (3)  c/s: 3733  trying: bingsolo
guesses: 1  time: 0:02:31:30 (3)  c/s: 3734  trying: culciouw
guesses: 1  time: 0:02:31:33 (3)  c/s: 3733  trying: binangly
guesses: 1  time: 0:02:31:34 (3)  c/s: 3733  trying: abachkai
guesses: 1  time: 0:02:41:38 (3)  c/s: 3733  trying: mcinbldt
guesses: 1  time: 0:13:16:32 (3)  c/s: 3812  trying: stho5530
guesses: 1  time: 0:14:59:39 (3)  c/s: 3797  trying: 15696*
guesses: 1  time: 0:14:59:40 (3)  c/s: 3797  trying: jolve!
```


John the Ripper is powerful and fast. John combines several modes into one program and is fully configurable.

John has built-in support for many types of passwords. If your cipher algorithm isn't supported by John, you can extend John to support it, thanks to its modular design. John itself takes advantage of this architecture to bring you optimized modules for different architectures, some of which use assembly routines for greater speed.

John has several modes, each of which goes about cracking passwords in a different way. To use any of the supported algorithms, you'll have three modes from which to choose for your audit. The mode you choose depends on the strength of the passwords themselves and how much time and processor power you have. A fourth mode, external mode, lets you use external modules for algorithms that aren't directly supported. These modes can be selected on the command line or by editing the john.ini file.

The four modes are described in detail next.

Wordlist Mode
Wordlist is the simplest of the crack modes offered by John the Ripper. John does not do any type of sorting of the wordlists, allowing you to further optimize its performance by putting the most common words at the beginning of the list. Like Crack, John will perform substitutions and other transformations on each word if configured to do so in the john.ini file.

Single Crack Mode
Single crack mode uses the username and GECOS information to guess passwords. John also adds previously guessed passwords to the list, helping to detect users with the same password on several accounts. As with wordlist mode, John will transform each guess as configured in john.ini. Because it doesn't take long to try this short list of guesses, single crack mode is much faster than wordlist mode and should be used first.

Incremental Mode
Of the three standard modes, incremental is the most powerful, but also the most time-consuming. It tries every combination of letters, numbers, case, and special characters. Because it tries every combination of every possible length, it runs indefinitely.

External Mode
Though John is powerful enough for most purposes on its own, it can be extended to include whatever custom routines you can define. These routines are developed using a subset of C compiled at runtime.

John has no problem handling the MD5 format. Notice in the following example, Maggie's encrypted password starts with a "1" indicating that it has been hashed with MD5. The portion of the string following the "1", but before the "$" (A2fa6G8h), is Maggie's salt value. The rest is the encrypted password itself (in this case, "maggie"). John takes virtually no time to crack this simple password (in this case, the clear text is maggie).

maggie:1A2fa6G8h$P/mI.DjciPfud6hnKdm6F.:501:501::/home/maggie:/bin/bash

Suppose this entry and two others are in the file test. When we run:

john -format:MD5 -w:password.lst test

...we see that John takes no time at all to figure out that Maggie's password and username are the same and continues working on the others:

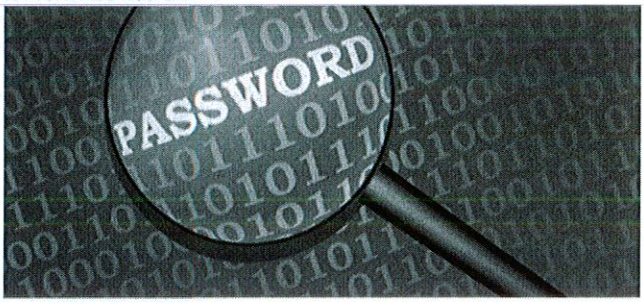
```
Loaded 3 passwords with 3 different salts (FreeBSD MD5 [32/32])
maggie          (maggie)
guesses: 1  time: 0:02:31:24 (3)  c/s: 3733  trying: bingsolo
guesses: 1  time: 0:02:31:30 (3)  c/s: 3734  trying: culciouw
guesses: 1  time: 0:02:31:33 (3)  c/s: 3733  trying: binangly
guesses: 1  time: 0:02:31:34 (3)  c/s: 3733  trying: abachkal
guesses: 1  time: 0:02:41:38 (3)  c/s: 3733  trying: mcinbldt
guesses: 1  time: 0:13:16:32 (3)  c/s: 3812  trying: stho5530
guesses: 1  time: 0:14:59:39 (3)  c/s: 3797  trying: l5696*
guesses: 1  time: 0:14:59:40 (3)  c/s: 3797  trying: jolve!
```

As John makes its way through the wordlist, it keeps you apprised of its progress. After each word it tries (with all the associated substitutions and combinations configured in john.ini), it tells you how long it took and the number of combinations it tried per second.

More information on attacks against DES and salt functions with an interesting chart depicting the number of encryptions per second with the advancement of processors is available at http://www.usenix.org/events/usenix99/provos/provos_html/node13.html.

What Determines the Strength of a Password Hash?

- Quality of algorithm
- Key length (hash length)
- CPU cycles
- Character set support
- Password length



SANS SEC401 | Security Essentials Bootcamp Style 41

The strength of a hash used for password storage primarily depends on five factors:

1. Quality of Algorithm
A hashing method is only as good as its algorithm. Even an infinite key length will not protect your password if the algorithm used to produce the key has been proven to be an insufficient protection mechanism. In general, however, it is very difficult for cryptographers to define the quality of an algorithm; many algorithms are believed to be reasonably secure until proven otherwise. It is recommended to use an algorithm that has been available for a long time and well tested.

2. Key Length
A larger hashing key means that there are more possible keys to uniquely identify the input of the hash function and, therefore, a smaller chance of collisions. Although this is considered the most important feature of hashes with respect to data integrity, the limited length of passwords employed by users makes key length only a minor contributor when considering the strength of password hashes. Related to password hashes, the key length of a hash could be considered a real issue if it is not able to uniquely identify all possible passwords within the minimum password length and complexity requirements that have been determined in the company's password policy.

3. CPU Cycles Used to Calculate the Hash
Because hashes cannot be reversed, an attacker must hash possible passwords and compare these hashes with the obtained password hash of an existing user in order to determine the credentials that can be used to log in. The number of password tries per second an attacker is able to launch against a hash depends on the number of CPU cycles the hashing function uses to compute the hash of a password. The more tries per second, the sooner all possible combinations can be tested.

4. Character Set Support
The strength of a chosen password depends heavily on the length of the chosen password and the character set used when choosing the password. A 7-character password made out of a maximum of 26 characters results

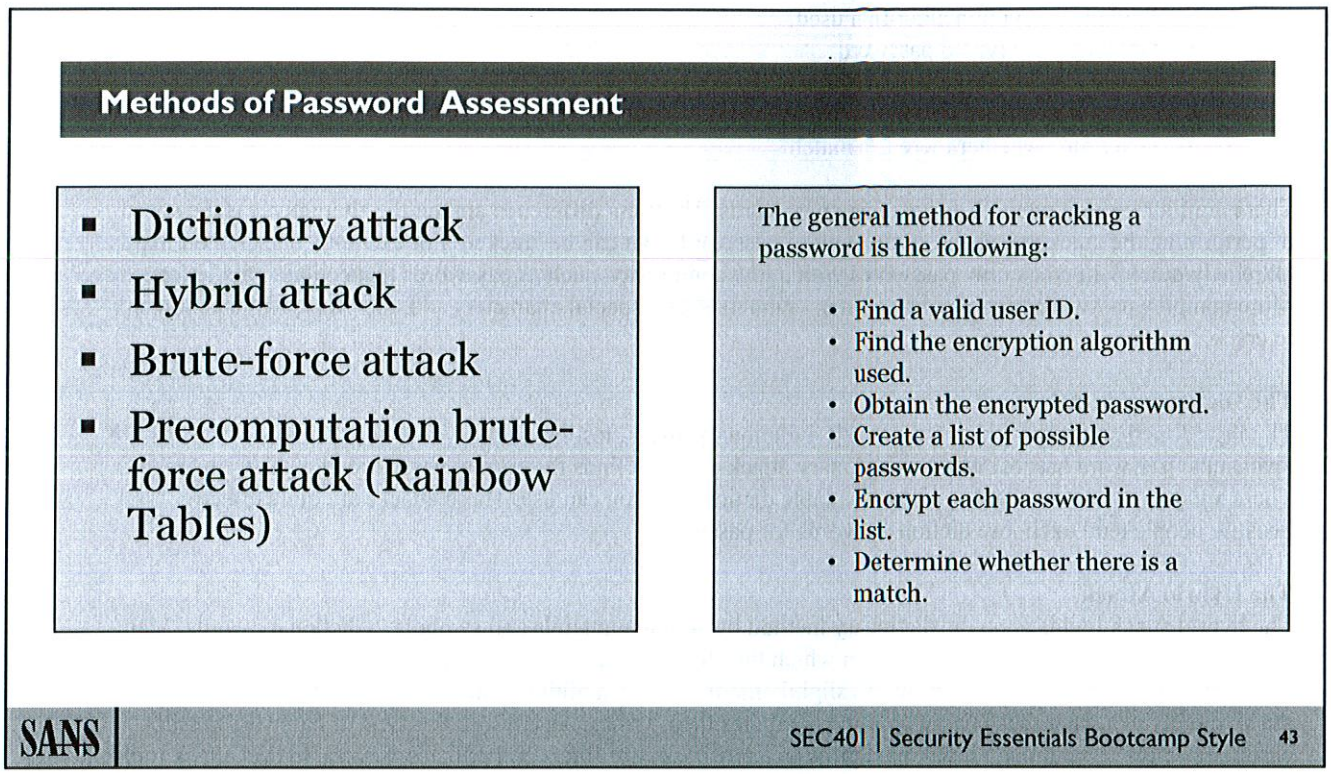
in 8 billion possible passwords (which is generally easy to brute-force within hours). When a 52-character set can be used, it will take 128 times longer to brute-force.

5. Password Length

Because most users do not like to be forced into using characters that need <ALT> keys to access, the effective password character space is generally limited to alphanumeric characters and punctuation marks. This limited character space makes the length of a password important. Through the length of the password, the possible password combinations can be increased and brute-forcing can be made more difficult.

References

1. Crypted Security - http://cryptedsecurity.com/unix-password-cracker/



Most of us are aware that we shouldn't use passwords that are too short (because all the possible character combinations could easily be tried) or write passwords on sticky notes and put them under our keyboards. Beyond this basic understanding, however, can we quantify what makes a password difficult to guess when a computer is used as the guessing engine? The answer is: It depends on the particular method used to protect the sensitive information.

Computers use one-way hashing algorithms to encrypt passwords for storage. A one-way hash is mathematically easy to compute in one direction (for encryption), but impossible to compute the other way, even for computers. This is important because someone who recovers a password file can't use the hashed values to reverse the one-way encryption function and recover the original passwords. But how, then, does the computer use the encrypted information to authenticate users?

The technique is simple. Even though hashing functions can't be reversed, they always can produce the same output given the same input. Thus, the computer stores only the hashed passwords (rather than original passwords) on disk. When a user attempts to authenticate to either the machine itself or the network, the computer applies the hash algorithm to the password the user has supplied for authentication. If the hash of the user-supplied password matches the hash stored on disk, the password is correct, and the user is successfully authenticated.

Password cracking is the process of trying to guess or determine plaintext passwords, given only encrypted passwords. The process does not actually break the encryption; it mimics the actions that would take place if a user tried passwords until guessing the right one. Each guess is hashed and compared to the stored value. When a match is found, the user is authenticated. The cracking operation is usually performed offline against a recovered password file.

The general method for cracking is the following:

- Find a valid user ID.

- Find the encryption algorithm used.
- Obtain the encrypted password.
- Create a list of possible passwords.
- Encrypt each password in the list.
- Determine whether there is a match.

There are four general methods for cracking passwords. The main difference among the alternatives is the speed of performing the crack versus the complexity of passwords that can be cracked. For example, one method that is extremely quick will crack only passwords with a low complexity, such as passwords that contain only letters. More complex passwords might contain letters, numerals, and special characters and, therefore, will take longer to crack.

The Dictionary Attack

The fastest method for cracking passwords is a dictionary attack, testing all the words in a dictionary or word file against the password hashes. When a dictionary attack program finds the correct password, it displays the result. There are many websites that have downloadable dictionaries you can use. These attacks are quite effective because people tend to choose dictionary words for passwords.

The Hybrid Attack

The hybrid attack builds upon the dictionary method by adding numerals and symbols to dictionary words. Many users choose passwords such as he1lo!! (in which the ells are replaced by ones) just to satisfy policies and filters. These passwords are just dictionary words slightly modified with additional numerals and symbols. The hybrid attack rapidly generates these passwords and computes their hashes. As a result, this type of password is still easily crackable, even though it will pass through many password filters and policies. Several hybrid attack tools have configurable rule sets that allow the attacker to specify combinations and permutations of dictionary words to try. These tools are surprisingly effective and easy to use.

The Brute-Force Attack

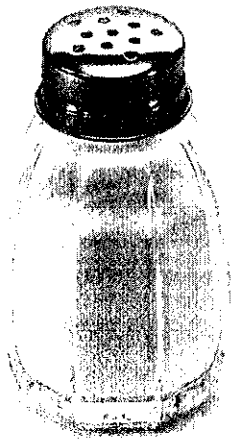
The most powerful cracking method is the brute-force method. It will always recover the password no matter how complex it is—it's just a matter of time. Very complex passwords that contain characters not directly available on the keyboard might take so much time that trying to crack them is not feasible on a single machine using today's hardware, but most complex passwords can be potentially cracked. This is usually much shorter than the password expiration time used by most site administrators. Using a real-world cracking tool is the only good metric for setting password expiration times.

The Pre-Computation Attack

As we previously explained, the strength of a password hash largely depends on the time it takes to generate a certain hash. Efficient password cracking is not possible if it takes a lot of CPU time to determine the hash that goes with a certain password. By pre-computing hashes of possible passwords and storing the results in a database or table, the CPU time can be invested at times when there is plenty of processing power available. By the time the real cracking needs to be done, matching hashes with passwords is only a matter of searching through the pre-computed tables, which requires more memory, but takes only a fraction of the time needed to brute-force a hash. The files containing the pre-computed password hash values are called "Rainbow Tables."

Fighting Pre-Computation Attacks

- Prevent hashes from being exact representations of passwords
- Unix: Salted hashes
- Windows: NTLMv2 uses domain name, server challenge, and other variables to randomize final hash to protect against pre-computation
  - NOT salted in local storage
  - Salted in transmission across the network



Pre-computation is only successful if you are certain that the hashes in the pre-computed tables will match the hashes of passwords discovered during future audits. You can be certain this is the case when a hash is an exact representation of a password. If two people have the same password, they will have the same hash and a pre-computation attack works.

The way to defeat this is to assign to each account a random value, known as a salt. Since each account has a unique salt, now if you combine the salt with the password and run it through a hash, even if two people have the same password, they will have different hashes because they have different salts. Since a salt is unique to each account and adds an element of randomness, it will defeat a rainbow level attack.

Rainbow Tables

- Based on research by Philippe Oechslin
- Pre-compute hashes into sorted table
- Trade CPU for memory
- Takes a lot of time to pre-compute, but makes cracking faster

Rainbow crack and the Rainbow Tables are an implementation of a faster cryptanalytic time-memory trade-off, developed by Philippe Oechslin.

Research on pre-computing password hashes was not entirely new. In 1980, Martin Hellman described a cryptanalytic time-memory trade-off that reduces the time of cryptanalysis by using recalculated data stored in memory. Philippe managed to further reduce the number of necessary calculations. Reducing the necessary CPU cycles obviously makes this method even more suitable for cracking password hashes.

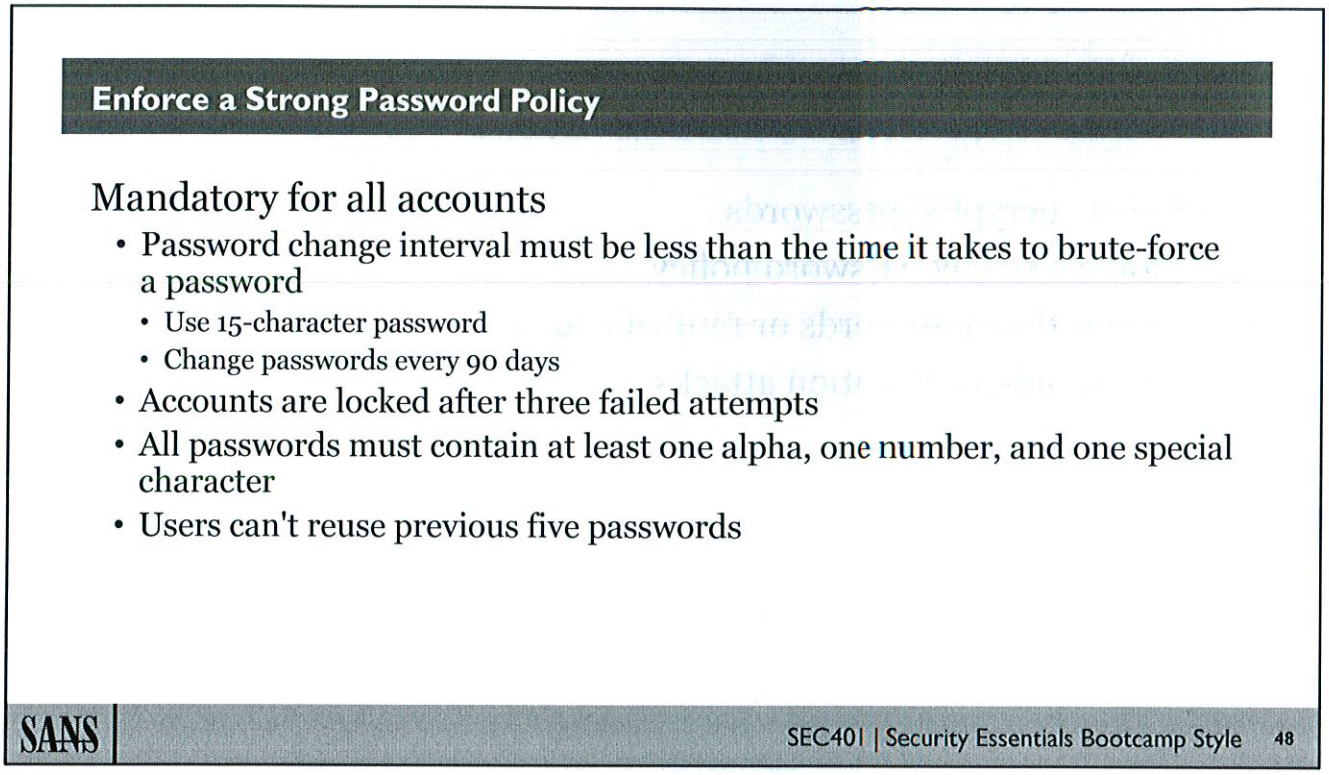
Rainbow Tables is the name given to the files that are produced by pre-computing password hash values and storing the data in an optimized manner to reduce the amount of disk space needed. After pre-computing the hash values, the cracking tool has to search only the tables for a given hash value to determine the corresponding plaintext password.

How to Protect Against Password-Cracking Attacks

- Protect encrypted passwords
- Enforce a strong password policy
- Use one-time passwords or multi-factor authentication
- Prevent pre-computation attacks

Now that we've examined some of the techniques that we can use to audit passwords in our own organizations, the next logical question is how do we prevent other people (attackers) from doing the same? Let's examine several techniques to protect against password-cracking attacks. The following are the techniques that we will review over the next several slides:

- Protect encrypted passwords
- Enforce a strong password policy
- Use one-time passwords or multi-factor authentication
- Prevent pre-computation attacks



A password policy is critical for a secure system. Even when you cannot strictly enforce a policy through automation, it's good to have a published policy so your users at least know what you expect. Here are a few general guidelines:

- Password change interval must be less than the time it takes to brute-force a password:
  - Use 15-character password
  - Change passwords every 90 days
- Accounts are automatically locked after three consecutive failed login attempts.
- Passwords must contain at least one letter, one numeral, and one special character.
- None of a user's previous five passwords can be reused.

Even the most strictly enforced policy won't help if users don't protect their passwords. Your users' habits are the first line of defense against password theft.

Choose a Hard-to-Guess Password

You can limit dictionary attacks by automatically enforcing a strong policy when users change passwords and by cracking passwords periodically. But these countermeasures cannot weed out every easy-to-guess password—it's always possible the attacker will have access to some piece of information about the user that your countermeasures don't. Again, we must rely on user education to fill in the cracks. Here are some dos and don'ts on the art of choosing a hard-to-guess password.

Do:

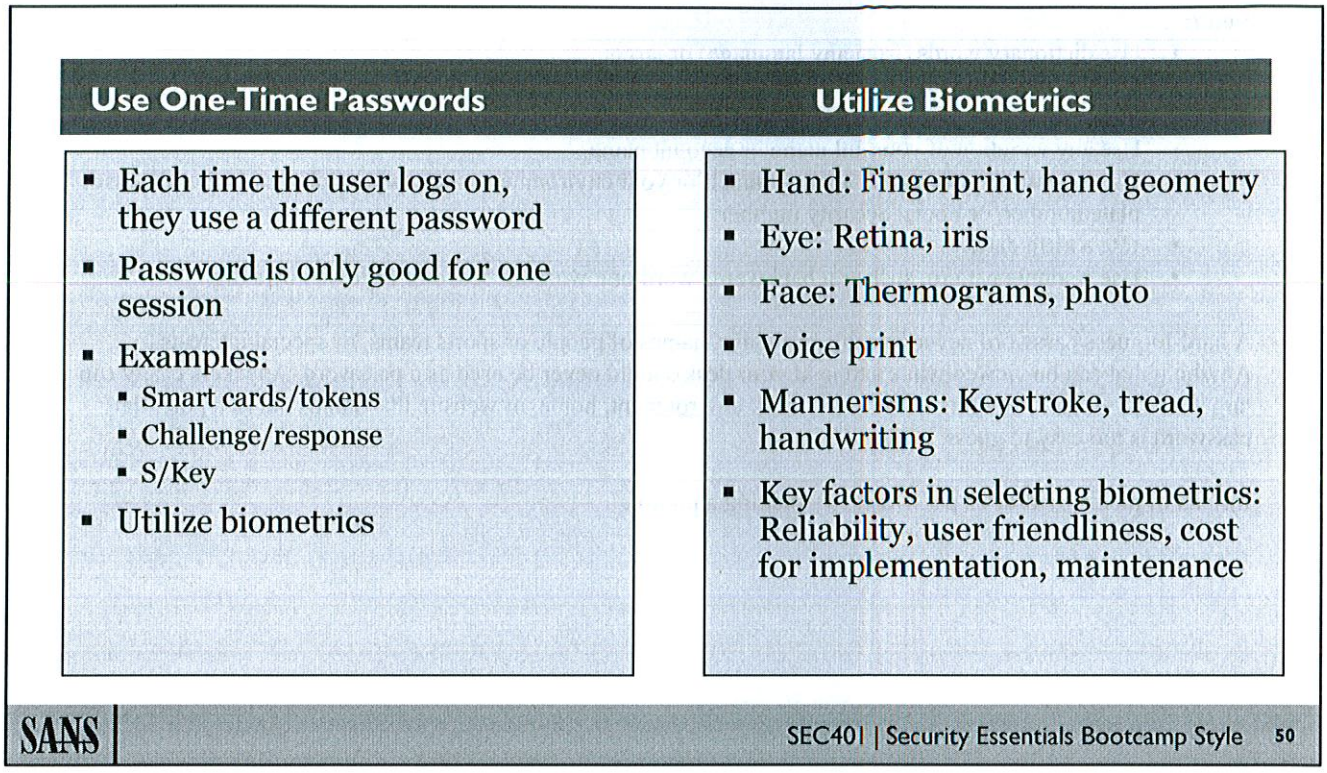
- Use a mixture of uppercase and lowercase letters as well as numerals or punctuation.
- Make sure new passwords are unrelated to any previous password.
- Use long passwords (at least 8 characters).
- Use a word pair with punctuation inserted, a passphrase (an understandable sequence of words), or the first letter of each word in a passphrase.

Don't:

- Use dictionary words (from any language) or jargon.
- Use a name, including that of a spouse, parent, child, pet, fantasy character, famous person, or location.
- Use any variation of your full name or account name.
- Use accessible information about yourself or your environment, such as your phone number, license plate number, or Social Security number.
- Use a birth date.
- Use a simple pattern, such as a backwards word or a word preceded or followed by a digit.

A hard-to-guess password never contains birthdays, names of people or sports teams, or special interests. Anything that can be viewed while sitting at your desk should never be used as a password. Attackers easily can target an individual. Anything in the user's work environment, home, or website that stands out as a potential password is too easy to guess.

Instead of picking words as passwords, try picking a phrase.



One-time passwords are very effective against password-guessing. Because the passwords change each time the user logs in, there is really no password to guess. The drawbacks are implementation costs and complexities and ongoing operating costs.

There are strengths and weaknesses to each of the approaches to one-time passwords. You will want to research each possibility to see which one is right for you. This section covers the approaches briefly to explain some of the options available to protect against password-guessing.

The most common way to implement one-time passwords is using token-based devices such as SecurID tokens. A user must have such a device handy when logging on to the system. The device is triggered by the time of day, so every minute the password changes. When the user wants to log on, she reads the current password from the token's display and types it in at the password prompt.

Instead of a time-based algorithm, some devices also use what is known as "challenge/response." The user presents her user ID to the system, which responds with a challenge. The user then types the challenge into the device, which generates a response. The user then types the response into the system at the password prompt.

In general, one-time passwords are a good countermeasure against keyboard and network sniffing. Because each password can be used only once and cannot be used to deduce subsequent passwords, it doesn't usually matter whether the password gets sniffed while the user types it.

One-time passwords are especially useful when a user is taking a trip to a place where encrypted communications tools are not available. The user could take along the token-based device and generate passwords at will. With a software-based system, the user could print a list of the next several one-time passwords.

Biometric mechanisms use people's physical characteristics to identify them uniquely. The fingerprint is a commonly used physical characteristic that varies from person to person. We're probably all familiar with the finger and thumbprint scanners that take advantage of this human trait. Fingerprint scanners have advanced to the point of being incorporated into PC cards, which can be inserted into a desktop or notebook computer to authenticate users, or being incorporated into the laptops themselves.

Even more common than the fingerprint biometric is the photo ID; this is beneficial because it can be used with "manual" biometric mechanisms. No computers need to be involved to compare the photo to the face of the person standing there in the flesh.

Effectively, biometric authentication boils down to a very long password that can never be reset. Biometric authentication systems that rely solely on biometrics for authentication are at risk when the identifying credentials for a person are stolen (such as reproducing a fingerprint left on a glass bottle) because it is not generally possible for someone to change his credentials after he has been compromised. Some biometric authentication systems require a two-factor authentication (such as biometrics and a PIN) to mitigate this risk.

Summary

- Access control is an important part of security
- Authentication: Biometrics and passwords
- Password cracking and countermeasures
  - John the Ripper
  - Rainbow tables

SANS | SEC401 | Security Essentials Bootcamp Style 52

Access control is an important part of any security model that deals with sensitive data or resources. Models have been developed to provide confidentiality (secrecy) of data as well as integrity (trustworthiness).

This module covered two ways to authenticate users: biometrics and passwords. Biometric mechanisms analyze a physical attribute of a person and compare it to recorded data known about the person. One determining factor in choosing a biometric mechanism is the users' acceptance of such a system. An uncomfortable or time-consuming enrollment process can be dreadful to users. Also, users don't like systems that are intrusive or produce too many false rejects. Even though biometrics can reduce user workarounds and password sharing while increasing security, the cost of the systems can be significant and must be weighed.

Often, passwords are the only protection against identity and data theft. Hence, several techniques have been developed to audit and attack passwords.

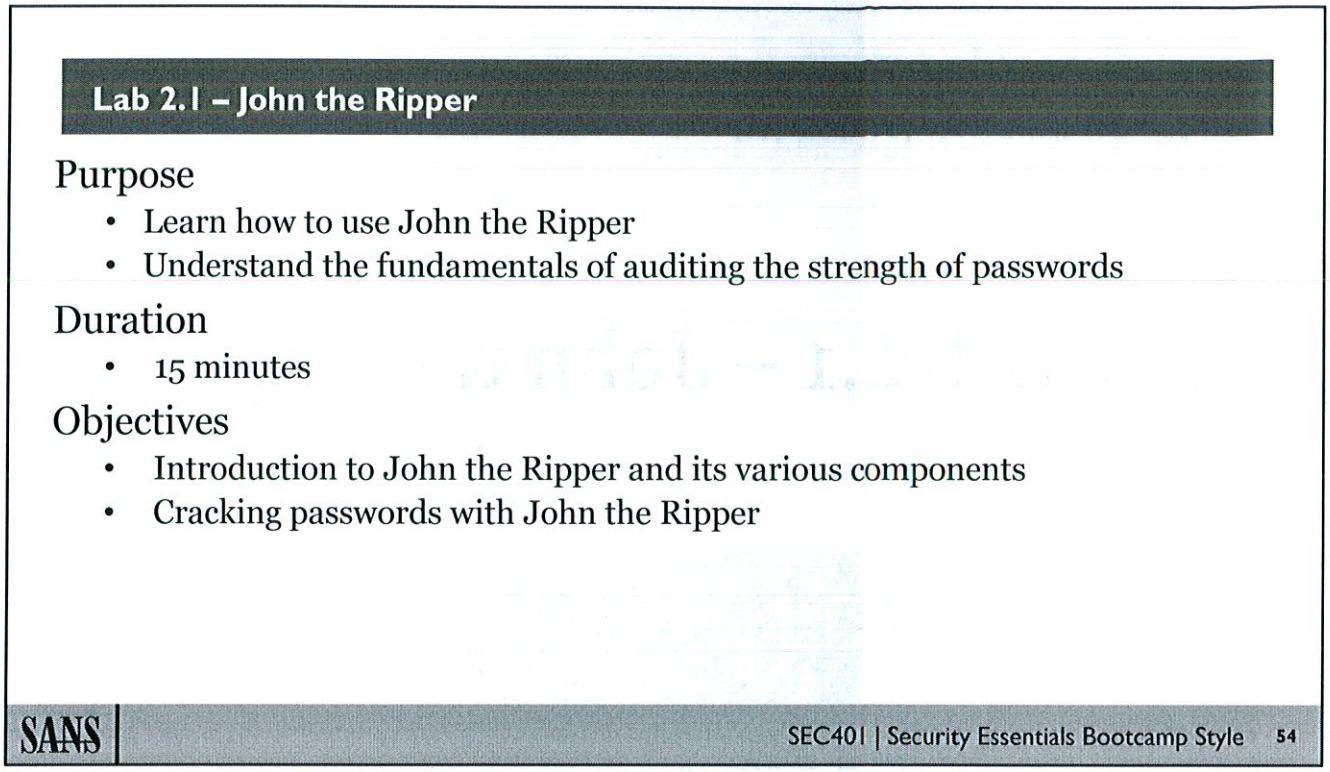
These techniques are surprisingly useful, despite the fact that operating systems protect password data with strong encryption and other countermeasures. The dictionary attack tests all the words in a wordlist, encrypting them one-by-one, against each of the hashed passwords. The hybrid attack also uses wordlists, but transforms each word, adding numerals and symbols. The brute-force attack is the most powerful but also the most time-consuming, trying every combination until a match is found.

SANS

Lab 2.1 – John the Ripper

John the Ripper

Now we will perform a lab using John the Ripper, a widely used password cracking tool, which was covered in the module. John the Ripper was written by a developer who goes by the name "Solar Designer." The project is maintained at http://www.openwall.com/john/. John the Ripper is a freely available tool that runs on a large number of operating systems and is included by default on your kali Linux VM. There is also a commercial version called "John the Ripper Pro," available at http://www.openwall.com/john/pro/ for a fee. John the Ripper supports various modes of password cracking, such as Simple Mode, Wordlist Mode, and Incremental Mode. We will discuss each of these modes in this lab.



Purpose

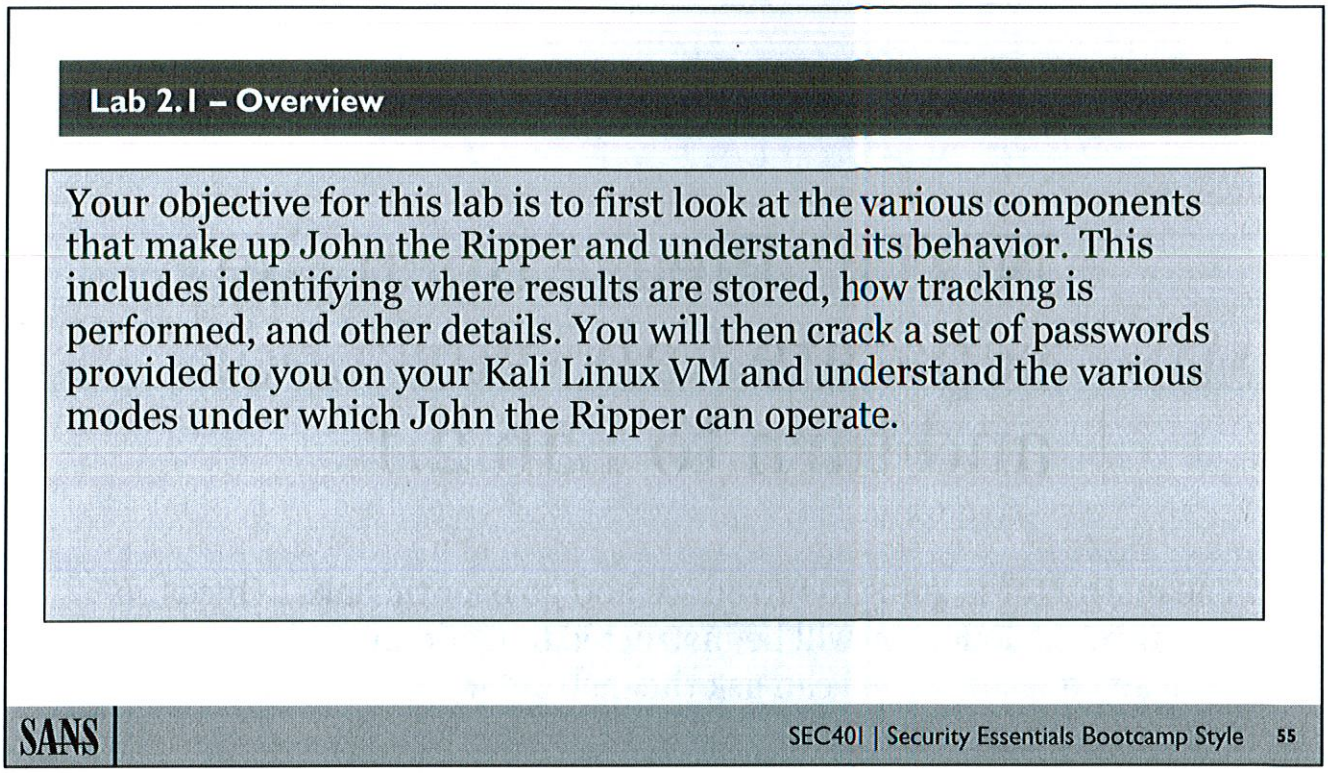
- Learn how to use John the Ripper
- Understand the fundamentals of auditing the strength of passwords

Duration

- 15 minutes
- The estimated duration of this lab is based on the average amount of time required to make it through to the end. All labs are repeatable both inside and outside of the classroom, and it is strongly recommended that you take the time to repeat the labs both for further learning and practice towards the GIAC Security Essentials Certification (GSEC).

Objectives

- Introduction to John the Ripper and its various components
- Cracking passwords with John the Ripper



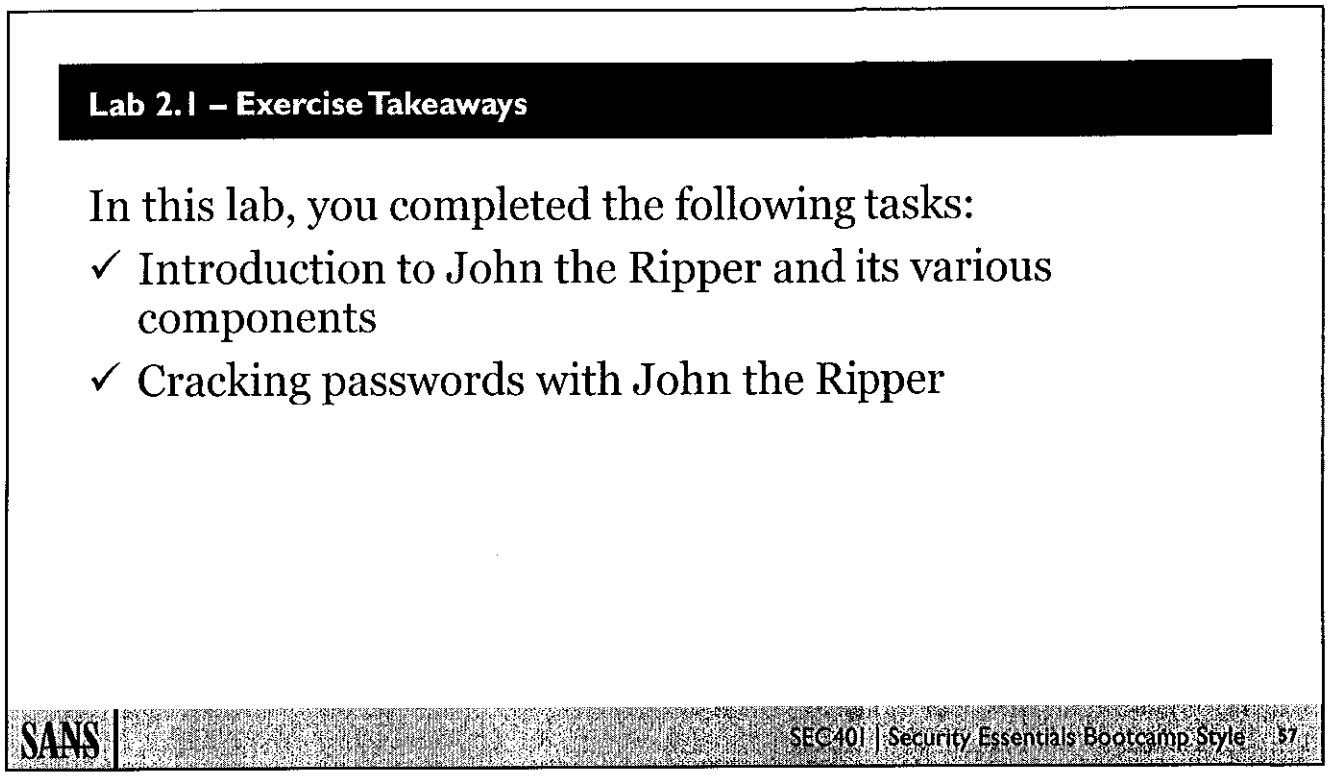
Your objective for this lab is to first look at the various components that make up John the Ripper and understand its behavior. This includes identifying where results are stored, how tracking is performed, and other details. You will then crack a set of passwords provided to you on your Kali Linux VM and understand the various modes under which John the Ripper can operate. Finally, you will take a look at the performance of John the Ripper and how long it took to crack the various passwords, as well as further look at the word mangling performed.

SANS

NOTE: Please open the separate Lab Workbook and turn to Lab 2.1

The instructor is going to introduce and go over the labs. Once the instructor is done, you will be instructed to work on the lab. If you have any questions, you can ask the instructor.

This page intentionally left blank.



In this lab, you completed the following tasks:

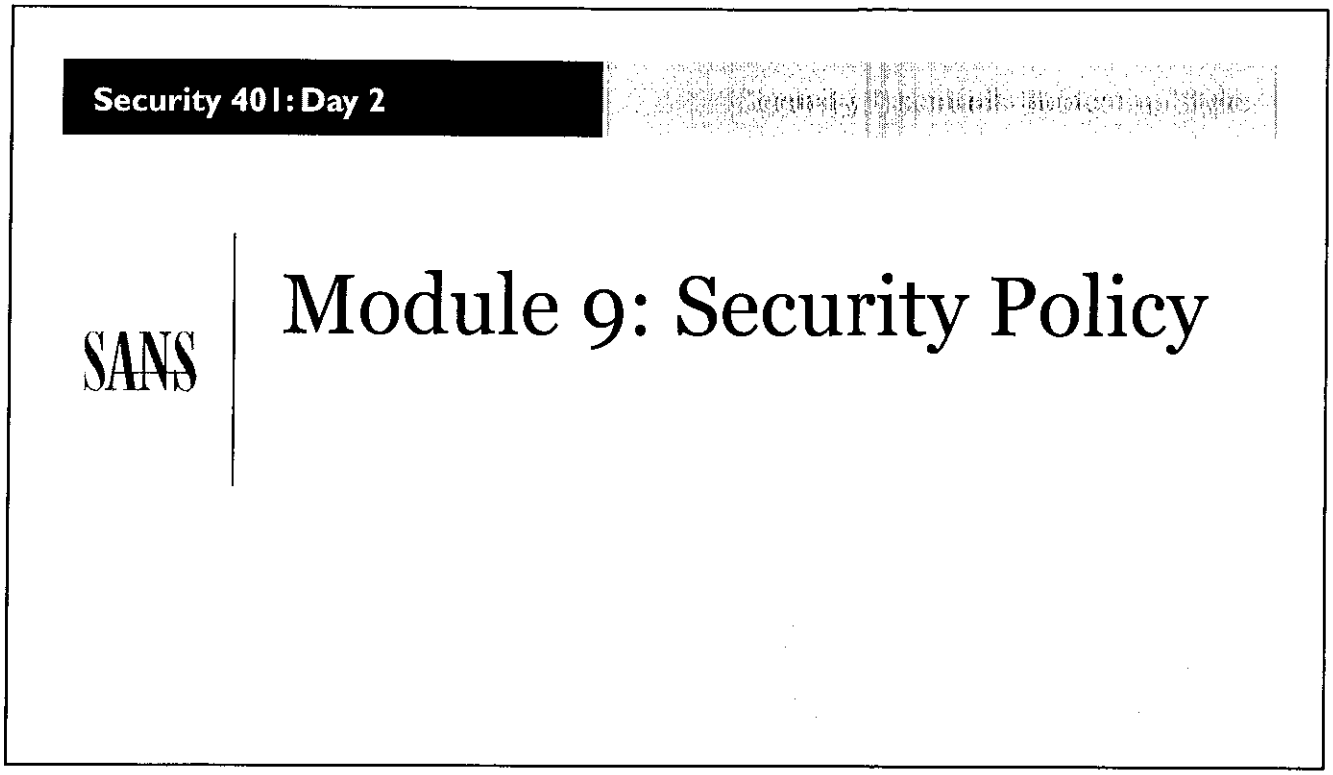
- ✓ Introduction to John the Ripper and its various components
- ✓ Cracking passwords with John the Ripper

In this exercise, we looked at another widely used password-cracking tool called "John the Ripper." The tool comes installed on Kali Linux by default and is simple to use. John the Ripper was originally written on UNIX, but has been ported over to many other platforms such as Windows and Mac OS X. It is another example of a tool that must be used only with permission and possessed by those with authority. Cracking passwords using a combination of wordlists and various forms of word mangling to test the strength of user-created passwords is a great way to validate your security policy. Brute-forcing passwords is typically reserved and used as a last resort when all other measures fail.

