Welcome to Security Lifecycle – Detection.

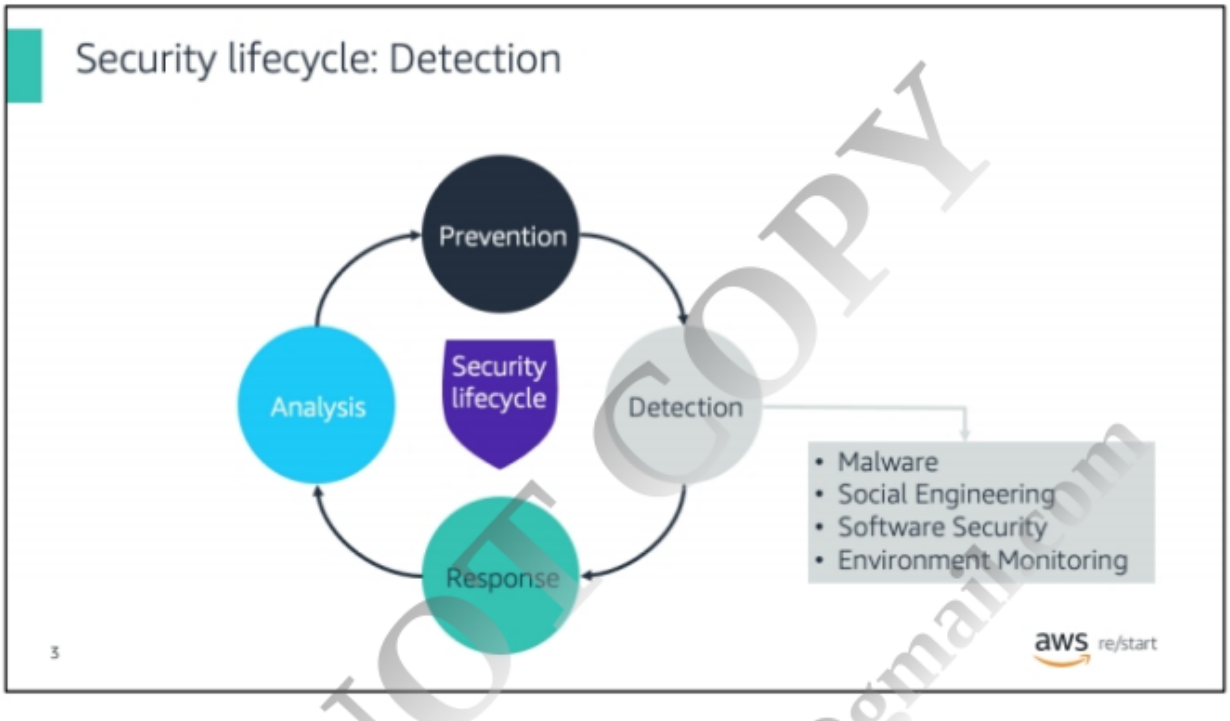
What you will learn

At the core of the lesson

You will learn how to:

- Explain what malware is and the different types of malicious programs
- Explain how to protect against malware threats
- Identify the goals of social engineering and list common social engineering attacks
- Identify measures to prevent social engineering attacks
- List security problems affecting applications
- Identify software development security principles to reduce the risk of exploitation

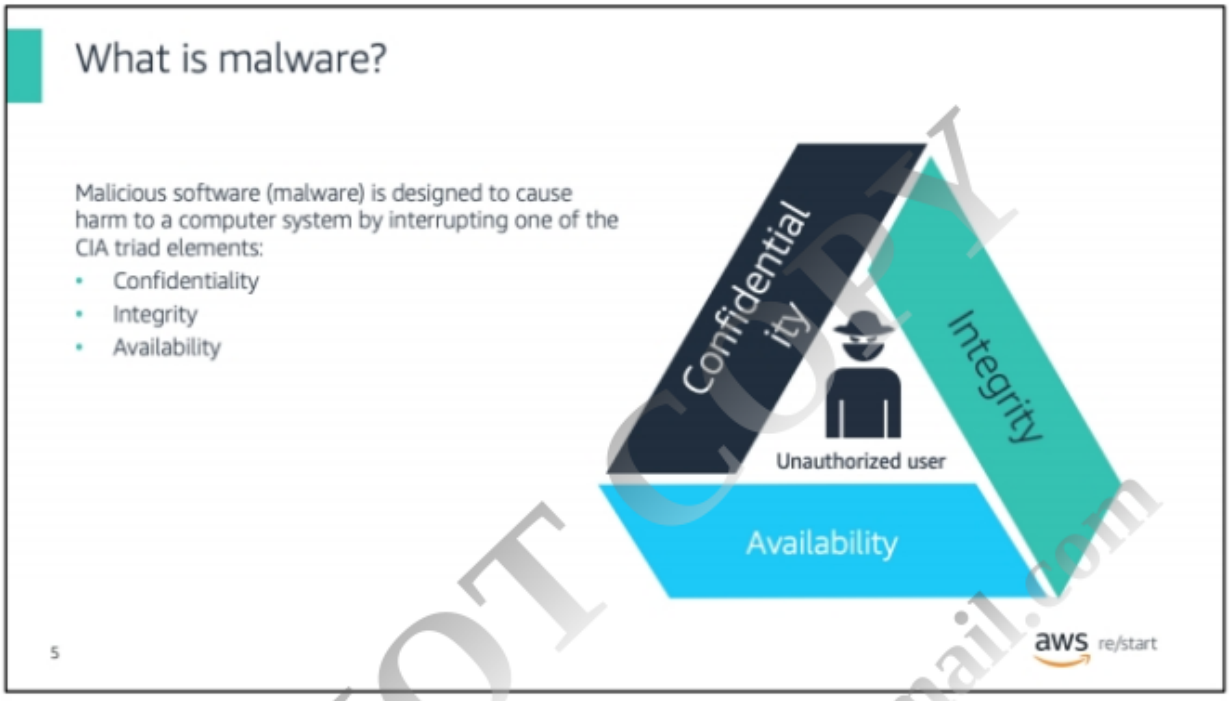
aws re/start



As a review, the phases of the security lifecycle consist of:

- **Prevention** – The first line of defense.
- **Detection** – Occurs when prevention fails.
- **Response** – Describes what you do when you detect a security threat.
- **Analysis** – Completes the cycle as you implement new measures to prevent the incident from occurring again in the future.

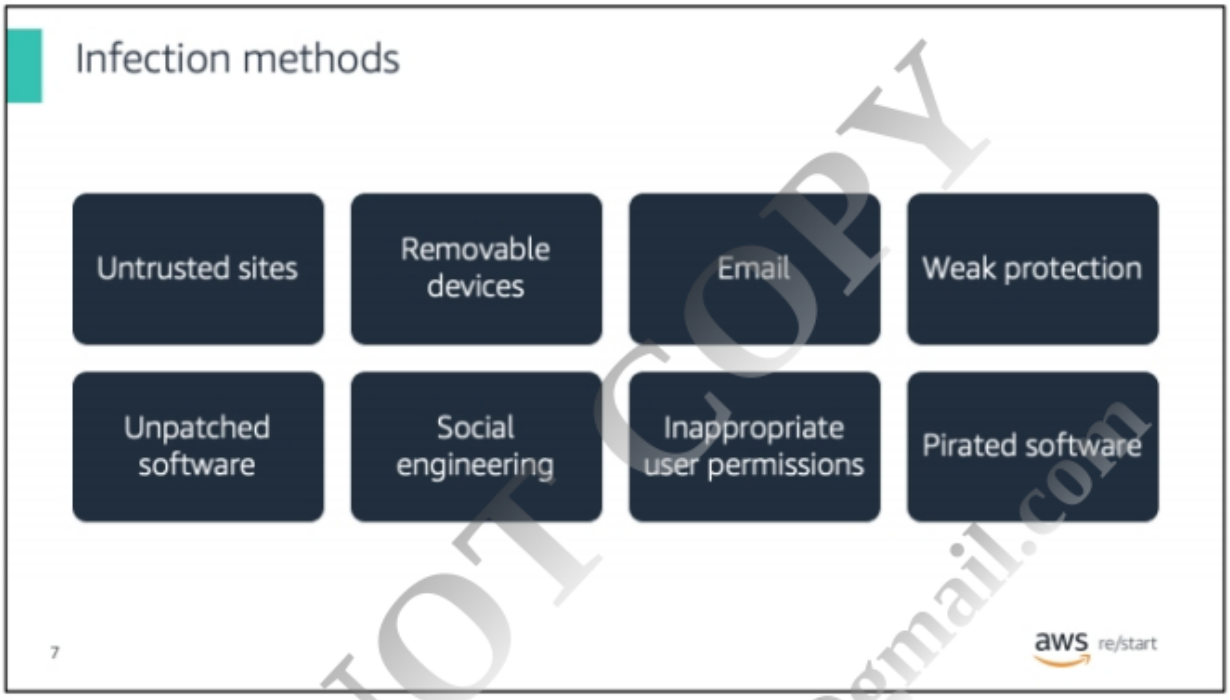
In this lesson, you will learn about the *Detection* phase of the security lifecycle. The topics cover monitoring and detecting an attack that gets past the security controls that are implemented as part of the prevention phase.



Malware is an application that causes harm to a computer system. It interrupts one or many of the CIA triad elements: confidentiality, integrity, or availability. Knowledge of malware, how to avoid infection, and how to respond to corrupted systems are key elements of security management.

Computer anti-malware includes McAfee, Norton, Kaspersky, and AVG. Device anti-malware includes:

- Kaspersky Security Cloud
- Kaspersky Total Security
- McAfee LiveSafe
- McAfee Total Protection
- Bitdefender



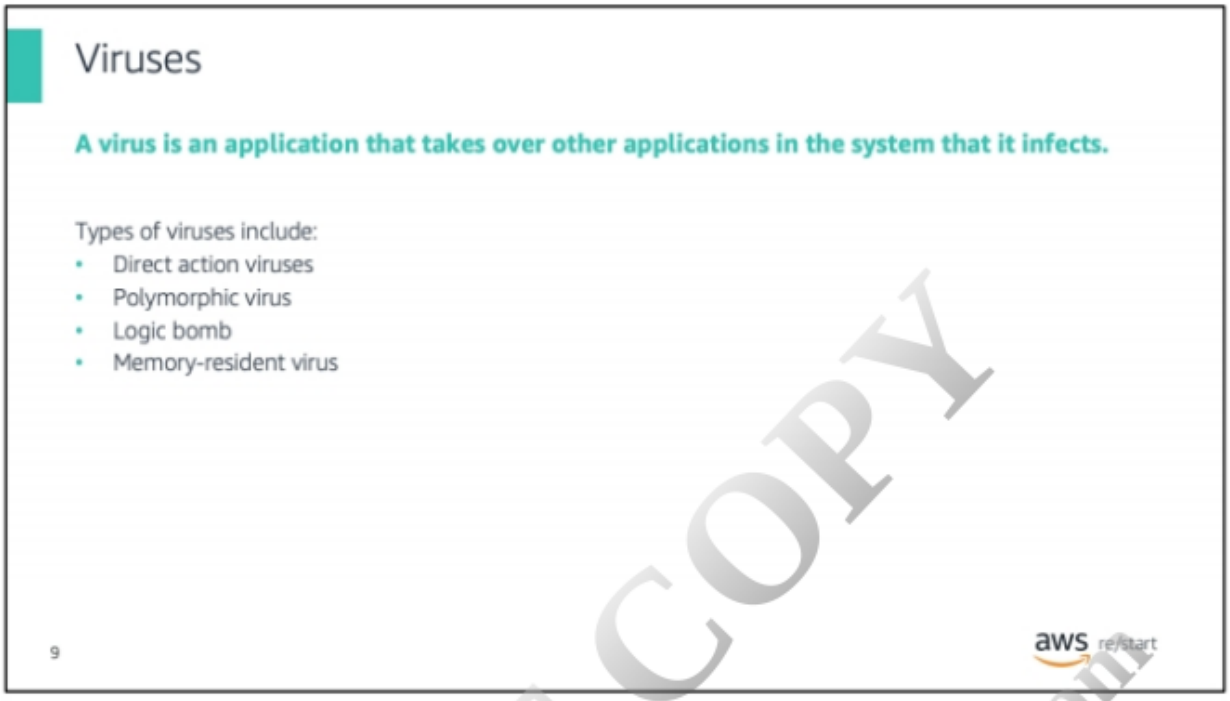
Example of infection using a removable device

A USB device is mailed to you. You open it, and it contains a backdoor that gives remote access to your system to an unauthorized user or third party.