SANS

Lab 2.1 is now complete

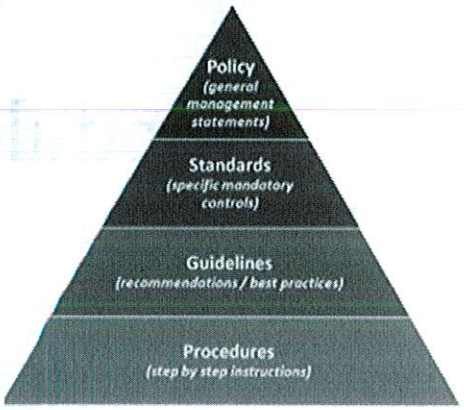
This page intentionally left blank.



Module 9: Security Policy
This page intentionally left blank.

Objectives

- Security Policies
  - Need for policies
  - Policy Framework
  - Enforcement
- Issue-specific policy examples
  - NDA (Non-disclosure agreements)
  - Copyright



Objectives

In this module, you will learn how to assess a policy by establishing a baseline framework to work within, and by establishing a mission statement that defines your policies. You learn how to assess and repair critical policies, and then how to repair others, one at a time. Involved in policy assessment is the need to understand how policies work throughout the organization and the need to understand unwritten or missing policies. In assessing policies, you should understand that there are roadblocks. Delays should be expected and overcome.

References

1. What are Policies, Standards, Guidelines and Procedures? - http://mindfulsecurity.com/2009/02/03/policies-standards-and-guidelines/

Security Policies

The student will understand the purpose and components of policy

This page intentionally left blank.

Why an Organization Needs a Security Policy

- Protects the organization, the people, and the information
- Establishes what must be done to protect information stored on computers
- Protects people who are trying to do the right thing
- Policies are the laws of an organization that set the boundaries of what is and is not acceptable
- No policy or an ineffective policy means an organization is in a state of security anarchy

Safeguarding information is a challenge when records are created and stored on computers. We live in a world where computers are globally linked and accessible, making digitized information especially vulnerable to theft, manipulation, and destruction. Security breaches are inevitable. Crucial decisions and defensive action must be prompt and precise.

A security policy establishes what you must do to protect information stored on computers. A well-written policy contains a sufficient definition of "what" to do so you can identify, measure, or evaluate the "how."

An effective security policy also protects people. Anyone who makes decisions or takes action in a situation where information is at risk incurs personal risk as well. A security policy allows people to take necessary actions without fear of reprisal. A security policy compels the safeguarding of information while it eliminates, or at least reduces, personal liability for employees.

Convincing the Organization

- Selling security policy to executives and users involves understanding their concerns
- To sell to executives, speak their language—money
- To get users on board, talk about how to make their job easier

Only executives can change corporate culture

If corporate culture is adverse to security, the culture must be changed before the policy can be updated

When it comes to selling the organization on the importance of a security policy, the more you understand the organization's concerns, the better you are able to sell your ideas. Knowing your company's overall security posture is vital. Understanding any specific areas of concern is also important. Perhaps intellectual property or web applications have recently become an area of interest. Try to understand what the organization's primary concerns are, and then look for ways to address those concerns.

On a more detailed level, people in different groups within an organization have different levels of concerns. Generally speaking, senior executives are concerned about finances. Understand this basic truth and you will start packaging your ideas in a way that provides senior management with information that speaks their language-saving money.

Users generally are interested in things that make their jobs easier. When selling security policy to user groups, try to identify with how the policy can help them perform their jobs in less time or with less effort.

Mission Statement

- What is the reason your organization exists?
- The "Top" of the security policy pyramid:
  - Helps to identify the critical assets across an organization
- When you encounter difficulties and crisis, a mission statement can help you refocus
- Cookie cutter policies do not work – policies must be aligned with the corporate culture

Is the risk tolerance in your organization aligned with the corporate culture?

A mission statement is the idea behind a brand. It is a statement to your customers and suppliers of what they can expect from you. Peter Drucker defines it in this way:

"A mission statement has to be operational, otherwise it's just good intentions."

A mission statement embodies your organization's reason for being—your purpose. If your organization does not have a mission statement, you should attempt to develop one and get it approved. You need an approved mission statement as we move forward to evaluate policy.

What does a mission statement have to do with information security? One of the biggest criticisms of security workers is that they are not sensitive to the needs of the business. So, we start with the heart and the soul of an organization—its mission.

Overall Security Posture

- Is the overall security posture more conservative or liberal?
- Some issues to consider include:
  - Allowing home use of laptop
  - Installing software
  - Sending personal information via e-mail
- Policy must be realistic, accurate, and enforceable
- Corporate position is the "why" and should be congruent with security posture

Many times, a mission statement points to what the expected overall security posture of an organization will probably look like. We discuss a number of "hot button" points, and you can probably add some that you've observed in your career to the list. One of the things to be sensitive to are "hot buttons"—the organization reacts differently to these than its normal approach to business. For instance, if a company was normally relatively liberal and informal, but required an incredible amount of documentation for sales, that might be worth discussing with management. We once worked with a high-tech company that ran seminars, and for every item it sold, it had to staple the cash money, check, or credit card receipt to the receipt for the purchase. The accountants spent hours unstapling all the receipts, depositing the money, and managing the records. After discussing this with management and explaining that it ran completely contrary to the corporate culture, it changed this policy.

There is no right or wrong security posture. A conservative approach might offer security benefits to the company by placing more controls on what an employee can do. However, this approach might take more security and IT staff to implement and monitor. A liberal approach might provide a more informal work environment but might put the company assets at a higher level of risk.

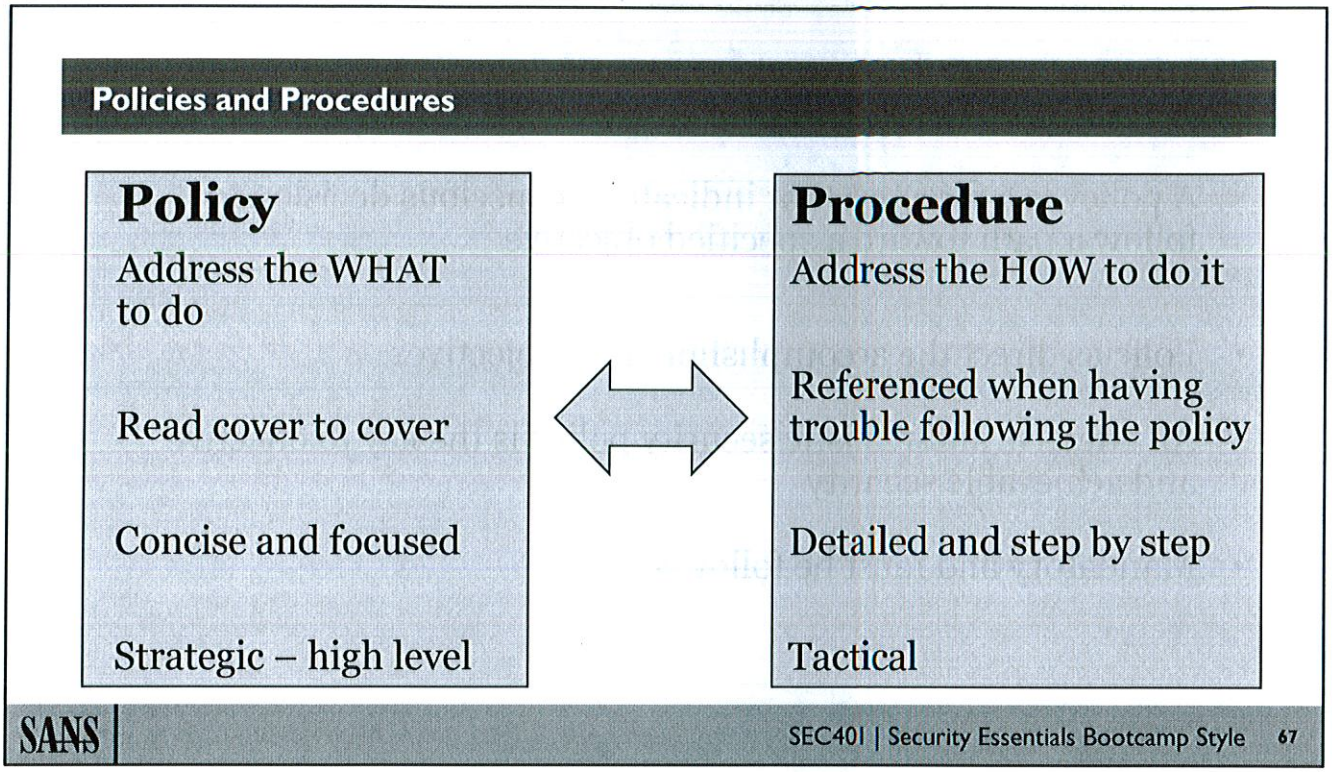
Establish a Documentation Baseline

- An organization survey for everything that is written down
- Key documents: All applicable policies at all levels, checklists, procedures, and management directives
- Acceptable use policy (AUP) and system-specific (hardening docs)

It is important to identify all of the "active" policies that are in place and make sure that there are no contradictions between active policies

A baseline is our foundation for evaluating policy, and it is made up of several components. We have the mission statement that defines what customers, suppliers, and employees should be able to expect from the organization. We have the assessment of the organization's security posture, which is a bit like looking in a mirror. Our mission statement is the way we hope people view us; the security posture is what we actually look like. Now, we can begin to evaluate policy. As you know, policy exists on several levels. Hopefully, everything is organized and up to date; if not, you need to begin the search for written guidance.

Enterprise-wide or corporate policy is the highest level of policy and consists of a high-level document that provides a direction or thrust to be implemented at lower levels in the enterprise. One approach to this for information security is to get a letter of endorsement from senior management. This policy must exist to properly assess lower-level policy. If this policy does not exist, begin work to create this policy document and get it approved before attempting to assess lower-level policy. This enterprise- or corporate-level security policy is the demonstration of management's intent and commitment for the information security in the organization. This should be based on facts about the criticality of information for business as identified during our assessment and evaluation of security posture. The security policy statement should strongly reflect the management's belief that if information is not secure, the business will suffer.



What do you do when some work does not seem to be covered by an organizational policy? Procedures are derived from policies; if you can characterize the procedures you follow (and you should be able to do that easily), then you can derive the parent policy. This is true even if it has not yet been written and signed. By walking through the who, what, when, where, and why, the parent policy is derived from an understanding of the procedure.

In your organization, what procedures can you list for which you need to document the policy? Make notes on the who, what, when, where, and why of your procedures. You will be able to derive any missing policies based on these notes.

Policy

- A policy is a directive that indicates a conscious decision to follow a path toward a specified objective
- Policies direct the accomplishment of objectives
- An effective and realistic security policy is the key to effective and achievable security
- Mandatory and must be followed

A *policy* is a *directive* that indicates a conscious decision to follow a path toward a specified objective. Often a policy can initiate the institution and empowerment of resources, or direct action by providing procedures or actions to be carried out. The policy itself should be effective and realistic and have achievable security goals.

It is critical to write down in a clear and concise manner what is expected of everyone in the organization when it comes to security. It is also helpful to inform people about what is expected of them, what the organization does, and what others in various roles within the organization do.

Procedure

- Detailed steps to be followed by users, system operations personnel, or others to accomplish a specific task
  - Preparing new user accounts and assigning privileges
- Complements the policy by focused on the HOW to do what is listed in the policy
- Step by step and very detailed
- Mandatory

A procedure is a step-by-step document that is used for operations. Procedures can be daily operations, such as nightly backups or infrequent operations like recovering from a disaster. These steps must be clear, complete, and concise. They should also be reviewed on a regular basis to ensure they are accurate and that no changes have occurred. While they are separate documents, the policies and procedures complement each other. The policy specifies what to do and the procedure specifies how to do it.

Policies are often read cover to cover, but procedures are typically referenced to provide the detailed steps in how to perform an activity listed in the policy. While policies are recommended to be 3-5 pages, procedures can be much lengthier because they contain a lot more detail and must list every specific step that must be followed.

Standard

- Organizational – Strategic
- Specifies uniform use of specific technologies or parameters
  - Focuses on WHAT technology to utilize
- Usually refers to specific hardware and software

Standards are applied to the organization as a whole. As with policies, these are mandatory. Standards are more specific than the overarching policies. They provide additional definition to the policies and tailor them to specific technologies. Unlike a policy, a standard does not state what is expected of a user from an organizational security stance. Instead, a standard specifies a certain way something should be done or a certain brand or type of equipment that must be used. A simple example of a standard is that all computers purchased must be a certain model and from a certain vendor.

Baseline

- A baseline is a more specific implementation of a standard
- A baseline definition gets into specific technical details of how a system should be configured from either a software or hardware standpoint
  - Hardening guides
- When adopted by the organization, baselines are compulsory

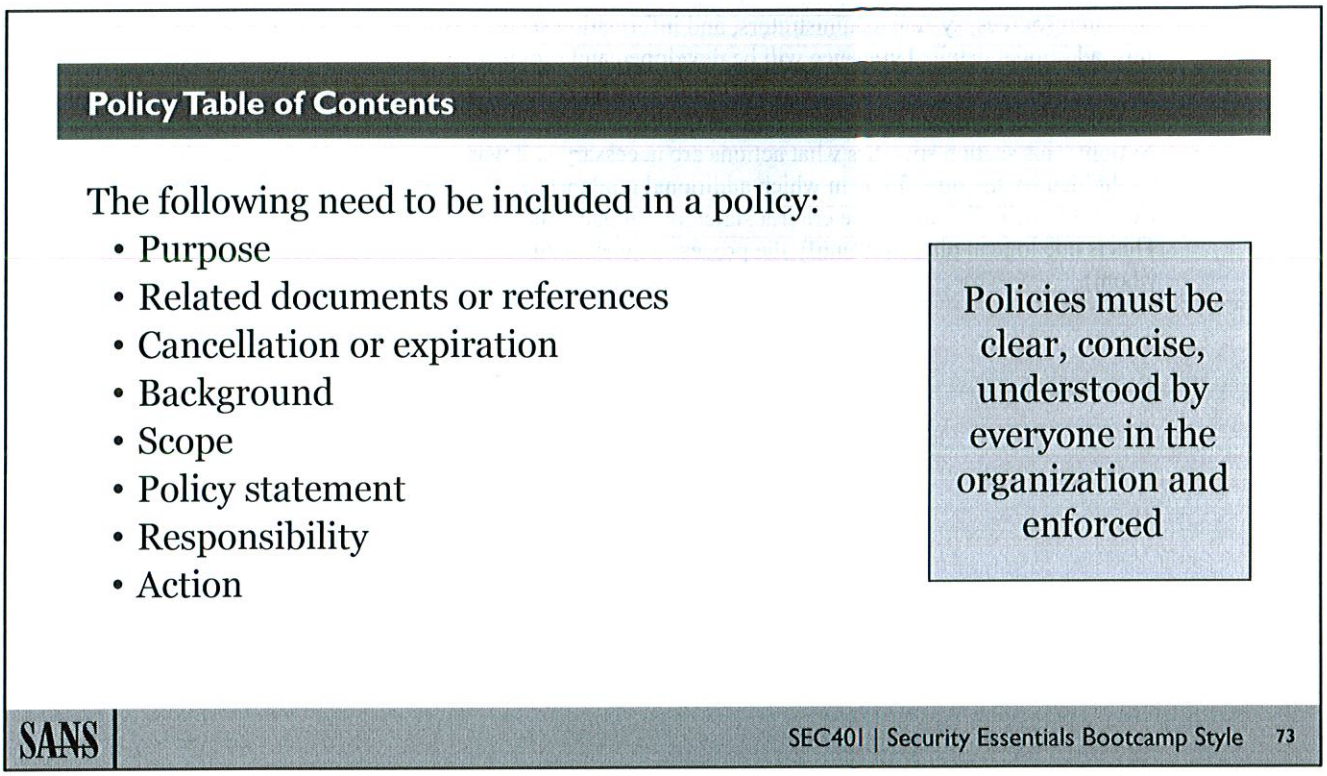
A baseline definition is essentially a more specific implementation of a standard. A baseline definition usually gets into specific technical details of how a system should be configured from either a software or hardware standpoint. After these documents have been thoroughly tested, they become mandatory for someone to enforce. Usually, a baseline starts off as a guideline until it has been properly modified to meet the needs of the organization. Hardening rules for setting up a new server is an example of something that starts off as a guideline and quickly turns into a baseline.

Guideline

Suggestions:

- Tips and tricks to make it easier to follow the other documentation
- Assists users, systems personnel, and others in effectively securing a system
- Helps ensure that specific security measures are not overlooked
- Applies to security measures that might be implemented in more than one way
- Not compulsory

Guidelines, unlike standards and policies, are not mandatory. Best practices are examples of guidelines that many organizations try to achieve; however, there is not a penalty if guidelines are not met. A guideline is more like a recommendation of the way that something should be done; however, people can choose whether they want to follow it or not. A best practice might start off as a guideline, and if analysis shows that there is a great benefit to following this guideline from either a security or efficiency standpoint, the guideline might become a standard, which would then make the guideline mandatory to follow.

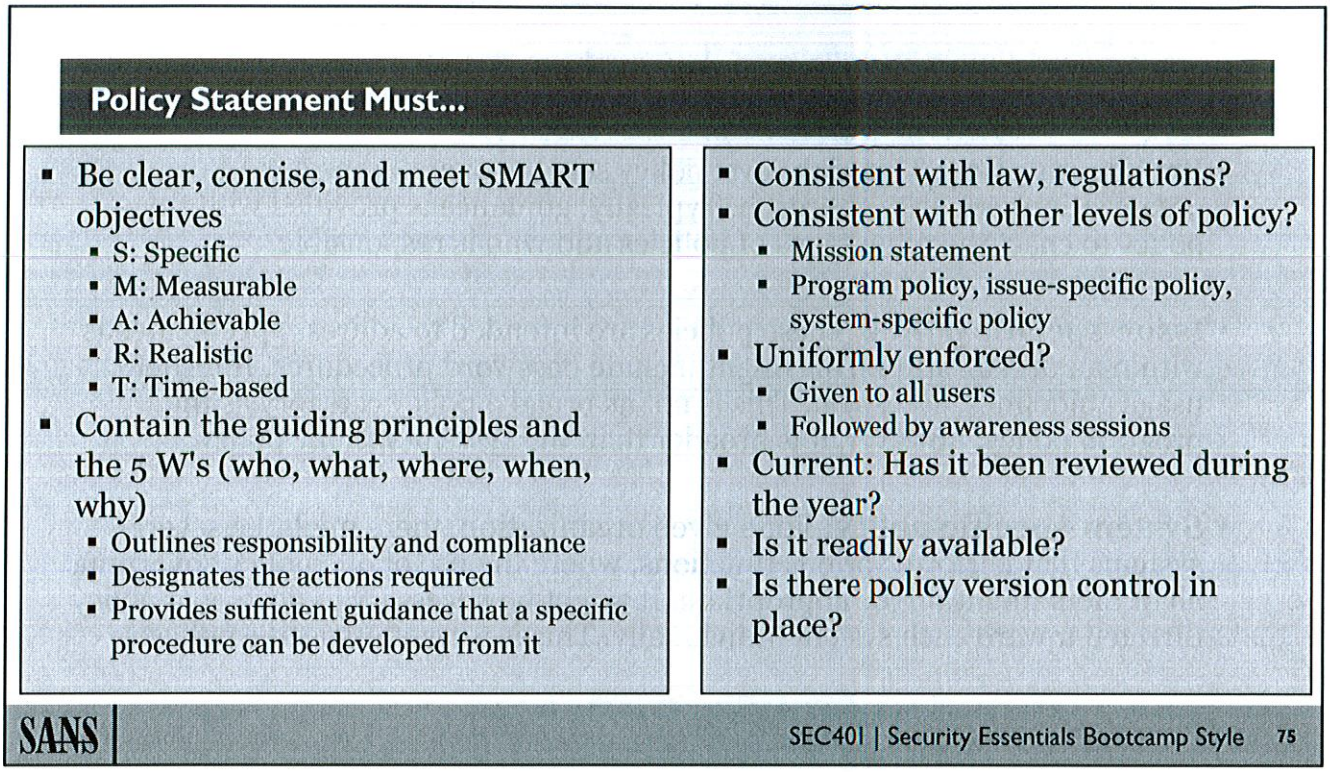


It is good to follow an approach where if the policy you need does not exist, or is badly out of date, you create, or update it, and then have it approved. The final step in establishing a framework is to assess critical policies. Even though you just wrote or updated them and you know they are correct, policies should be checked. It is better if you do not check your own work. We can spot only our own errors a small percentage of the time. Any detail-oriented person can check to see that the format of the policies is correct. In fact, a person that is not familiar with the subject the policy covers is ideal. People who are subject matter experts might not notice omissions in procedures because they have the procedure memorized. When you ask a person to help you by verifying policy contents, ask that she include the most common elements in her analysis. These include:

- **Purpose:** The security policy usually contains a statement, often at the beginning, describing the reason the policy is being established and any associated goals.
- **Related documents:** This is often entitled "References" and usually cites higher-level policy or implementation guidance.
- **Cancellation:** A new or updated policy might supersede existing (perhaps outdated) policy. This section identifies those policies and clarifies what is actually in effect.
- **Background:** This optional section provides information amplifying the need for the policy. It might also provide historical information relevant to the subject.
- **Scope:** This section identifies the depth and breadth of coverage (to whom or what the policy applies). Is it for one element of the organization, or will it also apply to contractor agencies who work for your organization?
- **Policy statement:** This identifies the actual guiding principles or what is to be done. The statement(s) are designed to influence and determine decisions and actions within the scope of coverage. The statements should define actions that are prudent, expedient, or advantageous to the organization.
- **Responsibility:** The security policy document states who is responsible for what. Typical positions that might be addressed include the head of the corporation, the CIO, people in the legal department or in

human resources, system administrators, and information security officers. Subsections might identify how additional detailed guidance will be developed and provided, as well as the frequency of policy review. Methods or techniques for measuring compliance might also be included in this section (as well as identifying parties responsible for the audit).

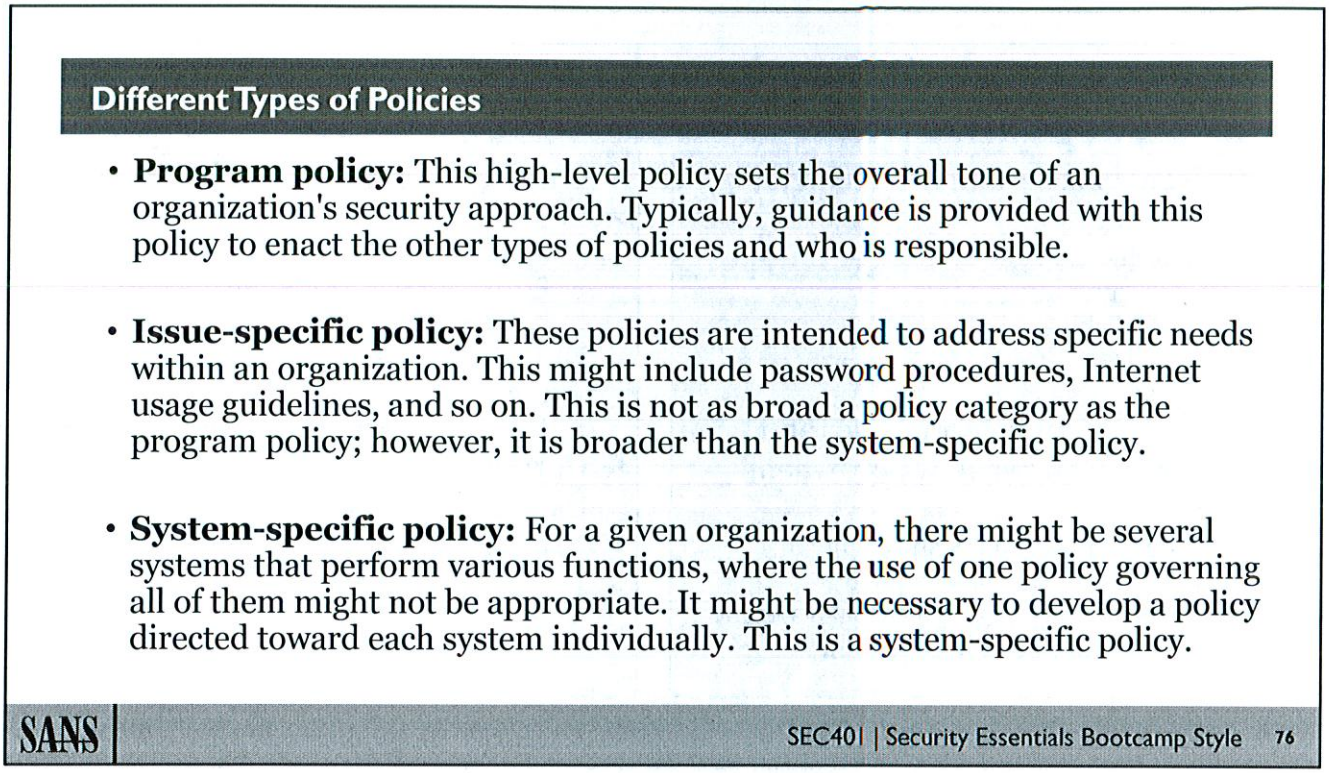
- **Action:** This section specifies what actions are necessary and when they are to be accomplished. It might identify the time frame in which additional guidance (mentioned previously) will be forthcoming. Hopefully the policy meets the criteria stated previously, but there might be a need for a waiver process. This is one logical place to identify the process as well as the time frame for policy review (and by whom).



Policy is not detail-oriented; it is a high-level focus of who has to do what. It should not have step-by-step information that is provided in the procedures. The rule of policy assessment is that the policy covers the "who" and "what needs to be done." Procedures cover how to do it. For example, the policy would state that each user must change his or her password every 90 days. The procedures would give you the details of how to change a password on a given operating system. In general, procedures can be updated with far less review than policy. The core characteristics of a policy are:

S: Specific
M: Measurable
A: Achievable
R: Realistic
T: Time-based

The security policy must also be in accordance with all relevant laws.



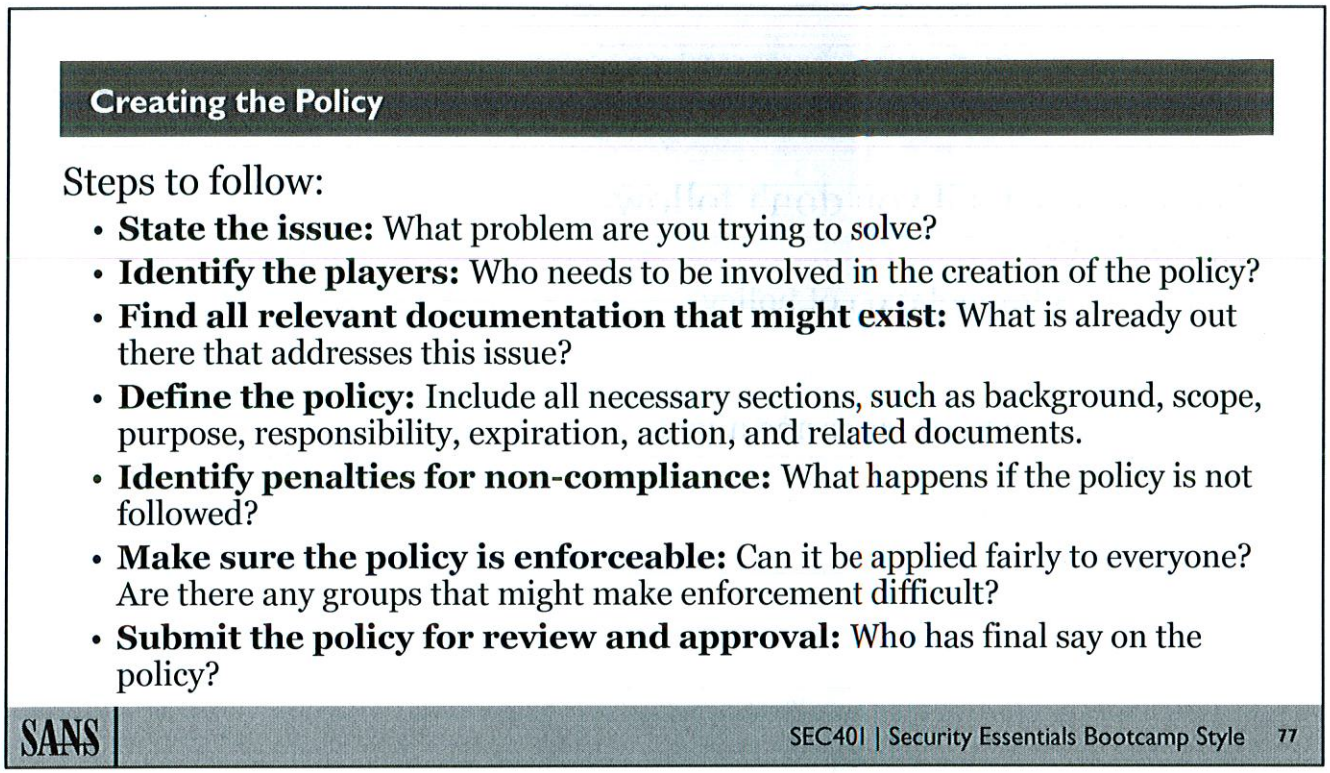
Considering the information in need of securing and the level at which it must be secured, examine the policy to see whether it is consistent with the mission statement and with the following other policies:

- **Program policy:** This high-level policy sets the overall tone of an organization's security approach. Typically, guidance is provided with this policy to enact the other types of policies and who is responsible.
- **Issue-specific policy:** These policies are intended to address specific needs within an organization. This might include password procedures, Internet usage guidelines, and so on. This is not as broad a policy category as the program policy; however, it is broader than the system-specific policy.
- **System-specific policy:** For a given organization, there might be several systems that perform various functions, where the use of one policy governing all of them might not be appropriate. It might be necessary to develop a policy directed toward each system individually. This is a system-specific policy.

If you discover any discrepancies, note them because you will need to resolve them for the policy to be meaningful... Again, note any contradictions you discover so you can get the document corrected.

Examine the policy for revisions to keep it current. Security policy should be reviewed regularly. Revisions in implementation should reflect lessons learned from recent incidents and new threats to the organization's security.

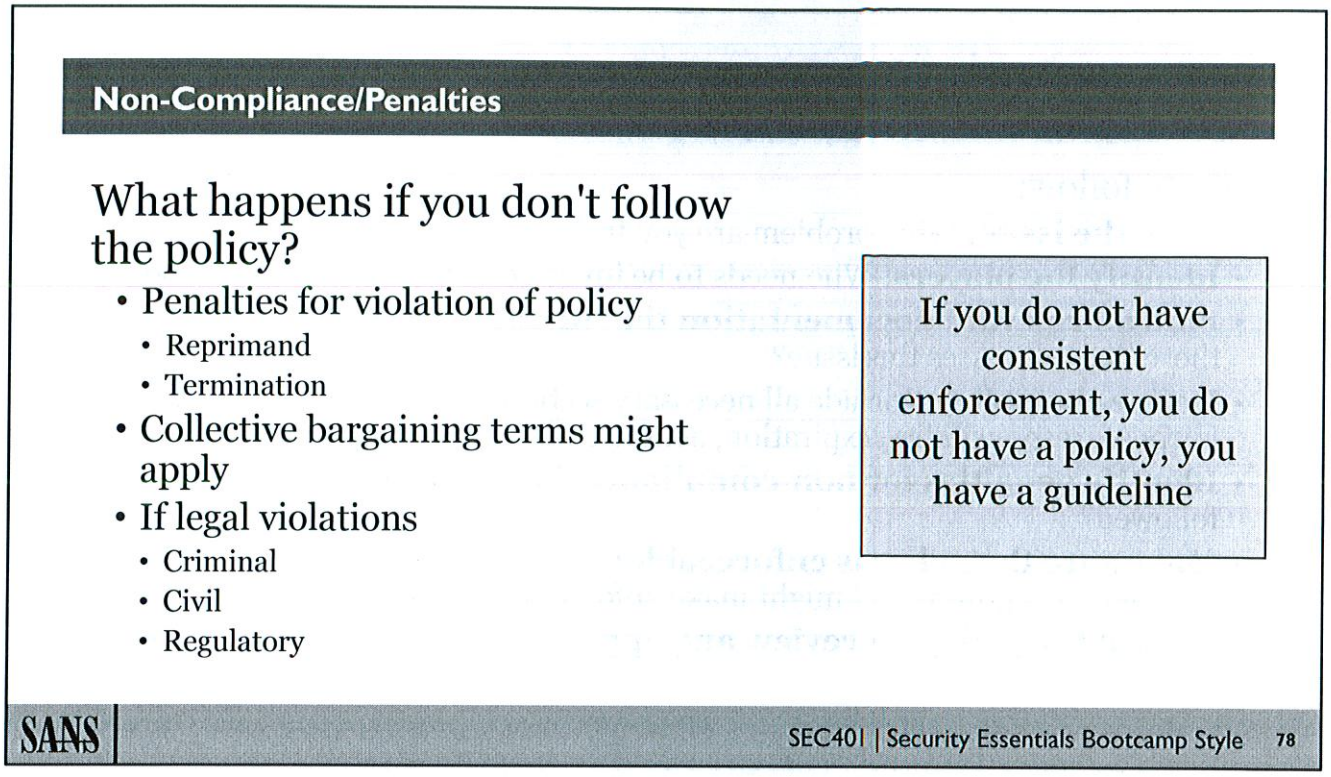
Check whether the security policy is readily available. The policy development guide might provide information regarding the responsibility for publishing and making available specific policy documents. Make sure security policy is incorporated in employee handbooks and posted for reference. It must be required reading as part of the employee orientation process. We recommend that security officers consider building an intranet Web page.



You have the basic information that you need to get started. As the policy is developed, you can continue to research the technical controls. As you see each section, one of the important questions to ask yourself is, "Does this policy set the correct tone or posture for my organization?" Obviously, a university will have a different tone to its AUP from a government entity. However, in every case, if the employees cannot easily read and understand it, the policy will fail to protect information and people, and it might not even prove to be enforceable.

The steps to follow when creating policy are:

- **State the issue:** What problem are you trying to solve?
- **Identify the players:** Who needs to be involved in the creation of the policy?
- **Find all relevant documentation that might exist:** What is already out there that addresses this issue?
- **Define the policy:** Include all necessary sections, such as background, scope, purpose, responsibility, expiration, action, and related documents.
- **Identify penalties for non-compliance:** What happens if the policy is not followed?
- **Make sure the policy is enforceable:** Can it be applied fairly to everyone? Are there any groups that might make enforcement difficult?
- **Submit the policy for review and approval:** Who has final say on the policy?



Penalties for violation of this policy will vary depending on the nature and degree of the specific violation. Penalties range from reprimand through termination for employees in accordance with the provisions of any collective bargaining agreement, to the extent such agreement applies to the employees. If violations of law are involved, users might incur civil liability to the organization or third parties and might also be subject to prosecution.

It is important to remember that if you do not have consistent enforcement, you do not have a policy, you have a guideline!

Issue-Specific Policy Examples

The student will be introduced to two issue-specific policies: non-disclosure agreement (NDA) and copyright

This page intentionally left blank.

Non-Disclosure Agreement

- Policy covers use, control, and enforcement of NDA
- An NDA protects both parties; it must not be one-sided
- An NDA protects sensitive information; the individual receiving information agrees to keep it confidential
- A legal document has certain specific requirements: Write clear, readable text

Let's start with an example using a non-disclosure agreement (NDA). This is an extremely wise tool for the protection of people and information.

Essentially, an NDA is a legal document that protects the parties involved in specific circumstances. The owner of the information is protected, and the individual who signs the NDA is also protected.

The NDA is an agreement between parties. The policy covers the use, control, and enforcement of the NDA. When creating the policy, remember that it must contain all of the components essential to any good policy. The policy should include *who* the NDA applies to, *what* the NDA is, *when* the NDA should be used, *where* the NDA is applicable, and *why* the NDA is important.

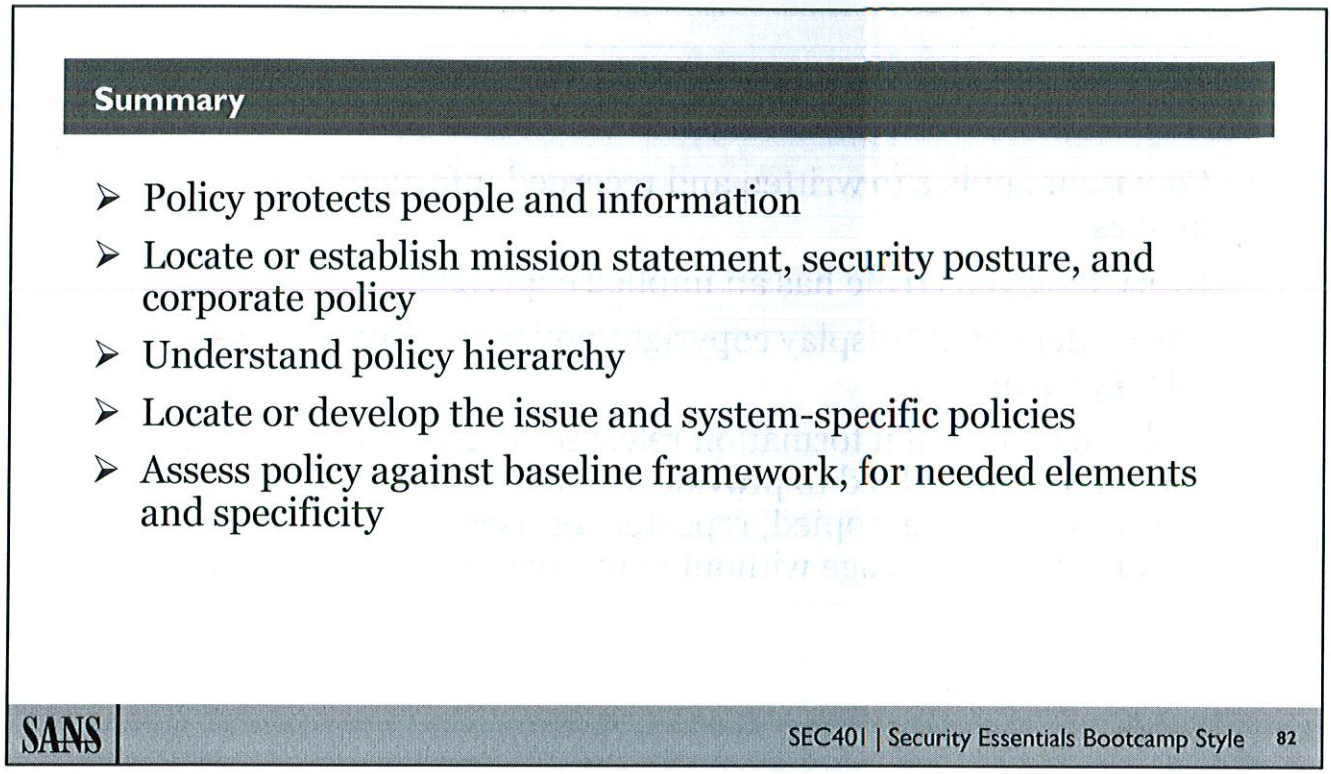
Intellectual Property – Copyright

- Copyright applies to written and recorded information and images
- Everything you create has an implied copyright
- The owner should display copyright notice to avoid "innocent infringement"
- Web pages and all information released to the public can and should be copyrighted to provide a measure of protection should they be duplicated, copied, reposted, or used within another document or Web page without your organization's permission

Copyright law is fairly consistent globally based on the work of the Berne Convention, but the information on the preceding slide is based on the US Library of Congress interpretation.

Copyrights primarily protect the author or the owner of a piece of information. If two parties agree that a written piece is work for hire, then the rights become that of the owner. Any organization's document that is published, such as a flyer, manual, or Web page, should have a copyright notice to identify the year of publication and name of the copyright owner.

Registering a copyright is also not required but gives additional protection. If there is any reasonable chance your organization would legally pursue a violation of your copyrighted material, then you should register that work.



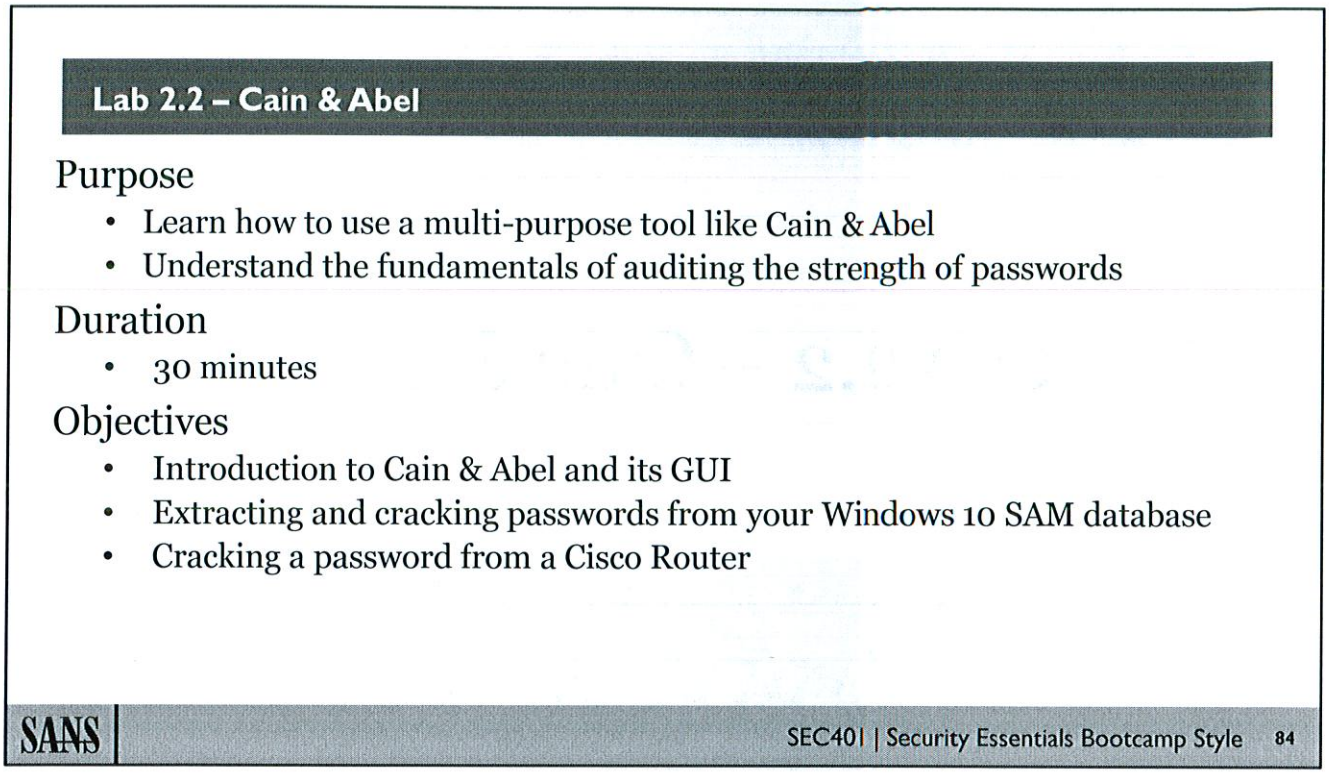
You now have a big-picture approach for evaluating policy. When you get back to your organization, we hope that you will put what you have learned into practice! Take the steps in order and the framework will work for you. Revisit your mission statement and ask questions to see whether your organization is living up to its mission. Start with a corporate policy, ensure you have the support of senior leadership, and help them state that good security is good business in a manner that cannot be misunderstood. Make sure you have required policies, that they have the required elements and that they are clear, concise, and SMART. Use a detail-oriented person to help you assess the specificity of the policy. Someone familiar with the topic might "autocorrect" or "fill in" any errors or omissions. If you wrote the policy, employ someone that knows the organization and fundamentals of information security to review the content. Remember that the threat level can change and policy should be reviewed in the light of major changes.