Types of malware include:

- **Viruses –** A virus attaches itself to system applications and runs every time a normal program runs.
- **Worms –** Different from viruses. They have no executable file and rely on application weaknesses to deploy themselves. A worm allows its author to control the infected computer remotely. It can be difficult to isolate because it spreads quickly. Examples: Morris, MyDoom, Sobig, Stuxnet
- **Bots** – Used to control computers or launch distributed denial of service (DDoS) attacks against vulnerable systems. Example: Poison Ivy
- **Backdoors –** A backdoor (also known as a Trojan horse) is often a secret server that steals information from the victim's system. It allows an intruder into a system. You can know about the backdoor if you scan the system and the network to find patterns of traffic. Examples: Sub7, GirlFriend, wack-a-mole, Zeus
- **Rootkits –** A rootkit cloaks itself by replacing system files that can reveal its presence. It is used to retrieve information. It is difficult to identify and remove because it can become part of the operating system. Removal often requires a system format. Example: Hacker Defender

- **Spyware –** Spyware jeopardizes privacy and typically comes embedded into applications that look free and interesting to use. As people are doing more finance and other personal activities online, these activities can be detected and revealed, and information stolen. Example**:** Real-time spy
- **Adware and scareware –** Adware deploys advertising content and monitors user activity, such as visited websites. It is similar to spyware, but focuses on ads and what is clicked. Adware often comes embedded in shareware applications. Examples**:** Spyware toolbars, Conduit Search
- **Ransomware –** Ransomware locks systems or makes data unavailable until the user pays.



Types of viruses

- **Direct action viruses** – A virus that attacks immediately to infect files or programs every time the code runs.
- **Polymorphic virus** – A virus that is self-encrypted to evade detection. It duplicates itself by creating working, yet slightly changed, copies of itself.
- **Logic bomb** – A virus that deliberately places code into a software system to set off a malevolent function after certain requirements are met.
- **Memory-resident virus** – A virus that installs itself and hides in the memory of your computer. After it runs, it looks for other files or programs to infect.

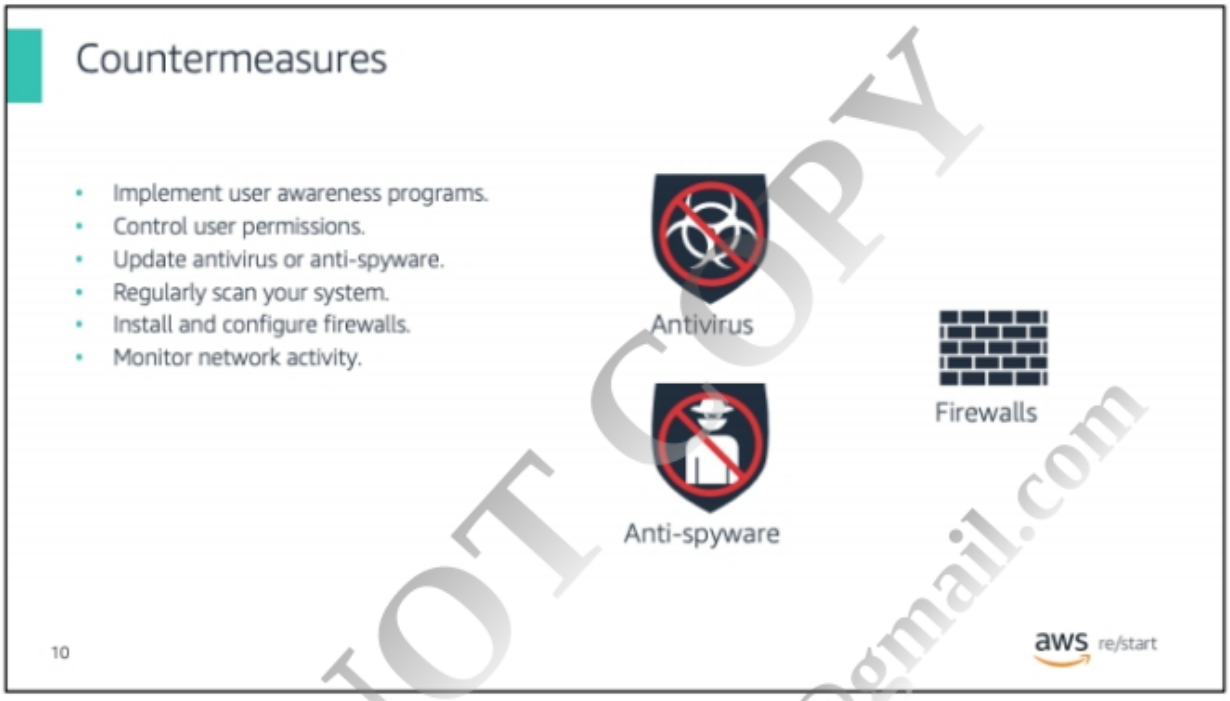
Other types of viruses include:

- **Cluster virus** – A virus that links itself to the implementation of other software programs.
- **Cavity virus** – A virus that tries to install itself inside of the file it is infecting by attaching to empty spaces inside the file.

Examples of viruses: ILOVEYOU, Klez, Chernobyl, Anna Kournikova, Flame, Michelangelo

Stuxnet is an example of a "successful" target malware. It was likely designed in collaboration between the United States and Israeli governments to harm the nuclear program of Iran.

For more information, see the "W32.Stuxnet Dossier" technical paper on Broadcom (https://www.symantec.com/content/dam/symantec/docs/security-center/white-papers/security-response-w32-stuxnet-dossier-11-en.pdf).



User awareness and layered protection are keys to detect (and prevent) malware. No single countermeasure will do it all.

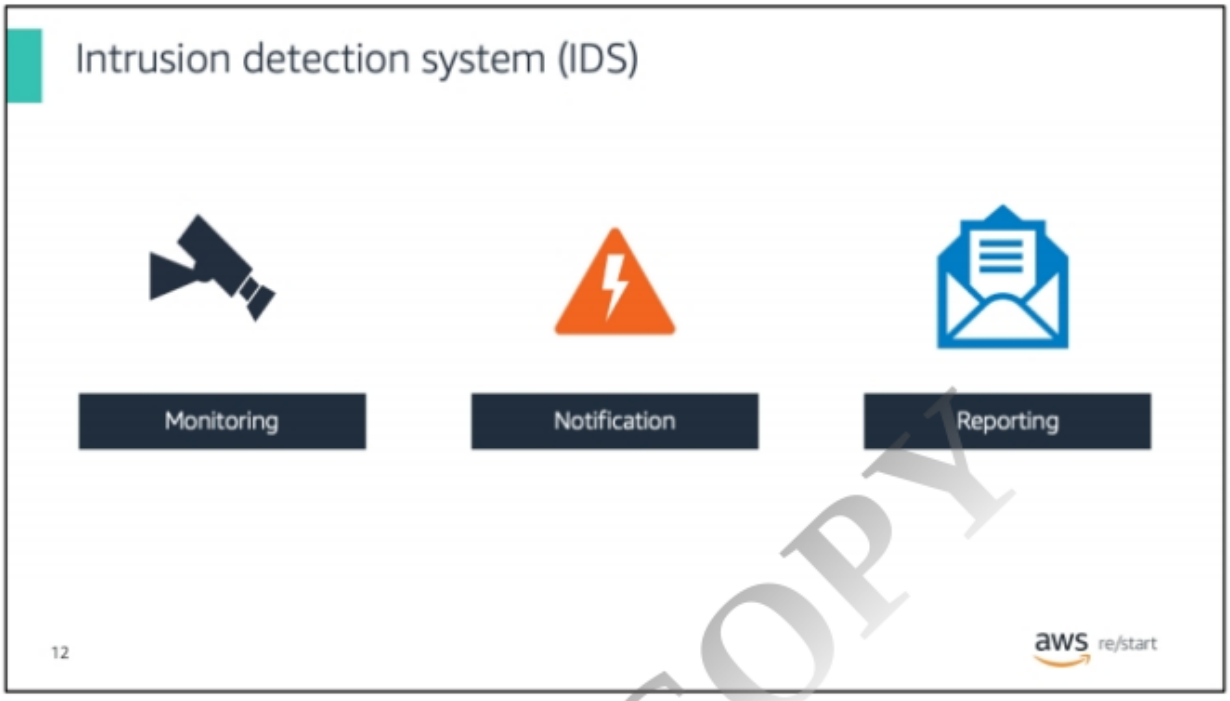
Countermeasures, continued

- Verify file integrity.
- Harden the system.
- Perform intrusion tests.
- Implement a baseline.
- Implement physical security.
- Establish policies.
- Scan incoming communications.

Intrusion detection systems (IDSs)

aws re/start

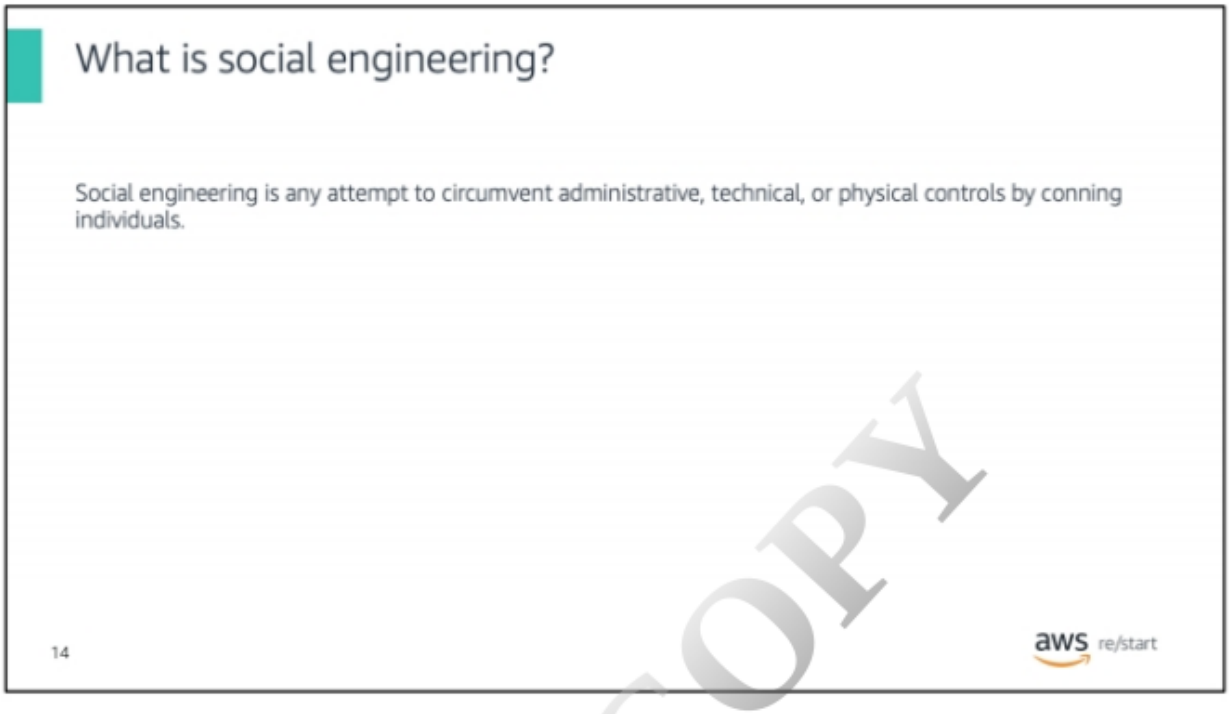
Technical, physical, and administrative countermeasures can be used to protect systems against malware.



Monitoring – Software application or device that monitors the network or systems for devious activity

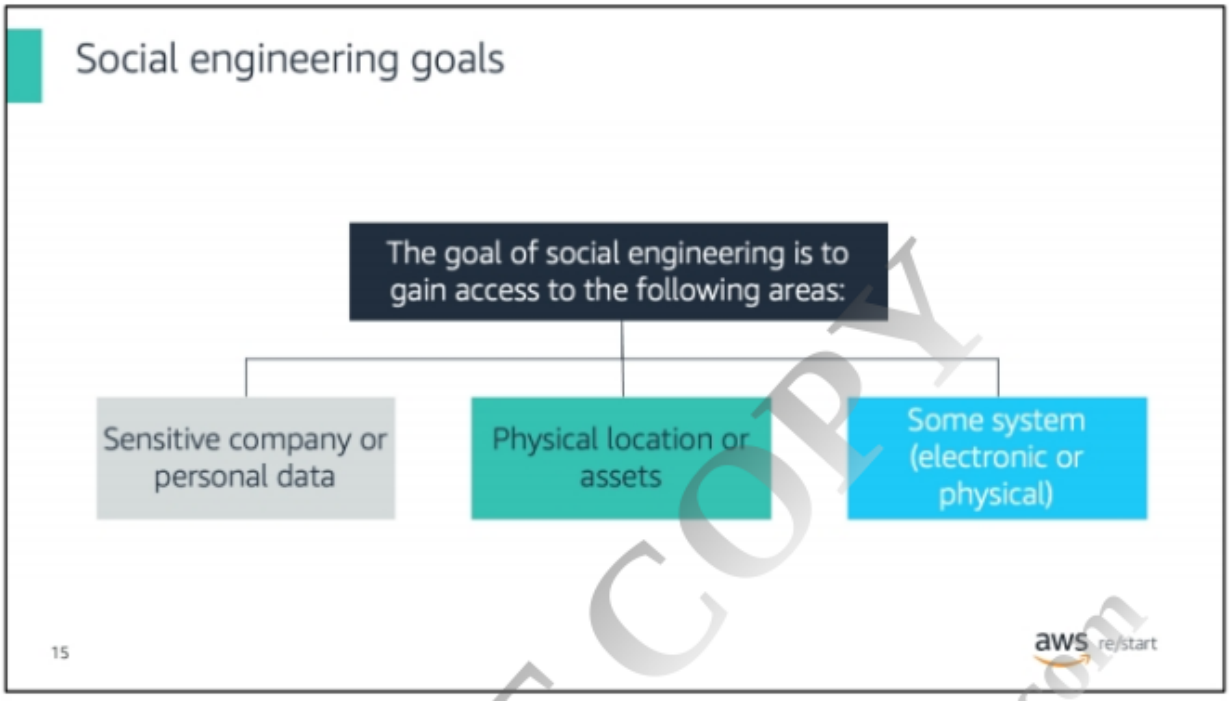
Notification – Sends alerts to administrator if malicious activity is found

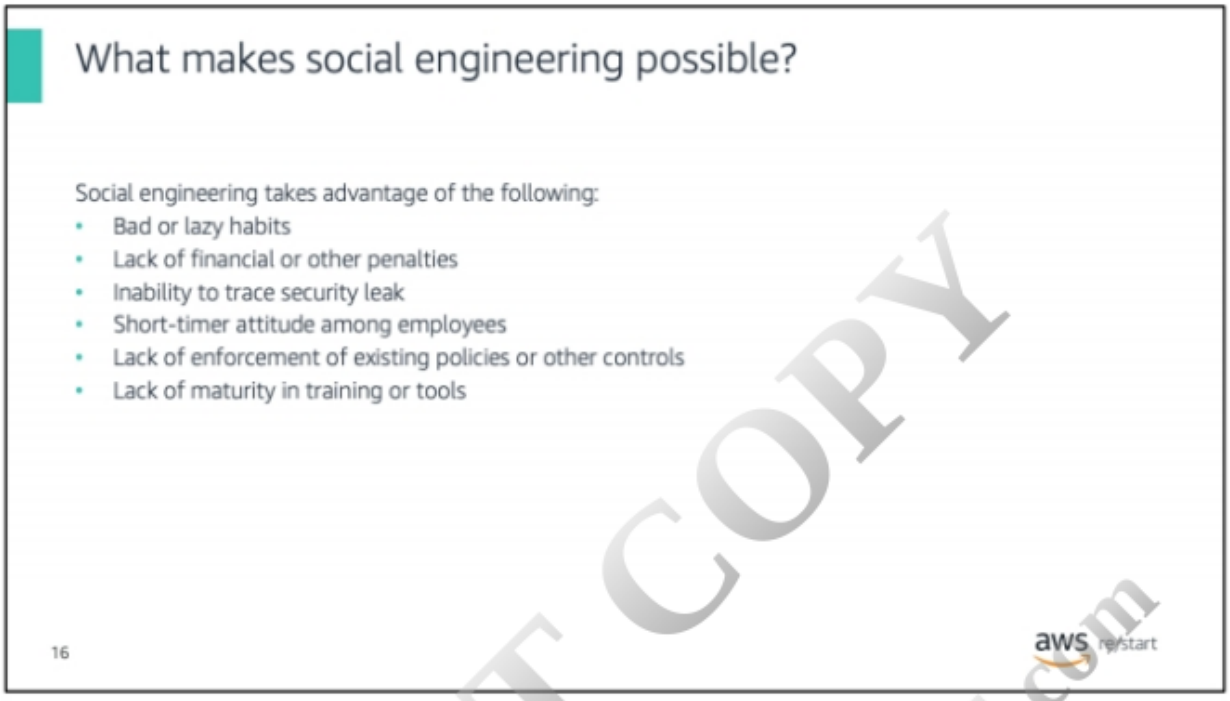
Reporting – Reported information includes the source address, victim address, and type of attack



Social engineering, in the context of IT and computers, is any human attempt or interaction to circumvent administrative, technical, or physical controls by conning or manipulating individuals. The objective of social engineering is to convince people to give up access to privileged data, a system, or a facility, or to reveal sensitive information.

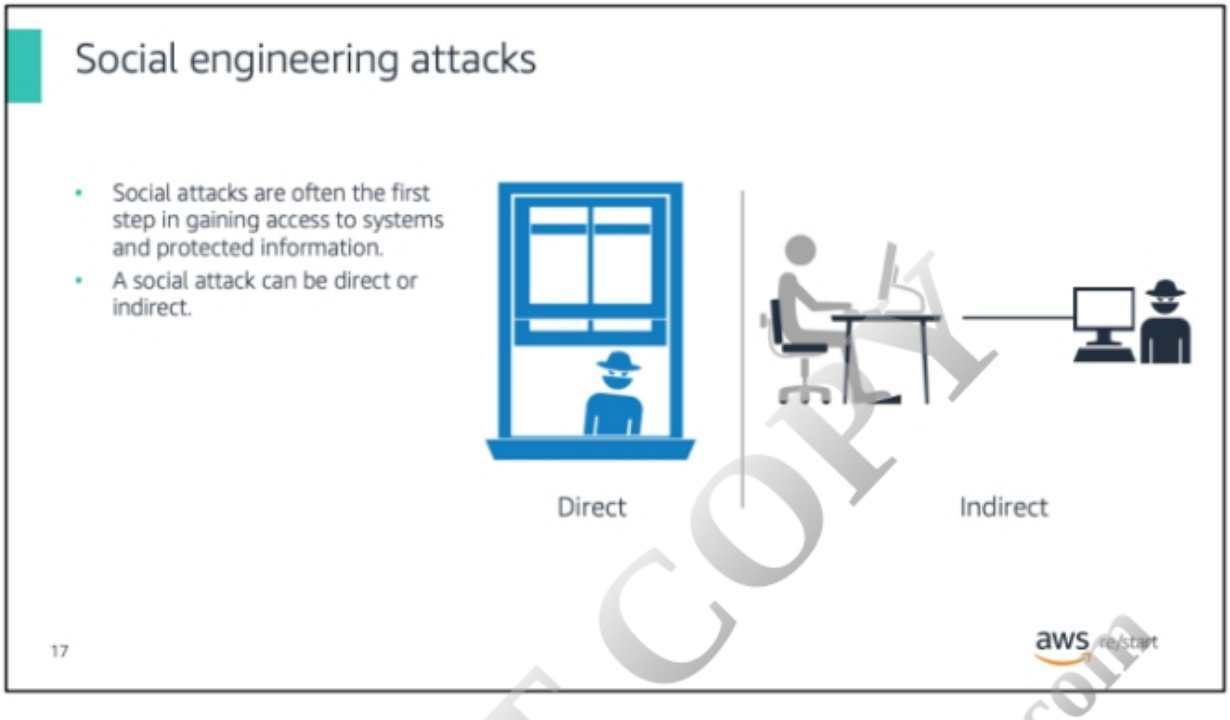
It is a method of attack and exploit that focuses on people instead of technology.





Social engineering takes advantage of the following circumstances:

- Bad or lazy habits that break normal security procedures.
- Lack of financial or other penalties for giving up information.
- Inability to trace security leak back to a specific employee.
- Short-timer attitude among employees (*I won't be here when they investigate.*)
- Lack of enforcement of existing policies or other controls.
- Lack of maturity in training or tools to combat social engineering.

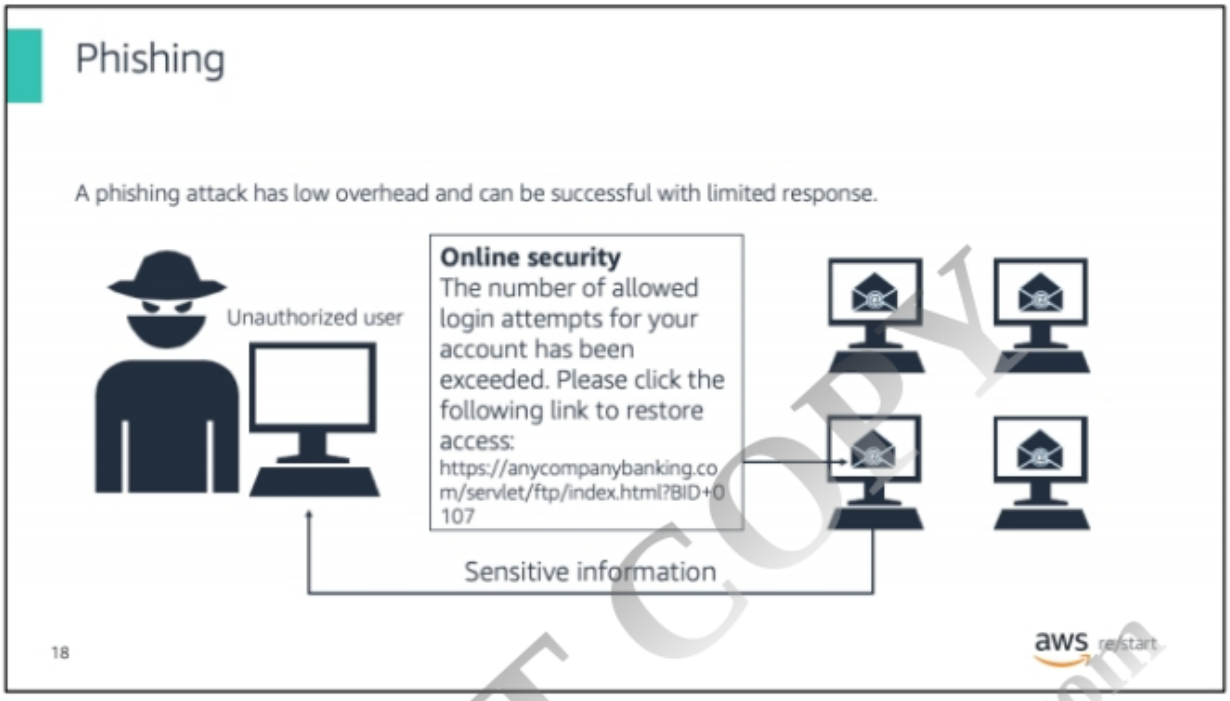


Social attacks get an unauthorized user past the first level of physical or technical controls.