SANS

Lab 2.2 – Cain & Abel

Authentication protocols vary in strength, and it is important to understand the pros and cons of each one. Furthermore, a strong password policy with enforcement must be in place to ensure users are choosing strong passwords and passphrases, including a minimum length, complexity, and other factors discussed in the aforementioned module.

There are a large number of password auditing and cracking tools available. In this lab, you will use the tool Cain & Abel, written by Massimiliano Montoro, which includes an enormous number of features. Cain & Abel is used by many types of security professionals including auditors, penetration testers, and administrators. It is already installed on your Windows 10 VM and is available online at http://www.oxid.it/cain.html. Besides password cracking, Cain & Abel can also perform sniffing, VOIP capture and RTP stream replay, remote desktop protocol (RDP) attacks, and countless others.



Purpose

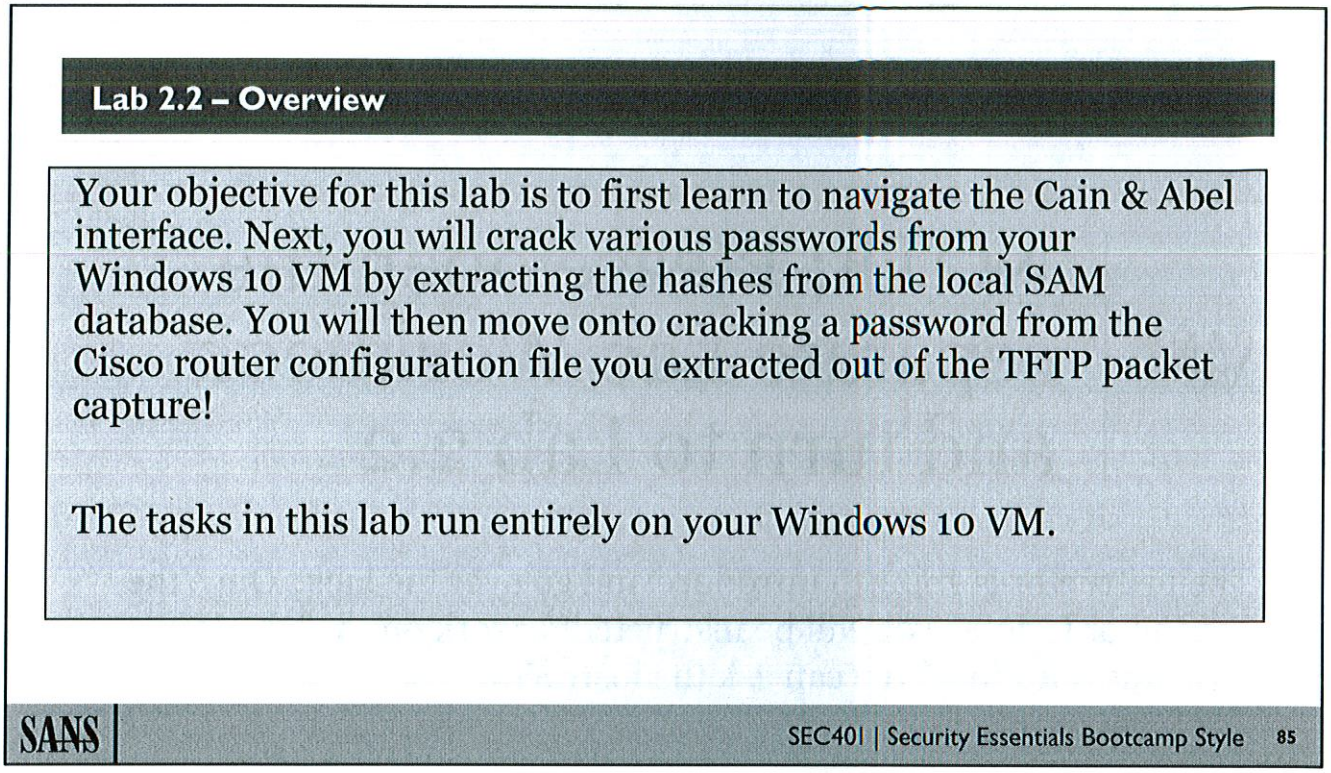
- Learn how to use a multi-purpose tool like Cain & Abel
- Understand the fundamentals of auditing the strength of passwords

Duration

- 30 minutes
- Depending on the speed of your computer, the copying and unzipping of the virtual machines from the USB drive might cause the duration of this lab to increase. If you are attending this class at a live event, you might have to allow the virtual machines time to extract during lecture and catch up during a break.

Objectives

- Introduction to Cain & Abel and its GUI
- Extracting and cracking passwords from your Windows 10 SAM database
- Cracking a password from a Cisco Router



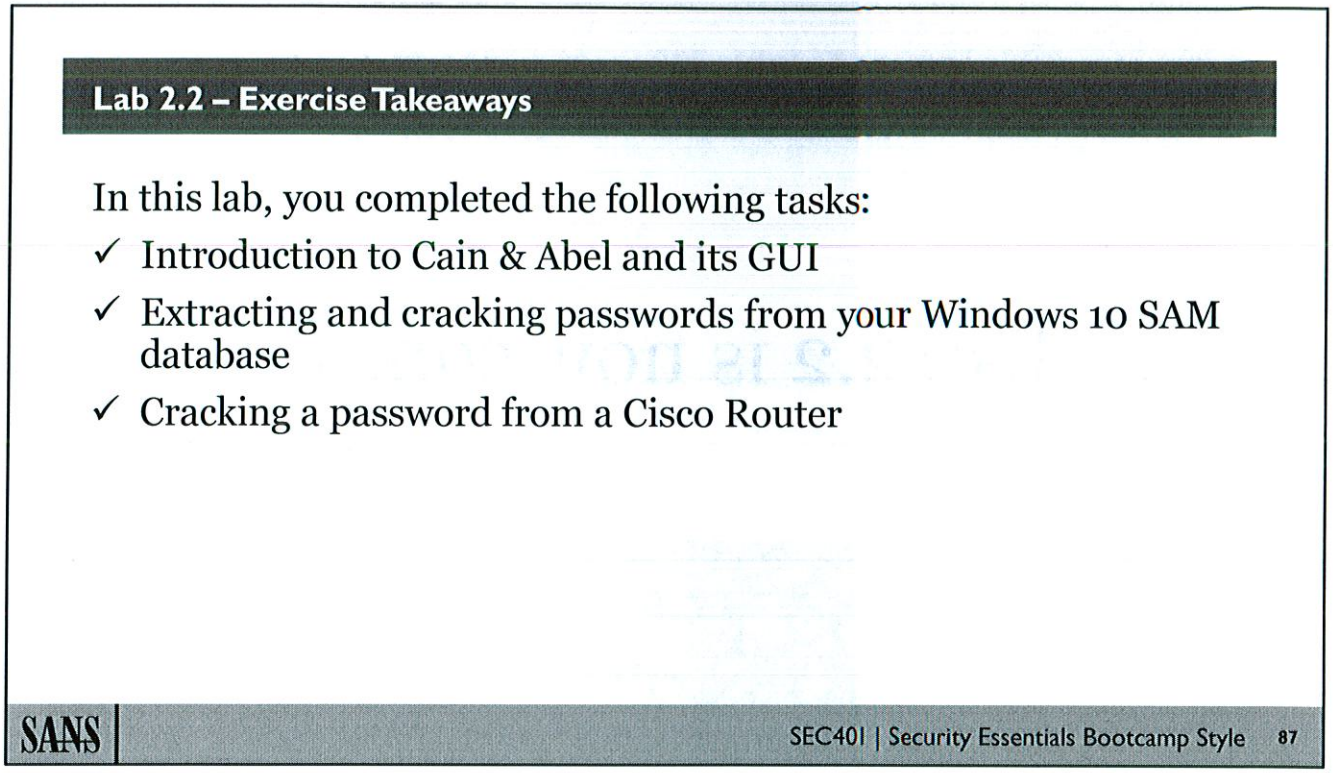
Your objective for this lab is to first learn to navigate the Cain & Abel interface. Next, you will crack various passwords from your Windows 10 VM by extracting the hashes from the local SAM database. You will then move onto cracking a password from the Cisco router configuration file you extracted out of the TFTP packet capture! The tasks in this lab run entirely on your Windows 10 VM.

SANS

NOTE: Please open the separate Lab Workbook and turn to Lab 2.2

The instructor is going to introduce and go over the labs. Once the instructor is done, you will be instructed to work on the lab. If you have any questions, you can ask the instructor.

This page intentionally left blank.



In this lab, you completed the following tasks:

- ✓ Introduction to Cain & Abel and its GUI
- ✓ Extracting and cracking passwords from your Windows 10 SAM database
- ✓ Cracking a password from a Cisco Router

As shown, Cain & Abel has a huge number of features beneficial to many types of security professionals. This powerful tool must be used only by approved professionals and only with permission. Antivirus will often detect Cain & Abel as malicious software and as such, an exception must be added. We primarily used Cain & Abel to crack different types of passwords; however, it offers much more such as wireless attacks, ARP cache poisoning, NTLM downgrade attacks, and VOIP-related attacks.

SANS

Lab 2.2 is now complete

This page intentionally left blank.

Security 401: Day 2 | Security Essentials Bootcamp Style

SANS

Module 10: Critical Security Controls

Module 21: Critical Security Controls

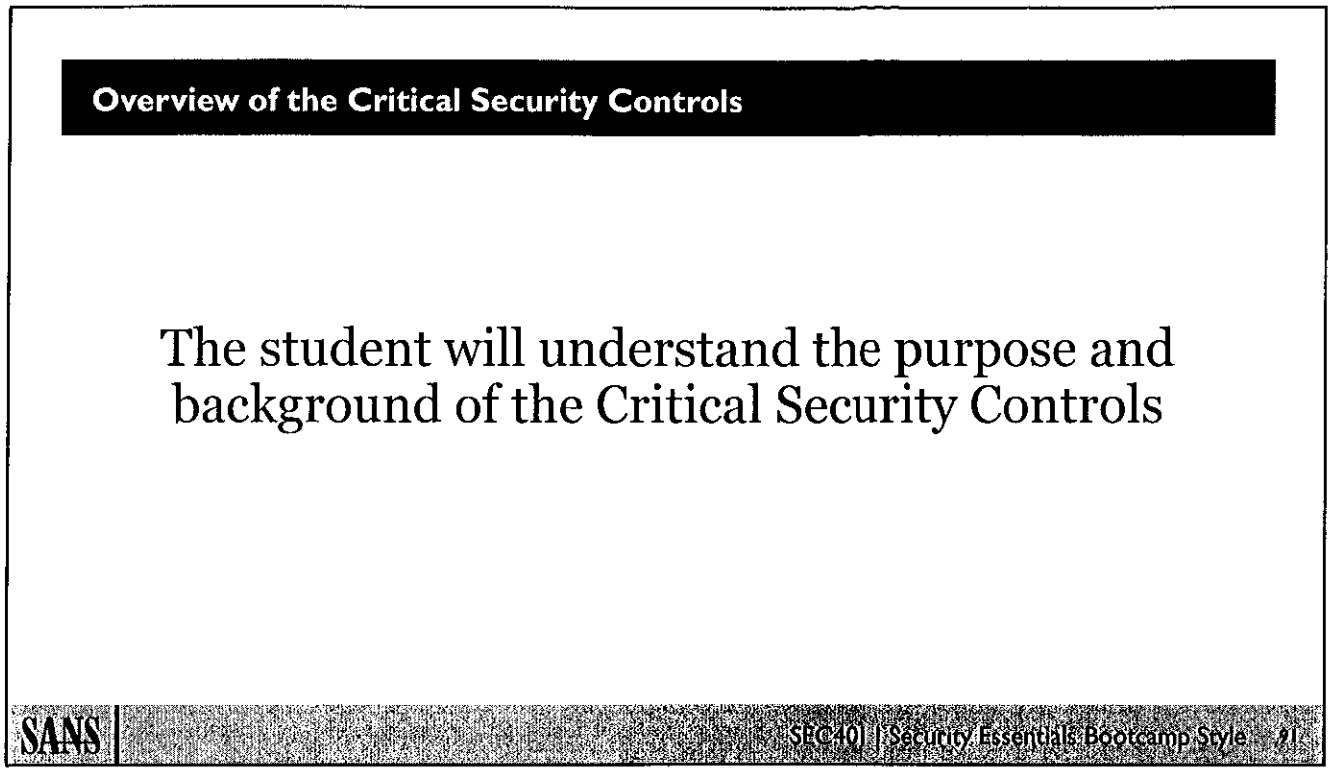
Material utilized from SEC566: Critical Controls by James Tarala

Objectives

- Overview of the Critical Security Controls
- The Critical Security Controls
- Sample Critical Security Control

In implementing security it is important to have a framework with proper metrics. As is often said you cannot manage what you cannot measure. The Critical Security Controls was created to help provided guidance for organizations on prioritizing what are the most critical risks to their organization. In addition to a framework, the Critical Security Controls documentation also contains details to help organizations put together an effective plan for implementation.

"This consensus document of 20 crucial controls is designed to begin the process of establishing that prioritized baseline of information security measures and controls. The consensus effort that has produced this document has identified 20 specific technical security controls that are viewed as effective in blocking currently known high-priority attacks, as well as those attack types expected in the near future. Fifteen of these controls can be monitored, at least in part, automatically and continuously. The consensus effort has also identified a second set of five controls that are essential but that do not appear to be able to be monitored continuously or automatically with current technology and practices. Each of the 20 control areas includes multiple individual sub-controls, each specifying actions an organization can take to help improve its defenses" (http://www.sans.org/critical-security-controls/).



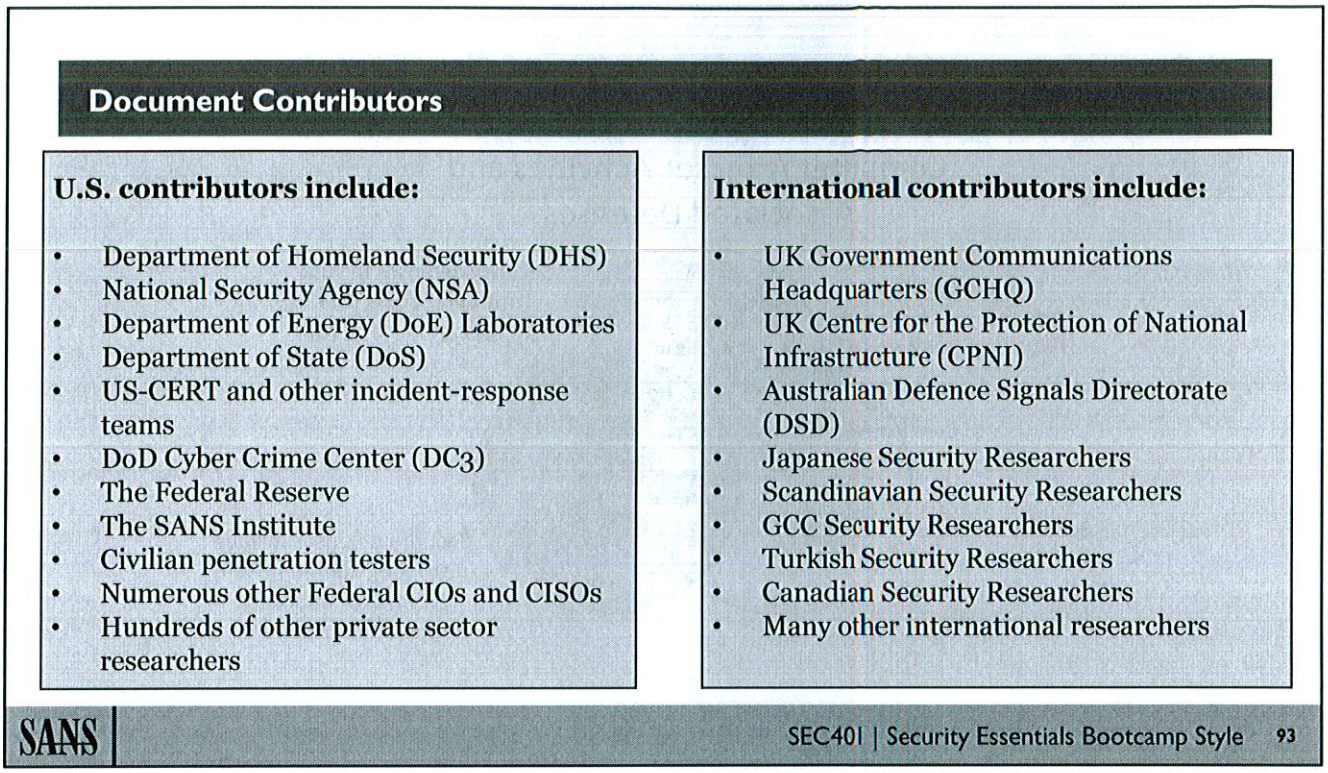
This page intentionally left blank.

What's the Point?

- Government and private-sector organizations are being attacked and compromised daily
- What we're doing today to defend systems is mostly not working
- Advanced vectors are proving that traditional security is not effective
- We need priorities and someone to take a stand and provide the industry with a set of real priorities for defense
- The goals of the Critical Security Controls:
  - Are that those with knowledge of threats and attacks help the groups defending systems as a part of a community risk assessment model to secure systems

Probably the best way to describe this purpose is from the authors of the guidelines themselves, who in version 3.0 of the CSC have the following to say:
"Securing our nation against cyber attacks has become one of the nation's highest priorities. To achieve this objective, networks, systems, and the operations teams that support them must vigorously defend against a variety of threats, both internal and external. Furthermore, for those attacks that are successful, defenses must be capable of detecting, thwarting, and responding to follow-on attacks on internal networks as attackers spread inside a compromised network."

"A central tenet of the US Comprehensive National Cybersecurity Initiative (CNCI) is that 'offense must inform defense.' In other words, knowledge of actual attacks that have compromised systems provides the essential foundation on which to construct effective defenses." (http://www.sans.org/critical-security-controls/).



Document Contributors

One of the best things about these controls is that they were not created in a vacuum. Instead, they are the result of numerous entities working together to provide feedback into the attacks that they are seeing and the controls that they've found, which are helpful in truly combating these threats. The individuals that contributed to the project have experience in fighting the threats directly and have given insight from their experiences.

Some of the groups that helped participate and contributed to this project are

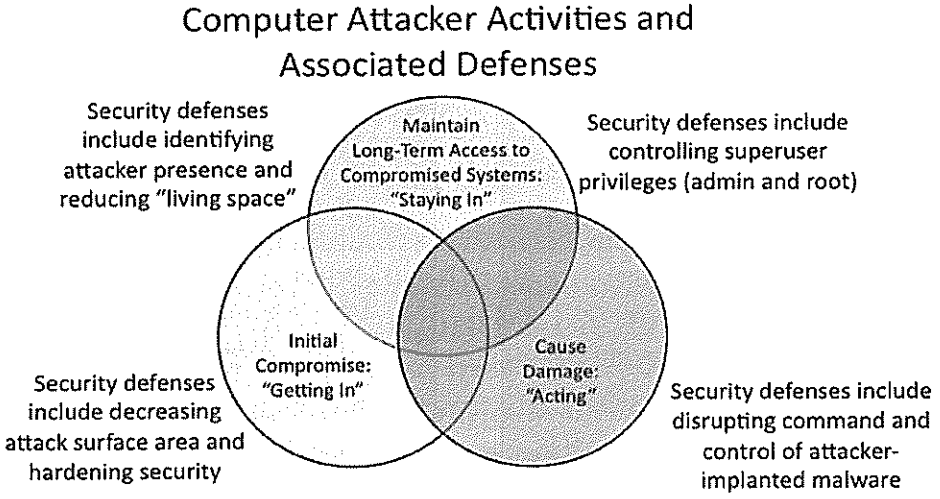
U.S. contributors include

- Department of Homeland Security (DHS)
- National Security Agency (NSA)
- Department of Energy (DoE) Laboratories
- Department of State (DoS)
- US-CERT and other incident-response teams
- DoD Cyber Crime Center (DC3)
- The Federal Reserve
- The SANS Institute
- Civilian penetration testers
- Numerous other Federal CIOs and CISOs
- Hundreds of other private sector researchers

International contributors include

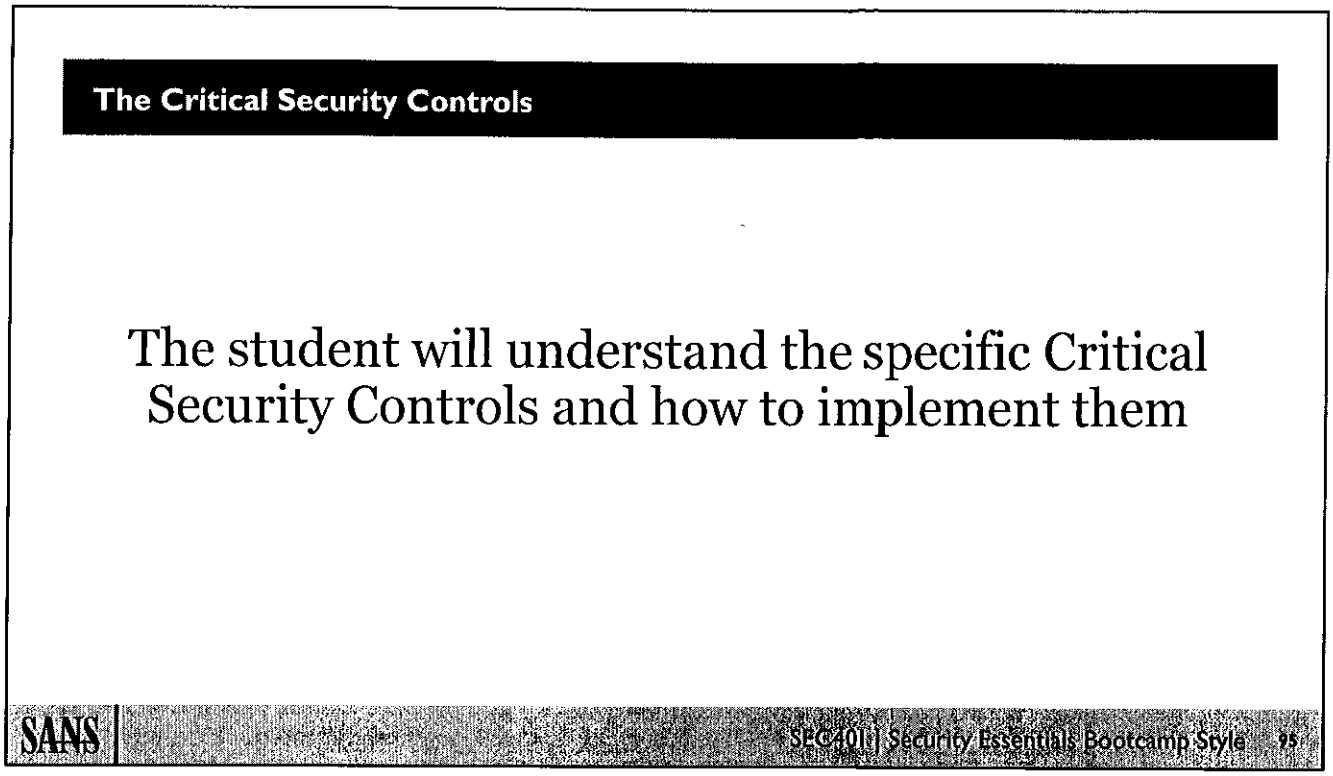
- UK Government Communications Headquarters (GCHQ)
- UK Centre for the Protection of National Infrastructure (CPNI)
- Australian Defence Signals Directorate (DSD)
- Japanese Security Researchers
- Scandinavian Security Researchers
- GCC Security Researchers
- Turkish Security Researchers
- Canadian Security Researchers
- Many other international researchers

Types of Computer Attacker Activities



These controls are not limited to blocking only the initial compromise of systems but also address detecting already compromised machines and preventing or disrupting attacker's actions. The defenses identified through these controls deal with decreasing the initial attack surface through hardening security, identifying already compromised machines to address long-term threats inside an organization's network, controlling superuser privileges on systems, and disrupting attackers' "command and control" of implanted malicious code. This slide illustrates the scope of different kinds of attacker activities these controls are designed to help thwart.

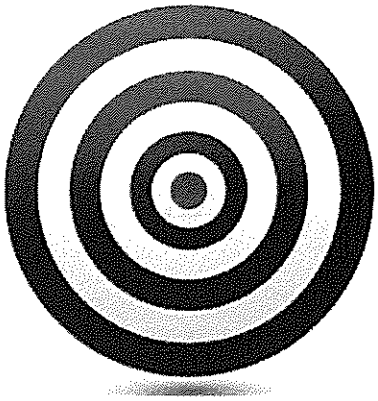
The rings represent the actions computer attackers often take against target machines. These actions include initially compromising a machine to establish a foothold by exploiting one or more vulnerabilities. Attackers can then maintain long-term access on a system, often by creating accounts, subverting existing accounts, or altering the software on the machine to include backdoors and rootkits. Attackers with access to machines can also cause damage, which could include stealing, altering, or destroying information, impairing the system's functionality to jeopardize its business effectiveness or mission, or using it as a jumping-off point for compromise of other systems in the environment. Where these rings overlap, attackers have even more ability to compromise sensitive information or cause damage. Outside of each set of rings in the figure, various defensive strategies are presented, which are covered throughout the controls described in this document. Defenses in any of the rings help to limit the abilities of attackers, but improved defenses are required across all three rings and their intersections. Note that the CSC is designed to help improve defenses across each of these rings, rather than merely preventing initial compromise.



This page intentionally left blank.

Project Guiding Principles (1)

- Defenses should be automated where possible and periodically or continuously measured using automated measurement techniques where feasible
- To address current attacks occurring on a frequent basis against numerous organizations, a variety of specific technical activities should be undertaken to produce a more consistent defense



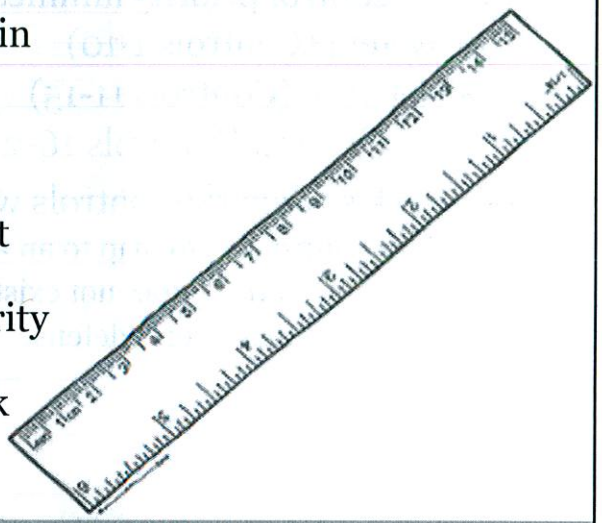
The following two principles are key components of the Critical Security Controls:

Defenses should be automated where possible and periodically or continuously measured using automated measurement techniques where feasible: While having controls is good, having controls that can be automated is even better. If an administrator has to constantly or manually work with the controls, then the likelihood of the controls being successful goes down. By adding the human resource element into the equation, it makes the likelihood of the control being circumvented higher due to neglect, mistakes, or simply lack or organizational resources to follow through on the control.

To address current attacks occurring on a frequent basis against numerous organizations, a variety of specific technical activities should be undertaken to produce a more consistent defense: There is certainly nothing wrong with operational controls. They bring value to an organization and can help the organization to better defend their systems. However, they are not the focus of the CSC. These are a set of technical controls that can help defend systems. There are other models that focus on process and operational tactics; this is not one of them.

Project Guiding Principles (2)

- Root cause problems must be fixed in order to ensure the prevention or timely detection of attacks
- Measures should be established that facilitate common ground for measuring the effectiveness of security measures, providing a common language to communicate about risk



In addition to the two principles mentioned on the previous slides, the following two principles were also developed for the same purpose:

Root cause problems must be fixed in order to ensure the prevention or timely detection of attacks: We are not simply attempting to address surface issues with these controls. We are trying to get to the heart of the issue. More money or more personnel is not always the solution to this problem. There may be other underlying causes that need to be addressed before we start to see success.

Measures should be established that facilitate common ground for measuring the effectiveness of security measures, providing a common language to communicate about risk: This is an area of maturity that many organizations are just starting to consider. Many organizations have heard the call for developing measures, but actually developing information assurance related measures may be well into the future. Organizations need to determine, whether they follow the Critical Security Controls or not, which set of measures they plan on following to measure their effectiveness.

Understanding the Controls

- Three control priority families:
  - System (Controls 1-10)
  - Network (Controls 11-15)
  - Application (Controls 16-20)
- Key rules when the controls were chosen:
  - Each control has to map to an actual known attack
  - If a known attack does not exist, it cannot be a control
  - "Offense must inform defense"

SANS | SEC401 | Security Essentials Bootcamp Style 98

The reality is this: Every day, defending information systems is becoming more and more complex as technologies expand, along with new features and systems being added to our networks at a growing rate. It seems like no matter how much money is spent on cyber defense, each year, the problem keeps getting worse. Even for those organizations that spend a larger and larger portion of their budgets on controls to protect their information systems, it seems like it is a losing battle against those trying to compromise their data.

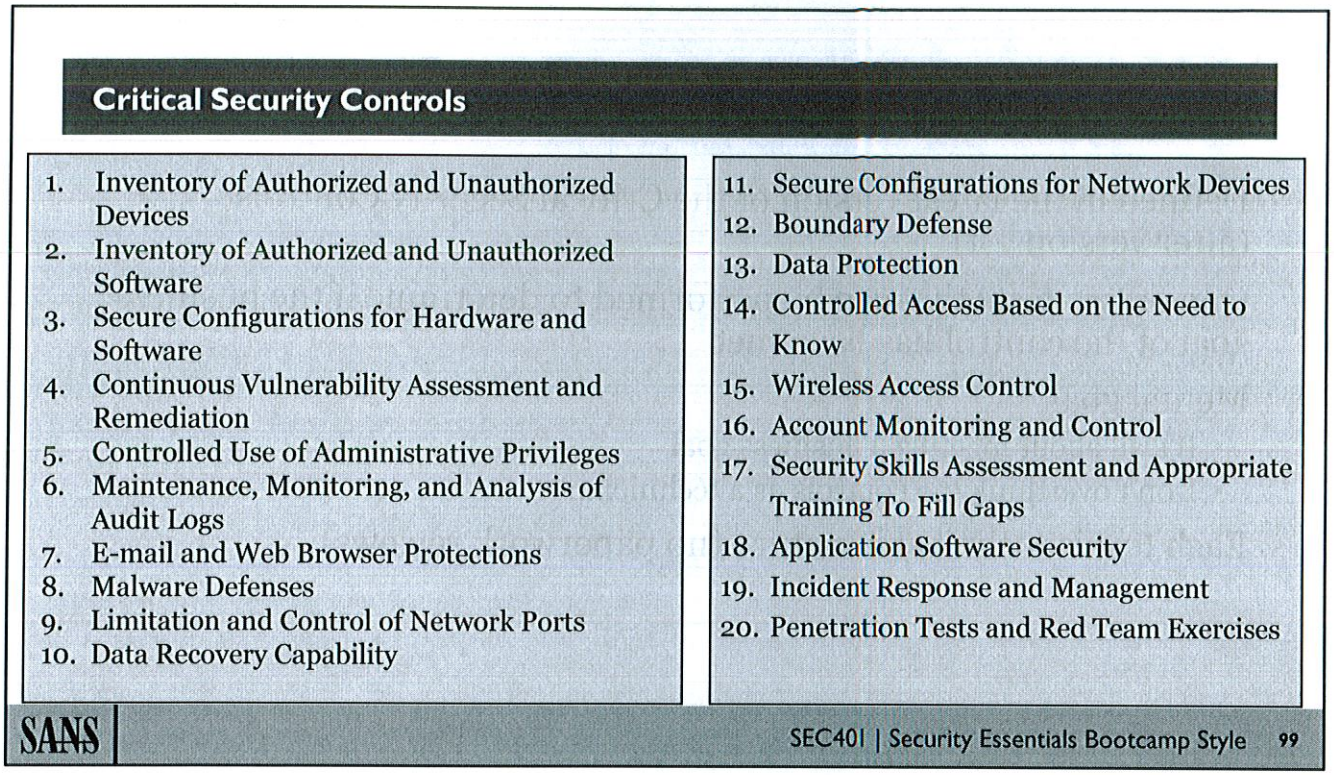
As a result, it seems like there is a real need that has been identified by CIOs and CISOs: They need a prioritized set of controls and associated measures that give them the biggest return on their security investment and give them a fighting chance at defending their systems. The reality of the issue is that rarely do more controls hurt the situation, but there are so many possible controls these days that it is difficult to know which controls make the most sense and returns the biggest gain for the investment. How do we know which controls to start with when defending our systems?

It is absolutely critical that we have priorities!

Note that there is a distinction between some controls. Most of the controls were meant to be automated. These defenses should be automated defenses that do not require constant attention on the part of system administrators. Although any system requires care and feeding, these defensive systems should not require manual efforts on the part of administrators in order to function. While other controls cannot be automated, those are labeled with (validated manually).

As said before, a few key rules were established when creating these controls:

1. Each control has to map to an actual known attack.
2. If a known attack does not exist, it cannot be a control.
3. "Offense must inform defense."



What actually are the Critical Security Controls?

The controls are as follows:

1. Inventory of Authorized and Unauthorized Devices
2. Inventory of Authorized and Unauthorized Software
3. Secure Configurations for Hardware and Software
4. Continuous Vulnerability Assessment and Remediation
5. Controlled Use of Administrative Privileges
6. Maintenance, Monitoring, and Analysis of Audit Logs
7. E-mail and Web Browser Protections
8. Malware Defenses
9. Limitation and Control of Network Ports
10. Data Recovery Capability
11. Secure Configurations for Network Devices
12. Boundary Defense
13. Data Protection
14. Controlled Access Based on the Need to Know
15. Wireless Access Control
16. Account Monitoring and Control
17. Security Skills Assessment and Appropriate Training To Fill Gaps
18. Application Software Security
19. Incident Response and Management
20. Penetration Tests and Red Team Exercises

Core Evaluation Tests (1)

Defined in the latest version of the Critical Security Controls (after version 2.1)

One or two tests that can be performed to determine if the business goal of the control has been met

Mental goal

- It's all about meeting a business goal
- Don't overthink the controls as a technician

Each test is technical in nature (no paperwork reviews)

In order to evaluate yourself as an organization, the writers of the Critical Security Controls had the opportunity to write core evaluation tests and measures that could be used to determine whether or not an organization meets the goals of the controls. These tests and measures have been added to the 20 Critical Control documentation itself in every version of the controls after version 2.1.

The purpose was to identify one or two tests that could be performed for each of the critical controls, which would help the evaluation team determine whether or not the defined business goal of the control has been met.

Note that each of these evaluations are meant to be technical reviews, not operational or paperwork reviews. It does us no good as an organization if we can accomplish something on paper but can't do it in reality. Therefore, each of the tests has been designed as a technical evaluation to see if an organization can actually accomplish a goal–not just perform it on paper.

Core Evaluation Tests (2)

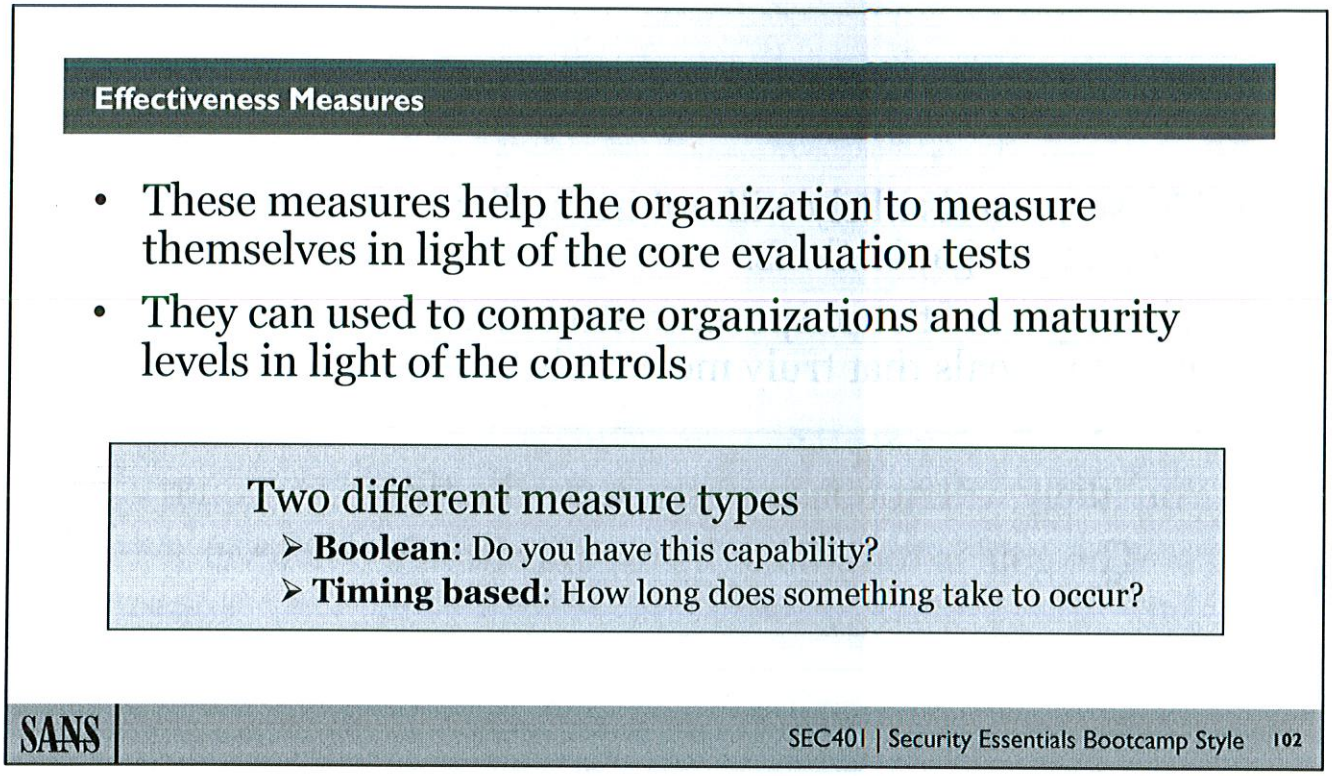
- These controls likely will not immediately be reachable for many organizations.
- That's part of the purpose: to stretch organizations into meeting goals that truly meet their business needs.

You likely will not pass all the tests the first time you try
The goal is to reduce the organization's risk level

Especially for organizations just getting started with these controls, there will be a tendency for people to read the tests and consider them unreachable or too difficult to perform. That's to be expected. These controls likely will not immediately be reachable for many organizations. That's part of the purpose: to stretch organizations into meeting goals that truly meet their business needs.

The mentality you need to have when reading these controls is that these are business goals that should be achieved. As a technician, you might think, "There's no way I can identify a new machine being added to my network within 2 minutes!," but, as a business leader, you should be thinking, "Why not?" If we are to truly protect our systems, then we need to establish goals from a business point of view that help us protect our data and not always be hampered by technology limitations. If the tools we have today don't solve our problems, then maybe we're using the wrong tools.

You should know that a process has already been started to work with vendors to help them prioritize their development time and make sure they're building tools to meet the needs of businesses, not simply what they think will sell. More will be coming out in the news soon about these efforts, so stay posted!



As a result of each of the core evaluation tests that are performed, there should be specific measures that are collected as an outcome of the core evaluation tests. These measures help the organization measure themselves in light of the previous tests. In addition, organizations will find that they will be able to use these measures to compare themselves to other organizations and to compare an organization's maturity level to other similar organizations based on the measure outcomes of these controls.

For each of the controls that have been defined, there are two types of measures that have been included in these controls: Boolean (yes/no) questions and timing-based questions. The Boolean measures help identify if an organization has the capacity to perform a particular task, as defined by one of the controls. The timing-based questions are meant to determine how long it takes for the organization to accomplish a particular goal.

Think of these measures, especially the timing-based measures, as similar to physical activities you might perform. For example, if you run a race, you will most often be judged by how long it takes you to finish the race. If you run a marathon, you are able to compare how long it takes you to run the 26.2 miles today with the length of time it took you to run the same distance in the past. In addition, you are able to compare your running time with the time it takes other runners to complete the same distance. As a result, you are able to measure yourself in the process.

The Critical Security Controls are meant to be measured in the same light. The lower your measure numbers, the more capable you are at meeting the business goals that the controls themselves represent.

Sample Critical Security Control

The student will understand the details provided for a given control and how to implement a sample Critical Security Control

This page intentionally left blank.

Sample Control: Critical Control #2

Inventory of Authorized and Unauthorized Software

Brief explanation

"Once a single machine has been exploited, attackers often use it as a staging point for collecting sensitive information from the compromised system and from other systems connected to it. In addition, compromised machines are used as a launching point for movement throughout the network and partnering networks. In this way, attackers may quickly turn one compromised machine into many. Organizations that do not have complete software inventories are unable to find systems running vulnerable or malicious software to mitigate problems or root out attackers (CSC 6.0)."

Business goal of this control

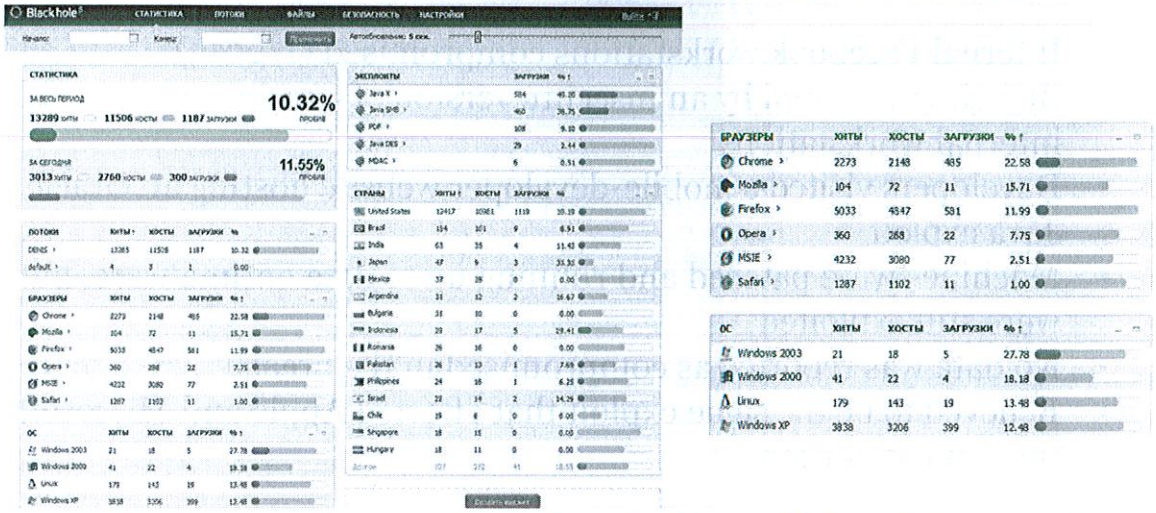
- Only authorized software should be installed on the agency's computer systems

SANS | SEC401 | Security Essentials Bootcamp Style 104

A business unit was operating normally, or so it seemed (its customers accessing online catalogs and placing orders), when suddenly, catalogs continued to be accessed but no more orders were coming in. Unaware of what has caused the anomaly, administrators are called upon to check the system and ensure that the ordering process has not been disabled. The administrators run a systems diagnostic and determine all computers and applications appear to be functioning normally. A vulnerability scan is requested, along with a traffic analysis (unfortunately, one had not been running) and it is determined that, indeed, catalog systems continue to be accessed regularly, but an unknown application is identified on the ordering system, which has what appears to be tunneled traffic destined for a site across the world, and not the company's.