Types of social engineering attacks:

- **Direct –** The unauthorized user is present onsite and directly participates in the attack. Examples include
  - Delivery of food or flowers
  - Job interview
  - Tailgating
  - Dumpster diving
  - Sales call
  - Picking up an unattended device from a desk and walking away
- **Indirect –** The unauthorized user is not onsite with the target. Or the unauthorized user might not be actively participating in the attack when the target decides to give up information. Examples include:
  - Spam or phishing
  - Technical assistance
  - Pretext (attacker uses an invented scenario or pretends to be someone else to gain access)
  - Product demonstration or sample

- Posting a help-wanted ad on a job site.



For more information about phishing, refer to the PhishTank website. The site is a collaborative clearinghouse for data and information about phishing on the internet. It also provides an open API for developers and researchers to integrate anti-phishing data into their applications at no charge.

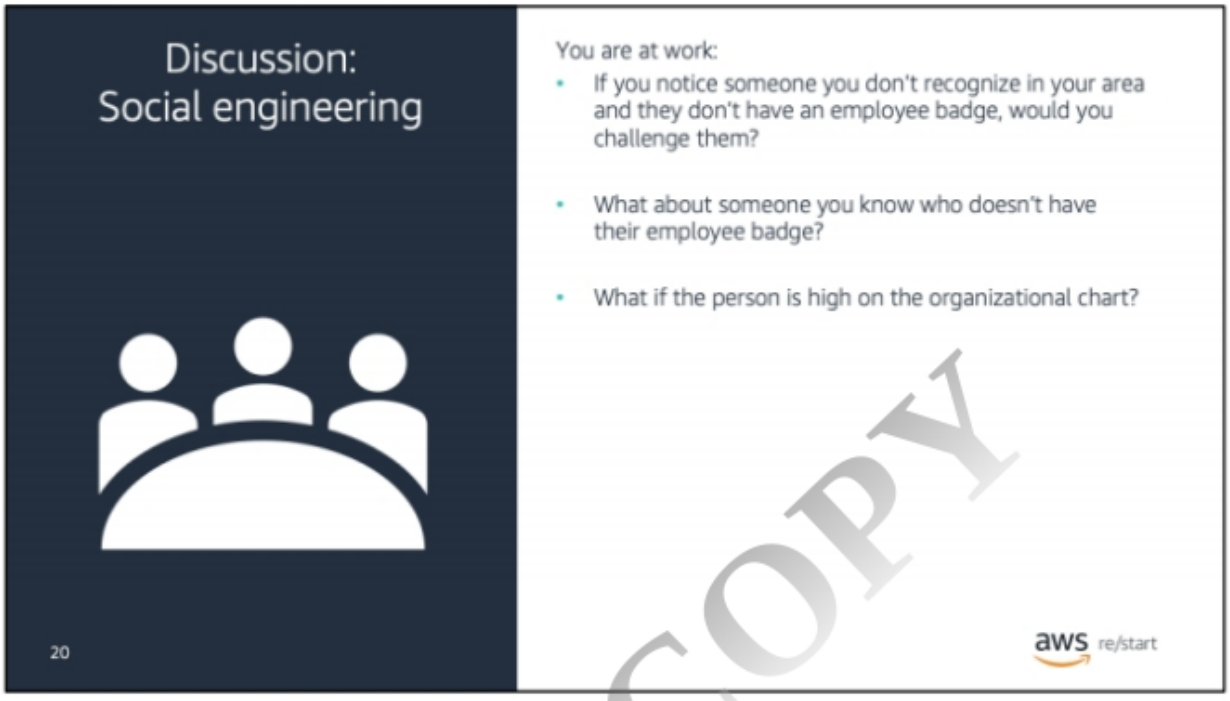
Features of the site include:

- If you suspect a site of being a phishing site, you can enter the URL, and search the PhishTank database to confirm your suspicion. If the URL is not in their database, you can add it.
- The site's FAQ page provides general information about the site, in addition to information about using some of its features.

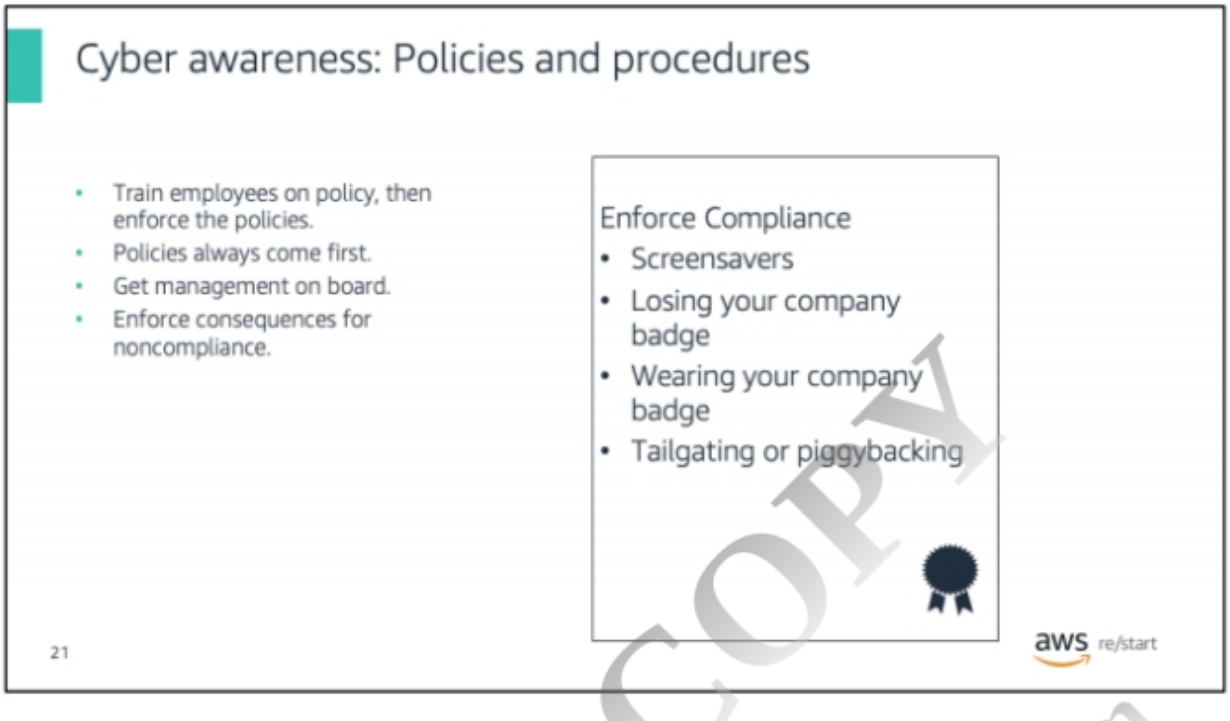
Preventing social engineering

Training and **education** are the most effective defenses against social engineering attacks. But sometimes, training does not work.

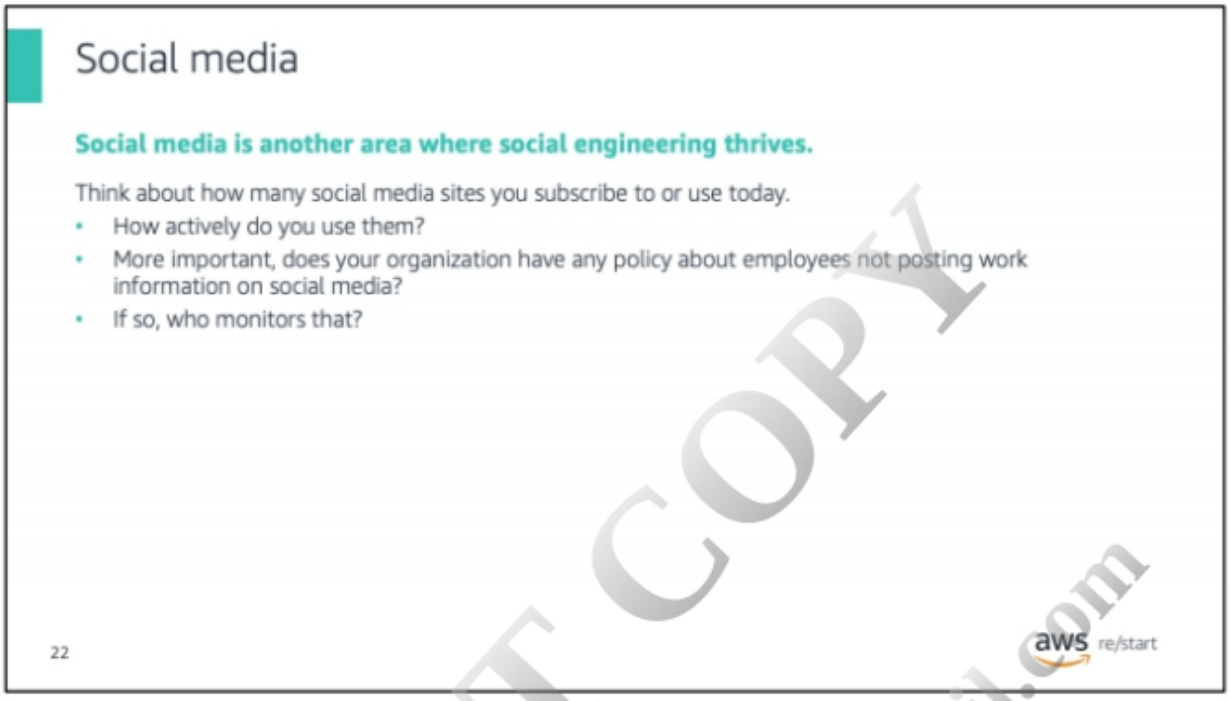
aws re/start

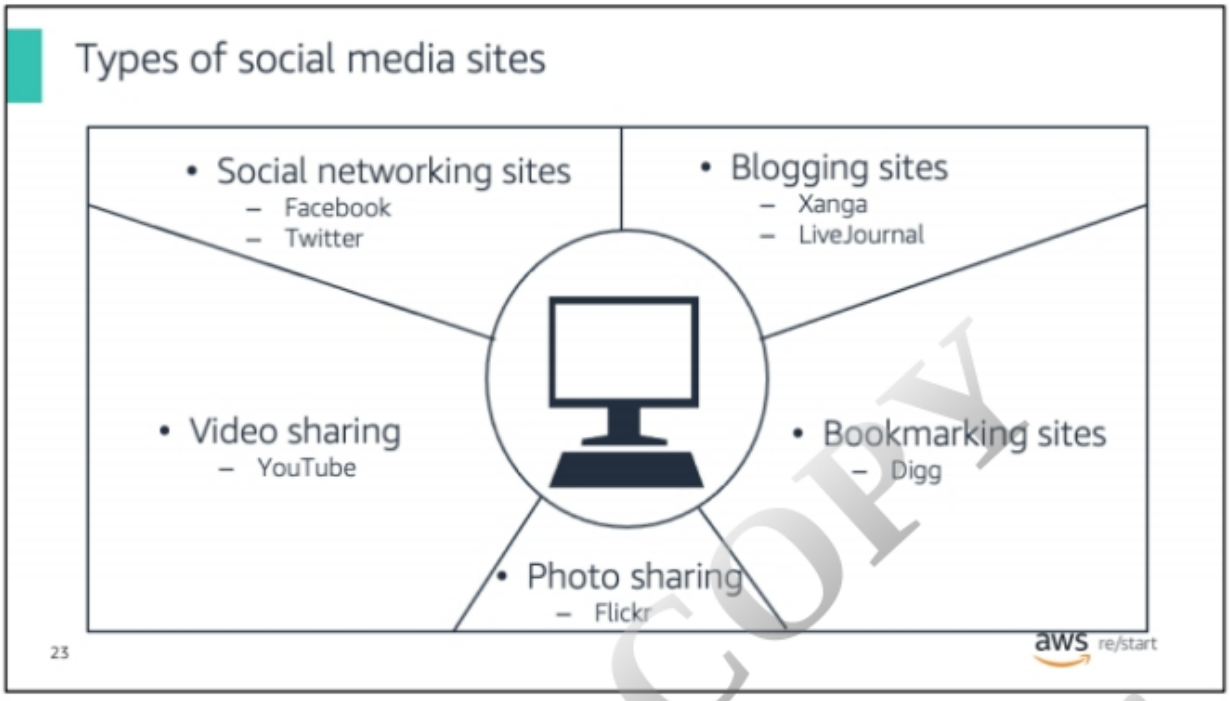


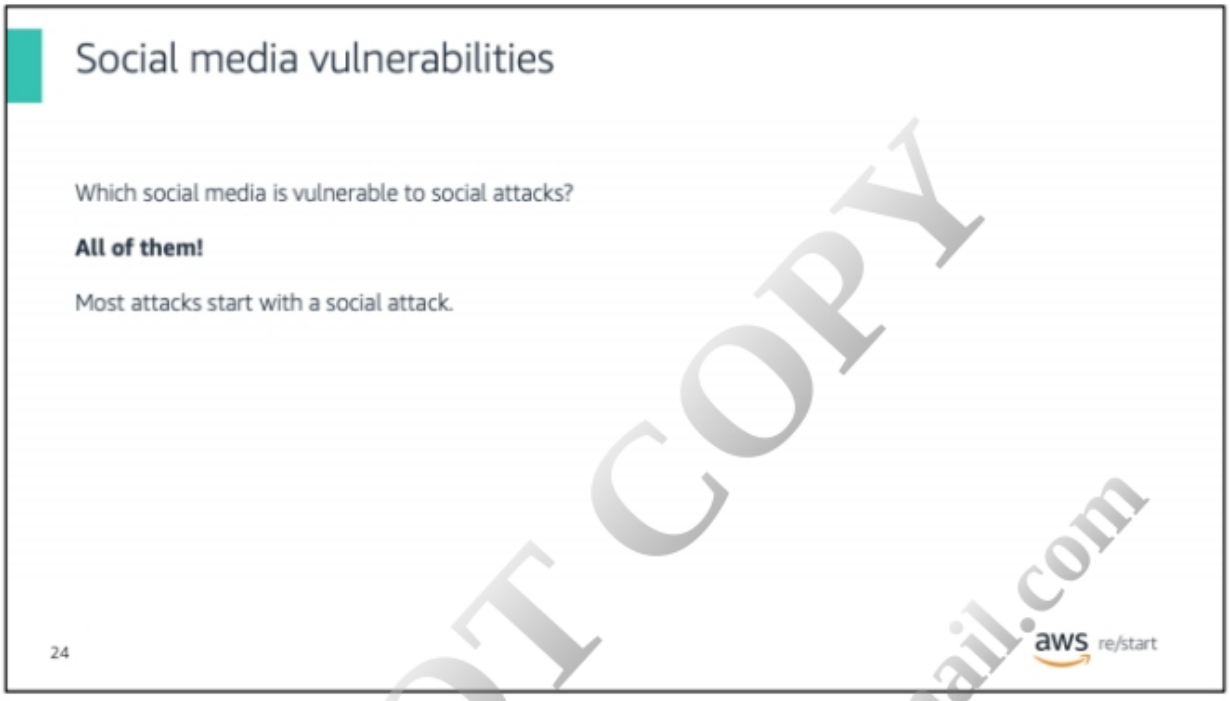
Think about the situations that are described here. They can relate to social engineering scenarios.



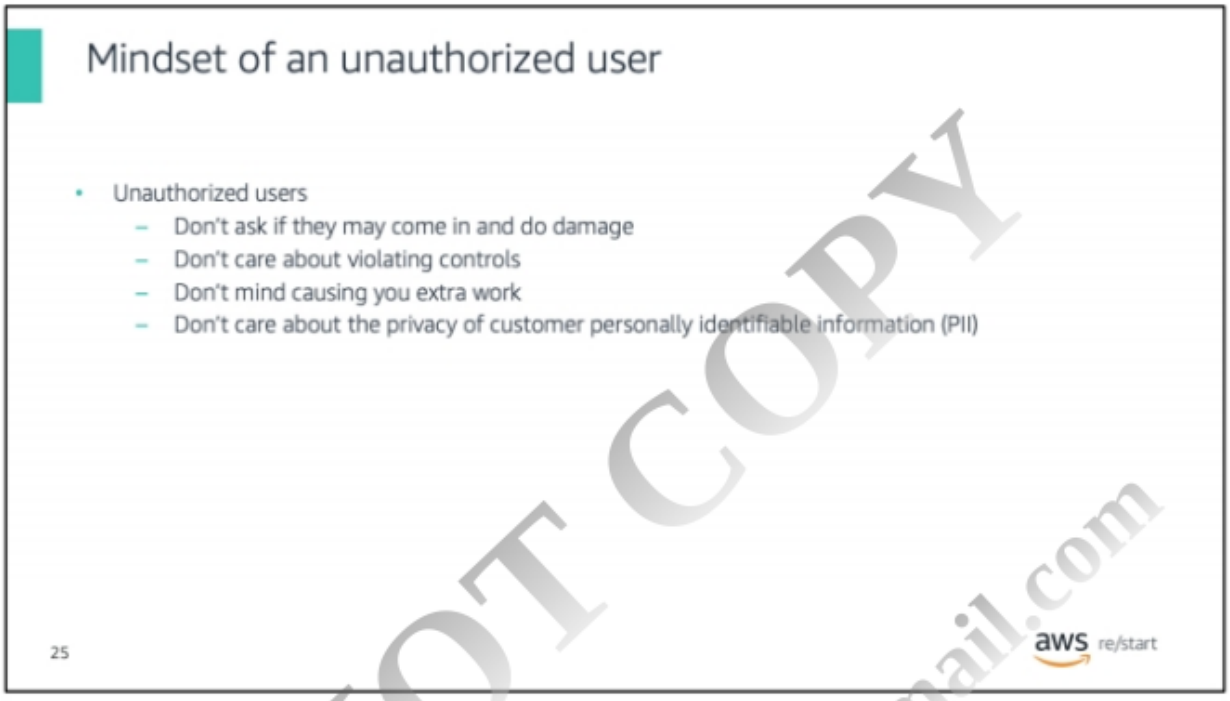
Enforcing a penalty for noncompliance is a deterrent to ignoring policies.







All social media is vulnerable to social engineering attacks.



Securing Social Media Content

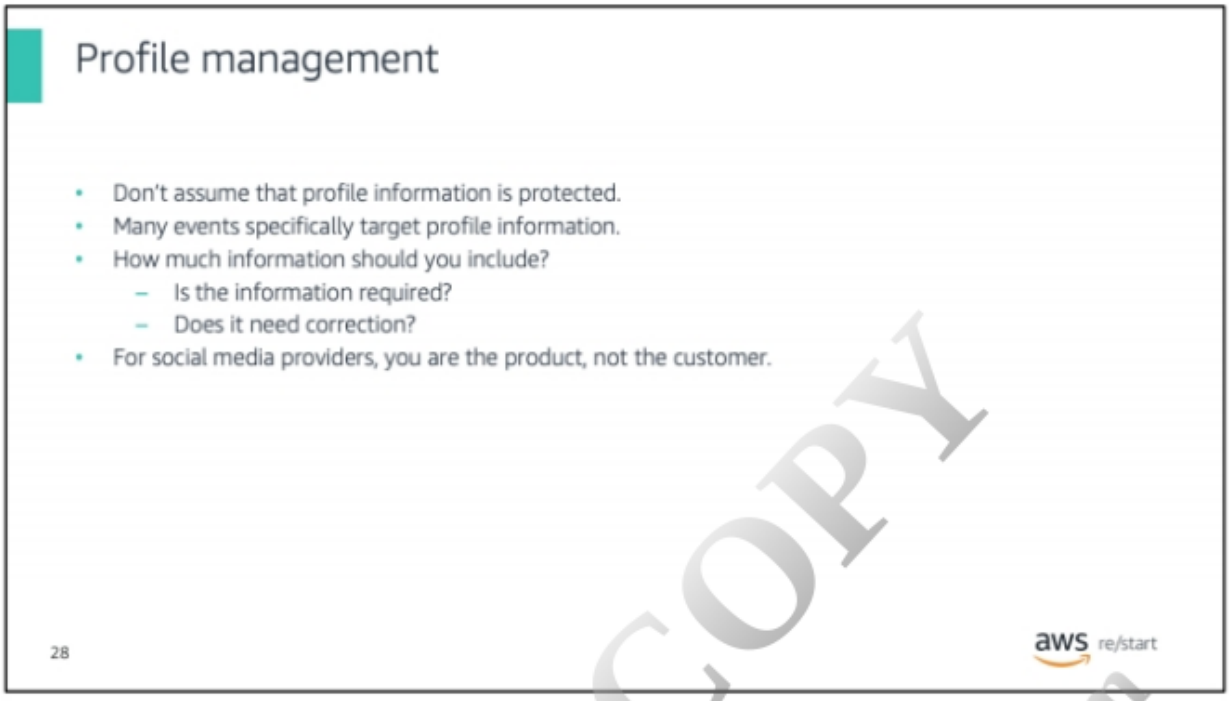
Can you point out some of the inappropriate or potentially compromising information on this Social Media app?

Here are a few vulnerable areas:

- In the first posting, Diego Ramirez reveals the **security firm** that ExampleCorp uses. He broadcasts a known security vulnerability with that firm.
- The second posting shares confidential information about the structure of his company's network, **including IP addresses**.
- The 3rd post near the bottom of the phone mentions that the day it was posted**, June 17th**, was **Maria's birthday**. In addition, it references that she has a **cat named Peanut**. Personal information like this could be used for a potentially attempting to guess a password. As we know that June 17th was Maria's 31st birthday, we could assume that Maria was born in 1989. People commonly combine pet names, birthdates, and similar personal info to create passwords. Would "Peanut89" be a valid guess at Maria's password?
- The 3rd post also suggests that Maria has posted photos of ExampleCorp's **Network Operations Center (NOC).** These may contain sensitive information that needs to stay private and secure.