It appears that the system had been compromised and a proxying application installed. The proxy application was rerouting traffic destined for the ordering application to an overseas site. The original ports associated with the application had been reassigned to the proxy service, and the orders were now being sent to another international company. After further analysis and a major loss in revenue, it was determined the systems had not maintained a proper inventory of applications installed, numerous applications installed had significant vulnerabilities, and most installed applications were not even necessary for operations.

Sample Attack Tool: Blackhole Exploit Toolkit



Screen capture from http://krebsonsecurity.com/2010/10/java-a-gift-to-exploit-pack-makers/

The Blackhole Exploit Toolkit (BET) is a toolkit most commonly installed on web servers and preconfigured to attack out-of-date code running on the machines of visitors to an infected website. Basically, an attacker would follow these steps with the BET:

1. Infect a website through a common flaw, such as SQL Injection.
2. Upload his packaged code created by the BET.
3. Wait for visitors to visit the infected site.
4. Visitors who visit the site run the hosted malicious code.
5. The payload defined in the packaged BET code executes on the client machine.

This process is possible because the tool targets out of date or unmanaged software on victim machines. If a client, on the other hand, knows what software is on their machine, properly configures and patches it, then he would not be vulnerable to exploits published via this tool.

Breach Case Study: Facebook

facebook

- Internal Facebook workstations compromised (1/2013)
- Breach was caused by an insecure version of Oracle Java running on internal workstations
- Developers visited a mobile-developer website hosting an Oracle Java exploit
- Machines were patched and running up to data antimalware, but were still exploited
- No data was reported as compromised in the breach
- Believed to be the same exploit that affected Apple and Microsoft in the same timeframe

For each of the Critical Security Controls, we have identified one organization that illustrates why other organizations should implement this control. It is generally not the case that this company engaged in gross negligence or did not know the right thing to do. Information security is most often about available resources, discipline, and a lot of luck. Rather than be embarrassed, representatives from this organization can be happy to know that their example can serve as a warning to other so, hopefully, it does not happen to others. Information security is a community and should act as such and seek to help others in their efforts.

Rather than try to tell the story of this breach in our own words, in order to stay unbiased and avoid unnecessary commentary, we use the words of the reporters and blog entries directly with knowledge of the case. The full story of the breach can be found at https://m.facebook.com/note.php?note_id=10151249208250766.

Critical Control #2 Defenses (1)

1. **Devise a list of authorized software** and version that is required in the enterprise for each type of system, including servers, workstations, and laptops of various kinds and uses. This list **should be monitored by file-integrity checking tools** to validate that the authorized software has not been modified.
2. **Deploy application whitelisting** technology that allows systems to run software only if it is included on the whitelist and **prevents execution of all other software** on the system. The whitelist may be extensive (as is available from commercial whitelist vendors), so that users are not inconvenienced when using common software. Or, for some special-purpose systems (which require only a small number of programs to achieve their needed business functionality), the whitelist may be narrow.

In addition to understanding what physical devices are attached to an organization's network, an organization must document what software they are attempting to defend. The first step in that process is simply to document what software the organization has authorized. Without an understanding of what software is supposed to be present, how can the organization defend that software and block the things not on the list? A software inventory is the first step.

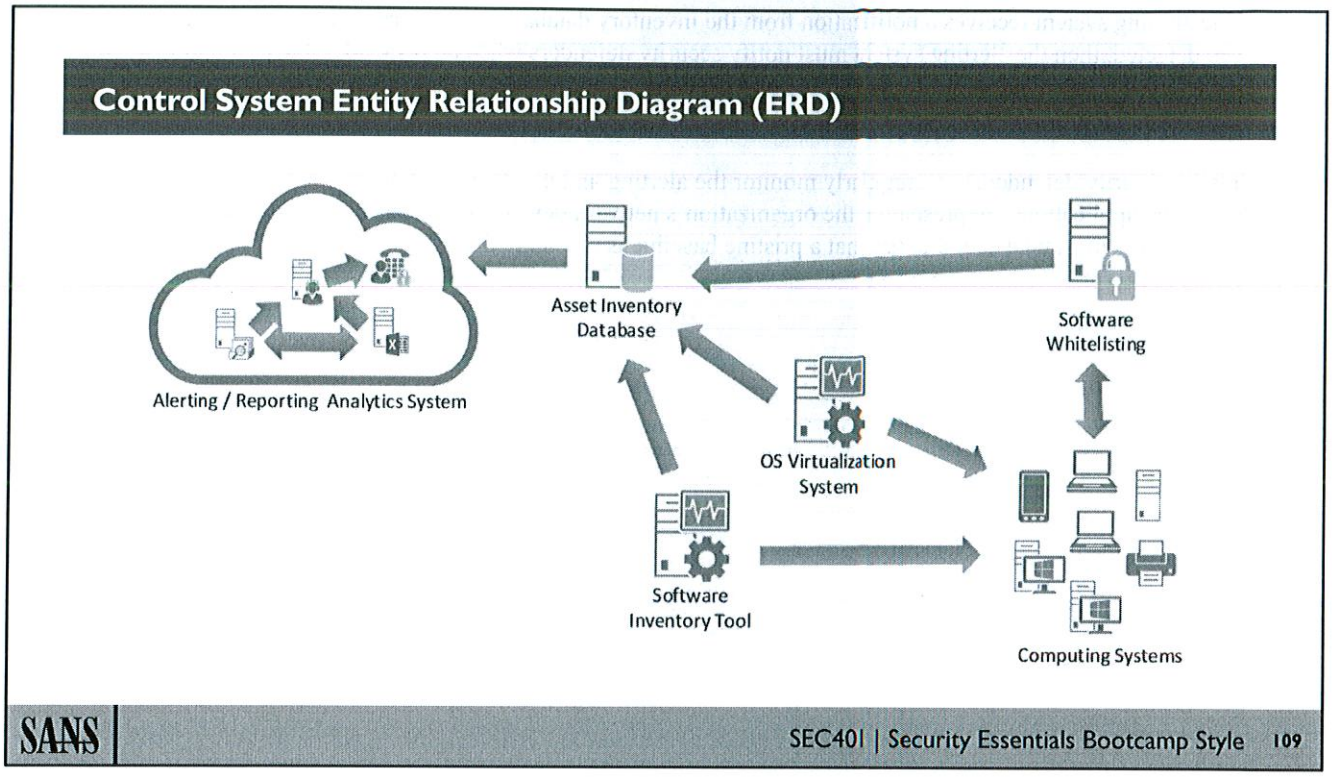
Most importantly, however, is the second principle: Once an authorized software list has been agreed upon, only that software should be allowed to execute. This principle cannot simply be enforced via policy statements and human resource principle. Instead, the organization must deploy technical solutions to ensure that only authorized applications are allowed to execute. Not only does this ensure that only authorized applications are used, but also blocks malicious applications that may attempt to execute.

Critical Control #2 Defenses (2)

3. **Deploy software-inventory tools** throughout the organization, covering each of the operating system types in use, including servers, workstations, and laptops. The software inventory system should track the version of the underlying **operating system** as well as the **applications installed** on it. The software inventory systems must be **tied into the hardware-asset inventory** so that all devices and associated software are tracked from a single location.
4. **Virtual machines and/or air-gapped systems** should be used to isolate and run applications that are required for business operations but based on higher risk should not be installed within a networked environment.

Not only should whitelisting solutions give the organization the capability to block unauthorized applications, but they should give an organization the capability to inventory what software is currently installed on a system. This inventory capability can help to ensure that the organization's understanding of what software is installed is in line with reality. This can also help to ensure that the whitelisting application is aware of new applications should be allowed or where malicious software is attempting to execute.

However, it is understood that there may be situations where legacy applications must be allowed for a temporary period of time even though they cannot be defended. Therefore, in situations where these applications must be allowed to execute as an exception, additional countermeasures must be deployed. Specifically, the use of virtual machines (in an isolated state) or air-gapped systems should be used to isolate these indefensible applications away from other production systems.



In this slide, we see a diagram known as an Entity Relationship Diagram (ERD), which can be helpful as we are trying to analyze the entities that must be used to implement and meet the goals of this particular control. Organizations may find that by diagramming the entities necessary to fully meet the goals defined in this control, it is easier to identify how to test the controls and identify where potential failures in the system might occur.

A control system is a device or set of devices to manage, command, direct or regulate the behavior of other devices or systems. In this case, we are examining software installed on the organization's network systems. These systems should be able to identify if new software is introduced to the environment that has not been authorized by enterprise personnel. The following are components in the slide's diagram and show how each entity works together to meet the business goal defined in this control. It also helps us identify what each process step is in an effort to help us identify potential failure points in the overall control.

The active device scanner has the responsibility of examining the software that is installed on a network device in order to determine if only authorized software is currently running on the system. Only authorized software should be present on a system and allowed to run. This active scanning system can help administrators by indicating which software is currently on a network device.

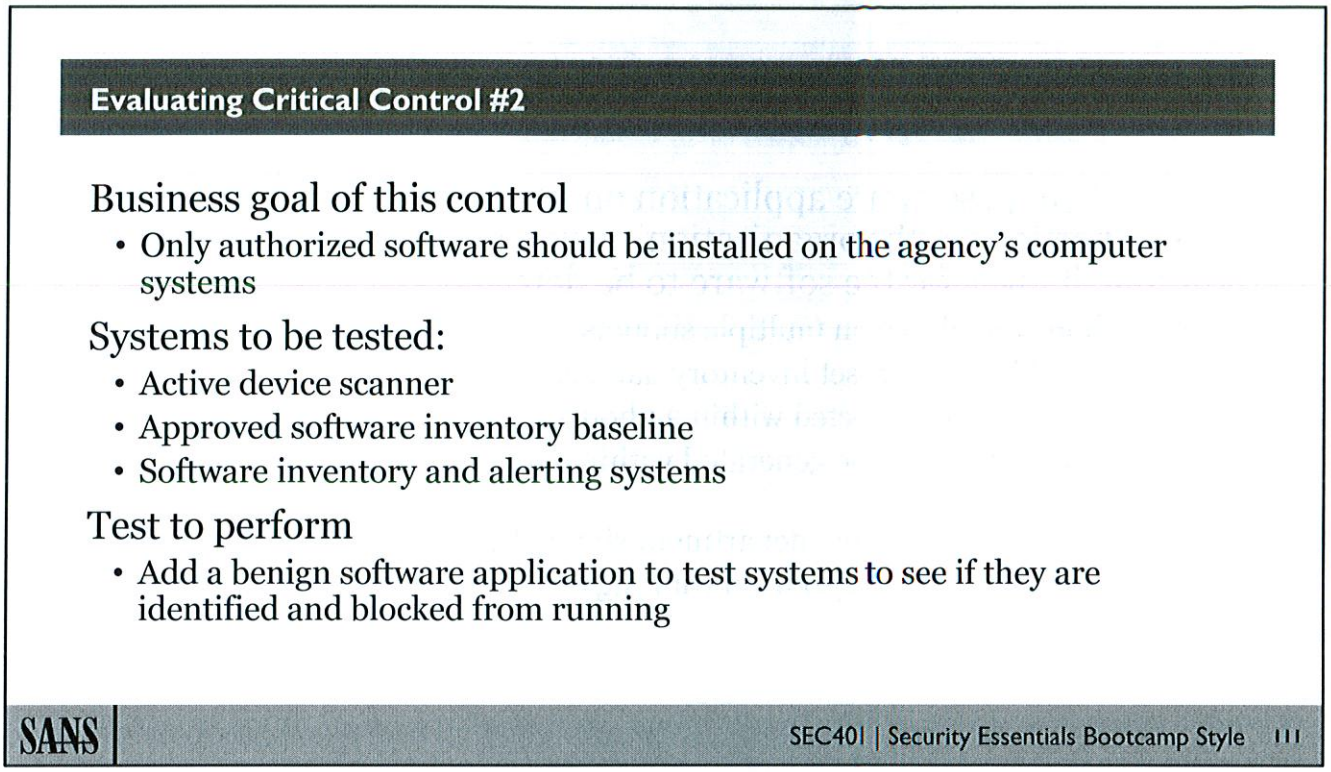
The active scanner must have the capability to report the list of software on a system to an inventory database, which records the inventory and the baseline of known good applications that should be installed on a given system.

In order to determine if there is consistency or not between the two lists, the inventory system has the responsibility to compare the list of running applications detected on a system to the authorized baseline of applications installed on a system. The inventory system should be the authoritative source of the authorized software baseline and be able to determine if the actual software list is different than the authorized baseline.

If there is a discrepancy between what should be installed on a given system and what is actually installed on a given system, then the inventory database must have the ability to initiate an alert to the alerting system. The two systems must be communicating and be able to exchange this information.

If the alerting system receives a notification from the inventory database system that unauthorized software has been detected, then the alerting system must notify security defenders who can respond to the new application. Defenders can either authorize the new software via change management or they can remove the application from the system to return it to the original baseline.

Finally, security defenders must regularly monitor the alerting and inventory system to ensure that only authorized applications are present on the organization's network devices. If there is additional software present, the defenders must take action to see that a pristine baseline is maintained.

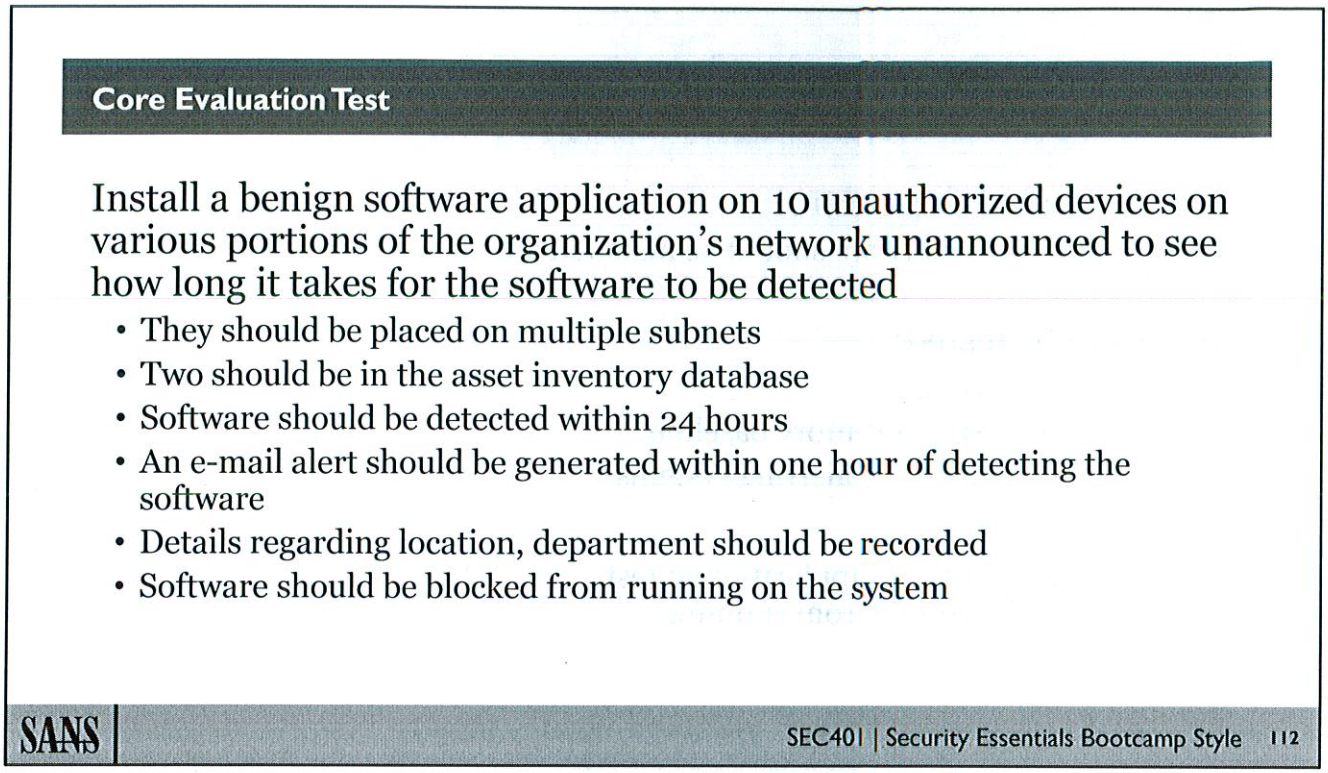


In this case, the goal is to identify software that has not been authorized on a given network device. The business goal is that only authorized software should be installed on a computer system. If there is unauthorized software, it needs to be detected and then removed or authorized on the system.

In order to meet this business goal, there are three systems that have been identified, which must function together:

- Active device scanner
- Approved software inventory baseline
- Software inventory and alerting system

To test these systems, an evaluation team must add a benign software application to test systems to see if they are identified and blocked from running. If the evaluation team has the capability to run software on a system that has not been authorized, then an alert should be generated and the software should be blocked from running. If this does not happen, then there is a breakdown somewhere in these systems and further investigation is needed to determine the root cause of the failed test.



For this control, the organization's evaluation team must install a benign software application on ten unauthorized devices on various portions of the organization's network unannounced to see how long it takes for the software to be detected.

A number of applications could be used for this test. Whatever application is chosen, however, it must not create a vulnerability on the system where it is installed or be used against the network or system later. Some tools, which are often labeled as "hacker tools" by antimalware vendors, are often times good choices for this assessment, as long as the tools will not likely be used to create a breach later. For standalone binaries, the ENUM utility is a fairly old application, which is labeled as a hacker tool but not likely to cause a vulnerability later. In addition, for a fully installed application, the evaluation team should find an application that is rarely used, updated infrequently, and unlikely to have a vulnerability associated with it in the near future (however at the end of the test, this application should always be uninstalled).

Similar to the previous control, the following testing criteria should be followed when performing this evaluation:

- They should be placed on multiple subnets
- Two should be in the asset inventory database
- Software should be detected within 24 hours
- An e-mail alert should be generated within one hour of detection
- Details regarding location and department should be recorded
- Software should be blocked from running on the system

Critical Control #2 Measures

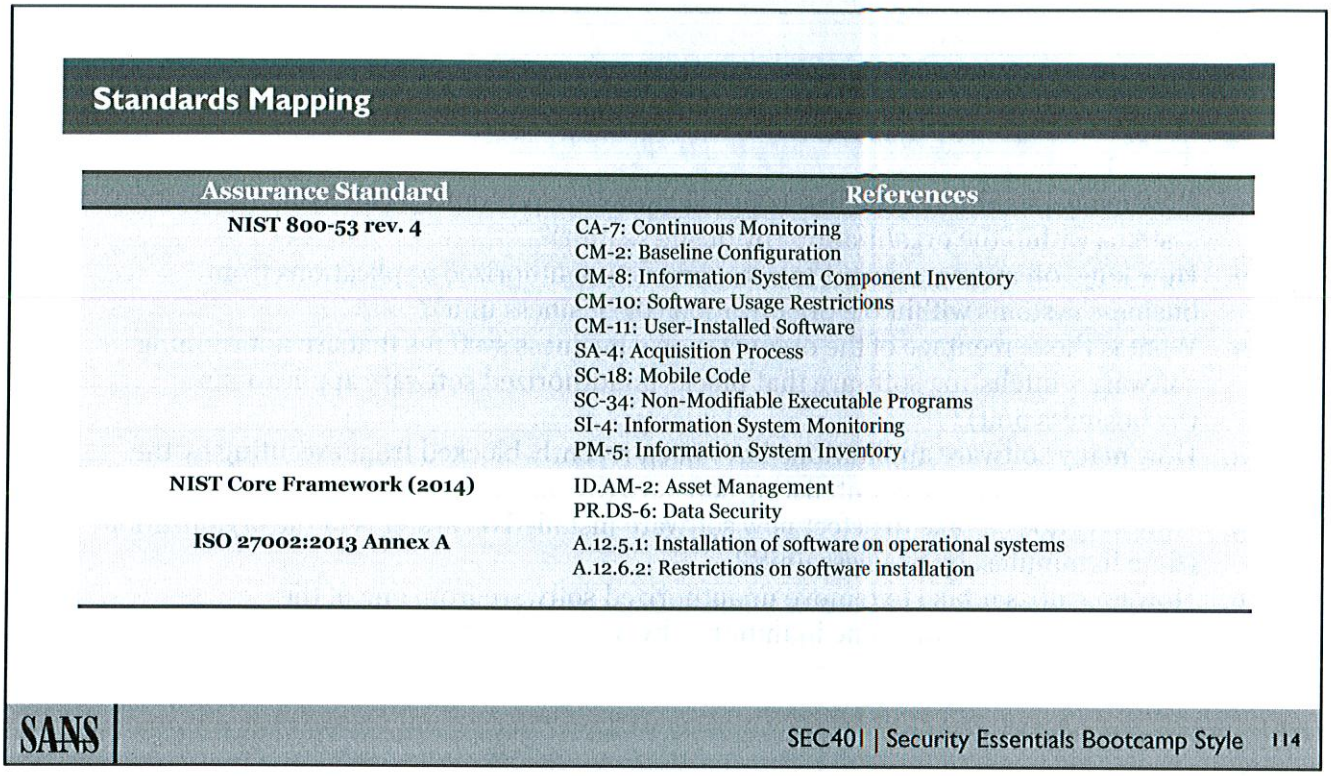
1. How many unauthorized software applications are presently located on business systems within the organization (by business unit)?
2. How long, on average, does it take to remove unauthorized applications from business systems within the organization (by business unit)?
3. What is the percentage of the organization's business systems that are not running software whitelisting software that blocks unauthorized software applications (by business unit)?
4. How many software applications have been recently blocked from executing by the organization's software whitelisting software (by business unit)?
5. How long does it take to detect new software installed on systems in the organization (time in minutes, by business unit)?
6. How long does it take to remove unauthorized software from one of the organization's systems (time in minutes, by business unit)?

The measures for this control focuses on discovering new software applications on systems that are unauthorized.

For this Critical Security Control, we have identified not only effectiveness measures, but also measures that we believe you can use to automatically collect information from your security sensors to give you a better perspective on the risk your organization faces. Starting in version 5 of the Critical Security Controls, we introduced these automation measures as a way for organizations to begin the process of automatically collecting data from security sensors in order to create a data-centric view of the risk facing their organizations.

Specifically for this control, we have identified the following automation measures that we believe organizations should consider implementing as a part of an initial measures program:

1. How many unauthorized software applications are presently located on business systems within the organization (by business unit)?
2. How long, on average, does it take to remove unauthorized applications from business systems within the organization (by business unit)?
3. What is the percentage of the organization's business systems that are not running software whitelisting software that blocks unauthorized software applications (by business unit)?
4. How many software applications have been recently blocked from executing by the organization's software whitelisting software (by business unit)?
5. How long does it take to detect new software installed on systems in the organization (time in minutes, by business unit)?
6. How long does it take to remove unauthorized software from one of the organization's systems (time in minutes, by business unit)?



The Critical Security Controls are complimentary to the NIST controls and maps to core areas in the documentation. The following is the complete mapping to the NIST controls:

CA-7: Continuous Monitoring
CM-2: Baseline Configuration
CM-8: Information System Component Inventory
CM-10: Software Usage Restrictions
CM-11: User-Installed Software
SA-4: Acquisition Process
SC-18: Mobile Code
SC-34: Non-Modifiable Executable Programs
SI-4: Information System Monitoring
PM-5: Information System Inventory

In addition to being mapped to the NIST 800-53 controls, the Critical Security Controls are also easily mapped to other information assurance standards. While mappings to other standards are rarely ever a one-to-one match, clearly, the ideas expressed in these guides often represent common ideas. Specifically, the Critical Security Controls also map to the following sections of the new NIST Core Framework, finalized in 2014:

ID.AM-2: Asset Management
PR.DS-6: Data Security

The Critical Security Controls also map to other standards, such as ISO 27002:2013. The following are the overlaps between this control and the ISO documentation:

A.12.5.1: Installation of software on operational systems
A.12.6.2: Restrictions on software installation

Summary

- Defenses should focus on addressing the most common and damaging attack activities occurring today and those anticipated in the near future
- Enterprise environments must ensure consistent controls across an enterprise to effectively negate attacks

The Critical Security Controls creates the framework for meeting these challenges

In the course of developing the Critical Security Controls (CSC), a number of guiding principles were also developed to guide the project and give direction to the writing and implementation of each of the controls. A couple of these guiding principles are

Defenses should focus on addressing the most common and damaging attack activities occurring today and those anticipated in the near future: In other words, the defensive mechanisms that are implemented today should be based on actual attacks that have been seen in the field during incident response activities. They should not be academic, but real attacks that have been seen and the controls that are developed should be based on tactics that can stop these attacks from being successful.

Enterprise environments must ensure consistent controls across an enterprise to effectively negate attacks: In addition to implementing these controls, they need to be implemented in a consistent manner across the enterprise. If they are not implemented consistently across the enterprise, then the organization is opening the door for risk. It is equivalent to building a house and putting locks on "almost" all of the exterior doors. The thought was good, but the home is still left partially unprotected.

Security 401: Day 2

SANS

Module 11: Malicious Code and Exploit Mitigation

Module 11: Malicious Code and Exploit Mitigation
This page intentionally left blank.

Objectives

- Mitnick-Shimomura
- Defensive strategies
- Common types of attacks

SANS | SEC401 | Security Essentials Bootcamp Style 117

The key to the future is to understand the past. The Mitnick attack is historic and can teach us a lot about modern-day security and provides us with key lessons learned. Just like in school you are required to take history classes, there can be a lot to learn from historic attacks.

It was Christmas 1994 when Kevin Mitnick executed his now-famous attack against security researcher Tsutomu Shimomura's home network. Using just a few basic attack strategies, Mitnick was able to affect a root-level compromise on systems owned by a skilled information security professional. The sad fact is **that many systems still remain vulnerable to these kinds of attacks; thus, this incident isn't so much ancient history as it is a modern case study**. In this module, we describe the attack in detail, discussing not only the conditions that made it possible but also some strategies that you can use to help manage the risks associated with these sorts of attacks.

Focusing on individual exploits used in real-world situations is useful, but it doesn't give you a very good look at the bigger risk landscape. The techniques Mitnick used certainly are not the only ones available to knowledgeable attackers. There are almost as many ways to abuse a system as there are ways to use it legitimately.

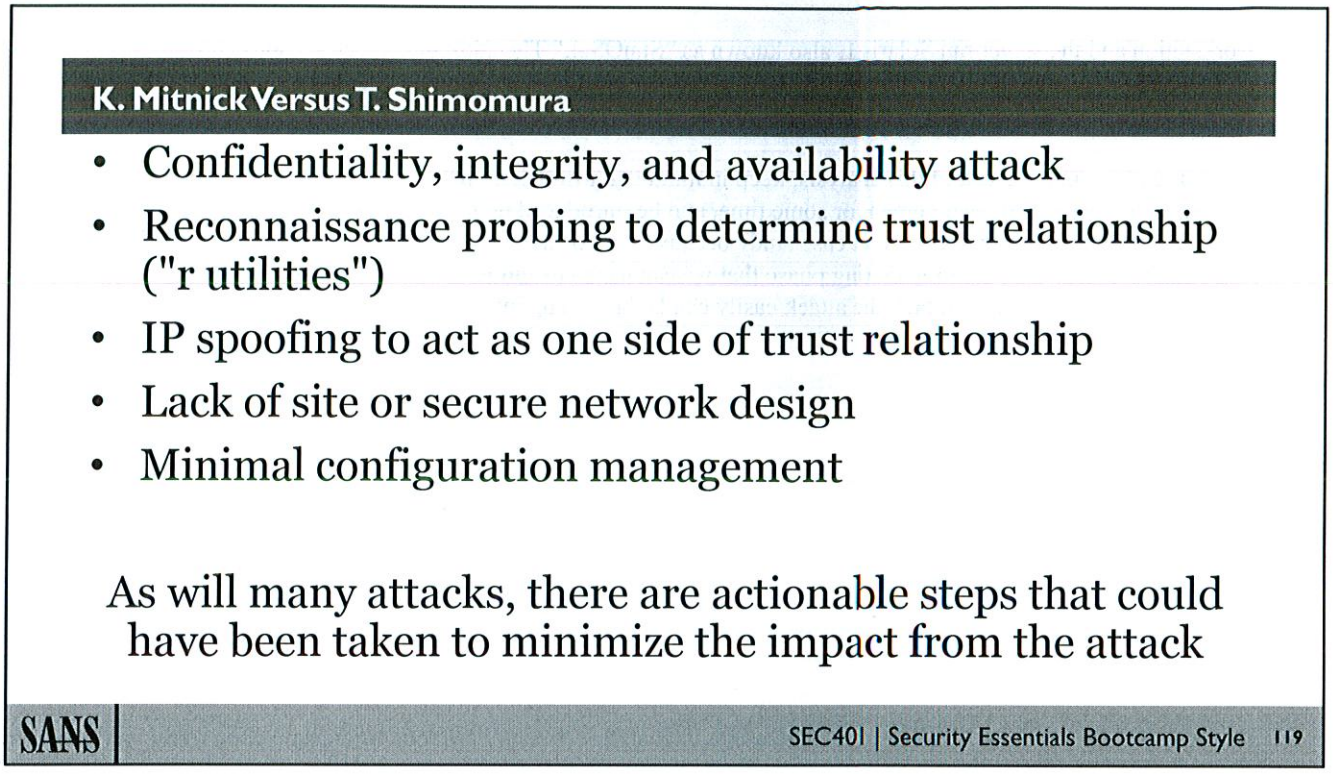
References

1) Cyber Attacks: 5 Ways Small Businesses Can Protect Themselves - http://www.forbes.com/sites/franksorrentino/2015/10/26/cyber-attacks-5-ways-small-businesses-can-protect-themselves/

Mitnick-Shimomura

The student will understand the details of the Mitnick-Shimomura attack, as well as what we can learn from this attack to appropriately protect our networks against these threats

This page intentionally left blank.



One of the reasons Kevin Mitnick's famous attack on Shimomura's network is a good example for analysis is its scope. The three major tenets of information security—the C-I-A triad of confidentiality, integrity, and availability—were all breached in this attack. By accessing files that were not his, Mitnick compromised the confidentiality of those files, and by penetrating network resources to which he was not granted explicit access, he compromised the integrity of that network. Finally, by executing a SYN flood as a denial of service against one of Shimomura's servers, Mitnick effectively rendered that machine unavailable. Another reason this scenario is an excellent demonstration is the step-by-step execution of Mitnick's attack. First, he scouted the resources he wanted to compromise using the Finger utility. From this, he determined his method of compromise, the Unix r utilities. Then, he used a denial of service to silence a trusted machine and imitated it by way of IP spoofing to gain access to the desired computer. From this system, Mitnick could obtain the files he wanted. Overall, the attack demonstrates the importance of perimeter defenses in planning an effective prevention strategy.

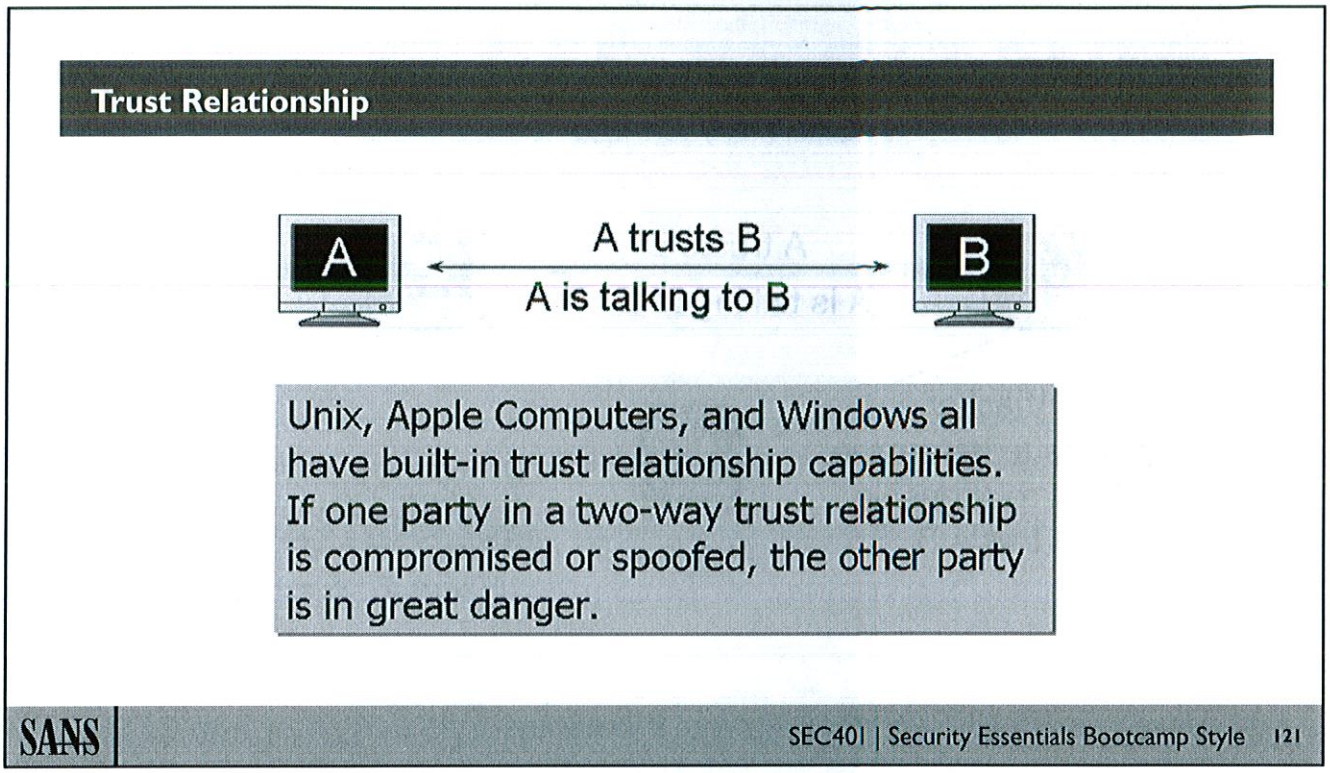
Shimomura, a respected computer security researcher, was relaxing and spending the Christmas holiday at the home of a friend in San Francisco. The home, known affectionately as Toad Hall, was a social experiment of sorts, exploring what it meant to live online all the time.

There was a substantial amount of computing equipment scattered around the various rooms, all networked and connected via a permanent link to the Internet. It even had its own domain name, toad.com. It might have been sheer coincidence, but at the very time Shimomura was relaxing upstairs, the servers in the basement downstairs were being used as a staging point to launch an attack on his own network, some 500 miles to the south.

According to a posting Shimomura made to a firewall mailing list the following month, one of the servers at Toad Hall had been compromised. The attacker, whose identity was unknown at the time, used the server to send a series of probes to Shimomura's home LAN in San Diego. Shimomura's network consisted of a workstation, a server, and another machine. The workstation, diskless, served as an X terminal. Both the

workstation and the server ran Solaris 1, also known as "SunOS 4." The additional machine housed Mitnick's ultimate target—some cellular telephone software Shimomura had reverse-engineered. For simplicity's sake, we refer to these three systems simply as the terminal, the server, and the target.

As you read through the rest of this analysis, keep in mind that Mitnick didn't invent this attack himself. The techniques involved had been known for some time, and he introduced no new innovations, except to perform it in the real world. Still, it makes a great case study because it is so easily understood. We particularly like the fact that there's a clear information-gathering phase that we can use to examine how clever intruders can map out trust relationships to abuse. In fact, the attack easily can be broken up into several clear phases, which we now examine.



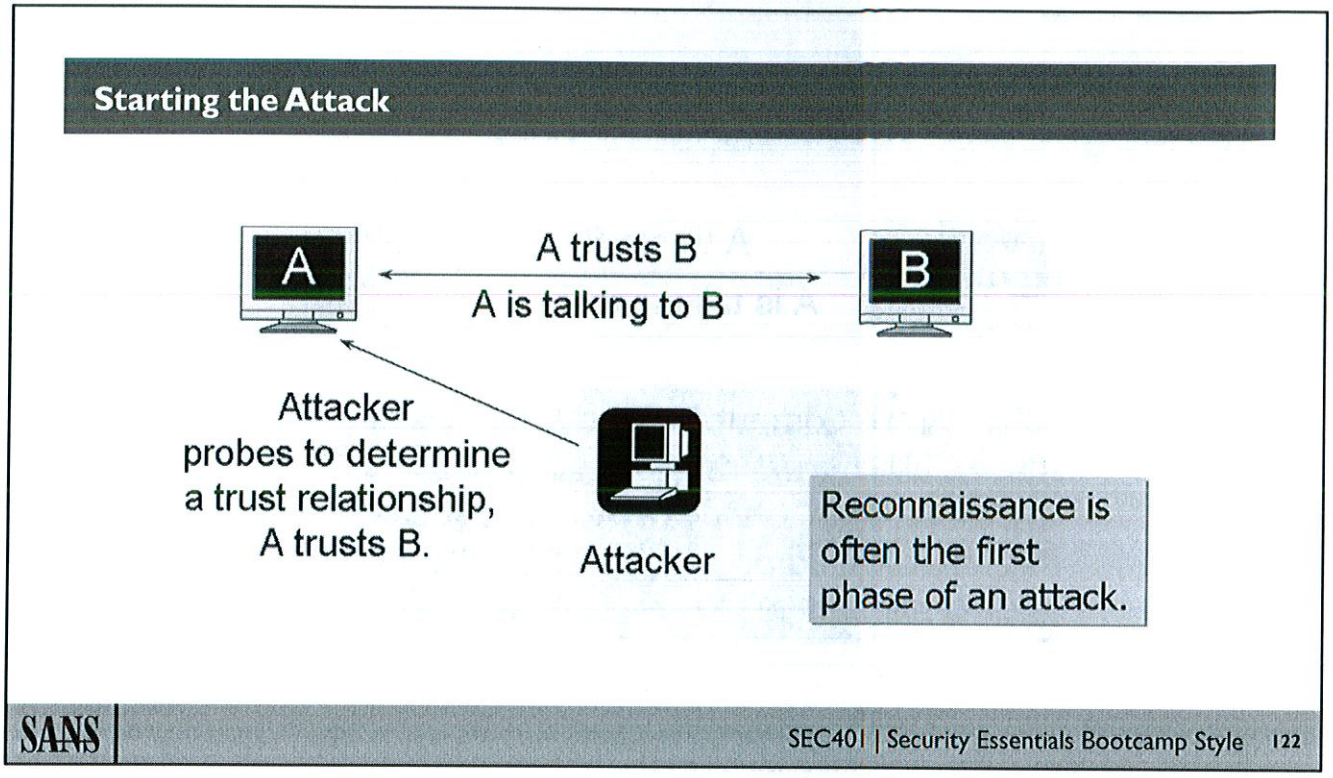
Phase 1: Scouting the Coast

If you're going to stealthily infiltrate an island fortress in the dead of night, the first thing you need to do is to scout the coast. You need to know the location and composition of the structures on the island, both man-made and natural. You want to discover all the bays, lagoons, rivers, and beaches to see where you can most easily come ashore without being detected, preferably someplace close to your eventual objective. Shimomura's LAN was this island fortress, so Mitnick's operation commenced with a short reconnaissance of the virtual landscape.

Like most well-planned attacks, this one started with a series of gentle probes designed to gather information about the victim's systems. Mitnick knew that he'd likely be unsuccessful in obtaining the cellular software via a frontal attack. The target was too well-secured for that. But if that machine could not be attacked directly, perhaps some other system could be compromised instead, and its trust relationship with the target could be abused.

According to Shimomura's analysis in the days after the attack, Mitnick started by sending out several feelers to determine just this sort of information. Their purpose was to discover possible trust relationships among the three computers on Shimomura's LAN. The trust relationships simply mean that a computer is familiar with another computer and trusts the information that is coming from it. If one of these hosts is compromised, the other is much more susceptible to attack.

Complex attacks against specific targets usually start with a reconnaissance phase in which the attacker maps out the lay of the LAN, so to speak, determining which hosts are present and gathering as much information about them as possible. After all, if the attacker hopes to achieve a specific goal, they will want to know what resources are available for their use.



Finger is a service that can return information about users on a particular system and is used for reconnaissance. If you supply a username, finger happily will tell you a lot of information about that user. However, what if you don't know the names of any valid users on that system? No problem! Finger will be happy to tell you all about anyone who's currently logged on to any host you name. The output can be quite informative, containing valid usernames, along with their home directories and shells. It can even tell the last time a particular account logged in and from which machine. To someone trying to gather intelligence about your network, this is a treasure trove of information. Because it can be so helpful to attackers, many system administrators routinely disable the finger service on machines for which they're responsible.

The finger probes Mitnick used were designed to look for patterns of remote logins between the three computers. The assumption is that if a particular user regularly logs on from a certain machine, there's a good chance that a trust relationship exists with that machine. In other words, the attacker hopes that the act of having to type an account's password each time someone logs in from one machine to another might have become so annoying that the user set up a .rhosts file to direct one computer to accept incoming login connections from the other computer on trust and not prompt for a password. If the user successfully logged in to the first computer, then tried to remotely access the second one, that machine would then accept the first computer's word about the user's identity and log him in without challenge.

Users often set up these trust relationships on their own accounts to make it easier for them to work on many machines at once. A system administrator might also do the same thing for all users on a machine by creating a hosts.equiv file. Although this sort of trust is convenient for users, it is usually a gaping security hole. Unfortunately, this is still common in many Unix environments.

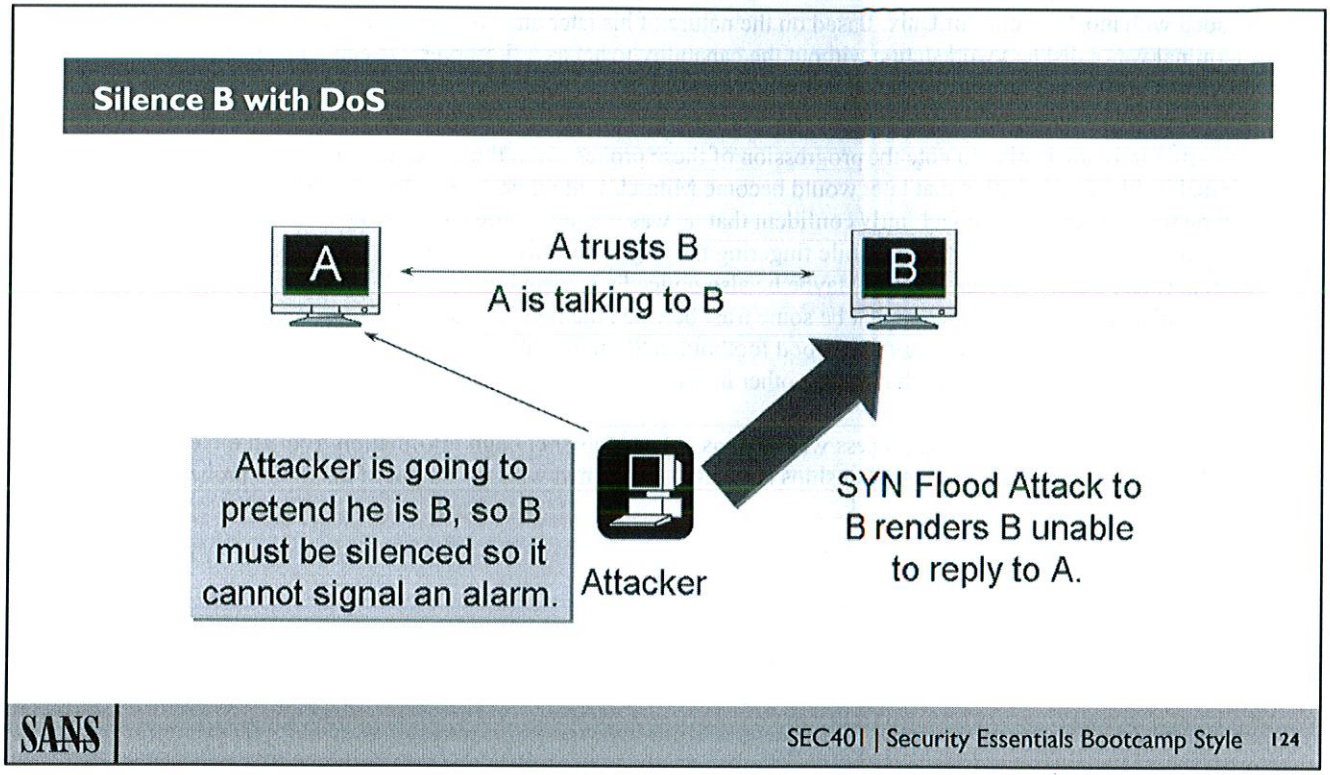
Now take a look at the other two commands Mitnick used during his reconnaissance, showmount, and rpcinfo. showmount lists the filesystems exported by an NFS file server.