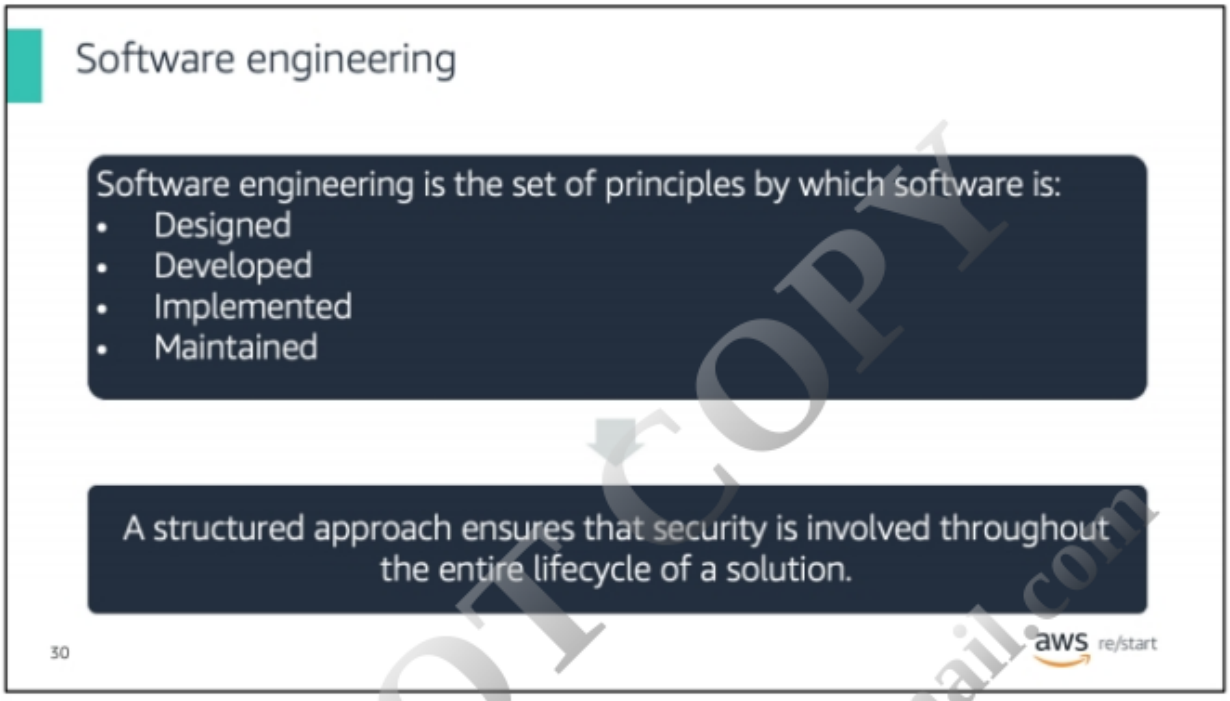
Some possible answers to these questions:

- Money
- Notoriety
- Fame
- Ego



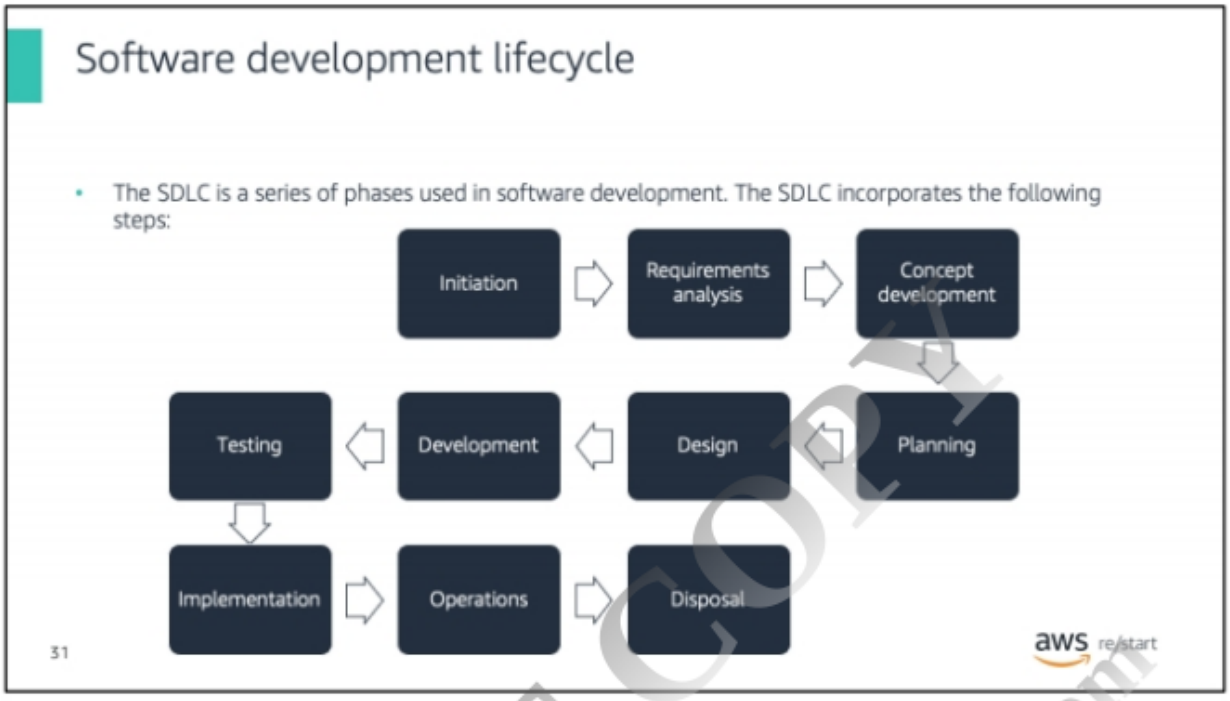
It is also important that you think about the type of information that you store in your social media profile.

Just as there are security threats that target individuals, so too are there security concerns around the software we may develop. Being aware of what precautions need to be taken in the development of software can prevent these threats from accessing everything from proprietary code, sensitive infrastructure information, and even customer data. Here we'll discuss some common methods by which software is



Software engineering refers to a set of principles applied in a methodological manner to aid the design, development, implementation, and maintenance of software solutions.

One of the key aspects of software engineering is the strategy that is used to develop applications. Opportunity exists for developing programs using different techniques. However, to maintain security throughout the entire solution lifecycle, you must take a structural approach.

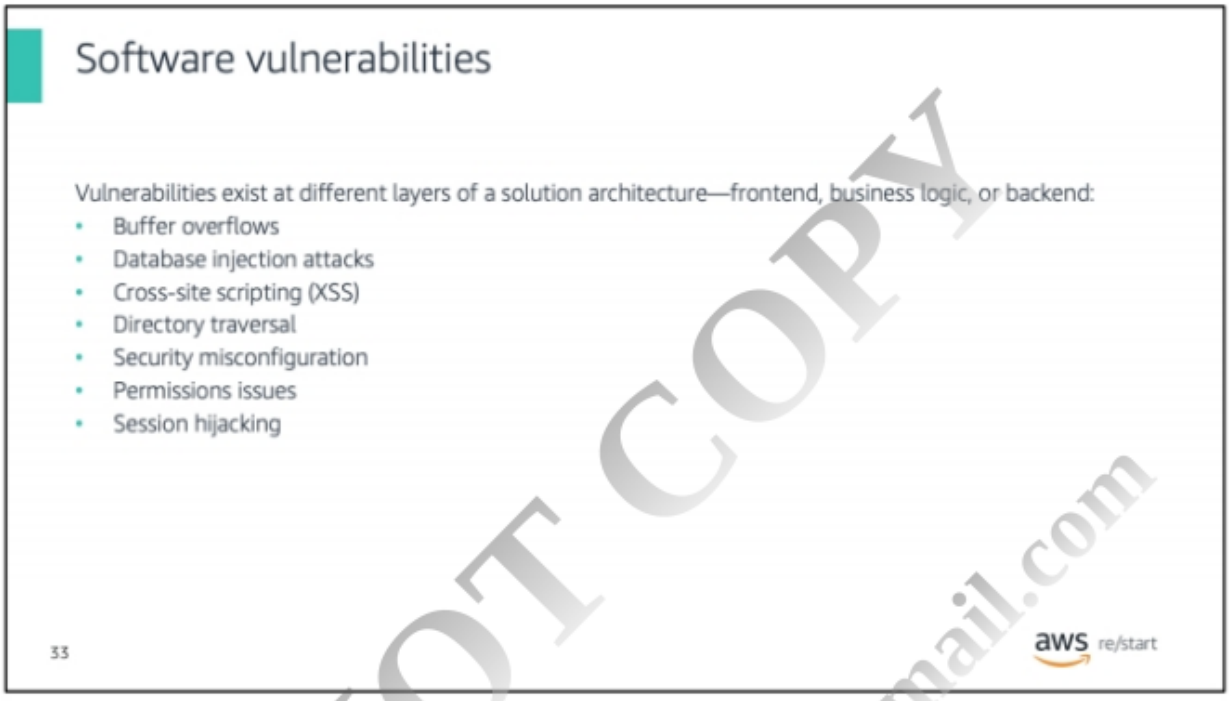


Consider security as early as possible in the software development lifecycle.

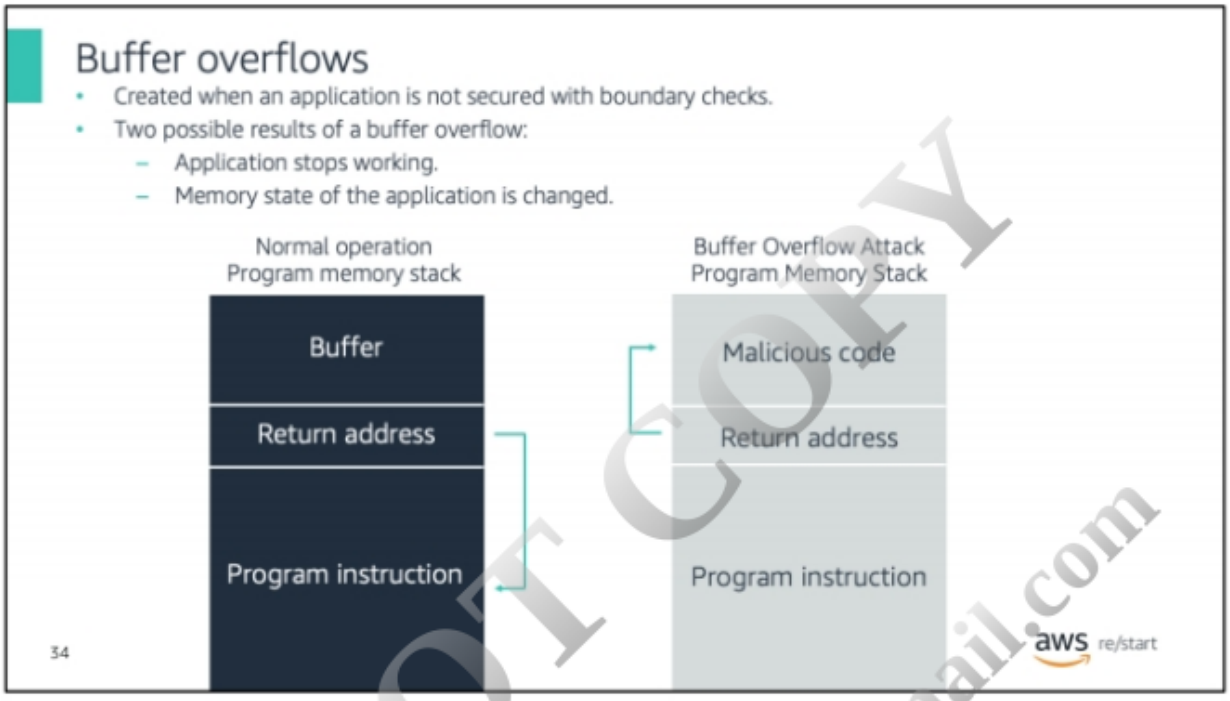
To minimize risk associated to application development, follow these security principles:

- **Change Management**:
  - Process by which preparation, support, and follow through in completing the work specified around a desired change is communicated to individuals, teams, and organizations. Ensures a common process and required transparency are implemented throughout the completion of work.
- **Separation of Duties**:
  - Restricts the amount of power held by any single person or team taking part in the development and delivery of software.
- **Peer Reviews**:
  - a type of software review in which a work product is examined by its author and one or more colleagues, in order to evaluate its technical content and quality.
- **Production and Development Teams**:
  - Ensuring that each team can provide feedback to the other with regards to quality and alignment of what the teams set out to build. The relationship between these teams is critical in ensuring that the software being developed performs as needed, to include doing so within the secure parameters required.