For its part, rpcinfo tries to enumerate the various RPC-based services on a remote machine. Most of the services listed would be related to NFS or to Sun's Network Information Service (NIS), the popular network user database

included with most varieties of Unix. Based on the nature of his later attacks, and from the fact that we know the X terminal was a diskless workstation without the capability to act as a fileserver, we can guess that these commands probably didn't turn up much useful information.

It is particularly interesting to note the progression of these probes, from the actual target of the attack to the eventual diskless workstation that later would become Mitnick's initial point of entry. Although we don't know his thoughts for sure, we can feel fairly confident that he was tracing a series of possible trust relationships between the three machines. Perhaps while fingering the target, he discovered an idle login session from the X terminal (there was one, it turned out). Maybe he also noticed a repeated pattern of logins from the server to the X terminal and inferred that there might be some trust between the two. Whatever his reasoning, he eventually decided that the diskless node would be a good foothold in Shimomura's network. That could be why he used rpcinfo and showmount against it but not the other hosts.

No matter what his exact thought process was, he was able to gather enough information from all three machines to make a pretty good map of the relationships they enjoyed. Armed with the output of his initial probes, Mitnick progressed to the next stage of his attack.



Phase 2: Cutting the Phone Lines

Back to our island infiltration mission! After you manage to slip onto the island undetected, you have to get through the compound's front gate. The rudimentary electronic door lock is connected to a computer system elsewhere in the compound, and it uses this connection to verify your identity. If you're not authorized for entry, you have to trick this lock into thinking otherwise. Fortunately, the designers of this building were builders and not cops. They run the lock's connection through a set of wires in a fairly obvious conduit along the outside wall of the building. It is a fairly simple matter to cut this connection so the remote system won't be able to communicate with the lock. This isn't enough to get you in the door yet, but it is a necessary first step. You've now removed the possibility of someone rejecting your false ID. This is exactly what Mitnick did for his next step, too.

Mitnick decided to try an IP address spoofing attack against Shimomura's X terminal. IP spoofing is just what it sounds like: You send packets to a remote computer but lie about your source IP address. Mitnick guessed that the X terminal might have a trust relationship with the server, allowing the server's users to log in locally without having to type a password. Therefore, his first goal was to open a TCP connection to the terminal's remote login service (port 513) that would appear to come from the server, so he wouldn't be challenged for a password. But first, he had to overcome a significant hurdle: The server machine had to be silenced.

Do you remember the three-way handshake TCP uses to open connections? The initiator of a connection is supposed to send a SYN packet to the destination, along with a randomized initial sequence number (ISN). The destination host ACKs that SYN with the initiator's ISN, and also includes its own SYN and another randomized ISN for the second half of the connection (the half that sends data back to the initiator). The destination expects to receive a final ACK from the initiator along with its randomized ISN (actually, the ISN + 1, but who's counting?). If the ISN in the ACK for either side of the connection is wrong, the whole process terminates and no connection is made. If you hope to be able to complete the handshake and make a new connection, you need to know the ISN that the destination host chose for its SYN.

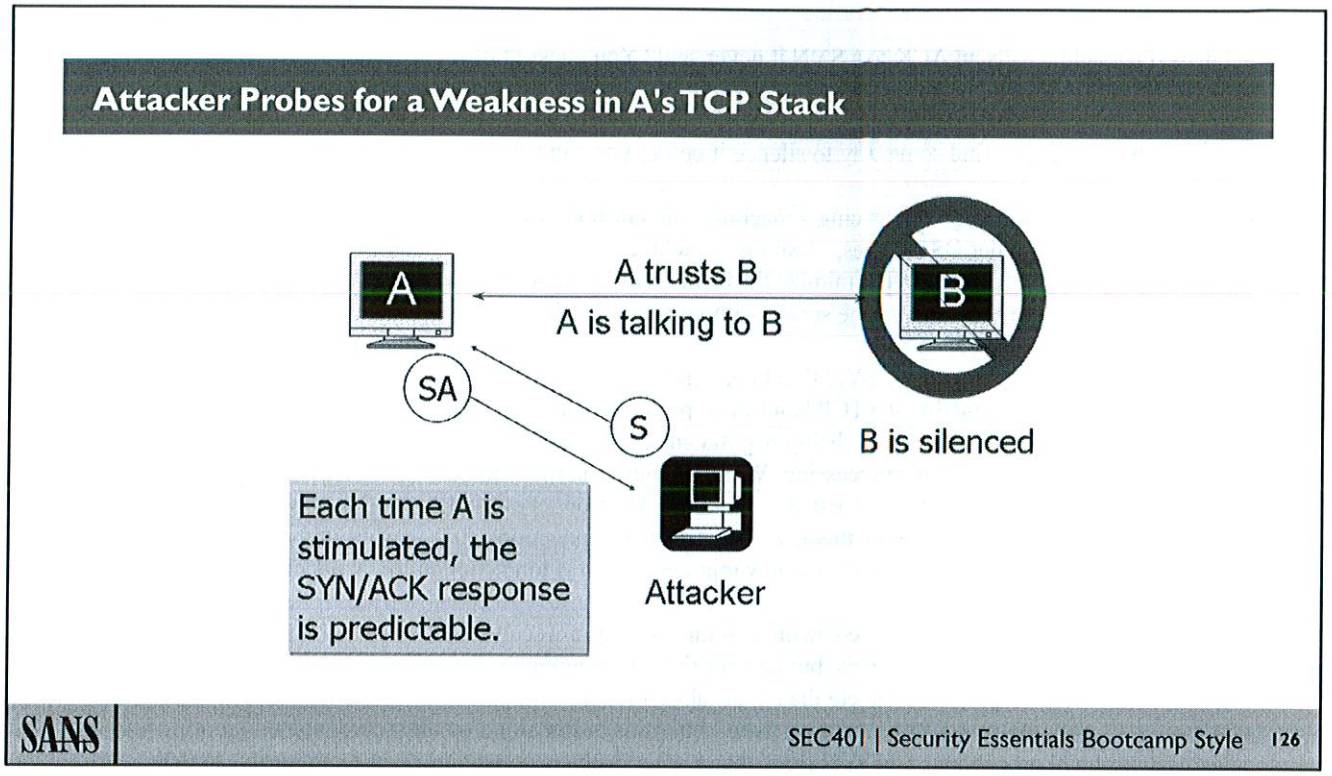
Here's the trick: If you're spoofing packets to the server, you're giving it the IP address of some other machine as the originator of the connection. Therefore, the ACKs will go to that other machine, and not you. What

happens if a machine gets an ACK to a SYN it never sent? You guessed it! It sends a reset (RST) packet in response to the ACK, which causes the intended victim to close the whole connection attempt right there, foiling your evil plot. Thus, if you're trying to spoof the address of a specific machine in order to exploit a trust relationship, you have to find some way to silence it before you can proceed.

Conceptually, the easiest way of silencing a machine is to crash it. Although there are numerous tools that can do this, especially for older OS releases, crashing a machine is pretty obvious because it is easy to notice when a machine is entirely down. To help minimize the chances of his attack being detected, Mitnick chose a stealthier way to temporarily silence the server without actually bringing it down.

The technique he chose, known as "SYN flooding," involves sending numerous SYN packets to the machine to be silenced but never completing the TCP handshake protocol. A lot of OSes allocate a fixed-size buffer to handle TCP handshakes while they are being negotiated. This attack sends so many SYNs that it fills the buffer and prevents any other handshake processing. While the buffer is full, the OS won't respond to any incoming connection attempts, not even the ACKs that are sent as a by-product of the IP spoofing attack. These buffers usually have timeouts associated with them, so the half-open connections don't stay around forever, but the timeout usually is pretty long, maybe even a few minutes, which is long enough for an attacker to do his job.

Let's pause for a moment to review the situation so far. The server received an overwhelming number of requests to create new TCP connections, but none of these faux attempts continued the protocol past its first step. The server kept hoping to complete the handshakes, though, so it placed these SYN packets in a buffer it could refer to when their ACKs eventually arrived. After this buffer filled up, the server started ignoring all other TCP protocol processing. The SYN flood attack effectively silenced the server temporarily, so that Mitnick could proceed to the IP spoofing phase of his plan without fear that the server would receive the ACKs to his false SYNs and cause the server to reject his spoofed connection attempts.

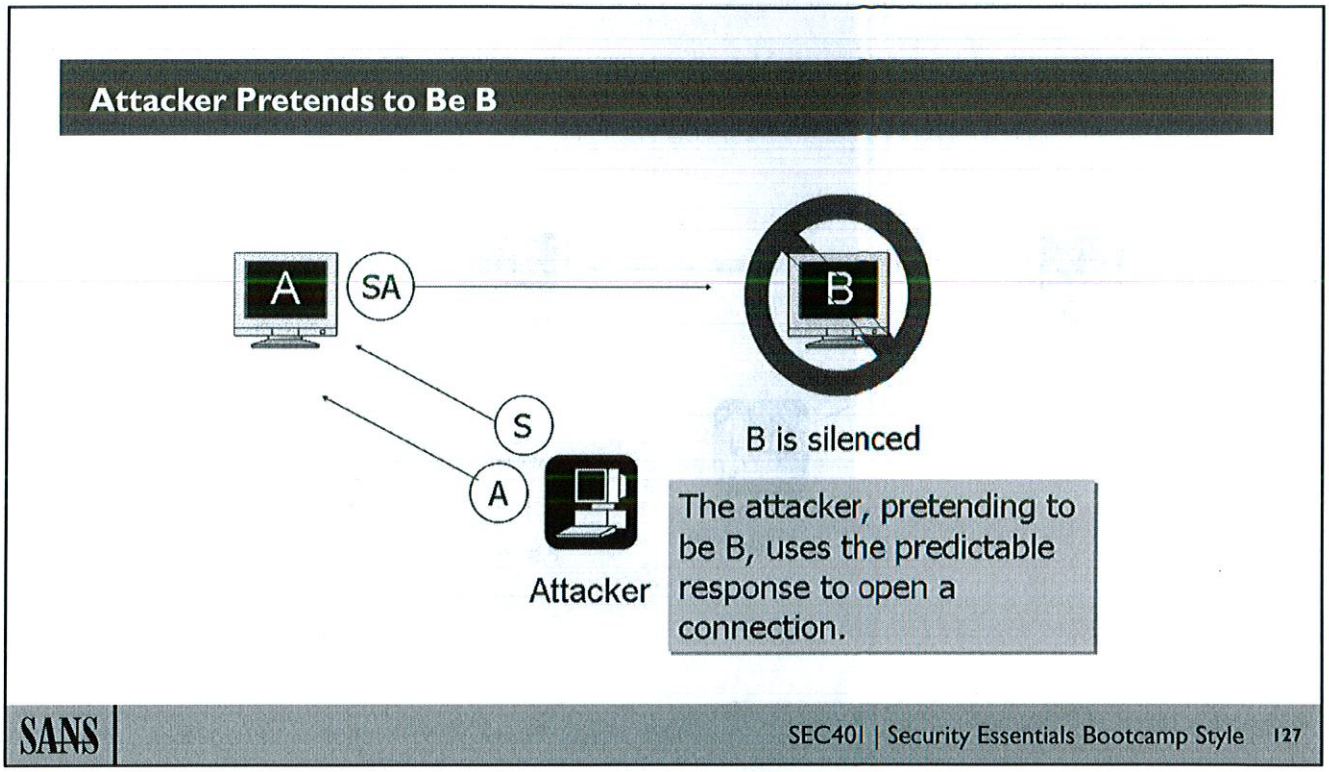


Phase 3: Analyzing the Lock

Okay, you're back on the island again. In the previous section, you prevented the nifty electronic lock from communicating with the central system that verifies IDs. This isn't enough to get you through the door, though. You still need to convince the lock that the ID you're going to present is a good one. Probably the easiest way is to hook up your own "central system" to the communication cables you just cut. You know the basic protocol, but there are a few little details of the communication you can figure out only "at run time," so you hook up your palmtop to the cables and experiment a little to find just the right combination. That's what Mitnick did next, too.

Think back for a moment when we were studying IP concepts. Do you remember the role of the initial sequence number in negotiating a TCP connection? When one computer sends a SYN packet, it generates an ISN and includes it in the packet. It expects that the other end of the connection will ACK with the same ISN (plus 1, remember, because the ACK indicates the next byte the ACKer expects to receive). If you think about it, this can be a problem for someone trying an IP spoofing attack. Because the attacker is sending their packets with the IP address of a different machine as the source, the receiver naturally will attempt to send the ISN to the spoofed address. Not only that, but if the attacker ACKs with the wrong ISN, the victim simply will reject the connection attempt. If the attacker never sees the victim computer's ISN, how can they complete the connection?

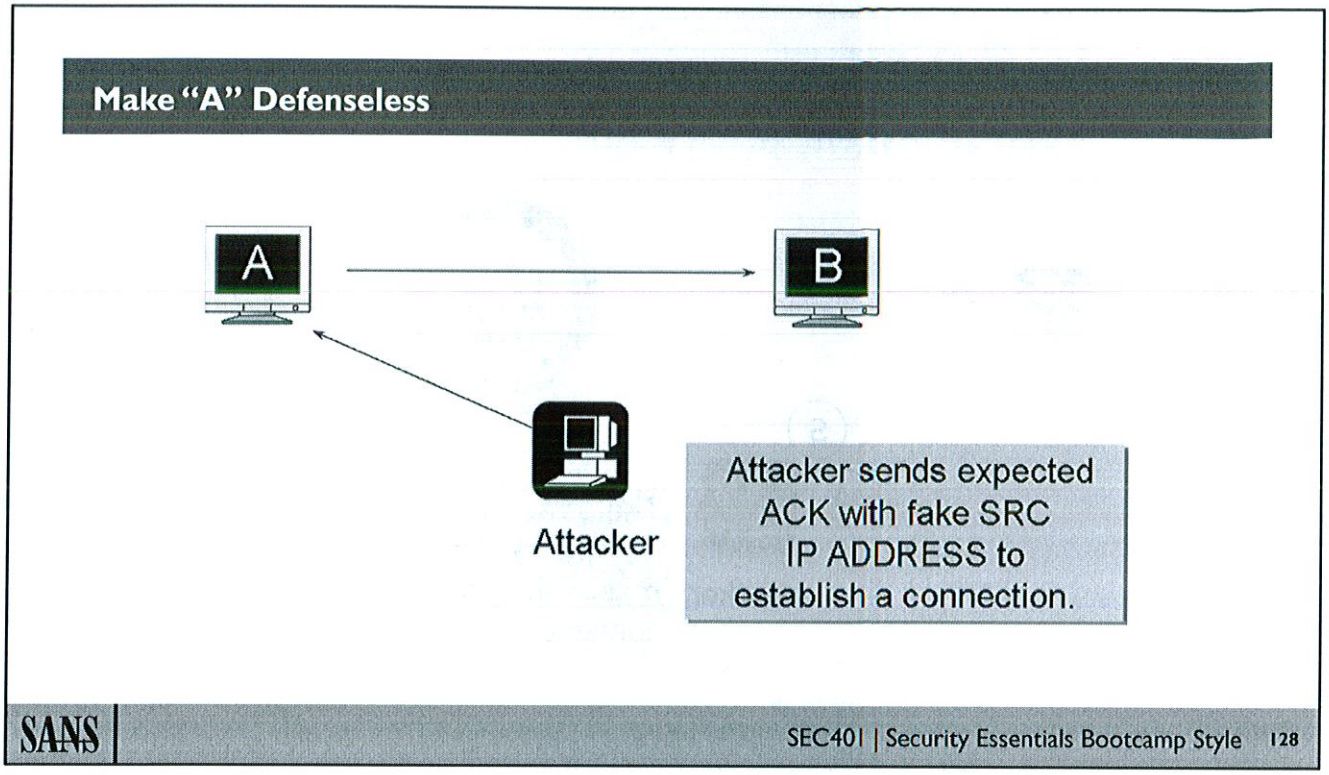
Like all really good ideas, the answer is obvious (after a lot of thought). The attacker has to predict the ISN that the target will use. It turns out that, although the TCP specification has pretty good protection against ISN guessing, many implementations either ignore the specifications or do not implement them correctly. The problem is guessing an ISN is supposed to be hard to do. At the time of the attack, however, most TCP implementations followed fairly static rules about how to generate ISNs, and most were vulnerable to simple guessing attacks. This remains the case today, but to a somewhat lesser degree.



Phase 4: Picking the Lock

From the previous slide, you can see what is happening at this stage of the attack. The attacker sends a SYN packet from an outside connection that is being spoofed as a trusted source (B). The target machine (A) replies with a SYN/ACK that the attacker never sees—it goes to the real B. However, armed with a predictable sequence number, the attacker completes the three-way handshake by sending an ACK to the target.

Now, back to our analogy. Your attack on the island fortress' bunker is nearly complete! You've figured out the inner workings of the lock and how it communicates with its back-end server. You've even connected your own phony server to the lock's communication lines, so you can trick it into allowing you access. All that's left is to present your fake ID and hope it works. You swipe it through the reader ... Success! The ponderous steel door slowly creeps open, and you're in!



Having gotten this far, Mitnick's next move was simple. Using the sequence number data gained from his probes, Mitnick was able to make a good guess as to the next ISN the X terminal would try to use. All he had to do then was to fake a connection to the X terminal's remote shell service that appeared to originate from the server. After waiting a short time, he could fake an ACK to the X terminal's SYN, too, using the guessed ISN. With a little bit of luck, he'd manage to convince the X terminal that the connection attempt had been successful. Unfortunately for Shimomura, luck was with Mitnick that night.

After connected to the X terminal's remote shell service, Mitnick mimicked the rsh protocol, lying to the terminal that his connection was coming from the server's root user. He hoped that the terminal trusted root to log on automatically from the server without having to supply a password. Apparently, this was indeed the case.

ISN guessing has one slight drawback: It works only against idle servers. If the machine you're trying to access is busy accepting other connections while you're attacking it, you can't guarantee that the next ISN it presents you with will be the one you guessed, because some other connection(s) might have been processed between the time you probed and the time you tried to use your guessed ISN.

Finish the Job

B sends rshell packet '"echo ++" >/.rhosts' to open A to attack

Attacker uses
rlogin -l root
to takeover 'A'

Attacker

Trust Relationship + Reconnaissance + Predictability = Hacked

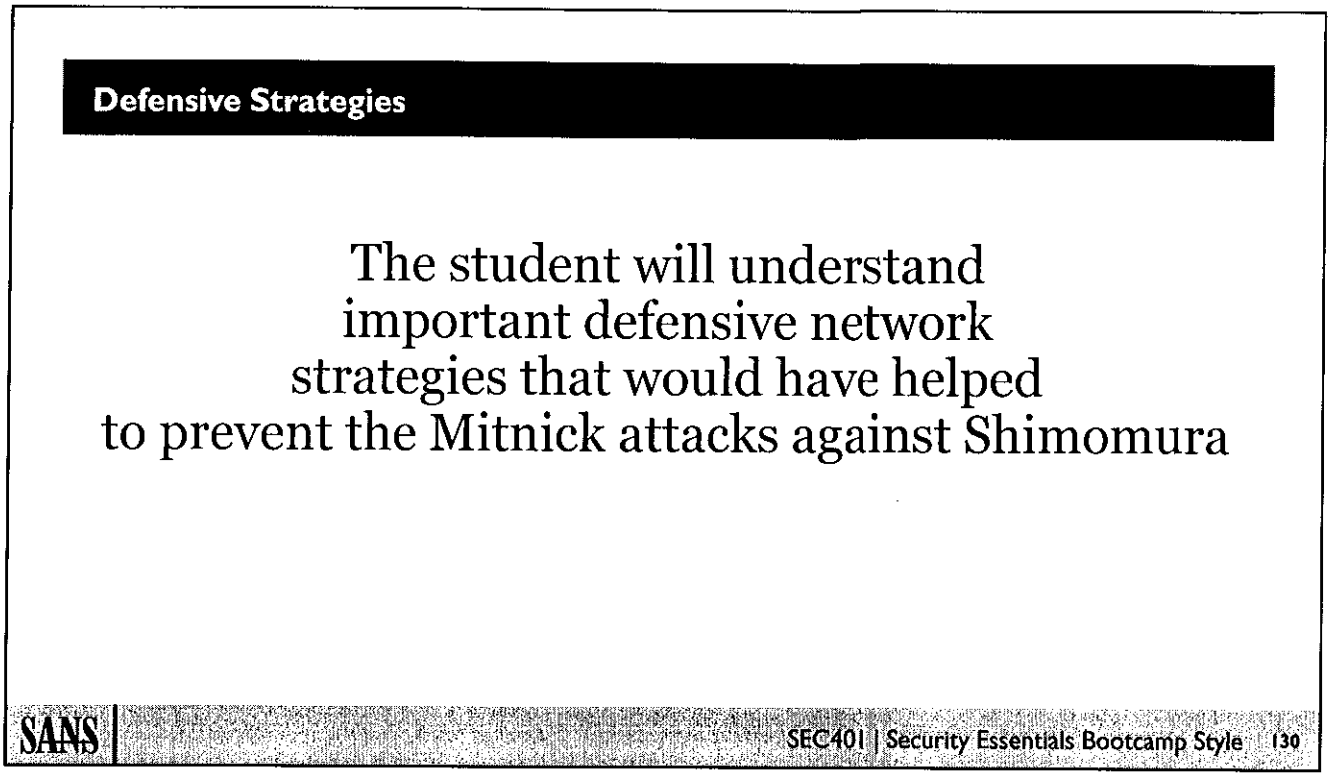
After successfully logged in as root, Mitnick had a simple goal: to preserve his root access and make it simpler to log in the future. To accomplish this, he modified root's .rhosts file to ensure that any user from any other machine on the Internet could log in as root on the X terminal without being challenged for a password. The command he issued was echo + + > /.rhosts. This overwrites the file with ++, allowing any user from any host to connect to the machine.

After this, it was a simple matter for Mitnick to clean up after himself. First, he shut down the spoofed remote shell connection and sent a flurry of RST packets to the server to clear up the temporary block he'd placed on it. After it was fully operational and responding to connection requests again, Mitnick hoped his slight modification to root's .rhosts file would have a better chance of going unnoticed.

What Mitnick did then was fairly straightforward. Using his new root backdoor to the X terminal, he logged on (using the command rlogin -l root, as seen in the slide) and brought over a nifty kernel module capable of tapping into an existing login session. Then he used the module to take control of a session Shimomura already had established from the X terminal to the target machine. Mitnick then simply browsed around until he found what he wanted and copied it for his own later use.

Fortunately, Shimomura's network was configured to keep good audit logs of Internet-based logins. An associate of his was watching over the network while Shimomura was out of town and noticed that the audit trail suddenly had gotten smaller! The only way for that to happen was for someone to manually edit the logs, probably as a way to help cover his tracks. Once alerted, Shimomura was able to analyze the attack and start the recovery process.

What followed was a long, arduous odyssey during which Shimomura partnered with the FBI and other law enforcement agencies to track down and arrest Mitnick. You can read more about this fascinating story, told in the Shimomura's own words, in his book, *Takedown: The Pursuit and Capture of Kevin Mitnick, America's Most Wanted Computer Outlaw – By the Man Who Did It* (ISBN 0786862106). Published by Hyperion in February 1996, this is a clear and interesting account of the whole process.



This page intentionally left blank.

Detection and Prevention Techniques?

- What common techniques (prevention and detection) could have prevented the attack?
- What risk management techniques could have detected the attack?

Goal: Make sure to fix the problem not address the symptom

It is certainly the case that an intrusion into your network can cause you no end of aggravation. Simply identifying the scope of the compromise can be a large task sometimes, so whatever energy and resources you can devote to protecting yourself ahead of time will almost certainly have large paybacks in the future. Let's look at a few techniques that you could use to help mitigate the risks associated with attacks like Mitnick's.

The information security cycle consists of three parts: prevention, detection, and response. The earlier you insert your countermeasures in this cycle, the more cost-effective they will be. Following that rule, the biggest return often comes with techniques designed to prevent attacks from occurring.

Not only is prevention usually the most cost effective way to deploy security resources, but it is probably also the most obvious. Sometimes it is too obvious. Organizations that rely solely on prevention often get taken for a bad ride when an attack eventually manages to get through all their preventative measures.

At a minimum, you should have well-defined incident response procedures in place to help you figure out what to do when your defenses are finally breached. And they will be breached eventually; it is just a question of when and to what extent.

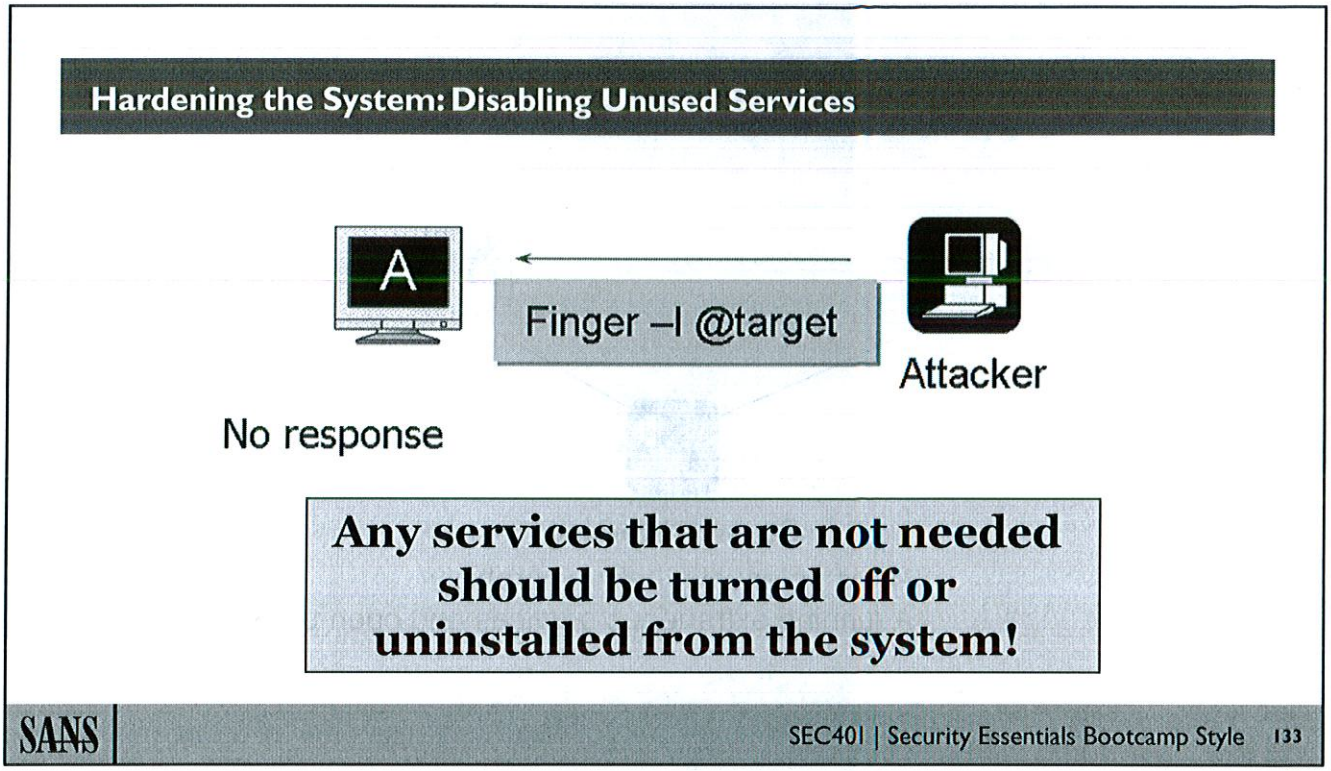
An ounce of prevention is worth a pound of cure.
– Benjamin Franklin

Patching Systems

- Although only partially relevant to Mitnick's attack, patching is important
- Timely patching can often prevent the majority of attack vectors from being successfully executed
- Patches are often available before or very soon after exploits are announced

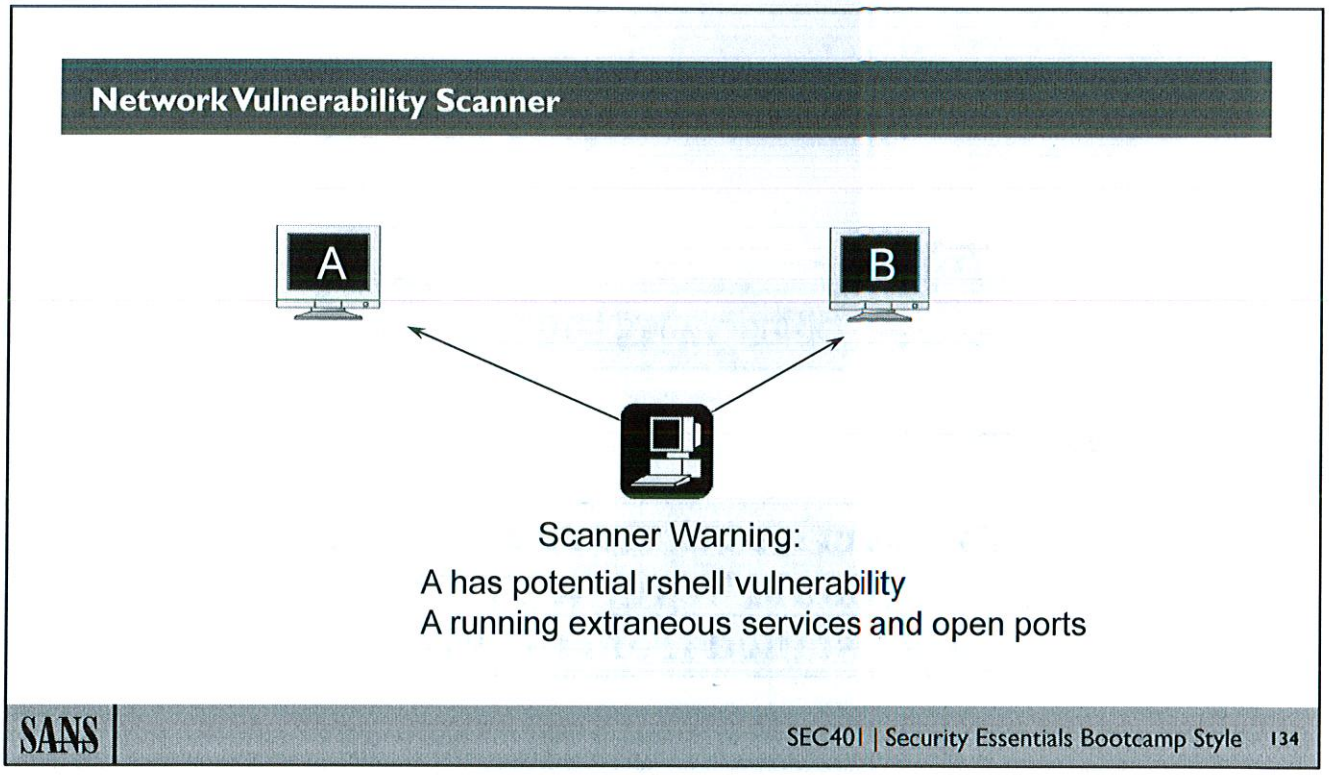
In executing his attack against Shimomura, Kevin Mitnick did not use unpatched systems directly. The inherent lack of security in the r* commands themselves enabled this compromise. However, in modern-day information security, patching of systems and applications is often considered to be one of the most important tasks for system administrators and security professionals alike.

When a vulnerability is discovered in an operating system's code, or an application is found to have a flaw, the software vendor usually releases a patch of some sort to plug the leak. Many times, these patches are available to administrators before, or shortly after, someone has actually created exploit code to take advantage of this vulnerability. By effectively managing the process of patching systems and applications, many attacks can be prevented before they really even get started.

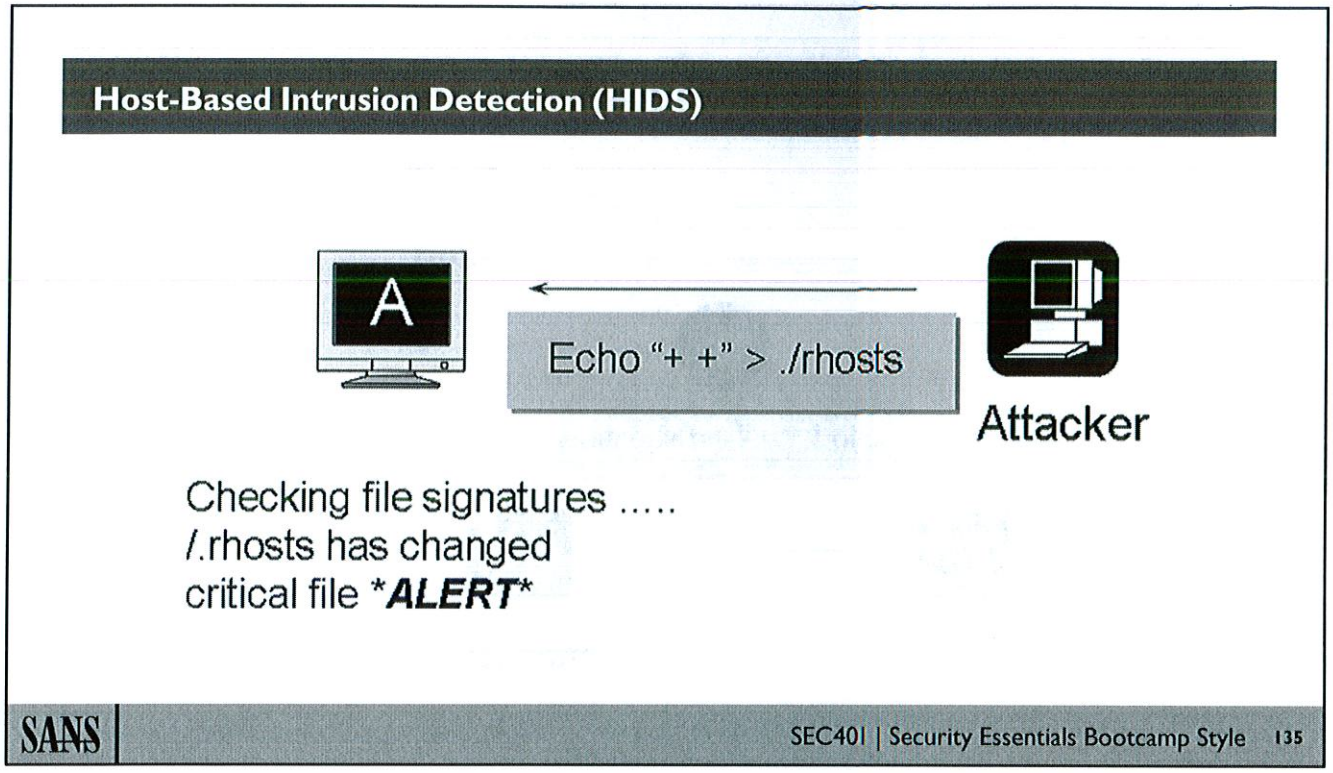


What if Shimomura's systems had not responded to the finger command Mitnick used for reconnaissance? This command, in its various forms, was instrumental in providing Mitnick with the necessary information to successfully stage his attack against Shimomura's network. Without this information, it would have been considerably more difficult to discover trust relationships, or anything else for that matter.

What about the r* commands? If Shimomura had disabled both finger and the various r commands, how would Mitnick have broken into the network? This all serves to illustrate a simple point: You should disable any and all services that are not essential to the system's function. Many operating systems come with unnecessary services installed and enabled out of the box, which can place a system at immediate risk once connected to a network. By carefully selecting options during installation and routinely auditing systems for unnecessary services or applications, you can significantly reduce the risk of a threat exploiting a vulnerability.

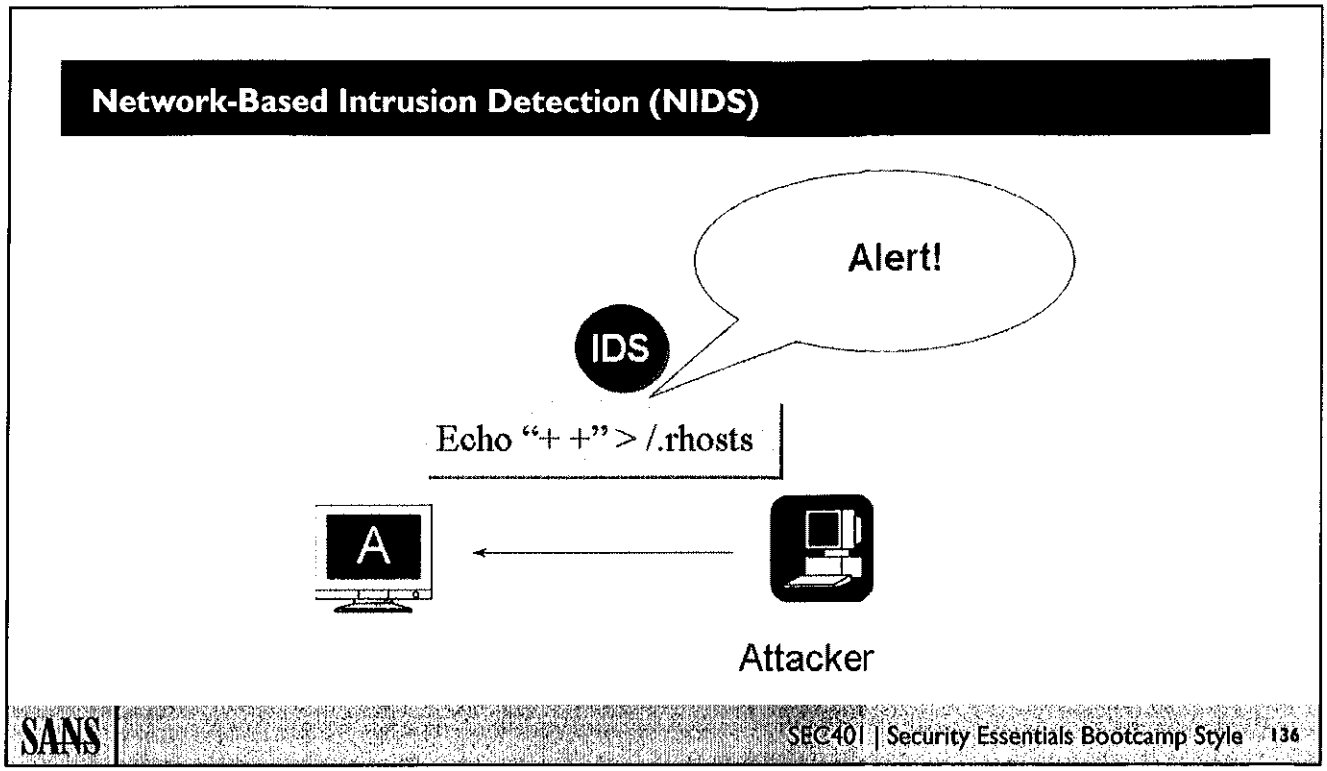


We're going to cover vulnerability scanning in more detail, but we felt it was important enough to merit at least a brief mention here. If you scan your own networks regularly, you stand an excellent chance of finding and closing vulnerabilities before attackers can exploit them.



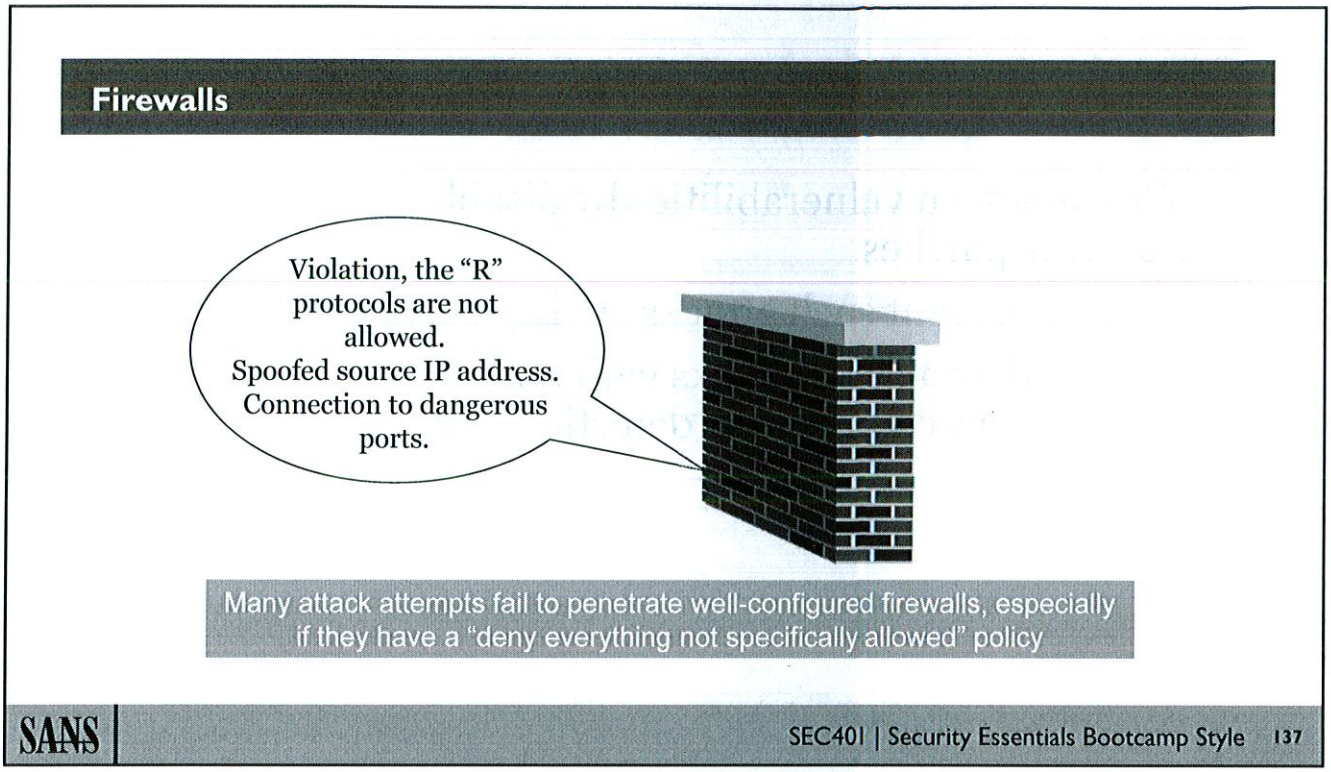
Host-based intrusion detection systems (HIDSs) usually consist of software that resides on a host machine and monitors the traffic in and out, as well as the integrity of the host's files. HIDS can be trained to record a system's initial baseline of users, running applications and services, and particular files to monitor, and it can then alert an administrator when any one of these elements changes unexpectedly.

HIDSs are often considered in the same class as host-based firewalls, and several current products can adequately perform the functions of both. Other types of HIDSs are distributed as software agents that can then be monitored from a central console; this type is more practical and suitable for a large enterprise. Typically, HIDS solutions are implemented with robust logging capabilities enabled on the host as well.



As attacks become known, it is common that NIDS (network intrusion detection systems) will have signatures for those attacks. This allows for an adversary using the same attack method to be detected. In addition to having signatures for known attacking tools, it is important for organizations to continue to develop their own signatures for new attacks that are detected. Keeping a NIDS up to date could be time-consuming, but is an important part of maintaining the device to properly catch adversaries.

Although we highly recommend starting with good preventive techniques, it is important to keep in mind that they are only a third of the total cycle of risk management. Be sure to follow them up with good detection and response mechanisms. That's security-in-depth. You will thank yourself later.



Probably the first thing any security analyst does when he designs a network these days is to plan for a firewall. It is almost impossible to have any kind of good internal security control without first establishing a secure network perimeter. In fact, the principle of security-in-depth practically demands that you be able to control the traffic entering and leaving your network. Fortunately, firewalls are very visible components of today's information security scene. They're usually the first thing management thinks of when it writes the security budget.

A good firewall (or at least a filtering router) can help prevent a variety of different types of attacks. In our scenario, it provides two very helpful functions: It prevents outsiders from accessing internal network services and from using spoofed IP addresses that should appear only on your own network.

Blocking access to non-critical services probably is the single biggest benefit of any of the risk management techniques we're going to discuss. Why offer to the entire Internet every service that's running on your internal LAN? Offering such an environment provides what the military would call a target-rich environment. If you narrow down to a select few range of services you offer, you can concentrate on configuring those services in as secure a manner as possible, while simultaneously denying an attacker any possibility of using poorly managed secondary services against you. In this case, had Shimomura deployed a firewall to block outside access to the r* command suite, finger, rpcinfo, and showmount, Mitnick would have had to search for some other way in.

Mitnick Example: Lessons Learned

- We can lessen vulnerabilities by disabling services and applying patches
- We can prevent such attacks as they occur with firewalls
- We can detect such attacks with both network-based and host-based intrusion detection systems

Having just described some basic prevention and detection techniques, you should now see some overall concepts and guidelines that can easily apply in your environment. First, you can significantly reduce your vulnerability exposure by patching systems as soon as possible, running scanners to detect vulnerable systems in your environment, and disabling services you don't need. You can also add a major element of perimeter (and possibly intra-network) security that can help prevent such attacks with a properly configured firewall. Finally, deploying host-based and network intrusion detections systems can alert you to any possible breaches in security or system anomalies much sooner.

Common Types of Attacks

The student will be able to identify the most common attack methods and understand the basic strategies used to mitigate those threats

This page intentionally left blank.

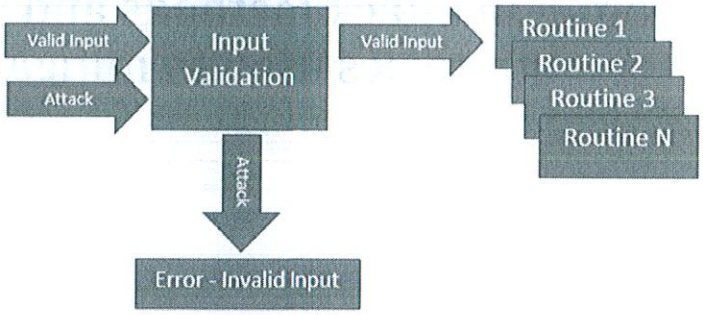
Input Attacks

Applications receive client data in many forms:

- Treat all user supplied input as potential attack points

Examples:

- OS command injection
- Buffer overflows
- SQL injection



One reason applications tend to have vulnerabilities is because they accept and process input from the user in a variety of different entry points. There are a number of ways an attacker can compromise an application by sending invalid or malicious data to these entry points.

Reference

1. Software Defenses to OWASP's Top 10 Most Common Application Attacks - https://securityintelligence.com/software-defenses-to-owasps-top-10-most-common-application-attacks/

Input Attacks: OS Command Injection

- Attacker sends OS commands as form or other input
- Relies on developer using input to build calls back to the OS
  - App that creates mailboxes using mkdir <username from form>
  - Attacker sends ksmith; rm -rf / as name
- OS runs rm -rf / command after mkdir

Some web applications use operating system level commands to perform certain functions. For example, a mailbox application might make a call to the operating system to create a new folder for a user's attachments, and name the folder to match the username supplied by the user. If the input is not properly validated, the user could have typed **ksmith; rm -rf /** as his user ID. When the create folder command was run within the operating system, the rm -rf / command would be run as a separate command meant to delete the entire filesystem.

OS Command Injection Defenses

- Avoid making system calls from within application
  - Especially based on user input
  - Use built-in application functions instead where possible
- Strip OS commands and characters from input
- Even better: Define valid characters for input used in this way; delete all others from input

It is generally a bad practice to make system calls from your web application, especially when the system call is built based on user input. In most cases, you should be able to find a function or library within your programming language that can perform the same action.

If you must make system calls in this way, you need to be aggressive about filtering and validating user input. Meta-characters and keywords that would be recognized by the operating system must be stripped out. In this situation, it might be best to define what the valid characters are (generally letters and numbers) and strip out everything else.

Input Attacks: Buffer Overflows

- Programs allocate a certain amount of buffer space to perform operations
- In poorly coded applications, no error checking is performed to ensure buffers are not overfilled
- The extra code placed in the buffer can sometimes be used to execute system commands and overwrite the return pointer

There are many types of malicious data attacks, most of which rely on poor programming practices with careless error checking on input data. An attacker might put an alphabetic string where a numerical value is expected, bringing down a popular database application. Worse yet, an attacker might insert Unix shell commands into the input in such a way as to trick the back-end web application into executing them on the server. These are both common examples of malicious data attacks that could be prevented by more careful validation and checking of input data.

One of the most well-known and popular examples of this sort of attack is the buffer overflow. To communicate with anything, be it a user or another program, an application has to accept some sort of input. It can be a password, a command to be executed, a record from a database, or just about anything else you can imagine. After the program reads this information, where does it go? Even if the information is extremely transient, the application has to store it in memory somewhere, if just for a few clock cycles. That means the program has to set aside a buffer, a region of memory in which to hold the data until it's no longer needed. When the buffer is allocated, the application tells the system how many bytes it will need to store, and the buffer is created to be exactly that size.

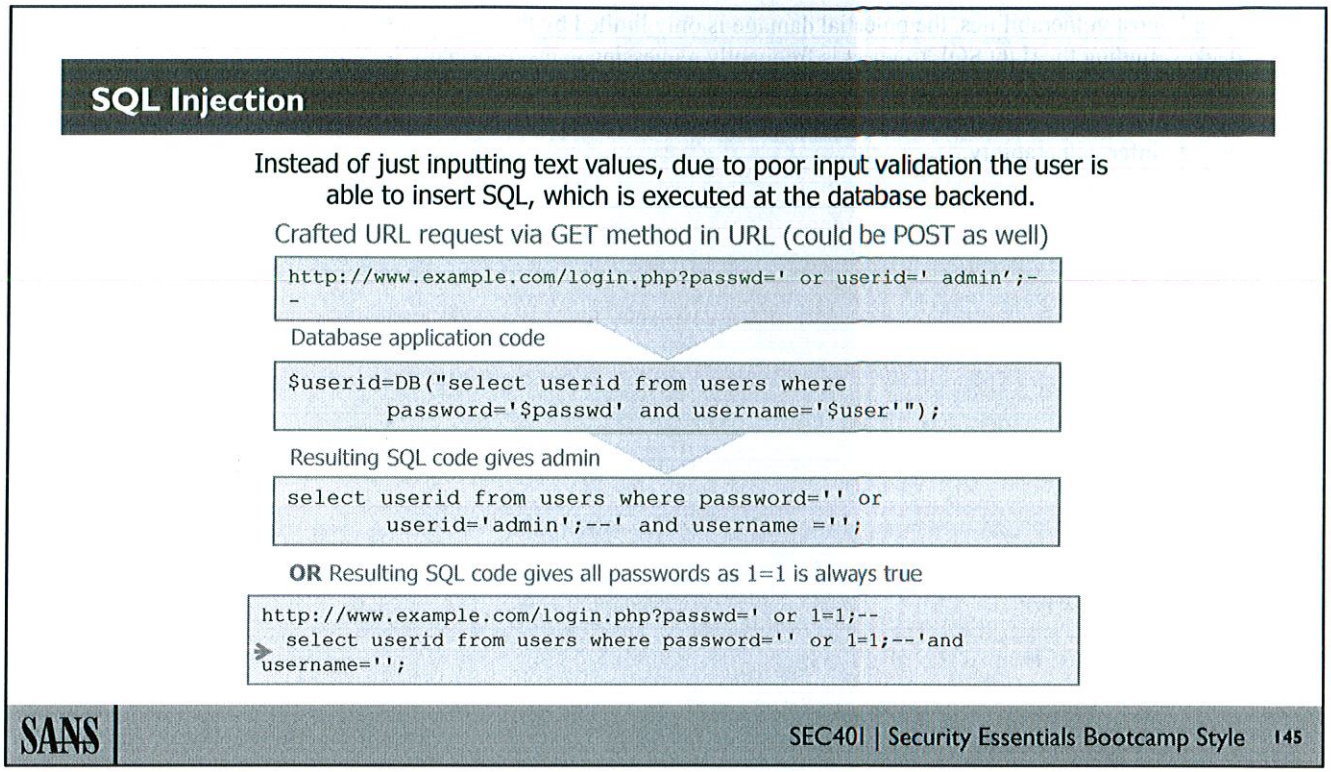
Buffer Overflow Defenses

- Run the latest versions of OS and web server software
- Update and patch your web server software
- Update and patch your languages/runtime environment/server add-ons
- Update and patch purchased or open-source web parts (bulletin board, shopping cart, etc.)
- Run a vulnerability scanner against your site
- Implement IPS or Web Application Firewall
- Validate and sanitize user input

Because the buffer overflows exist in the off-the-shelf components of the website, you can reduce your risk by keeping all parts of the website patched and updated. Don't forget to pay special attention to the open-source web packages that you are using on your website; these are frequently updated due to new vulnerability discoveries.

There are signatures for many of the buffer overflow attacks in web servers and application components. Running a vulnerability scanner against your site can tell you whether you are running vulnerable software, and implementing an Intrusion Prevention System (IPS) or Web Application Firewall can block or drop traffic that matches the signatures for these overflow attacks.

Finally, before using any input within your application, it should be checked to ensure it is an appropriate length and has valid content.



SQL injection is yet another vulnerability taking advantage of insufficient input validation. The technique, if successful, can be used to execute arbitrary SQL commands to which the web server application database account is authorized.

The key to the attack shown is the ability to input a single quote into the input field for SQL, which allows the attacker to finish the string being input. Then the attacker inputs his SQL and ends with the comment symbol (--). He is then able to cut off the end of the SQL so that the statement works. This can be done through the URL in the address bar of the browser or by manipulating the HTML of a client web page before it is POSTed. Once an attacker can execute SQL in this way, there are many further attacks that are possible. The attacks available depend largely on the vendor of the database.

SQL servers do support various access limits. For example, the web server might be able to alter only certain data and might have no access at all to particular critical data.

Some web scripting languages (for example, PHP with magic_quotes turned on) automatically escape user-supplied data. Nevertheless, it is highly recommended to carefully validate user data because many databases, such as Oracle, give read access on the user password table to low-privileged users.

In the previous example, the developer should validate the input provided by the user. For example, the userid is likely limited to alphanumeric characters (a-z and 0-9). Passwords might be trickier, but you can at least ensure that characters such as ' and \ are escaped so that they are not interpreted as SQL.

SQL injection can lead to OS compromise because some databases do allow the execution of system commands such as SQL server via the xp_cmdshell stored procedure. In this case, an SQL injection vulnerability can be used to execute system commands on the database server. In most cases, databases allow the user to write files. A file in the right location can be executed by an unsuspecting user later, in particular, if it is possible to overwrite an existing file.

As with most vulnerabilities, the potential damage is only limited by the aspirations of the attacker. For an attacker, finding the right SQL to inject is frequently a guessing game, especially for custom applications. In these cases, don't provide too much information to an attacker with overly verbose error messages. Attackers might still use blind SQL injection techniques where other factors, such as the time a response takes to happen, as a way to infer vulnerability.

SQL Injection Defenses

- Validate user input
  - Filter out SQL commands and special characters (' ; : ")
- Have length limits on input
  - Many SQL attacks depend on long strings
- More tiers: Add an application layer between the web server and the database
- Utilize stored procedures instead of SQL queries
- Database access: Web account should not have rights to add/drop/modify tables or stored procedures
- Do not display SQL errors to web users
- Monitor SQL error messages

If your application makes SQL calls back to the database, you must ensure that all SQL commands and characters are removed from all user input. Some characters to look for include single quote, double quote, slash, backslash, semi-colon, and extended characters like NULL, carriage return, new line, and so on. It might also help to have length limits or truncate input since many SQL attacks depend on long strings to do anything malicious.

You can keep from creating and running SQL commands from your web application scripts by utilizing stored procedures instead of SQL calls to interact with the database. You can extend this further by moving to a three-tier architecture, adding an application layer between the web scripts and the database.

The account that is used by the web service to access the database should have restrictive rights. It should not be able to add, drop, or modify tables; should not be able to create or modify stored procedures, and; should have access only to the necessary databases and tables.

One aspect of many SQL injection attacks is the amount of information the attacker can learn about the database from the error messages returned. Error messages often disclose table names, column names or data types, and other important information to the end user. Your production server should be configured to give vague generic errors to users and log detailed error information to a log file.

Finally, you can better detect whether someone is searching for or exploiting SQL vulnerabilities on your site by watching your SQL logs for errors that might indicate a user is sending invalid queries to the database.

Input Attack Defenses Summary

- Remove from user input all characters that have special meaning in scripting languages
- Be suspicious of all input, including HTTP headers and cookie data
- Validate on the server, not the client
- Enforce an "allow only necessary characters/strings" stance where possible (whitelist)
- Otherwise, filter out known bad characters or input (blacklist)
- Implement a library of input checking routines to use throughout the application
- Don't forget to check for encoded characters

If there is one phrase to drum into your head it's "Always validate and sanitize user input." Every piece of information the web server receives from a client browser application is suspect of being malicious. You must ensure that all user input is safe before using it.

Input checking should not be limited to form input and the query string. Clients have the ability to manipulate data in the HTTP headers and cookie values as well. Validation can be done using scripts on the client side, but this really should be used only to make the web application more usable. Client-side validation provides no additional security because a malicious user can modify or bypass any scripting or validation done on the client; all data must be validated on the server as well.

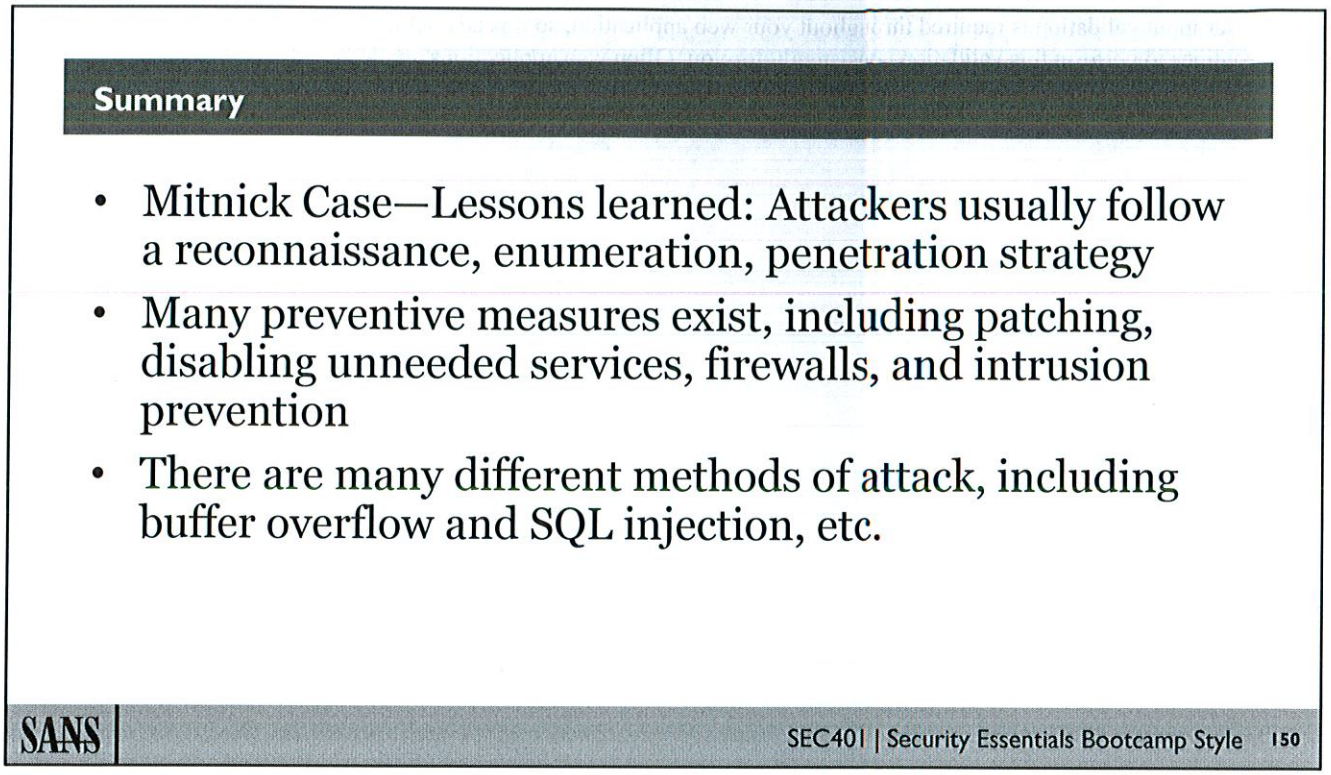
The preferred way of validating input is to whitelist or to allow only known good values.

Another method to validate input is to check for or sanitize known bad input. This is also known as blacklisting.

With blacklisting, you try to guess all the bad input that a user might try to send into the application, and delete or replace the bad input before it is processed by the application. Blacklists have a higher chance of treating bad input as safe or corrupting valid input. Sanitization while blacklisting can be done by deleting the unsafe characters or by changing them to safe characters. For example, the single-quote character ' is a string termination character for many SQL database query languages. If a user types this character into a bulletin board posting, she could cause an error when the data is written to the database.

The single quote character could be deleted from the input, but that would corrupt the grammar of the user's statements. Another challenge with blacklisting is the variety of ways a character can be encoded when sent to a web application. In different situations, the web server or application software might convert hexadecimal characters, Unicode characters, and URL-encoded characters into a form that could cause problems when processed by the application.

User input validation is required throughout your web application, so it is advisable to implement a library of routines to perform this validation consistently for you. Often web applications are designed such that some validation is done automatically as part of the initialization steps of each page of the application.



We started this module with a case study, and for good reason. To secure your systems, you have to understand what types of threats you need to secure them against. Thanks mostly to Tsutomu Shimomura's efforts, Mitnick's attack is a well-documented and thoroughly analyzed incident. Studying it allows you to virtually look over the shoulder of a skilled attacker and observe him as he does his dirty work, a valuable experience for any security professional.

His attack started, as most well-planned attacks do, with some basic intelligence gathering techniques. In this case, Mitnick's goal was to map out the patterns of user logins between the systems on Shimomura's LAN. With that information in hand, Mitnick was able to infer the trust relationships between the various machines and exploit them using a previously theoretical technique known as "IP spoofing." The purpose of his spoofing attack was to create on Shimomura's X terminal a backdoor, which allowed Mitnick to log in as root and to carry out an alteration attack by installing a kernel module that allowed him to hijack an existing login session to the target system. Once there, he browsed until he found the cellular phone software he was looking for and downloaded it to his own system. Along the way, he used his access to modify the system logs to try to hide his activity, which ironically was Shimomura's first tip-off that something was wrong.

Of course, there are a number of risk management techniques Shimomura could have used to help prevent attacks like this one. For example, had Shimomura deployed a firewall to protect the perimeter of his network, Mitnick's attack probably would have been foiled easily. Other techniques available today include disabling the Berkeley r* commands and replacing them with secure SSH equivalents, deploying some sort of host- or network-based intrusion detection system, and using periodic vulnerability scans of our own networks to identify and close potential problems before attackers like Mitnick can take advantage of them.

To give you a better understanding of some of the other risks your systems face every day, we ended this chapter with our own informal list of some of the more common categories of attacks. Its purpose is just to give you a taste of the possibilities and to drive home the fact that most systems have many different points of attack.

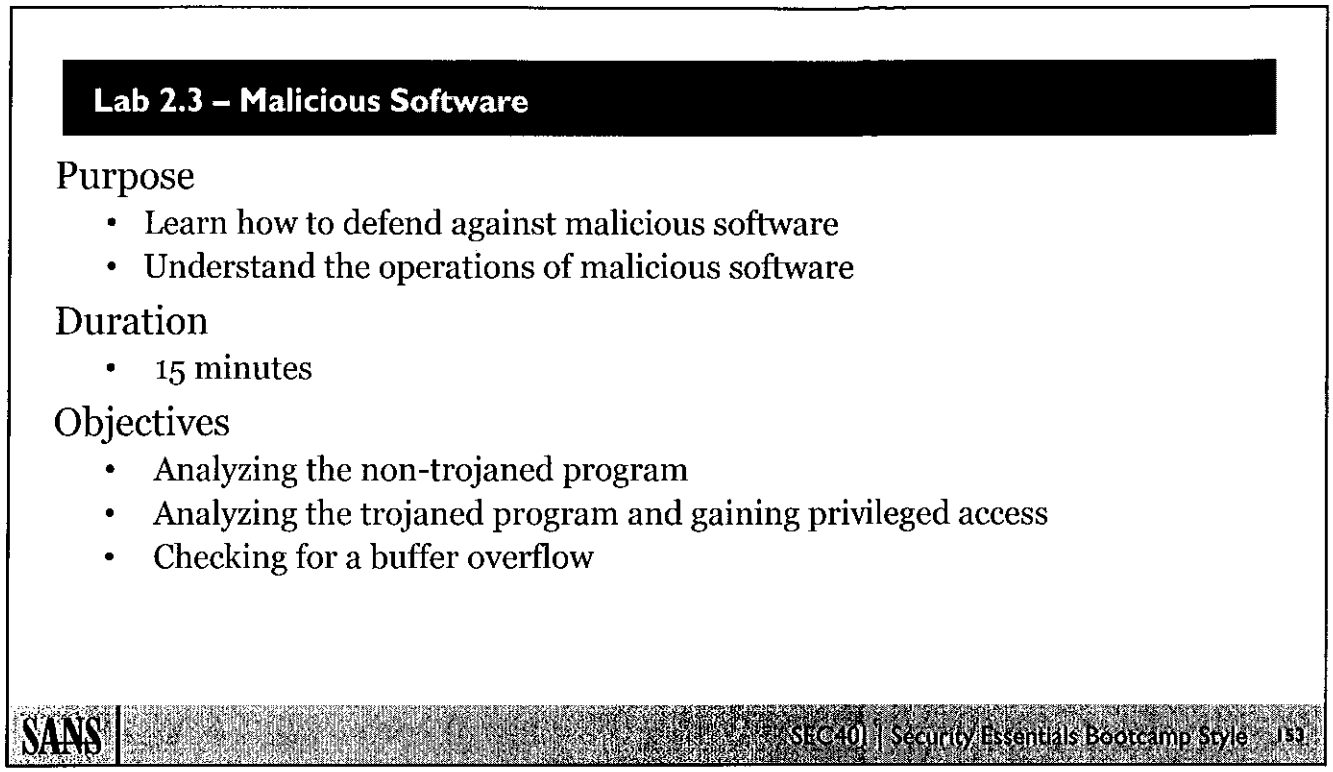
There are so many attack methods, in fact, that you really can't rely on just one or two preventive measures to protect yourself. You need to follow the entire risk management cycle, from prevention, through detection, all the way to response, and back again to the beginning. You should use several distinct countermeasures and address each phase of the cycle somehow. Only in this way can you hope to manage the risks associated with sophisticated attacks like Mitnick's. If it can happen to someone like Shimomura, it can happen to you.

Security 401: Day 2

SANS

Lab 2.3 – Malicious Software

In the module, you learned about various types of malicious software, also known as "malware." One type of malware is a Trojan Horse. This is a program that often performs some useful or entertaining function, but also does something unintended, such as opening up a network port or connection, downloading files, or simply providing some form of backdoor access. This type of malware is often a concern with Open Source Software (OSS) because the source code is publicly available. Attackers can add code and functionality to the program, compile it, and then make it available for downloading online, causing users who download and run the trojaned version of the software to fall victim.



Purpose

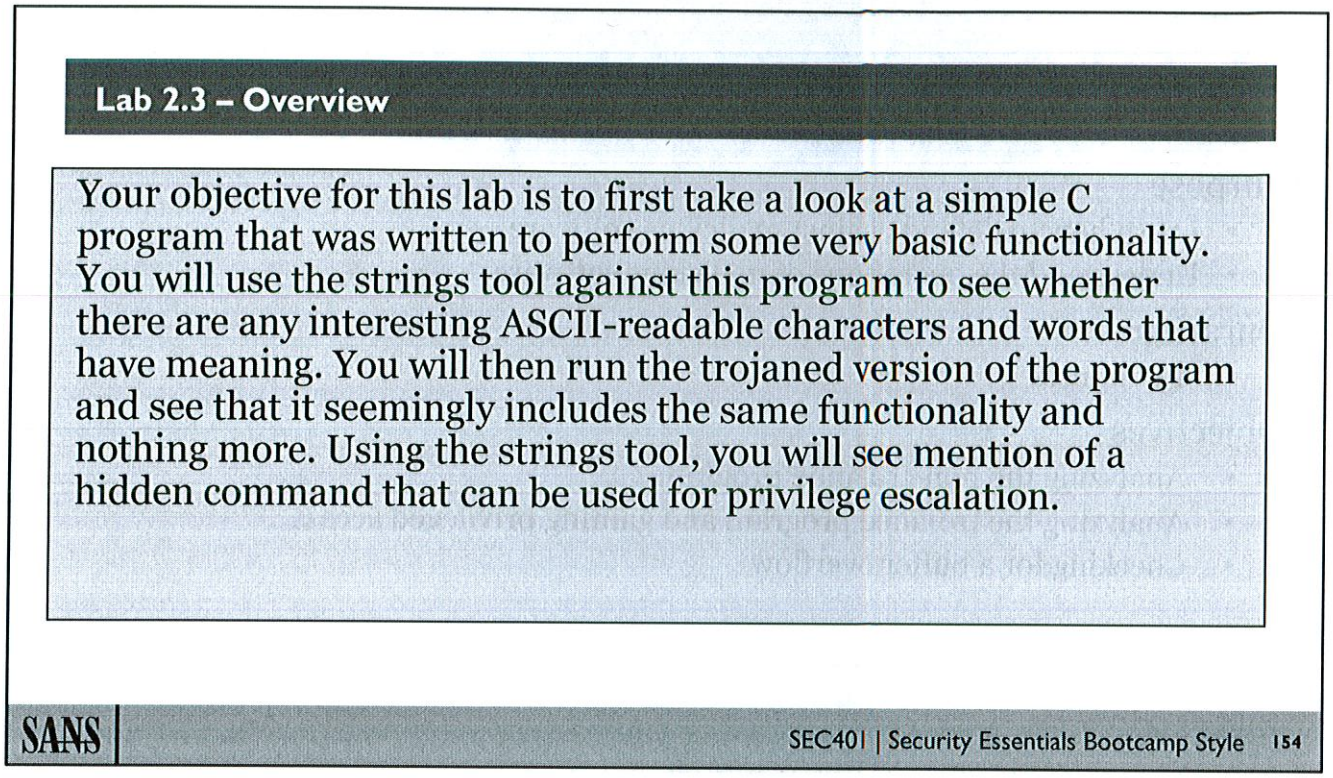
- Learn how to defend against malicious software
- Understand the operations of malicious software

Duration

- 15 minutes
- The estimated duration of this lab is based on the average amount of time required to make it through to the end. All labs are repeatable both inside and outside of the classroom, and it is strongly recommended that you take the time to repeat the labs both for further learning and practice towards the GIAC Security Essentials Certification (GSEC).

Objectives

- Analyzing the non-trojaned program
- Analyzing the trojaned program and gaining privileged access
- Checking for a buffer overflow



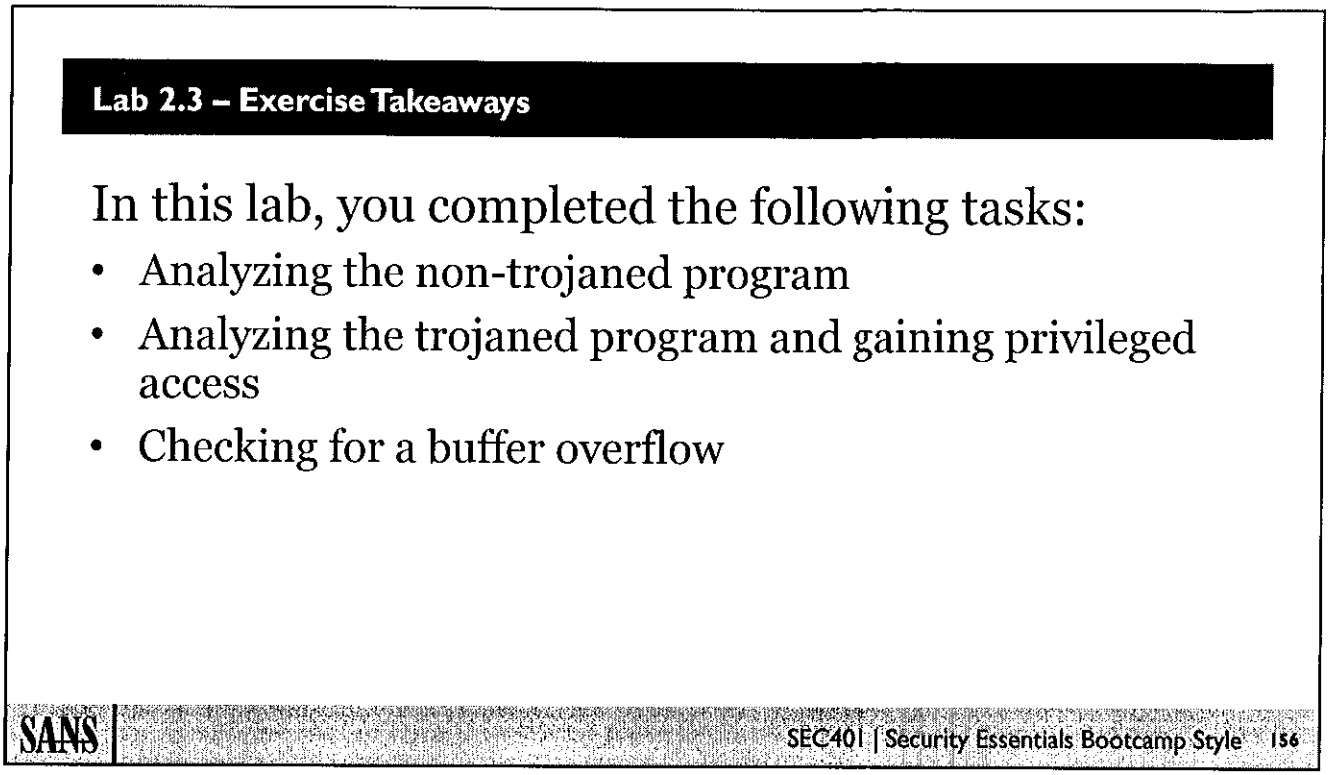
Your objective for this lab is to first take a look at a simple C program that was written to perform some very basic functionality. You will use the strings tool against this program to see whether there are any interesting ASCII-readable characters and words that have meaning. You will then run the trojaned version of the program and see that it seemingly includes the same functionality and nothing more. Using the strings tool, you will see mention of a hidden command that can be used for privilege escalation.

SANS

NOTE: Please open the separate Lab Workbook and turn to Lab 2.3

The instructor is going to introduce and go over the labs. Once the instructor is done, you will be instructed to work on the lab. If you have any questions, you can ask the instructor.

This page intentionally left blank.



In this lab, you completed the following tasks:

- Analyzing the non-trojaned program
- Analyzing the trojaned program and gaining privileged access
- Checking for a buffer overflow

As you can see, modifying a program and having it perform something to an attacker's advantage is quite trivial. It is important to always get software from trusted locations. If using open source software, it is highly advised that someone performs a code review to check for backdoors and other malicious code that might have been added. Code review is a highly skilled role that requires the reviewer to be proficient in the programming language of the program he is reviewing. These individuals also check for programming errors that might lead to vulnerabilities such as a buffer overflow.

SANS

Lab 2.3 is now complete

This page intentionally left blank.

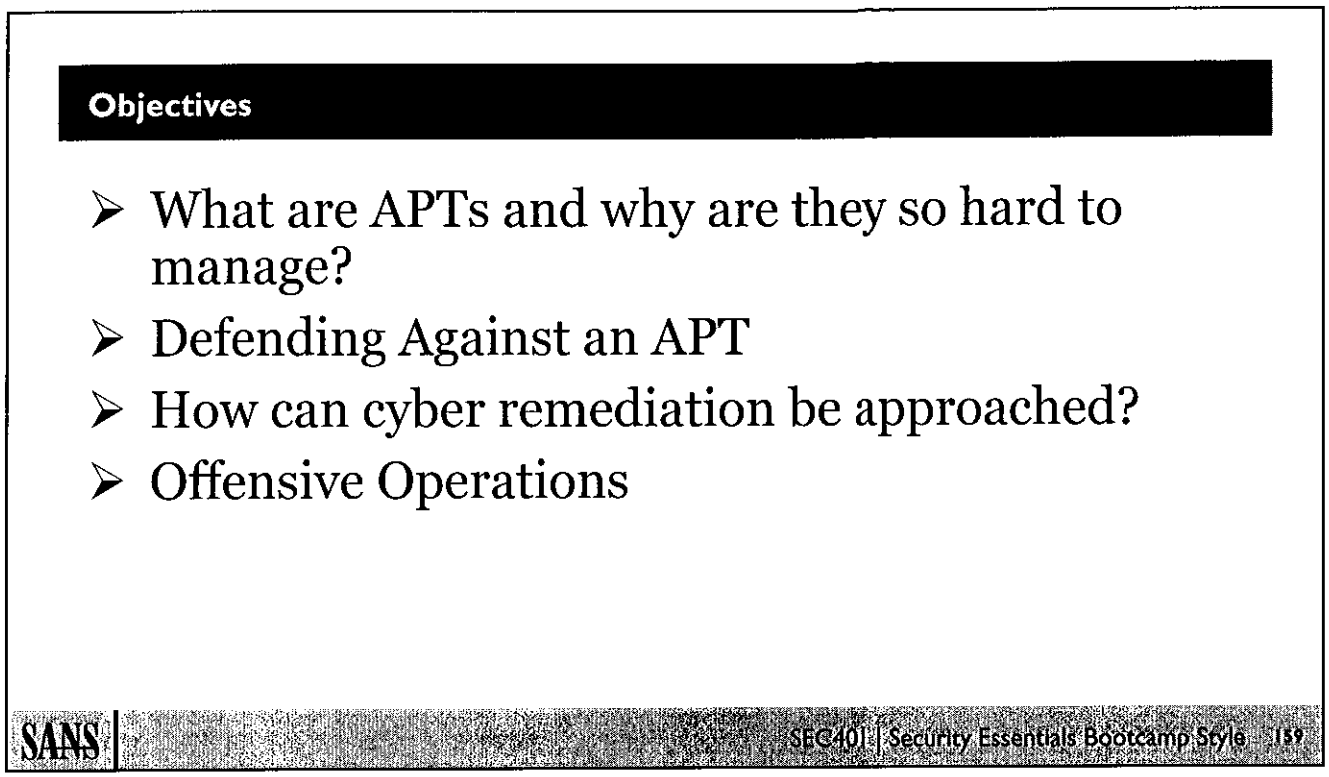
Security 401: Day 2

SANS

Module 12: Advanced Persistent Threat (APT)

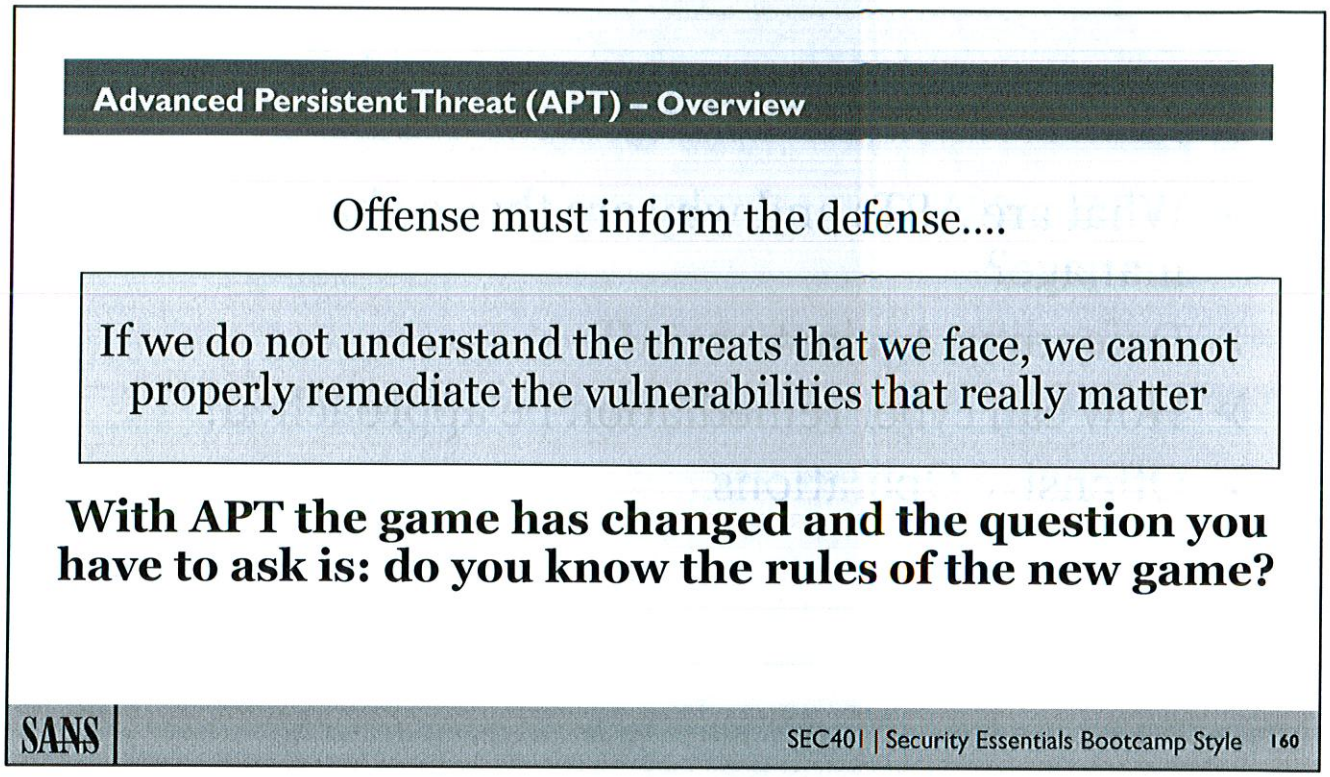
Module 12: Advanced Persistent Threat
In order to implement effective solutions, it is critical that you understand the threats that you are facing. APT or Advanced Persistent Threat is a relatively new adversary and is different than traditional threats and requires a new way to think about security solutions. This module will cover in detail what an APT is and more importantly methods for remediation.

Reference
1. The material for this module is based on the book "Advanced Persistent Threat" by Eric Cole



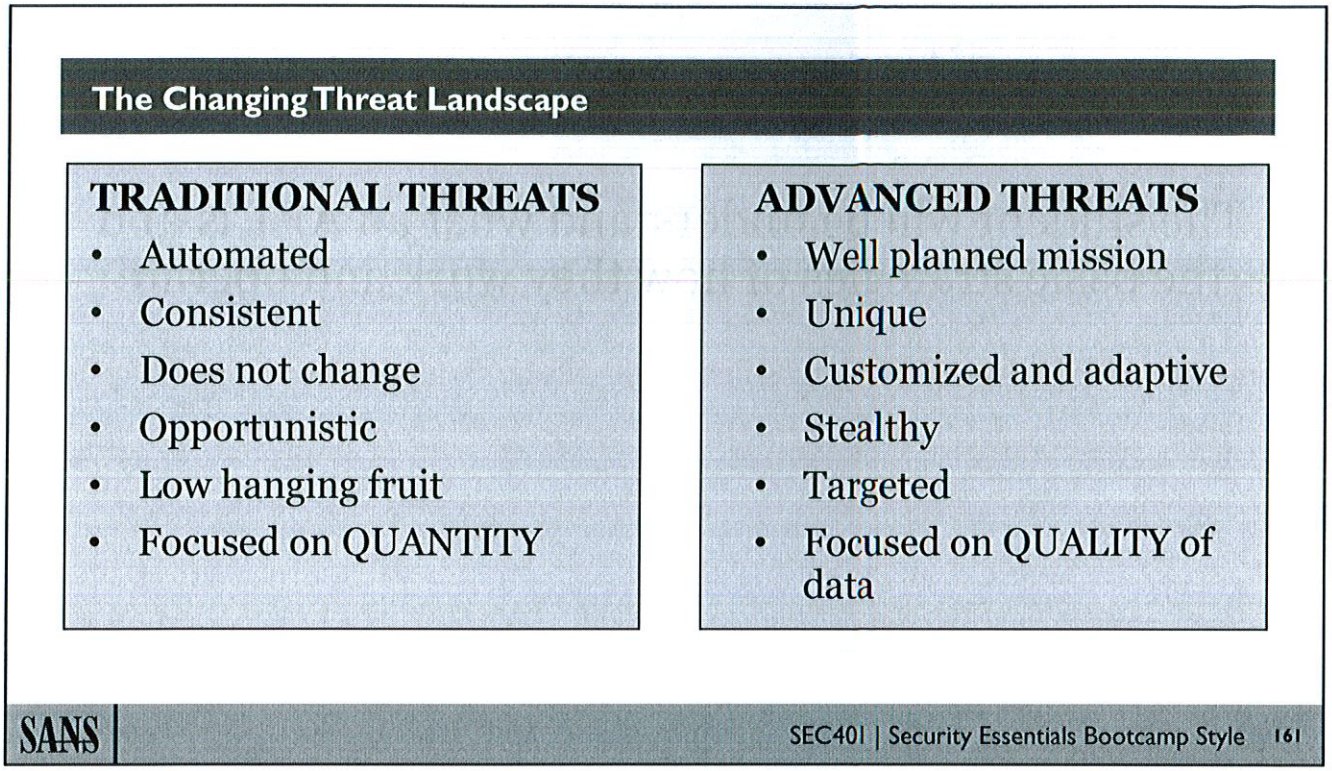
In this module, we will discuss the new threats that exist in cyberspace and effective ways for dealing with these threats. In particular, in this module we will look at the following objectives:

- What are APTs and why are they so hard to manage?
- Defending Against an APT
- How can cyber remediation be approached?
- Offensive Operations



One of the key mottos of a cyber defender is that the offense must inform the defense. It is critical to always understand the threat and use that to drive the risk calculation. Once you understand what the threats are that could have the highest likelihood and impact, you would then focus on what vulnerabilities are utilized to cause damage. If you do not understand the threats and how they work, you might be fixing the wrong vulnerabilities, which would have little impact on reducing the overall risk to the organization. This is why organizations spend significant amounts of money on security, but still get compromised. There could be many reasons for this but one of the primary reasons is because they are fixing the wrong vulnerabilities because they do not understand the real threat and how it works.