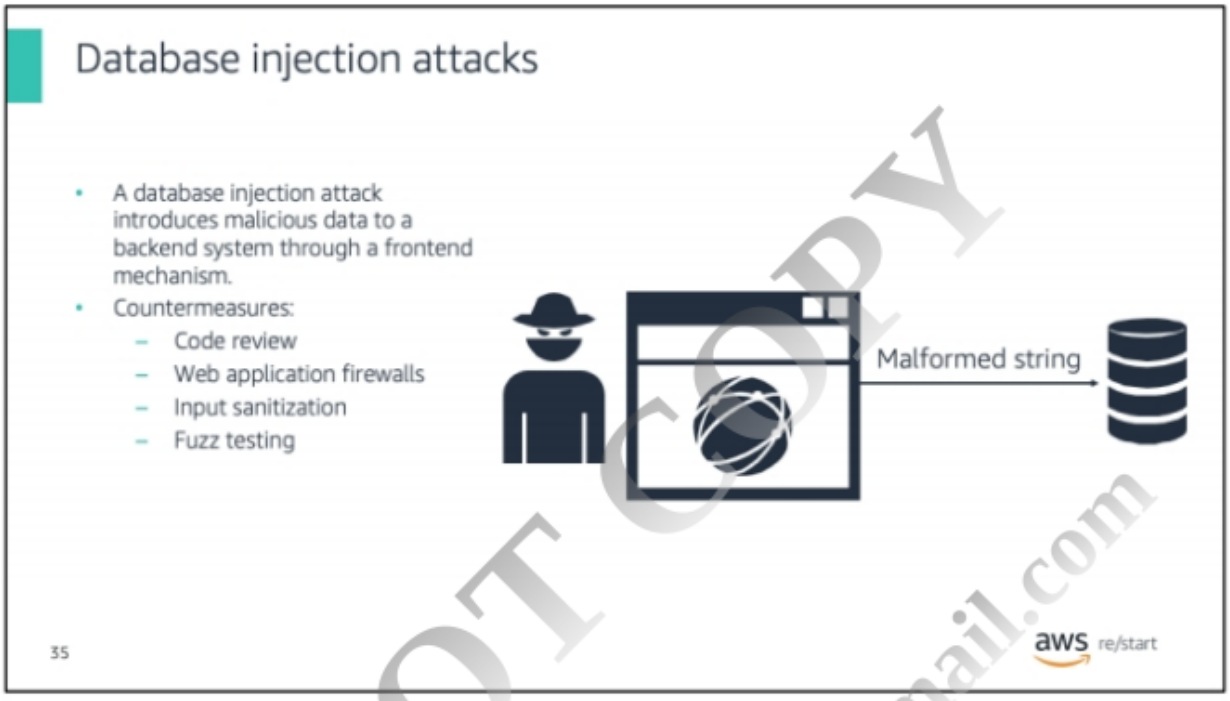
- **Quality Assurance**:
  - Guarantees a level of quality for the end client, and to help the software development team to identify problems early in the process.
- **Background Check on Programmers**:
  - Ensures that the contributing team members are vetted and free from potential influences that may coerce or conceal the motives of an individual.
- **Code Escrow**:
  - Ensures that the maintenance of code is prioritized so that it isn't ignored. This prevents code from being "abandoned" and becoming a security threat.



The next topic discusses these vulnerabilities in more detail.

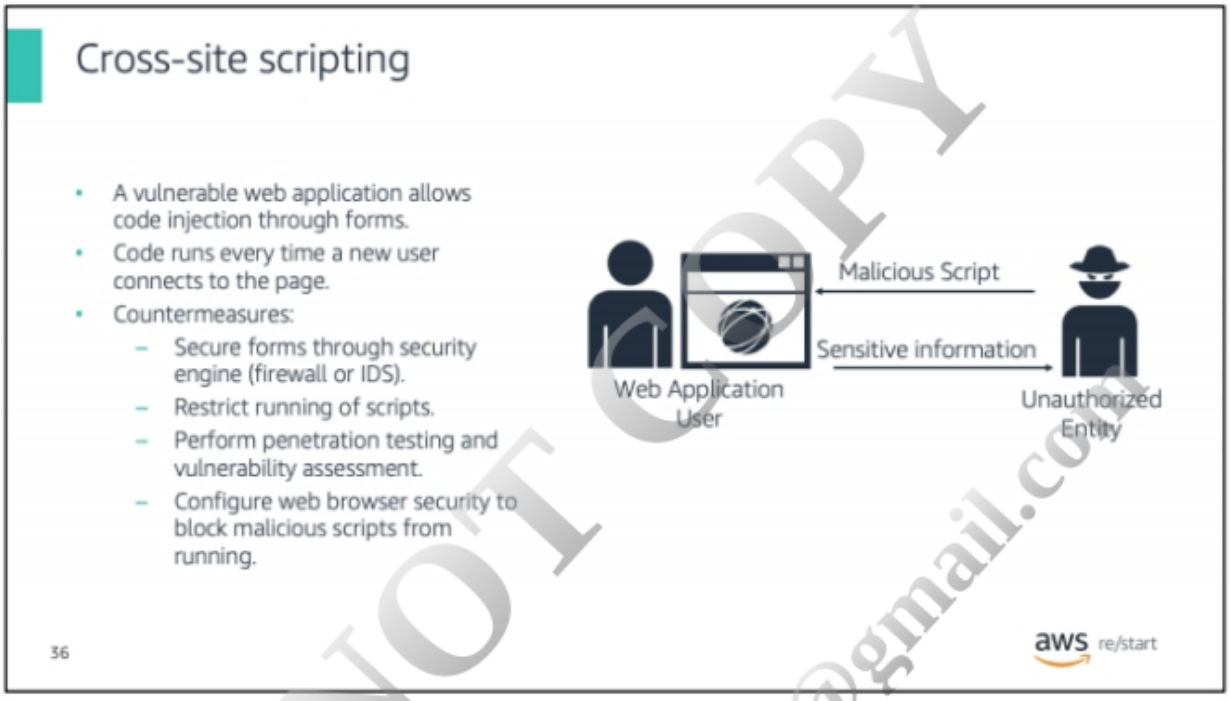


Unauthorized users can take advantage of a buffer overflow vulnerability to insert malicious code into a running program. By doing so, they gain access to protected information or gain control of the program.

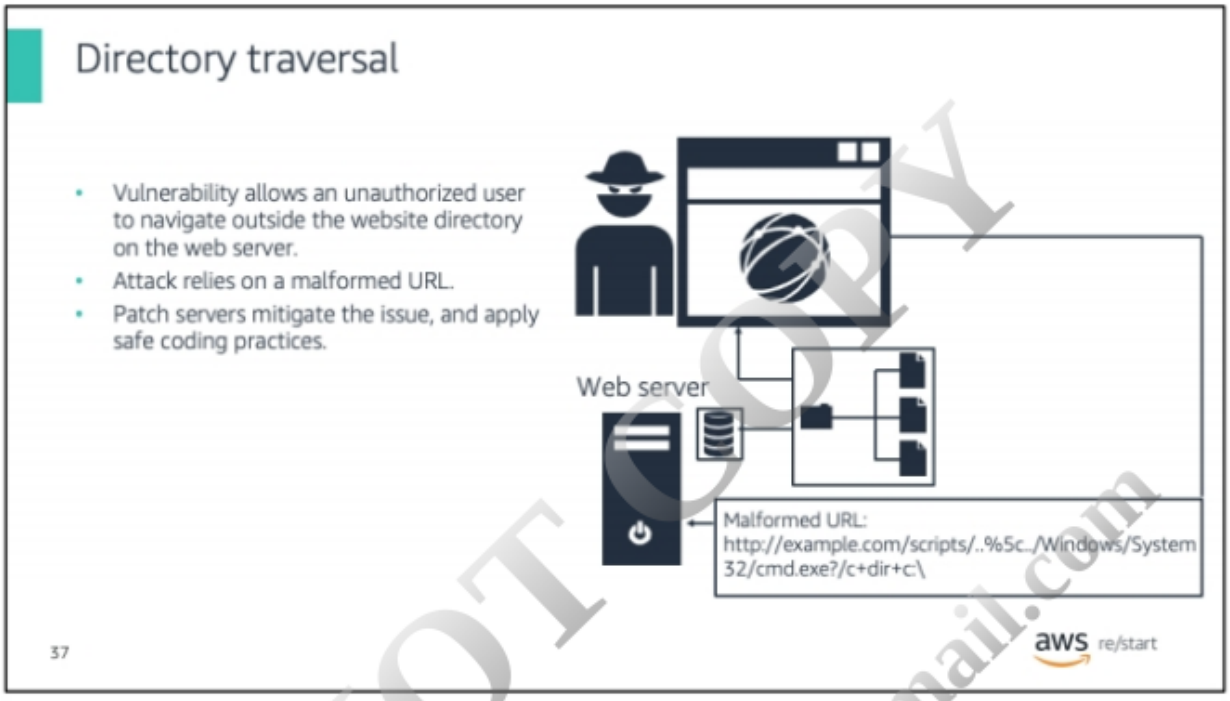


A database injection attack attempts to alter a backend database system through a webpage or other frontend interface. This vulnerability is often caused by a lack of (or incomplete) input validation. In turn, this points to a lack of code review or not considering security early in the development process.

- Being able to implement a thorough **code review** that looks for potential weaknesses can help identify where security measures need to be taken
- Implementing **Web application firewalls** can help ensure only the the traffic that is verifiably allowed to access data ensures a measurement of control around what can and cannot access sensitive data.
- Ensuring that data inputs are "sanitized" – so that only data that conforms to a known standard and isn't incorrectly formatted can pass through sensitive processes
- Fuzz Testing is a process by which the software is tested with malformed, unexpected, or random data to ensure that the software knows to not attempt to process data that doesn't conform to expected input standards.

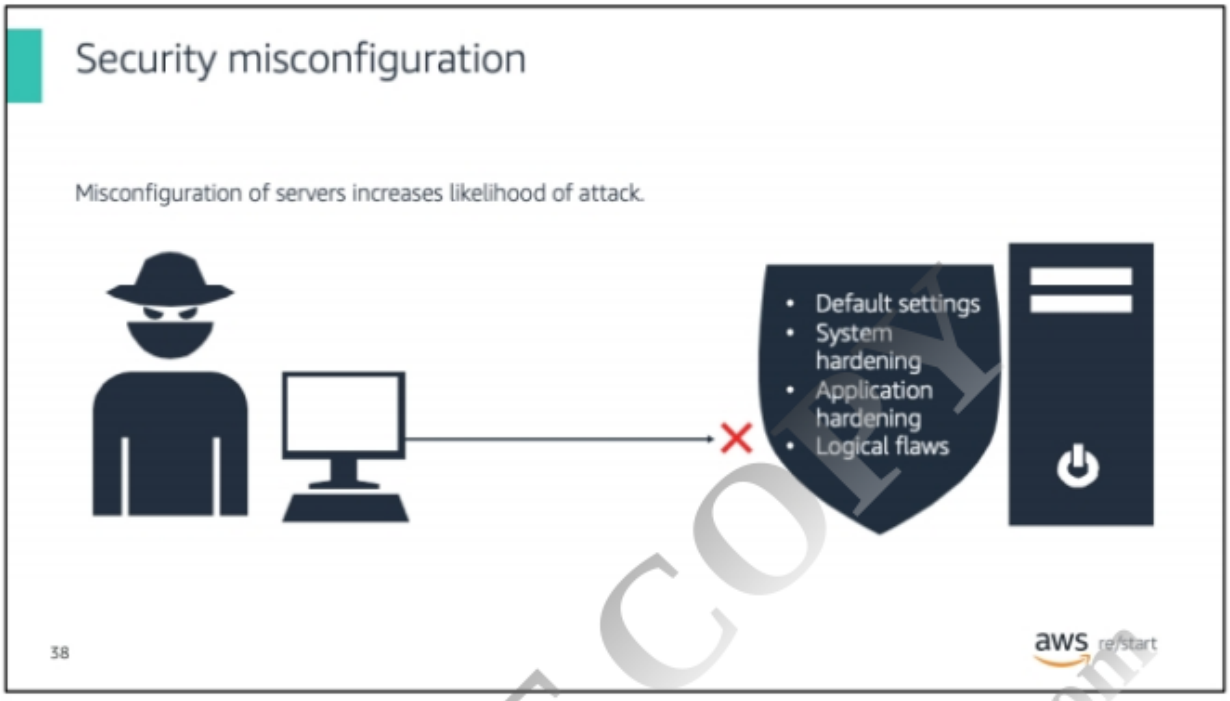


Cross-site scripting (XSS) is a client-side attack that aims at performing a malicious activity on the client computer. The unauthorized entity injects a malicious script that the browser runs when it loads the webpage.



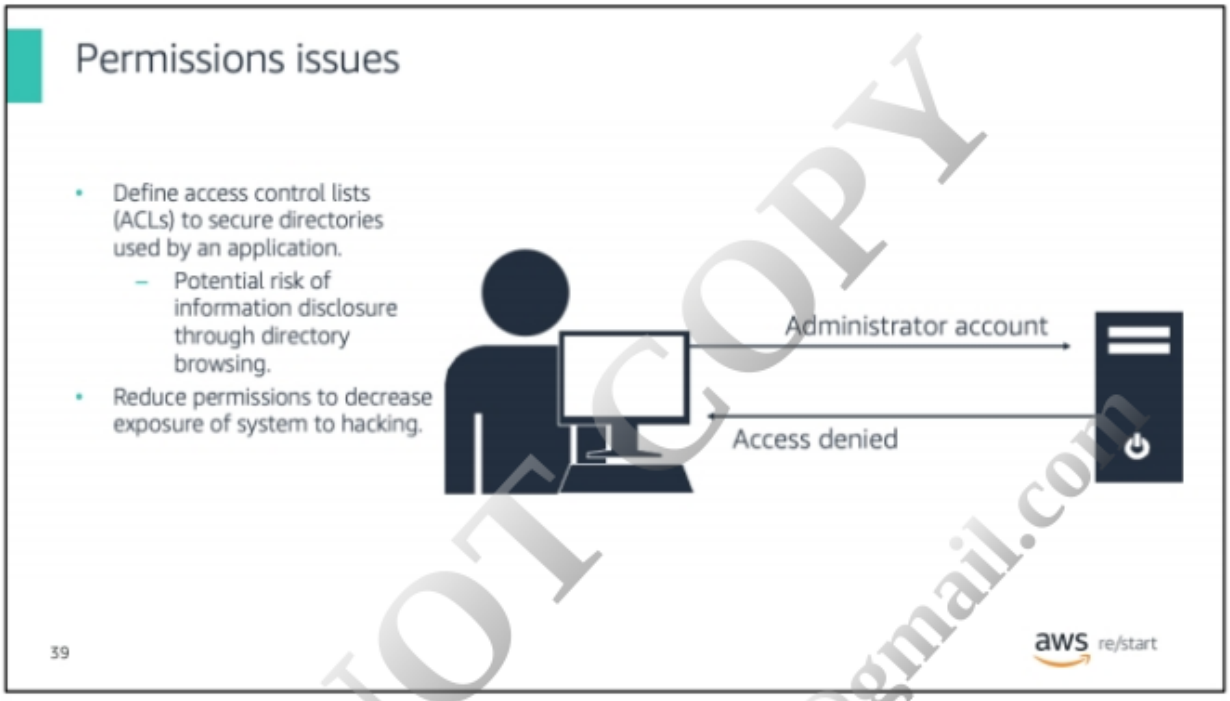
In a directory traversal, the unauthorized entity is able to go outside the root folder of the site to explore other folders of a system.

An effective remedy is to block specific URL formats and patch the system.

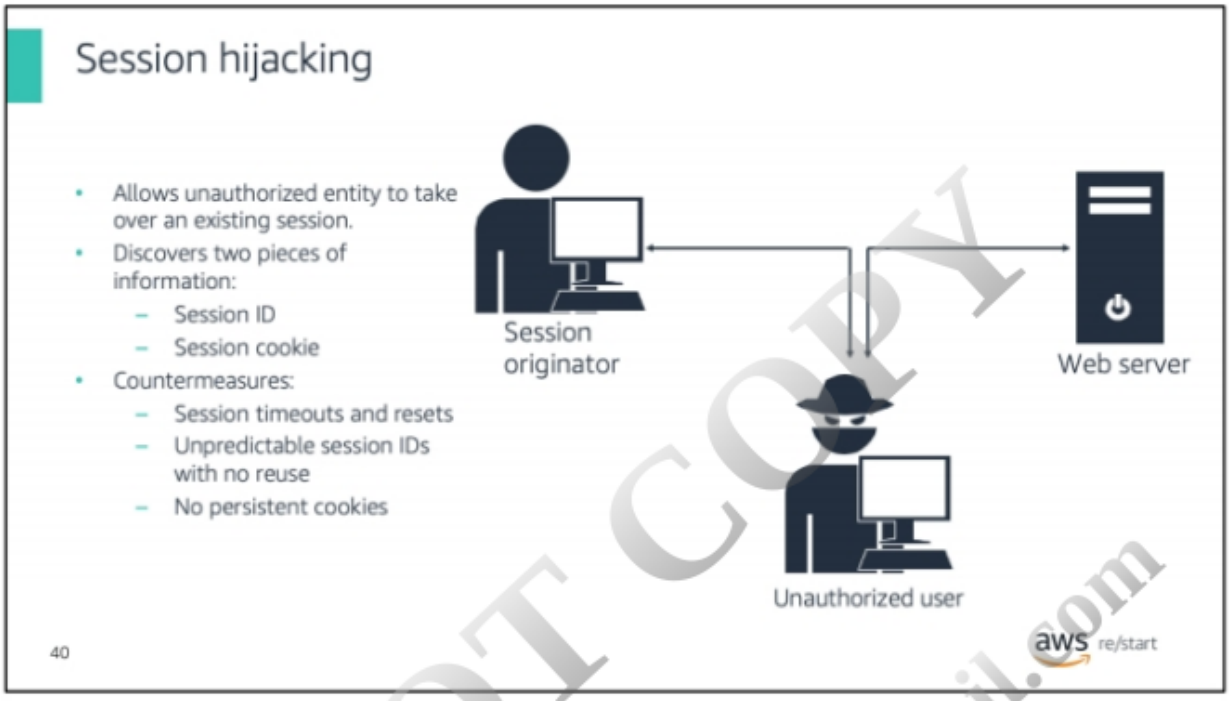


Security controls are often provided in the configuration of a server or piece of software. However, if they are not turned on or properly configured, the server or software becomes vulnerable.

Make sure to optimize security at the web-server level and the web-application level. For more information, see the Open Web Application Security Project (OWASP) website.

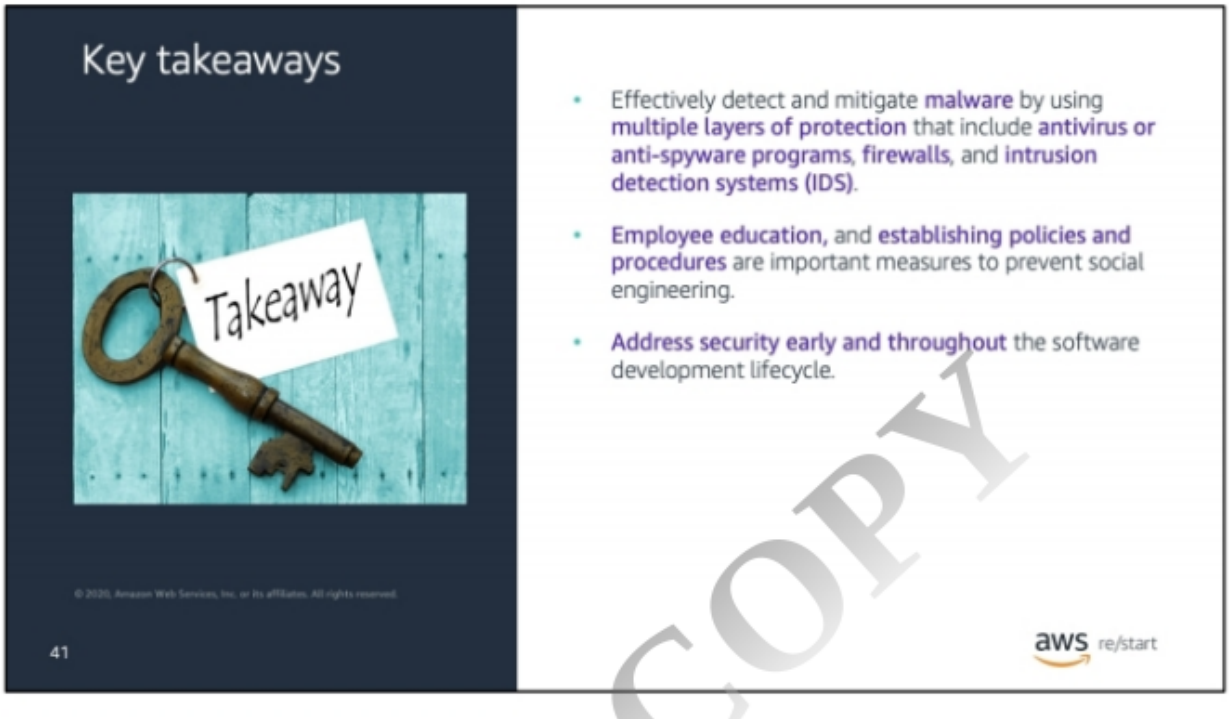


Granting too many permissions to access a remote system make the system vulnerable to attacks. Use the principle of least privilege when you define permissions.



In a session hijacking attack, a *man-in-the-middle* attacker intercepts and alters incoming and outgoing HTTP requests to take over a session. By using the session originator's session ID and cookie, the unauthorized user impersonates the authenticated user.

Ways to mitigate session hijacking include defining short timeouts on authorization and using transient cookies in the browser.



Key takeaways from this lesson include:

- Effectively detect and mitigate malware by using multiple layers of protection that include antivirus or anti-spyware programs, firewalls, and intrusion detection systems (IDS).
- Employee education, and establishing policies and procedures are important measures to prevent social engineering.
- Address security address early and throughout the software development lifecycle.