APT is a game changer in how the adversary targets, compromises an organization and causes damage. Many organizations approach to security is focused solely on viruses, worms, and traditional attacks. While these are still problems, they are not the only problem. Defensive mechanisms for traditional threats are not as effective against advanced threats because they behave and attack in a different manner. New threats require new approaches to defense.



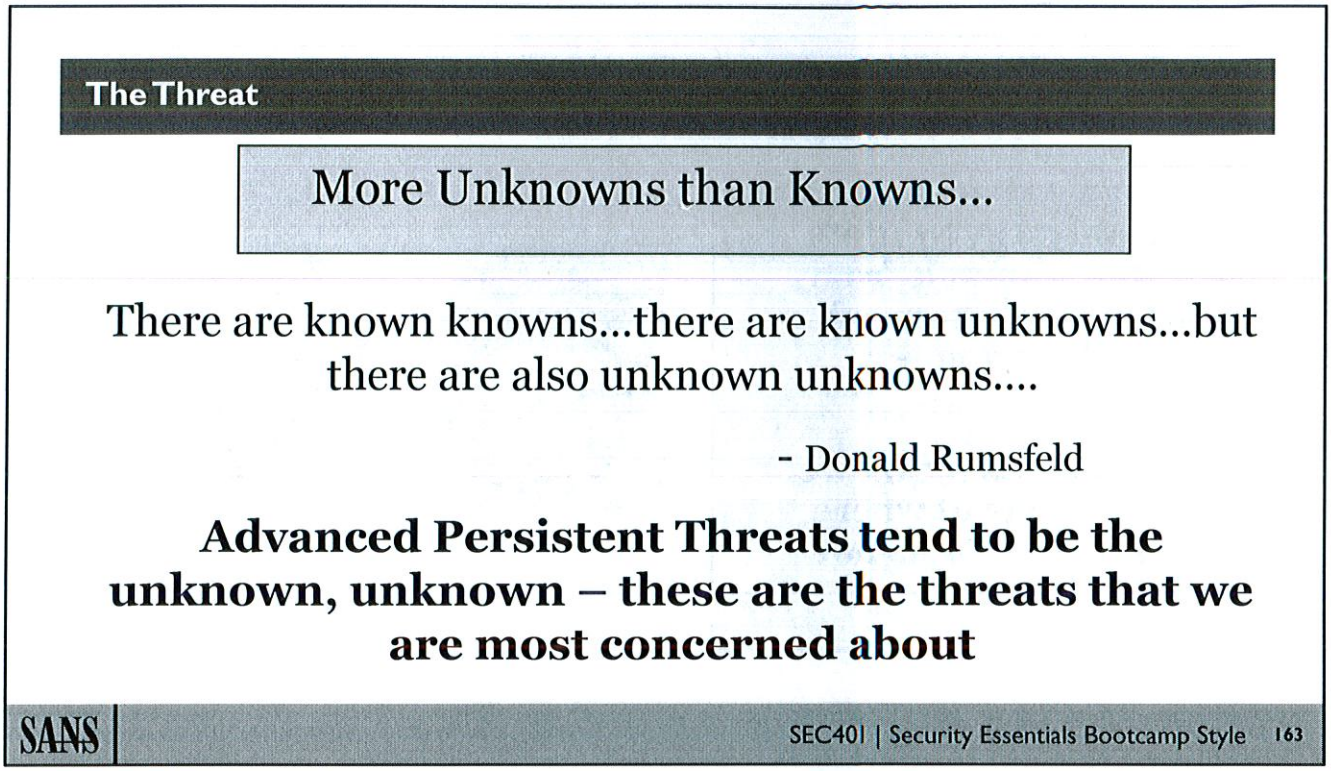
Many organizations and many people started addressing and working in security when the primary threat was traditional threats. While traditional threats can still cause damage and are devastating, they tend to operate in a predictable manner. Traditional threats such as a virus or a worm is written once and always runs in a consistent manner. Typically it will use the same set of exploitation methods to try and break in to multiple systems. Once the exploit code is known, defensive methods can be built to stop future attacks from the same piece of code. With traditional attacks, the adversary is concerned about breaking into any system, not one system in particular. Traditional attacks are often focused on the quantity or number of systems that it compromises. Since a traditional exploit will typically work in the same manner, signature-based detection is a common method of defense.

With advanced threats, it is no longer a piece of exploitation code, but a well-planned mission with humans actively involved. An advanced threat is often targeting a specific organization and will do anything possible to break into that organization. The attacks often have human involvement and instead of targeting a system to break into, it is often targeting a person as part of the attack. Since advanced threats are targeting a single organization, they are often customized, adaptive and unique to the specific organization that they are attempting to compromise. With an advanced threat, they are focused on the quality of the data not the number of organizations that are compromised.

What Are APTs and Why Are They So Hard to Manage?

The student will be understand what an APT is and the basic strategies of how they work and operate

This page intentionally left blank.



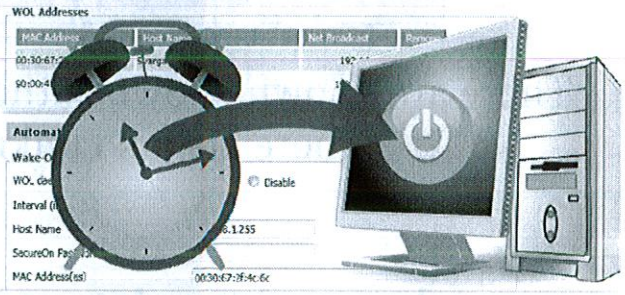
In cyber defense, the threats that should scare you and terrify you the most are not what you know, but what you do not know. In any situation it is always important to ask yourself, what am I not seeing or what am I missing. The way most organizations approach security is by looking at what they know and what they are aware of. While it is important to have a comprehensive approach to security and fix the known threats, advanced adversaries want to be stealthy and undetected. This is the reason why many advanced attacks going undetected for 18-24 months because organizations are not looking for these attacks and therefore will not be able to find them.

In order to win at security, it is important to have a comprehensive approach that looks at all possible threats and defends against them. With known attacks, cyber defense is straightforward because you can deal with the actual method of attack. For example, you can look at the specific method of exploitation and remediate the issue. With unknown threats, it is more difficult because you do not know the specific method of exploitation. Therefore, it is more difficult to deal with because you have to look at the general behavior and remove the vectors of attack.

Current Capabilities

The only way to 100% protect yourself from attacks is to turn off your computer

How scary is it that we live in a world where this statement is not true?



One of the statements that people who work in security use to make is that the only way to 100% protect yourself from attacks is to turn off your computer. The idea was to emphasize that 100% security does not exist if you want any functionality. Essentially, security and functionality have an inverse relationship—as you increase one, you decrease the other. 100% security would imply 0% functionality or a system that has no value. As soon as you increase functionality to be greater than 0, security drops to below 100. Therefore, the logic was that if you turn off the computer and make functionality 0%, you would have 100% security; however, we live in a time and place where that statement is not true. Even if you turn off your computer it can still be attacked.

You might be scratching your head; saying, if it is turned off, there is no way to access it over a network so it cannot be compromised. However, there is a technology called WOL (wake on LAN), that is built into most network interface cards that allows computers to be turned on over a network. The idea behind WOL is to be able to allow remote maintenance or patching after hours. Typically you do not want to patch a system during the day because it can impact operations, so ideally it would be preferred to do it after hours. However, after-hours computers are typically turned off. WOL allows for a signal to be sent to a computer to be able to turn it on remotely. Remember, anything that can be used for good can be used for evil. Therefore an adversary can also use this feature to remotely turn on computers and attack them, even if they were originally turned off by the system owner.

References

1. How to Schedule Your Computer to Wake Up at Specific Times - http://www.howtogeek.com/70084/how-to-schedule-your-computer-to-wake-up-at-specific-times-with-dd-wrt/

APT Defined

APT is a cyber-adversary displaying advanced logistical and operational capability for long-term intrusion campaigns with the goal of exploiting information in a covert manner

It is important to note that there are many different definitions that are used to describe APT. Instead of debating about subtle differences, let's use, as a starting point, our definition of APT to be 'a cyber-adversary displaying advanced logistical and operational capability for long-term intrusion campaigns with the goal of exploiting information in a covert manner'. It is also important to note that many people today refer to APT just as advanced attacks since the persistent nature is implied in having an advanced adversary.

The important point to emphasize is that it is an adversary, a group of people that are planning a mission that is targeting your organization. This is no longer a piece of malicious code that will hit random targets trying to break in. APT is a planned, organized and well-orchestrated attack against your organization. It is important to point out that the "A" in APT does not stand for the advanced nature of the attack; it stands for the advanced nature of the adversary. It is important to remember that if you have an advanced adversary, they are going to use the easiest, quickest, and most effective method to break in. Therefore many types of APT's are actually fairly simple, but they are very effective in how they work and operate.

Another important difference with an APT is that instead of targeting a computer, it often targets a human. It, in essence, is performing social engineering to convince a user to open an attachment or click on a link. Many users who receive an e-mail that they believe came from a boss or a colleague and looks legitimate, the probability of them taking action is very high. Therefore, a lot of APT's utilize various methods of spear phishing. Since every attack is unique and different, this is not inclusive of all attacks, but is meant to give you an idea of how the attacks work in general.

When an APT targets an organization, the ultimate goal is long-term exploitation of the target. They do not want to get in, extract a piece of information, and leave. They want to stay in for a long period of time. Therefore many APT's are meant to be very stealthy, fly under the radar, bypass common security devices and go undetected for a long period of time. This is a reason threat hunting is becoming necessary for all organizations because it is important to take a proactive stance against this aggressive adversary.

APT is often summarized as being stealthy; targeted and data focused are the main qualities that differentiate it from standard attacks.

Reference

1. Advanced Persistent Threat by Dr. Eric Cole

Understanding APTs

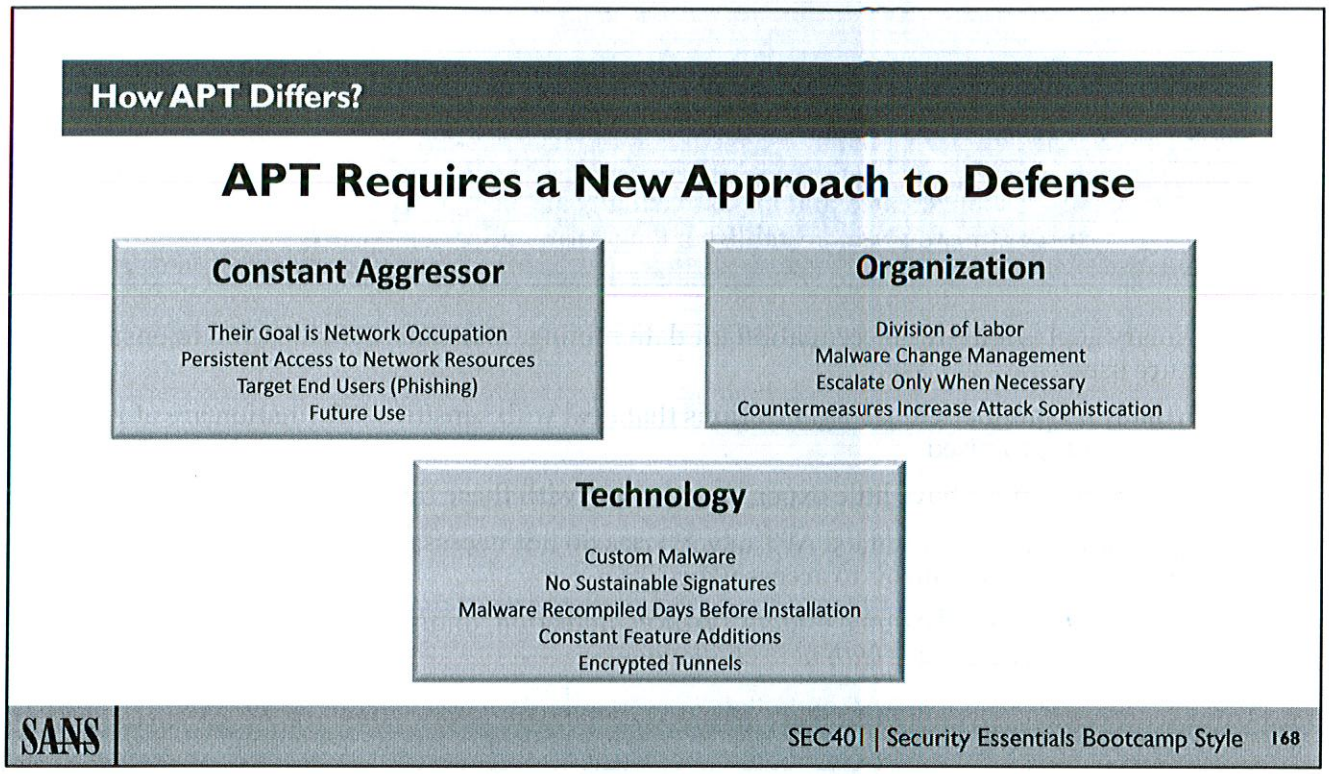
APTs are highly sophisticated and bypass virtually all "best practice" cyber security programs to try and establish a long-term network presence

- Primary goal is long-term occupation for data mining, malicious activities and to ensure future use
- Virtually *all* premier global organizations that deal with sensitive information are at risk of being compromised
- Most organizations have little experience dealing with these threats
- Sophisticated and well-funded APT adversaries do not necessarily need to breach perimeter security controls to access networks
- *APTs are changing the game – identification, detection, analysis and remediation must evolve to keep pace with new challenges*

APT is a game changer in terms of how you approach and implement security. As you will see throughout this module, traditional security measures are still needed to deal with traditional attacks, but are not very effective against advanced threats. This is because advanced threats are very sophisticated and understand what security measures organizations implement and have developed ways to bypass these measures. If the goal of an advanced attack is long term occupation, it is a high priority to go undetected. Therefore being able to slip under the radar and go undetected by "typical" security devices is a key objective of an advanced threat.

While one could argue over the true destructive nature of traditional attacks, many worms and viruses were focused more on being disruptive, but not typically focused on targeting critical data. Advanced attacks are data focused with one of the primary goals being to target and cause harm to an organization by disclosing, alerting or destroying critical, sensitive information.

By understanding how APT's work can allow you to re-design your security. While prevention is ideal, preventing an advanced attack is difficult. Therefore a key focus in implementing effective solutions is timely detection with a primary goal of controlling and minimizing the overall damage. Organizations need to accept that they are going to get targeted by an APT.

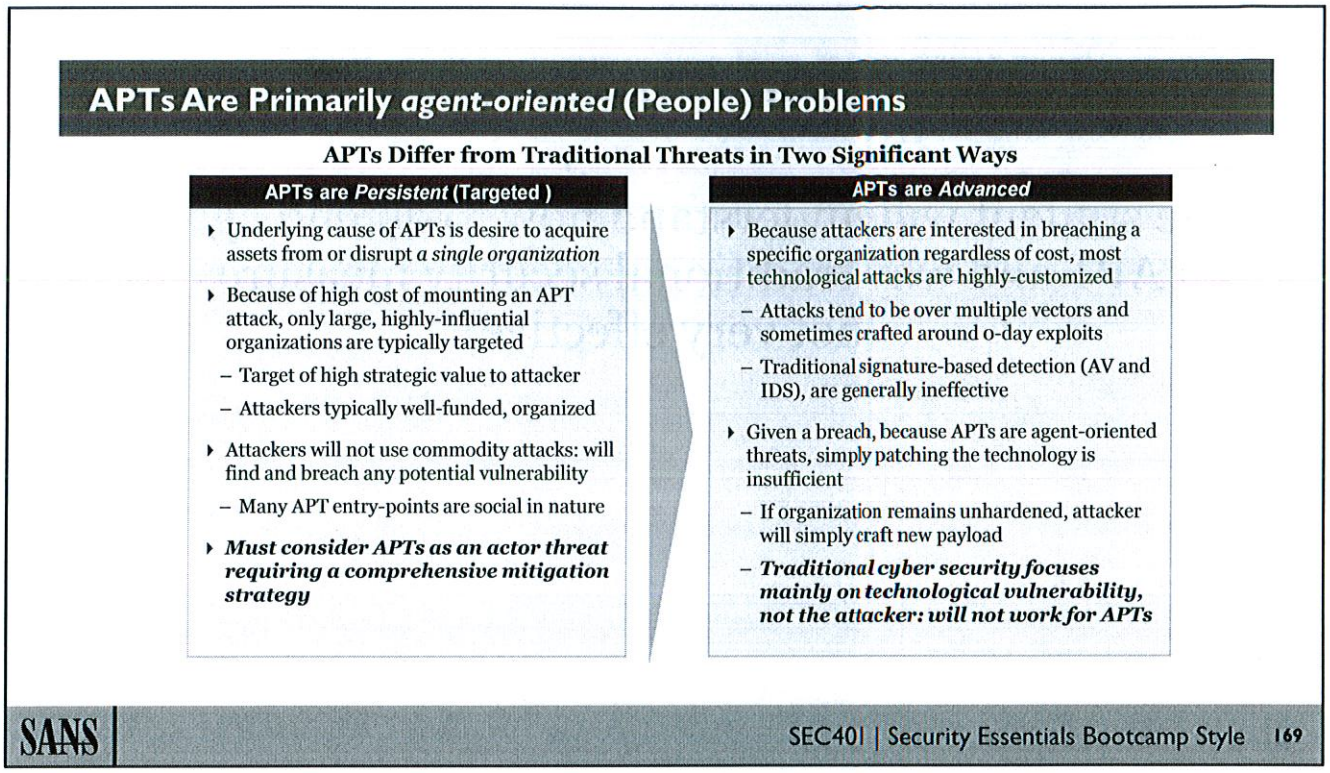


While there are many characteristics of an APT or advanced attack, there are three that stand out. It is important to remember that these characteristics require a new approach to dealing with APT's. Remember that offense must inform the defense. The more you understand how the offense operates, the more effective defensive measures you can implement.

Constant Aggressor – If your organization is being targeted by an advanced threat, in many cases, if you do not take proactive measures, the adversary will get in. This is why some security professionals say traditional preventive measures are just postponing the inevitable. Therefore detection is a critical component of an effective defensive measure with regards to advanced threats. It is also important to note that instead of trying to target servers on the DMZ which are often locked down and difficult to compromise, an APT will go after the end user via known protocols like e-mail. Since most organizations allow e-mail from the Internet to go all the way to a user computer on the private network, this is an effective way to be able to setup a pivot point on the internal network, bypassing many of the security measures that are in place.

Organization – As mentioned early in this module, an advanced attack is not a piece of code like a virus or worm, it is a well-organized group of people who are planning a mission. The mission is to compromise your organization is the most effective manner, using the minimal amount of resources required. This is a professional group of people whose full-time job is to organize and launch attacks. They are very sophisticated and have an array of tools and resources at their disposal.

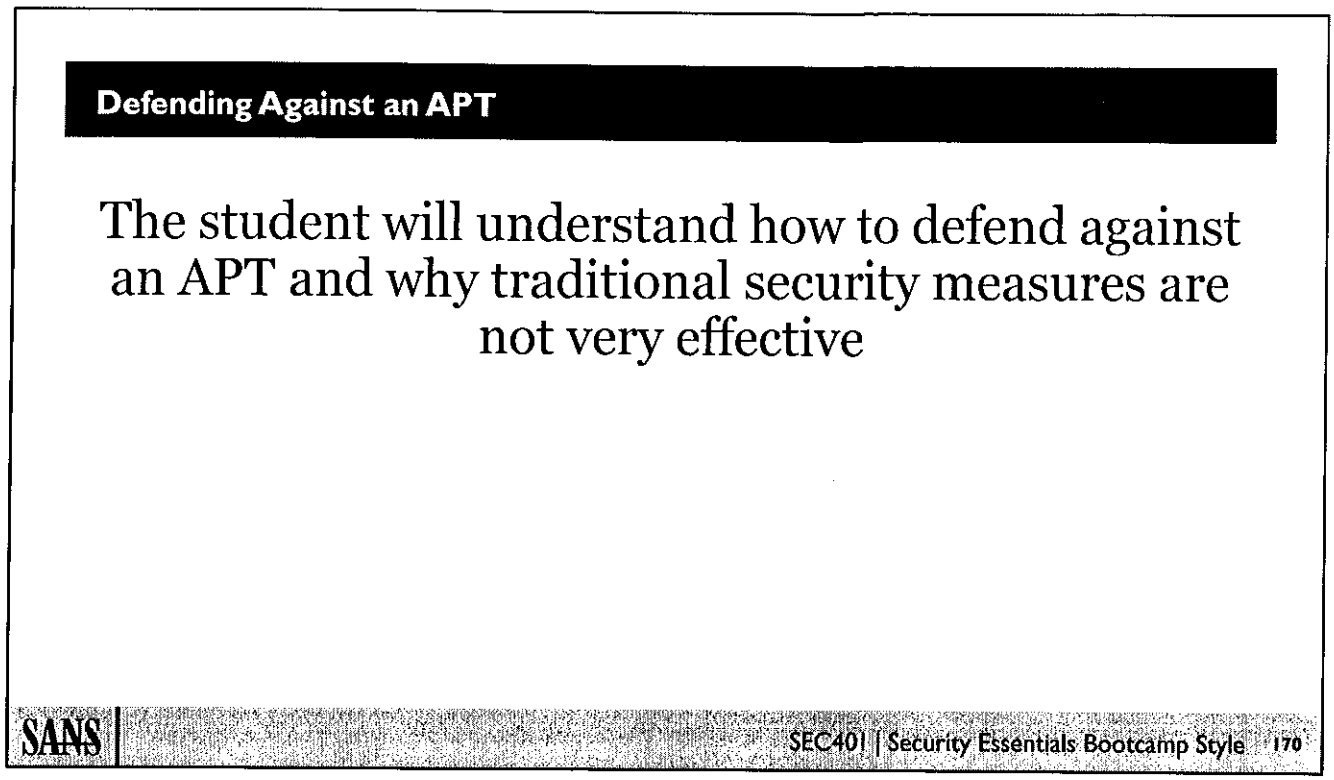
Technology – It is always important to remember that anything that can be used for good can also be used for evil. An APT utilizes technology to be covert and go undetected. Instead of signatures, they now use customized, one-off code to make detection very difficult for the defense. They also utilized one of the favorite tools of the defense – encryption. People often mistakenly say that encryption stops the adversary from reading an organization's information. This is only partially correct. Encryption stops anyone from accessing anyone's information. Not only will encryption stop the offense from reading information, but it will also allow the adversary to go undetected by the defense. Therefore, traditional security measures that utilize payload analysis will go undetected by advanced threats.



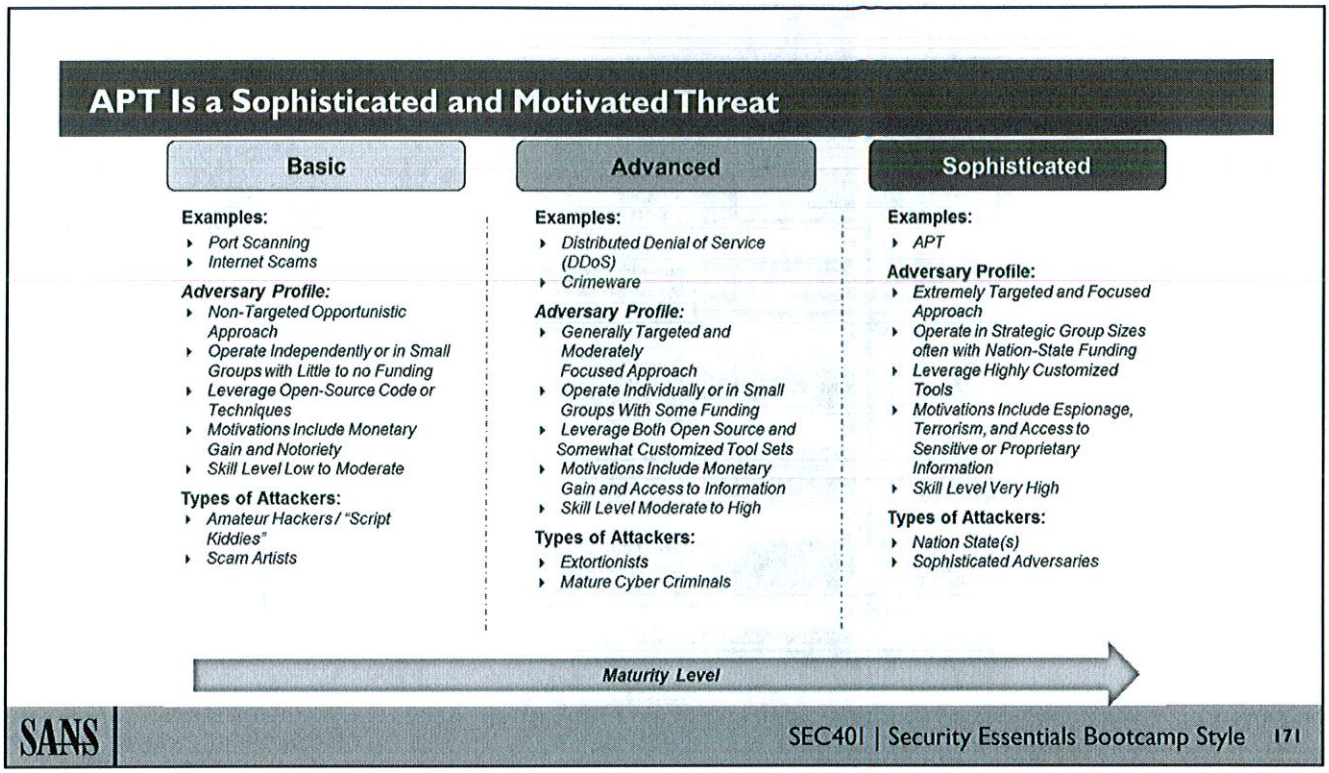
People can debate and argue over the security of various operating systems such as Windows, MacOS, Linux and Unix but at the end of the day, if you put enough energy and effort into how the operating systems work, they can be secure. Configuration management, patching, hardening and locking down services are all effective ways to protect a computer operating system and make it more difficult to compromise. However, advanced threats target a new operating system. One that is very difficult to lock down and secure and almost impossible to patch. This operating system is not made by Microsoft or Apple, it is (wait for it, wait for it), the human operating system (i.e. the brain).

Typically, APT's are people problems because the APT are not trying to find a vulnerability in a piece of software, they are targeting the vulnerable in a human. In most cases, if a person receives a legitimate-looking e-mail from a trusted source, the probability of them acting on the e-mail (i.e. opening an attachment or clicking on a link) is very high. Therefore, controlling and managing what external information makes it to the end user and what actions they are allowed to take are very important in controlling and managing advanced threats.

What makes APT's so difficult to protect against is that an organization is only as strong as its weakest link and based on how well organized an APT is, they will usually find it. Plus APT's are very adaptive. If you find a point of compromise and fix it, unlike a worm that would no longer be effective, an APT, will find a new weakness and exploit it. Therefore dealing with advanced threats requires constant vigilance and focus.

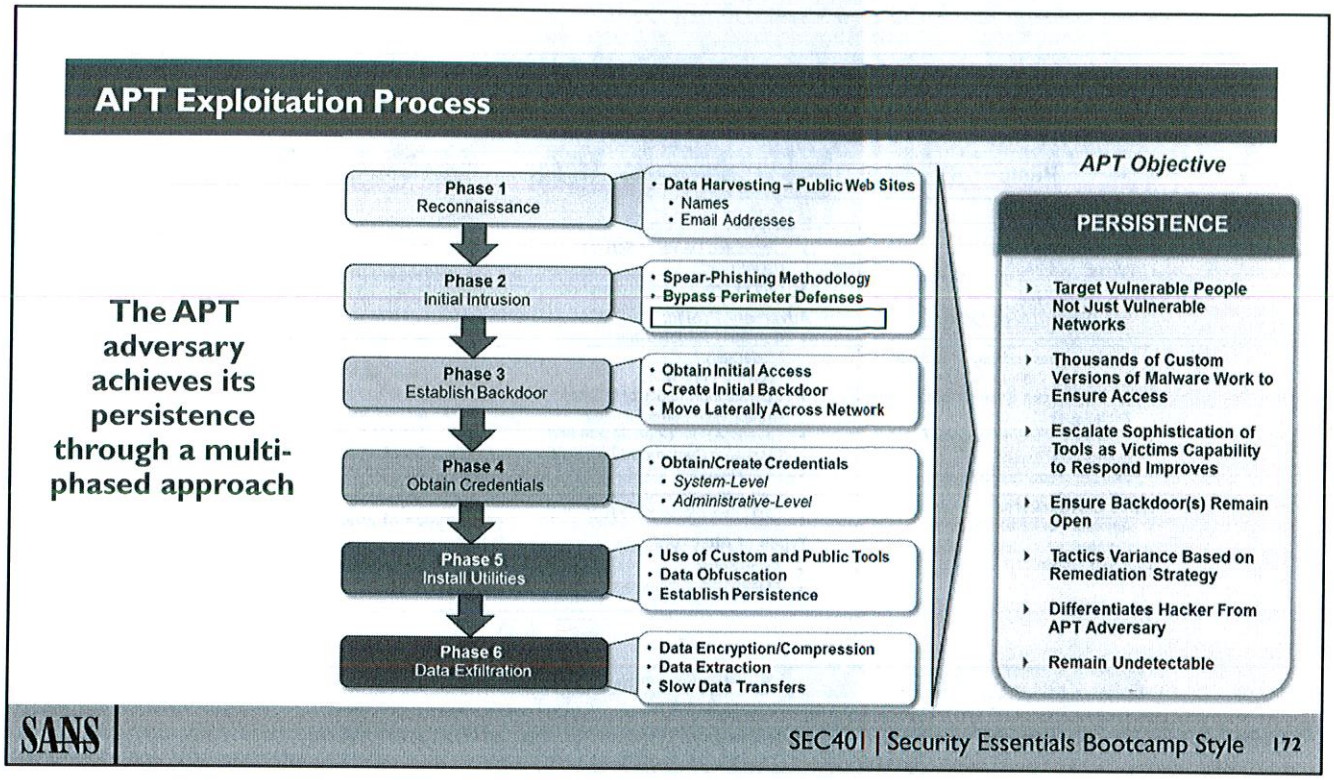


This page intentionally left blank.



The threat is always changing and adapting. Those organizations that are successful at cyber defense are the ones who understand the adversary and are constantly adapting how they implement security. Cybersecurity is a constant game of leapfrog. As the defense develops new methods of defending against the adversary, the adversary does not give up and stop attacking. Instead, they adapt and look at ways to bypass the new security measures that are in place. Just because an organization is secure today, does not mean they will be secure in the future. If an organization deploys a new method of defense and does nothing else, it will work for a period of time. While static attacks will still be blocked, the advanced adversary (who is constantly adapting to the defensive battlefield) will find out ways around the defensive measures and those measures will no longer be effective against all attacks. Therefore, the defense must be as adaptive as the offense. The bottom line is whoever is more adaptive ultimately wins.

To be adaptive, it is important to understand how the adversary is changing over time. Many defensive solutions that organizations deploy are very good at defending against the basic attack. This is your traditional virus, worm, or port scanning malware that is pre-programmed and always works in a predictable way. Since it does not change, if you can defend against it today, the defensive mechanism will continue to work into the future. All the while, these attacks are still prevalent and the reason traditional measures are still needed. The determined adversary will become more advanced. Advanced attacks are more targeted, but is traditionally an individual or small group targeting an organization. The main concern today is sophisticated, organized adversaries known as APT's. This adversary is very targeted and very determined. In many cases, if you are targeted by an APT, the probability that they will get in is very high. Therefore, a sophisticated defensive posture focuses on timely detection and controlling the overall damage.



Traditional attacks are fairly straightforward in how they work and operate. Find a visible IP address, locate an open port, identify a vulnerability in the service, and exploit the system. While this general technique still works and can be used by an advanced adversary, APT's tend to use a more sophisticated, multi-phased approach focused on persistence. One of the main goals of an APT is to be stealthy, targeted and data focused. APT's want to be invisible, bypass all of an organization's defensive measures, and go undetected for a longer period of time.

Note: It is important to point out that there are a lot of different models used to show the process an APT uses. While different models might have slight variations, the general process illustrated below will be the same.

The following are the general phases of how an APT exploitation process works:
Phase 1: Reconnaissance – Focuses in on gathering information and builds a profile of the target. This not only includes information about the organization itself, but also individual targets that can be used to setup an initial pivot point. Typically this phase is based on public or open-source information so it is very hard for an organization to even know it is happening. The main defensive measure for reconnaissance is to limit the information that an organization (and individuals in an organization) make publicly available. With all of the public repositories and social media, this step is very hard to stop. Typically the best an organization can hope for is to minimize the impact.
Phase 2: Initial Intrusion – Once detailed information has been gathered and a target has been selected, the target will receive some sort of message to gain access to the system. This is typically done through some form of direct e-mail or targeting. While there are many variants and they will continue to change over time, executable attachments and malicious links are two of the more common methods. The reason is simple: They work and they are very effective. While APT's have a rich arsenal of advanced attacks, their primary goal is to use the simplest, easiest and most effective way to compromise a system. If a well-crafted e-mail works and accomplishes the mission, why utilized more advanced methods of compromise.

Phase 3: Establish Backdoor – In many cases, the initial point of compromise is not the ultimate system that the adversary is targeting. Therefore, they will often set up a pivot point, which is an initial entry into the system. From there they will perform lateral movement to further compromise additional systems and get closer to the ultimate target. This is an area that adversaries try to be very stealthy in performing this activity. This is also a primary area of focus for network-based threat hunting. Regardless of how stealthy the adversary is trying to be, they are going to generate traffic and if monitored close enough, this activity can be detected.

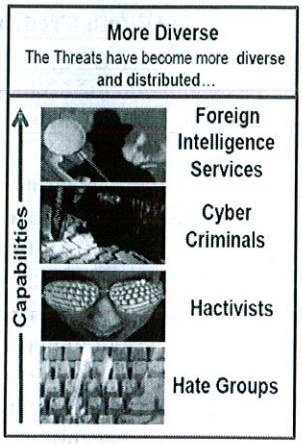
Phase 4: Obtain Credentials – Typically, in order to perform lateral movement or gain access to sensitive information, high-level credentials are often required. This is sometimes referred to as elevation of privileges. Depending on the account that is compromised, the adversary might have limited access. In those cases, the adversary would have to look for ways to gain additional access and obtain privileged access. In some cases, the adversary will find an account that is logged in as administrator and minimal work is needed in this step. While it is important to never log in as administrator and check e-mail or surf the web, unfortunately, this is allowed more often than not, and when it is done, allows for easy access by the adversary.

Phase 5: Install Utilities – In order for an adversary to be stealthy, perform lateral movement and exfiltrate information, they often need to upload and install tools on a system. If the security is really bad, the adversary can sometimes cause harm with minimal to no effort, but often additional tools are needed to exploit or compromise additional systems. While the adversary will try to hide these tools and be covert, the bottom line is the state of the system is changing. This is why application whitelisting in blocking mode is so effective because it makes it very difficult for the adversary to perform this step. Also, from a host based threat hunting perspective, because new tools are installed on a system, this provides indicators that can be identified to determine if a system has been compromised.

Phase 6: Data Exfiltration – Ultimately an adversary is breaking into a system to cause harm. With advanced threats, it usually revolves around exploitation of data, with the most common method being exfiltration of sensitive data. No always, but typically an adversary is targeting an organization because there is sensitive data that they want to access. This can be classified information, research plans, product information or personal data such as credit cards. For this phase, the adversary will typically set up a C2 (command & control) channel to exfiltrate the information and cause significant harm to the compromised organization.

Sample APT Compromise

1. Adversary collects non-traditional attack information
2. The adversary creates a highly socialized, targeted e-mail message that potentially contains previously unknown malicious code – ***spear phishing***
3. If phishing attempt successful, the adversary immediately connects to the victim's workstation
4. The adversary will quickly install additional channels to ensure access to the internal network
5. The APT will quickly entrench themselves at the enterprise level
6. Data is collected and exfiltrated from the network

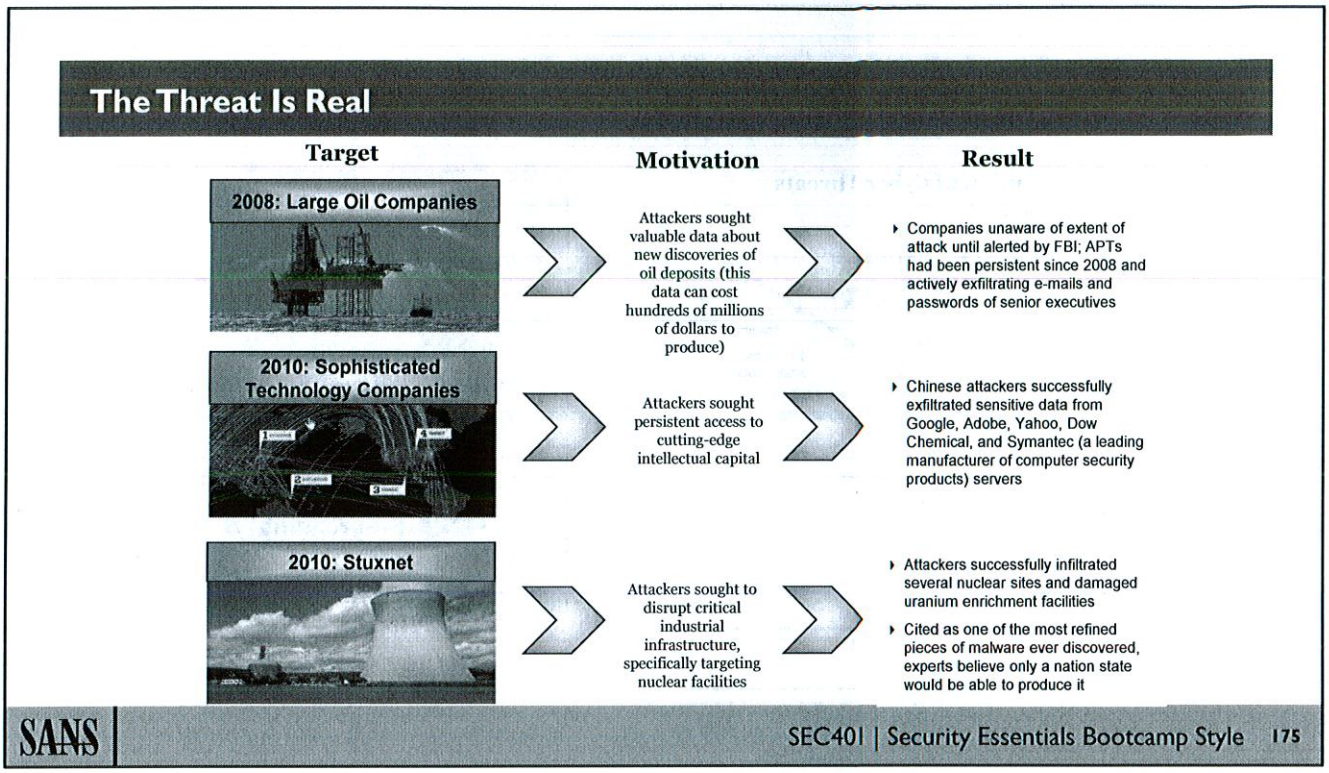


APT's are like humans, everyone is unique and different. Even though there is no such thing as a "standard" or "typical" APT, there are some general attributes that many APT's do follow. Once again, this slide is meant to just give a flavor of what an APT can look like, but everyone is unique and different.

The following are the steps in a sample APT compromise to give you a general idea of how the adversary works:

1. Adversary collects non-traditional attack information
2. The adversary creates a highly socialized, targeted e-mail message that potentially contains previously unknown malicious code – ***spear phishing***
3. If phishing attempt successful, the adversary immediately connects to the victim's workstation
4. The adversary will quickly install additional channels to ensure access to the internal network
5. The APT will quickly entrench themselves at the enterprise level
6. Data is collected and exfiltrated from the network

Typically an adversary will find and target an individual at an organization. The goal is to find someone in which public information is available. Based on social media and public presentations, if an adversary looks hard enough, they can often find this information. Once the information has been gathered, the adversary now puts together the "story" of how the target will be approached and compromised. While the direct method of delivery can vary, typically some form of spear phishing attack is used—this is where a legitimate-looking e-mail is sent to the user with an attachment or an embedded link. If done correctly the user will open the attachment or click on the link and the adversary will be able to gain a foothold on the system. From there the adversary will perform lateral movement, gaining control of as many systems as possible. Once the critical data that is often the goal of the attack has been located, the adversary will set up a C2 so they can communicate with the compromised systems.

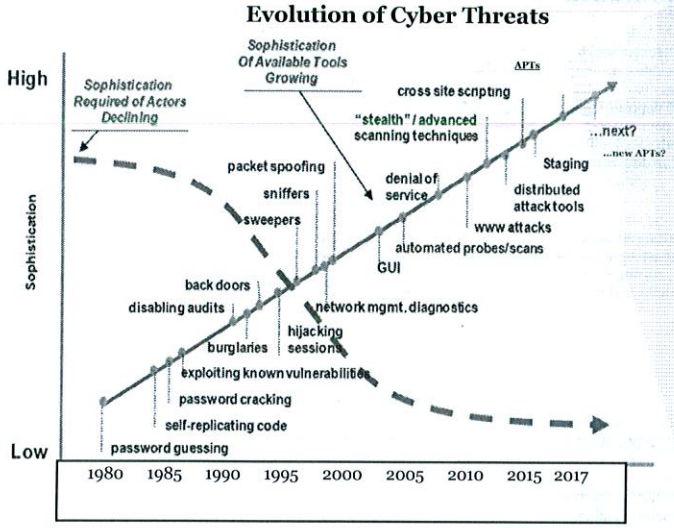


Some of us can probably remember working in cybersecurity over 15 years ago and we would tell people about the dangers that lurk on the Internet. The typical response was, "Prove it to me." Many people, especially executives would say that you are telling me a nice story but show me real, actual attacks. Back then, many of the damaging attacks were not public and the public attacks that were disclosed were more an annoyance and defacement of websites. As such, it was hard for people to believe that there was a real threat.

Today, the threat is real. Just read any newspaper or turn on any news channel and on a regular basis, the topic of breaches and cyber security is being covered. Not only is the threat real, but it is getting worse and worse. Many of us probably remember when 100,000 records compromised was a big deal. Today, millions of records have become a common occurrence.

It is also important to point out that type of industry or size of your organization does not matter. In the past, most attacks were focused on large organizations or government entities. Today from financial services to healthcare to utility companies, no one is safe. Even small companies are targeted because they are often connected to larger organizations and the smaller companies have less security. Adversaries will often find the easiest path of least resistance into an organization. The bottom line is if you are connected to the Internet, you are a target!

The Adaptive Nature of APT's



The number of threats has increased exponentially

Sophistication of attacks has only grown over time

- APTs are a manifestation of this trend
- New, increasingly sophisticated APTs are predicted

With an APT, the goal is to be stealthy and go undetected which means methods that work today will not work in the future. Therefore the adversary has to constantly find new ways to compromise a system. Since most organizations do not do an effective job with filtering or blocking e-mail, this is still a preferred method for an adversary. While breaking into a traditional operating system can be difficult, compromising the human operating system (i.e. the brain) is still very predictable and easy to do.

It is also important to note that while APT's are orchestrated by a group of people who are planning a mission, the tools that they utilize are becoming more and more sophisticated. While this is true, in many cases, because well-crafted e-mail is a very effective method of compromise, the advanced methods of exploitation are not required. However, it is important to mention that as soon as an organization starts blocking or implementing more effective methods of stopping phishing attacks, the adversary already has the next generation of tools ready to go.

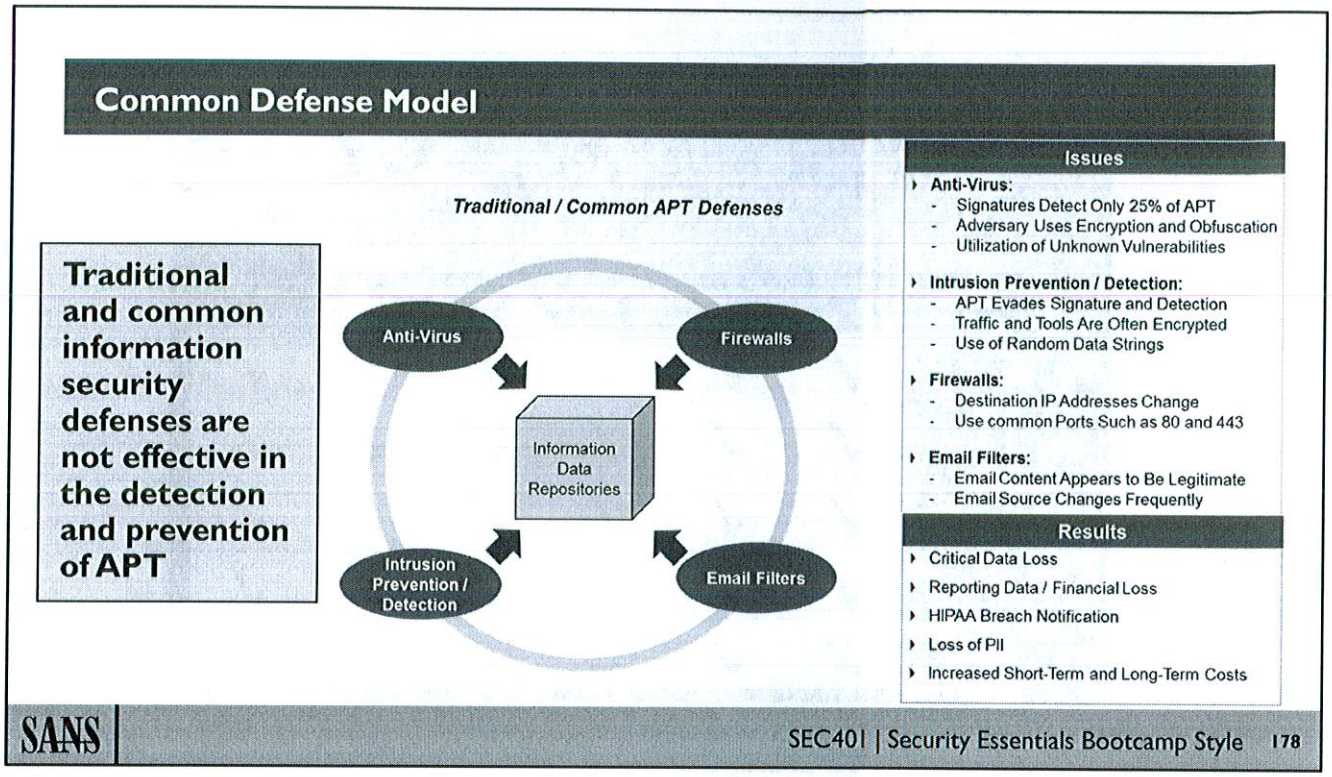
Security Measures in Place During an APT Breach

	Oversight Compliance	Firewalls / Proxy Servers	Host Auditing Enabled	Anti-virus	IDS	Endpoint Software Management
Government	✓	✓	✓	✓	✓	✓
Financial	✓	✓	✓	✓	✓	✓
Fortune 50	✓	✓	✓	✓	✓	✓
Manufacture	✓	✓	✓	✓	✓	✓
Law Firm	✗	✓	✗	✓	✗	✓

In watching numerous breaches on the news, the common question people often ask is, "What security measures did organizations have in place during a breach?" Based on how the media presents it, it is easy to think that organizations were negligent in security. Now, there are some cases in which organizations were grossly negligent in security but in many cases, organizations spent significant money on security and were still breached. Therefore, the question that people often ask is, "What security measures were actually implemented?" This slide is meant to give you a sample of the security measures that were implemented.

The important point of this slide is that there is no single technology that will make you secure. Every technology has a purpose and works well for the purpose it was designed for, but no technology will be effective against all threats. Traditional security measures are very effective at defending against traditional threats but are not as effective against APT's.

There are several important points to keep in mind when reviewing this slide: 1) These security technologies are still very effective against traditional threats that are still a real problem today – therefore this technology is still needed and still required; 2) If organizations did not have these technologies in place, things would be a lot worse; 3) Many of these technologies can be effective against APT's but must be designed and deployed correctly; often these technologies are deployed at an organization and never changed. With proper re-configuration, these technologies can still help defend against many different types of threats.



Traditional security defense measures are still needed on today's networks because they still catch some attacks; however, they are often not configured or designed correctly and are less effective at dealing with APT's. The primary reason is security technology is often designed to solve a particular problem, if the problem changes, the solutions become less effective. In addition, adversaries are very smart, which means they do not want to get caught. Therefore they will study how security defensive solutions work and figure out ways around it.

There are two primary design components of traditional security solutions that make them less effective against APT's: 1) signature based and 2) payload analysis.

Traditional attacks were based on a piece of code, known as an exploit, that ran in a predictable manner. Therefore, the executable code had a signature (unique sequence of binary code) that was unique to each exploit. Since the code did not change, the signature would always be the same. This allowed for security measures, once a new piece of malware was discovered, to develop a signature and use that signature to find all instances of the malicious code. Today's adversary uses unique code for each attack. Even when a similar exploit is needed, the adversary will use obfuscation techniques and re-compile the code, which will no longer contain the original signature. Therefore, signature detection is not as effective against APT's.

Most signatures and malicious content that needed to be analyzed was in the payload of the packet. The IP, TCP, and UPD headers were fairly standard and the malicious content was in the payload. Therefore, most of the traditional security devices perform payload analysis. The advanced adversary recognized this and with APT's the payload will be encrypted. If the payload is encrypted, it will render many of the traditional security devices unable to analyze the payload and render them ineffective at catching APT's.

Traditional Remediation Techniques Under APT's

	Traditional Remediation on Traditional Threats	Traditional Remediation on Advanced Persistent Threats
Password Reset	▸ Attackers have procured user passwords and have active access to user accounts ▸ Password reset removes access to accounts	▸ Password reset temporarily removes access to accounts ▸ Attackers utilize shared accounts to discover changed passwords ▸ Attackers have active access to user accounts again
Antivirus	▸ Attackers have planted common attack vectors on organization computers ▸ Anti-viral software detects and removes such vectors	▸ Antiviral software unable to detect custom-created exploits ▸ APTs require custom-crafted detection and removal solutions
Network Security	▸ Organization enacts strict firewalls and network security to exclude external traffic ▸ Internal access controls prevent wide data breaches	▸ APTs planted internally already open holes through firewall and network security ▸ Attackers have access to user accounts, bypassing internal access controls

Since we have been talking about the ineffective nature of tradition remediation techniques, we thought it would be appropriate to finish up this section by looking at some specific examples of how these techniques are not effective at catching APT's. The three areas we will look at are password reset, antivirus, and network security.

Typically, password resets would remove access to a system or temporarily block access; however, with APT's, the adversary will usually utilize shared accounts or have active access to a user's system. Therefore, even if the account is blocked, they can either reset it and/or they can use another account. Also, system or administrator accounts typically do not have account lockouts and is one of many reasons they are targeted by an adversary.

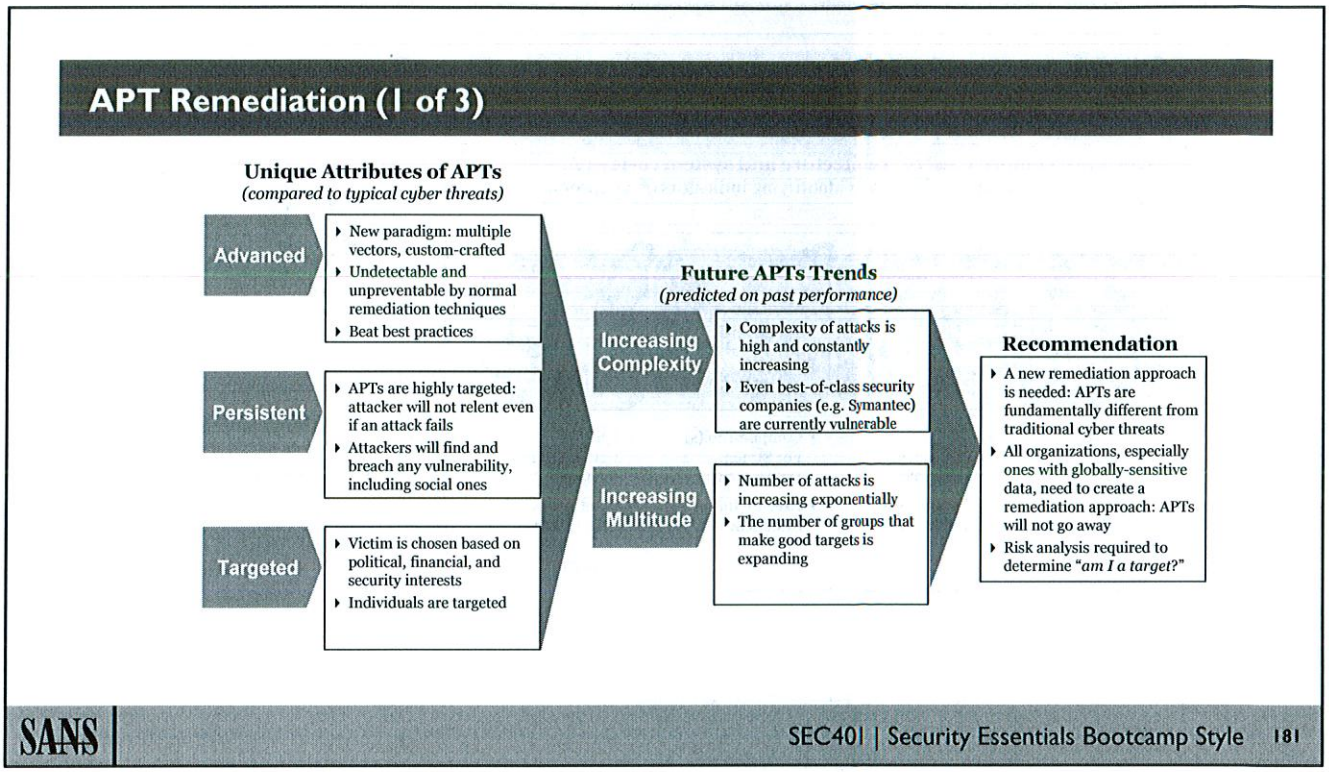
As mentioned on the previous slide, AV and network security devices are based on signatures and payload analysis. By utilizing unique attack vectors with encryption allows APT's to go undetected by these traditional security measures.

Because of the high level of sophistication, traditional cyber remediation techniques are insufficient to address the technological risks posed by APTs

How Can Cyber Remediation Be Approached?

The student will understand different approaches for remediating APT's

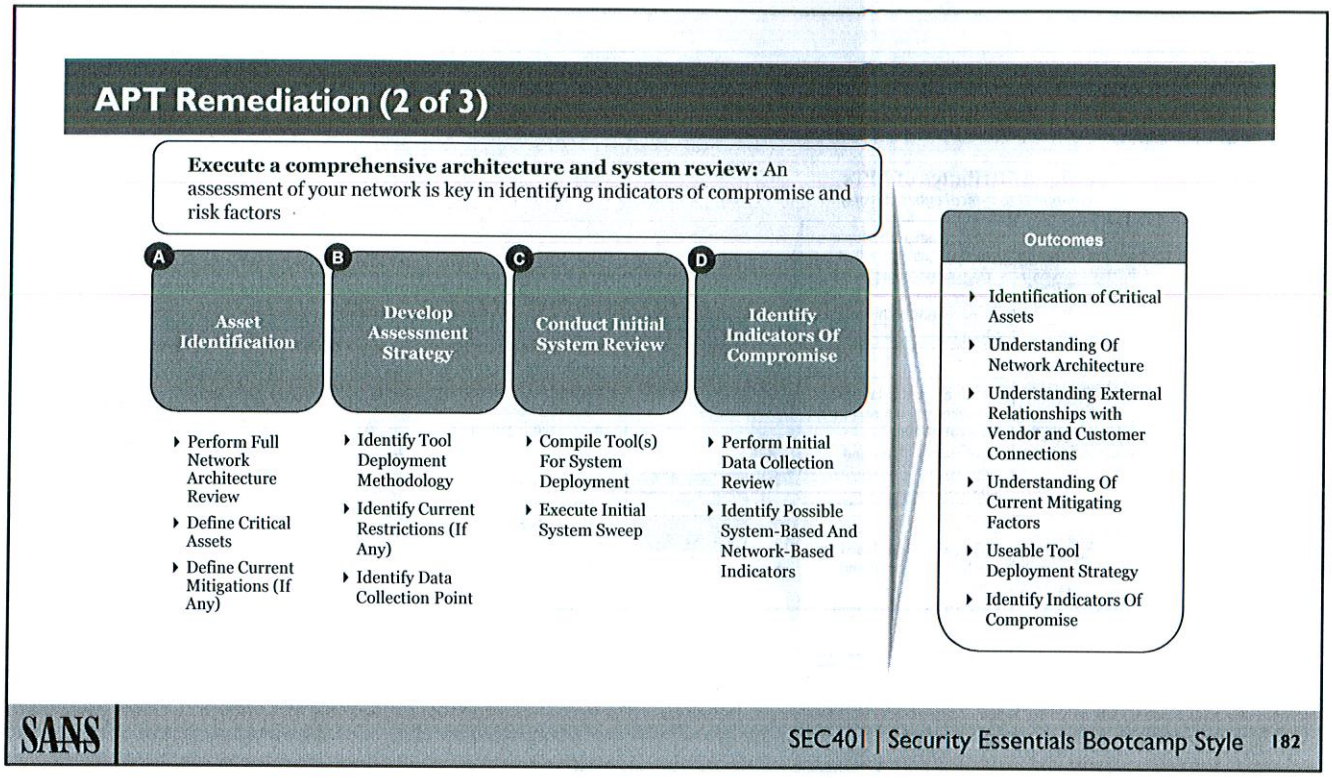
This page intentionally left blank.



APTs require a fundamentally different approach from typical cyber threats. With typical threats, most organizations take an outside-in approach when looking at ways to stop the adversary from getting into an organization with early detection. A typical security architecture involves having external firewalls with NIDS behind it. This provides an initial approach to mirroring prevention and detection together. This also creates a perimeter between the untrusted Internet and the internal network. This idea of a strong perimeter works fairly effective against traditional threats and the approach that most organizations take. While this is a good initial start it has two main problems: 1) it assumes that random malware will hit your site, but not that you are a target; 2) it assumes the perimeter will provide proper protection.

Based on those assumptions, the two questions that have to be asked with APT's are: 1) am I a target?; 2) will my perimeter be breached? In the current threat environment that we work in the answer to both of those questions are yes. Therefore a new approach is needed to deal with security. The important point to keep in mind is how the adversary works. APT's typically will target a user on the private network, perform lateral movement and create and encrypted outbound command channel. In designing solutions to deal with advanced threats, the question that has to be asked is how to with deal with an adversary that breaches the perimeter and in which we cannot analyze the payload because it is encrypted.

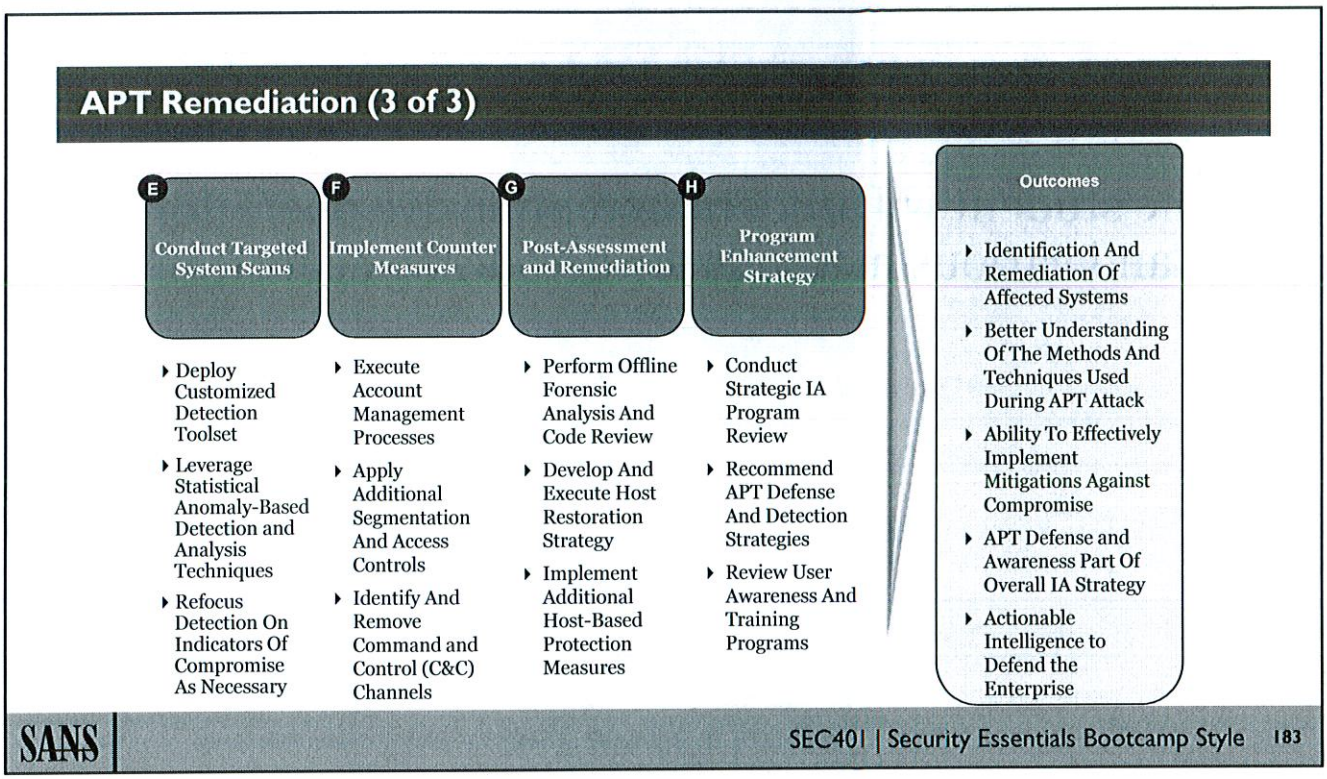
To deal with these challenges, organizations need to create strategic, comprehensive mitigation plans. One of the best ways to learn how to improve is to look at why the offense is winning and adopt some of their tactics. The offense is winning because they have a group of organized people whose mission it is to compromise your organization. What if we took a similar approach? What if you had an organized group of people whose entire mission is to find and combat the advanced adversary?



In the previous slide, it was mentioned that most organizations take an outside-in approach in dealing with the adversary. However, with advanced threats, what if we start with an inside out approach, focusing on what can and will cause the most damage? What is the difference between a major breach and a minor breach? Contrary to what many people think it is not because someone breaches your perimeter. Regardless of whether your perimeter is breached is not directly related to the impact it has on your organization. It is focused on the data that was compromised. The difference between a major and a minor breach is the type of data and the amount of data that was compromised. As Dr. Cole states many times in his lectures, it is all about the data.

If the data is the key component to proper security and it is what the adversary is ultimately after, shouldn't a remediation plan start with asset identification? If you do not know what your critical data is, where it is located, and how you can protect against it, then, the first question that must be asked is ultimately what will cause the most harm to your organization, and then, what is the information that the adversary is going to target? The answer to those questions should be similar if not the same. It is important to note that during this first stage of asset identification, an organization will need to develop multiple plans. In many cases, if the organization is not properly designed and segmented, the long term plan will be designing and implementing a new network architecture, but this could be a significant undertaking. Therefore, you would also develop a shorter term plan to provide the monitoring and protecting of the key data flows.

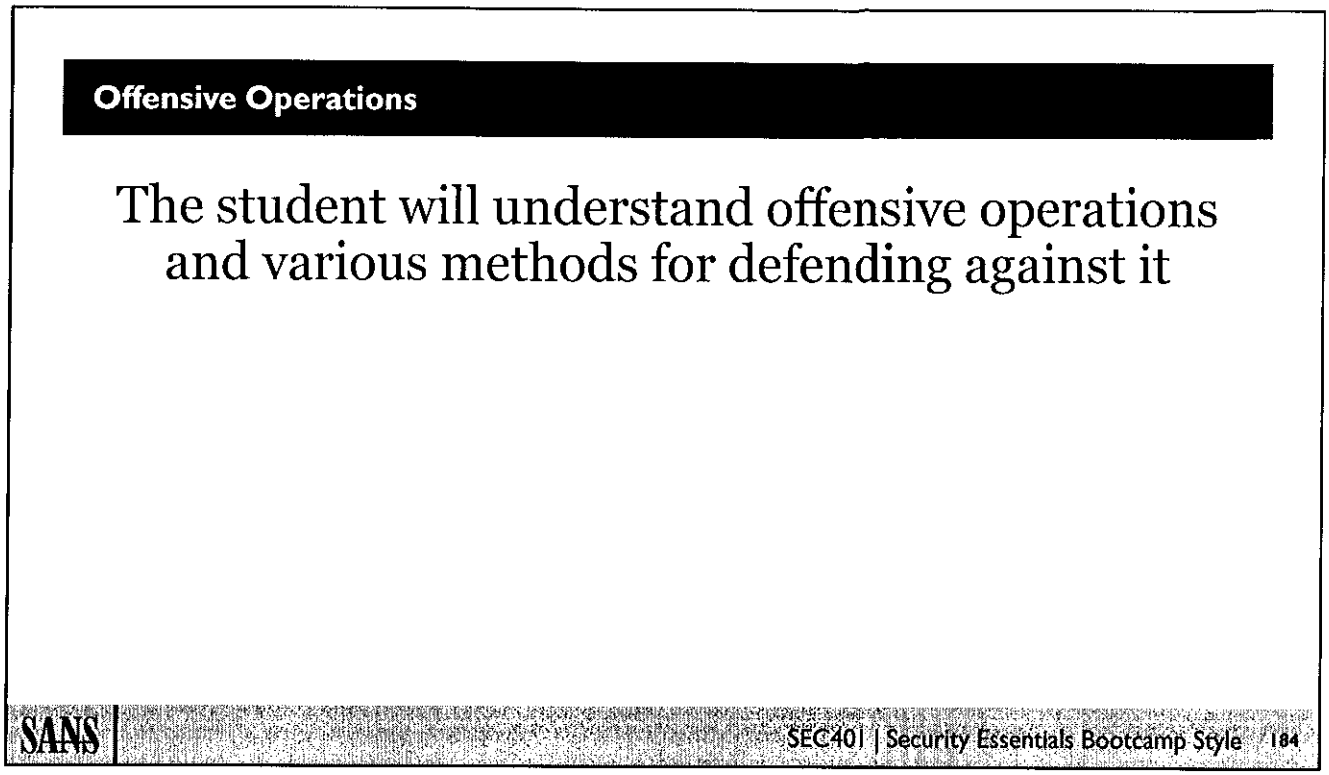
Once the critical data is identified a plan of attack known as an assessment strategy needs to be developed. At this stage, it is important to make sure that you have the proper resource (both people and tools) to properly assess and identify exposure points. An initial system review follows the development of the strategy to start to understand the environment. The goal is to look for not only vulnerabilities but also potential points of compromise. At this stage of assessment, is closely tied with threat hunting. The idea is to not only minimize future damage but also determine if you are already compromised to stop the damage that is occurring at the present.



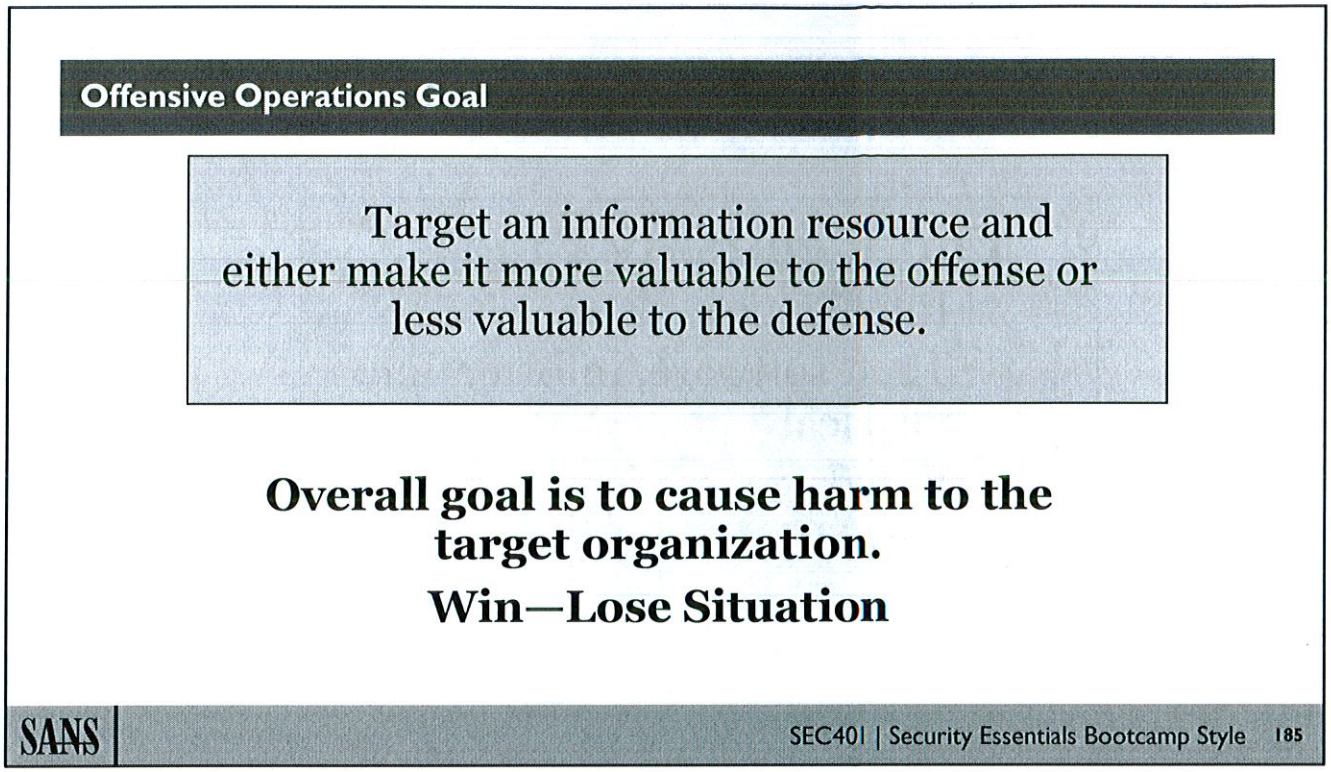
Once the initial plan is put together and analysis begins, it is followed by targeted system scans and analysis. Since an organization has limited resources, it is important to always prioritize and de-scope the problem. One of the challenges is that organizations try to do too much and end up doing very little. Instead of trying to secure the entire organization and scan every system, focus on the most critical business processes and key data stores.

Understanding the problem and exposure points is important, but ultimately you want to deploy countermeasures to minimize the impact a compromise has on your environment. Remember that the countermeasures should be broken down into items that can be done short term, medium term, and long term. Having a great plan that takes 5 years to implement is not acceptable because you cannot wait that long to be secure. It is also important to make sure that any countermeasures are reasonable from a budget and resource perspective. Some effective countermeasures for dealing with APT's are segmentation, access control, host hardening, application whitelisting and outbound proxies.

With APT's it is important to recognize that prevention is not 100% and that you will be compromised. Therefore part of the remediation plan has to involve forensics and cleaning up the system. Remember that the goal of security is to control and minimize the damage which requires timely detection and response. Precise planning will allow for strategic removal of APT and help reduce the risk of any future system compromise.

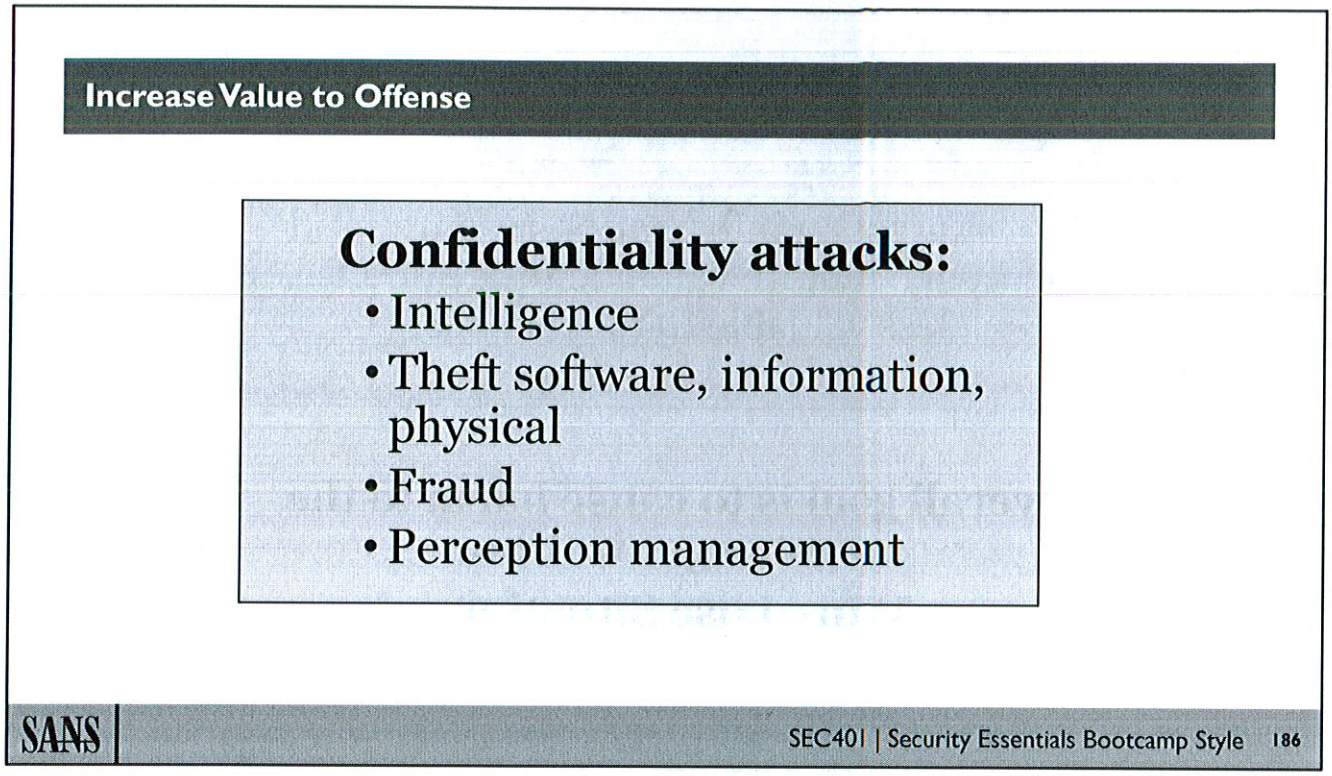


This page intentionally left blank.



Although poker fits nicely into the notion of zero-sum—whatever one poker player wins, another (or the rest) loses the exact same amount—this doesn't apply so neatly to business, economic warfare, and information warfare. Intense competition drives us to search for non-zero-sum solutions. In the book, *NONZERO: The Logic of Human Destiny*, the author Robert Wright leads the reader to believe that all parties will seek win-win situations. This is *not* the mantra of the information operations worker. We have the opposite perspective. We win; you lose, but perhaps not in zero-sum fashion. In our world, you size up your opponent with reconnaissance, and you find an information resource opportunity with the goal of making it more valuable to you, less valuable to them, or both.

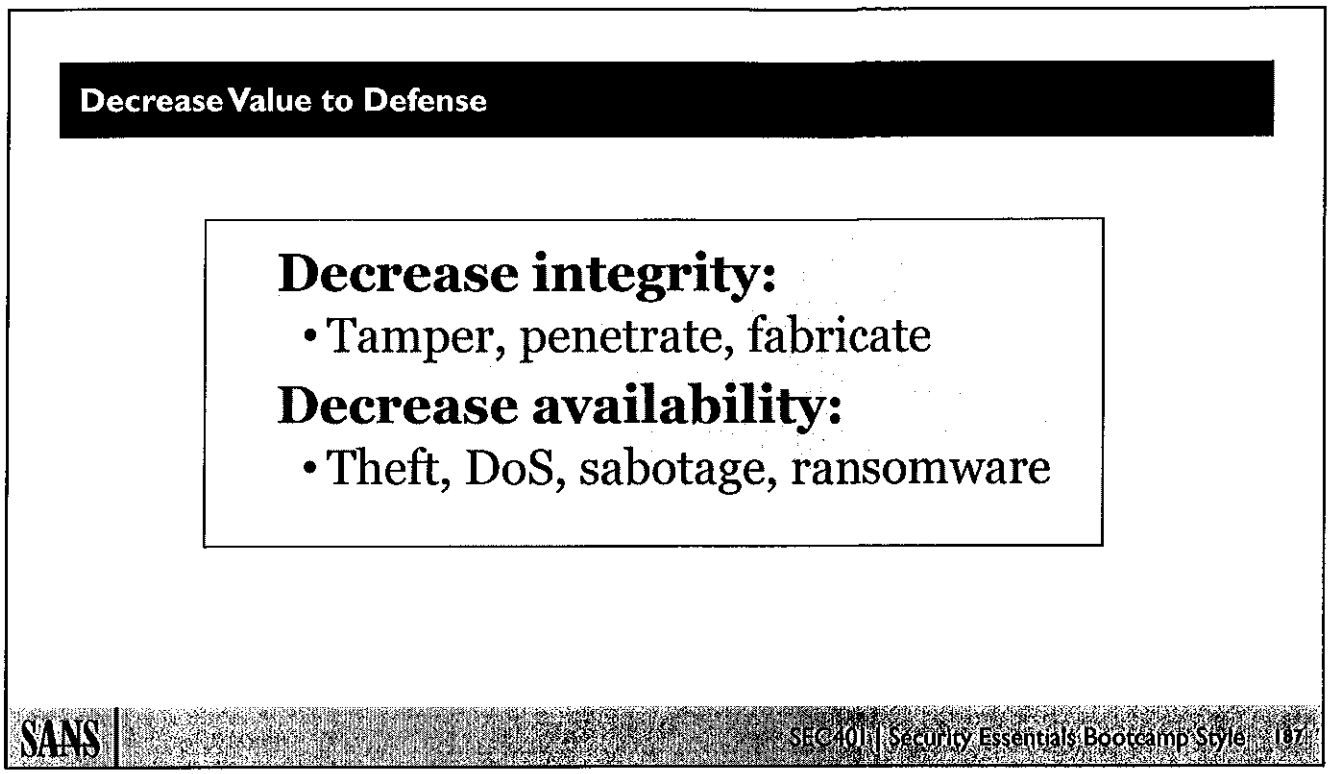
You do (or better said, "you ought to do") the same thing if you are on defense. You size up your own organization, you find the targets of opportunity, and you determine how as an attacker you could co-opt these. Then, of course, you design and implement the countermeasures to make such an attack harder to accomplish.



The attacker is the offense. One of the goals of attackers is to take a piece of data/intellectual property your organization owns and steal it. When the attackers steal a piece of sensitive data, they are increasing the value of the information to themselves. However, in doing so, they have decreased the confidentiality of your information so they can obtain benefit from it.

What is a pound of information worth? This phrase was coined by Perry Luzwick, a DoD information warfighter in the military at the time he said it. He used this phrase to entitle a paper that SANS helped research and develop. If you had a dollar in your wallet and I took it, you would not have the dollar, so you would know that it was missing. If the offense is able to acquire access to the opposition's information, he might or might not know that it is missing. If he does not know the information has been accessed, there are a number of possibilities that could occur.

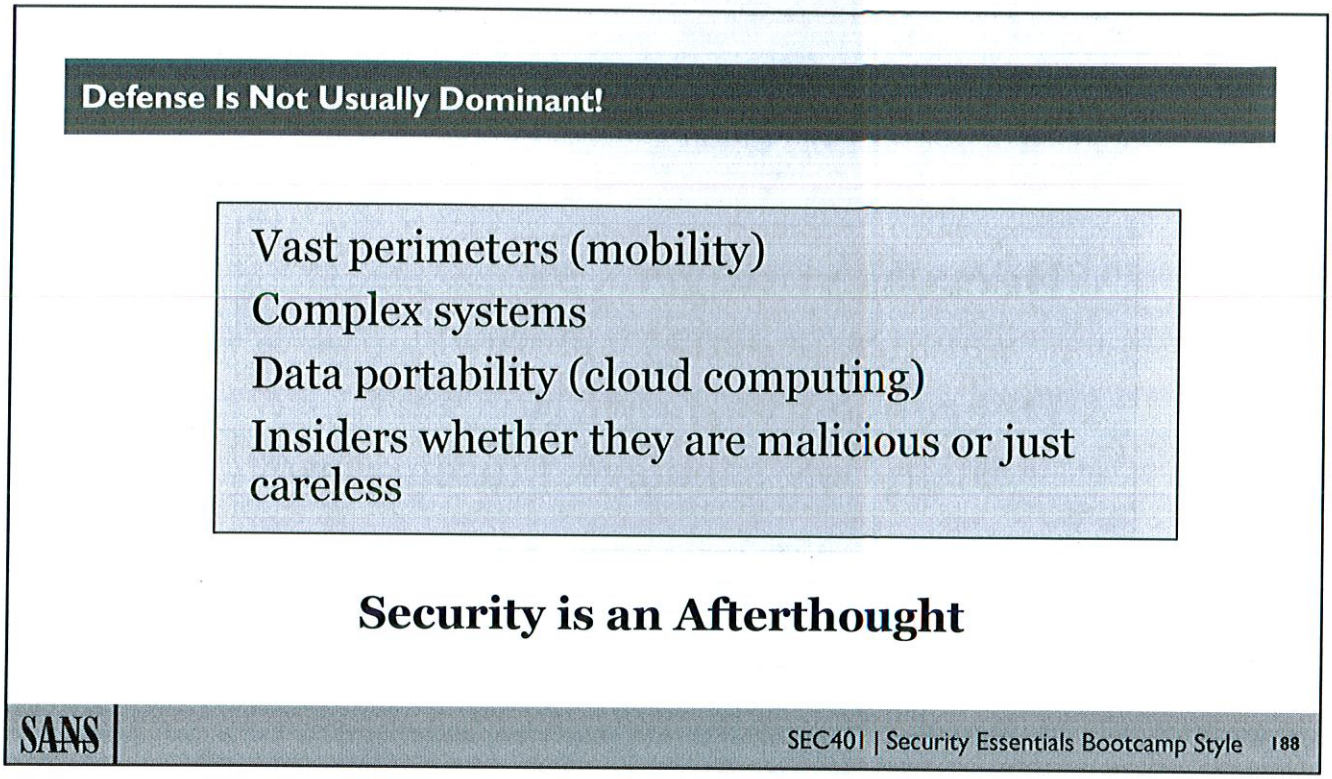
It turns out that there are very few models to accurately quantify the value of information, and yet, intuitively, we know information is at the heart of the value of a company. In the SANS incident-handling course, we teach the notion of critical program information—the crown jewels of information that make or break a company. This is a central concept in information operations as well.



On the defensive side, the challenge is to protect the value of our information and hopefully reduce the value of the offense's assets. The most common approach to defense is to establish effective perimeters and seals to maintain and to be able to validate integrity. If the information containers are breached, then a number of things can happen—most of them bad.

Availability has been a whole new ballgame officially starting in the year 2000, with DDoS attacks against yahoo.com and cnn.com; however, there is evidence of serious denial-of-service activity going back to at least 1997. Availability is a simple way to decrease the value of the defense's assets and requires fairly little sophistication. On the other hand, to defend against a large denial-of-service attack is challenging. Ransomware attacks are becoming a norm, which directly focuses on availability of information.

If the attacker can penetrate system integrity and copy the information, the defender might never know a breach has occurred. This leaves the defense open to make predictable moves. The attacker might also modify the information, perhaps leading to mistakes that also decrease the value of the defensive player's own information assets.



Although defensive dominance, like win-win non-zero-sum games, is a nice thought, it seems unlikely in the real world. The advantage belongs to the attacker. We are all sitting ducks in some senses. If you think in terms of each IP address potentially having 65,536 TCP ports and the same number of UDP ports to defend, you realize the perimeter we have to defend is really big!

Defense-in-depth is one of our most important tools as a defender. If we can learn to think like the attackers, to see through their eyes even just a bit, we can be alert to the opportunities to attack us asymmetrically and attempt to place defenses to prevent those. Sometimes we might be able to use tools like honeypots to understand their attacks and motives, and we might be able, on occasion, to release some incorrect information and cause the attackers to make a predictable response.

However, if the odds are in the favor of the attacker, then we need to learn to attack. Clearly we want to stay well within the law and ethics of our country and profession, but, hopefully, this module has given you the foundation to think about competitive intelligence. The principles will apply nicely.

Dumpster diving *per se* might be beyond the pale of legal, but companies post tremendous amounts of information on their Web pages and in their papers. Perhaps the best way to gather information on a company, though, is to pose as a potential customer; salespeople seem to feel obligated to share everything they know.

Asymmetry and non-zero-sum game opportunities are out there. One last book to consider is Malcolm Gladwell's *Tipping Point: How Little Things Can Make a Big Difference*. It is not a book on information warfare, but it is an interesting twist on asymmetry. No one has yet solved the problem of teaching people to think in a non-zero-sum fashion step-by-step, but it can be done.

Summary

- If you are connected to the Internet, you are a combatant
- Organized crime is becoming increasingly interested in cyber methods
- Expect blended attacks that combine cyber and physical means
- Defensively focus on risk reduction
- Cycle time needs to be as fast as the attacker
- Plan business continuity in the event of DDOS disruption

As demonstrated by the examples and anecdotes provided in this chapter, the threat of being a victim of an APT is real. Whether a direct target of attack, a target of opportunity, or randomly chosen, everyone is at risk. Further, the level of risk is expected to increase in coming years.

The aggressor's goal is to achieve the greatest impact from the least investment possible (asymmetrical result). Fortunately for the adversary, the ongoing expansion of network perimeters and increasing system complexity aids greatly in the impact available from a relatively minor investment.

The defender's goal is to increase the required investment to a level where the target is no longer attractive. Although achieving a true zero-sum result is unreasonable to expect, there are tools and techniques that can be used to begin approaching balance. Your task is to understand and use these tools to your best ability.

"As usual, SANS courses pay for themselves by Day 2. By Day 3, you are itching to get back to the office to use what you've learned."
Ken Evans, Hewlett Packard Enterprise - Digital Investigation Services

SANS Programs
sans.org/programs

GIAC Certifications
Graduate Degree Programs
NetWars & CyberCity Ranges
Cyber Guardian
Security Awareness Training
CyberTalent Management
Group/Enterprise Purchase Arrangements
DoDD 8140
Community of Interest for NetSec
Cybersecurity Innovation Awards

Search SANSInstitute

SANS Free Resources
sans.org/security-resources

- E-Newsletters
  NewsBites: Bi-weekly digest of top news
  OUCH!: Monthly security awareness newsletter
  @RISK: Weekly summary of threats & mitigations
- Internet Storm Center
- CIS Critical Security Controls
- Blogs
- Security Posters
- Webcasts
- InfoSec Reading Room
- Top 25 Software Errors
- Security Policies
- Intrusion Detection FAQ
- Tip of the Day
- 20 Coolest Careers
- Security Glossary

SANS Institute
8120 Woodmont Avenue | Suite 310
Bethesda, MD 20814
301.654.SANS(7267)
info@sans.org