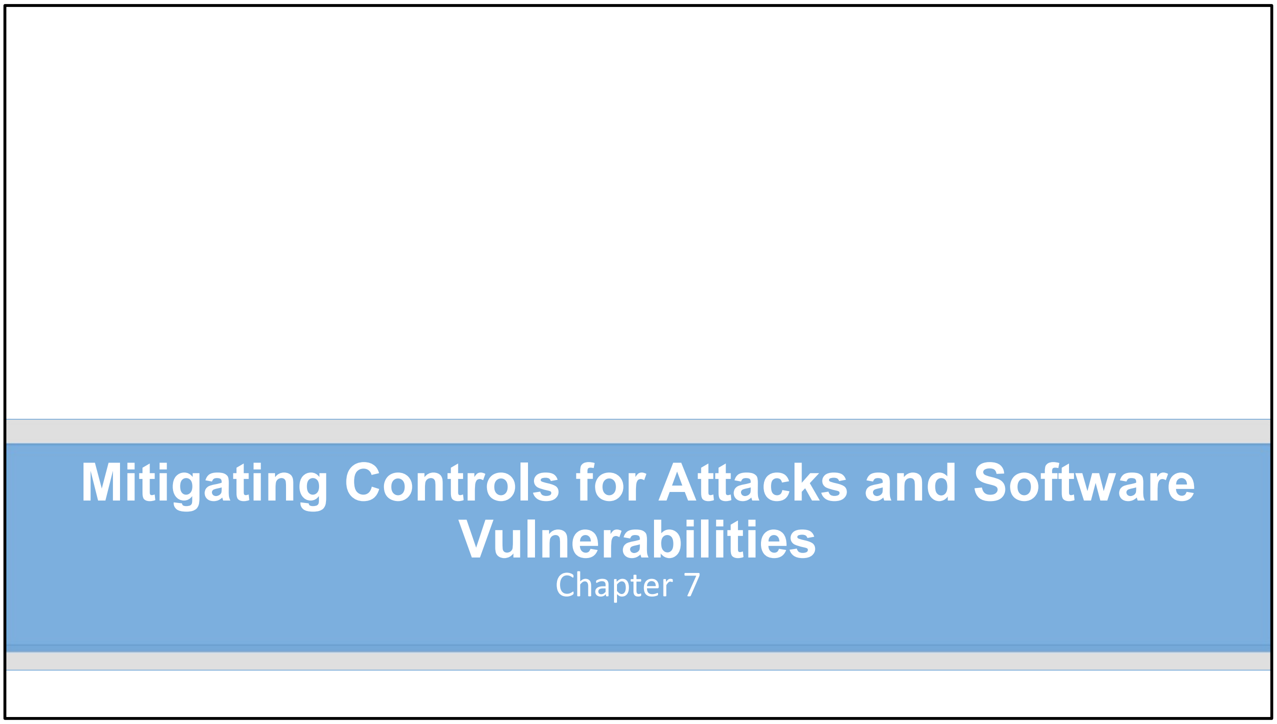


Mitigating Controls for Attacks and Software Vulnerabilities

Chapter 7

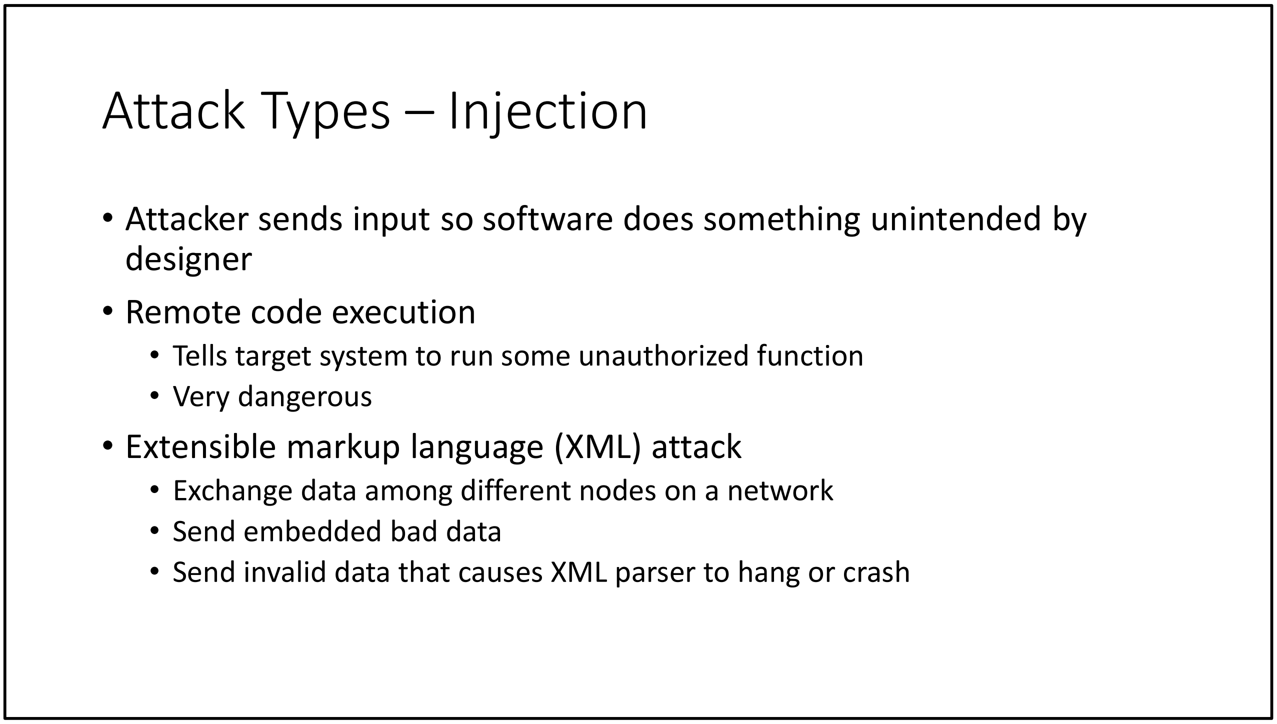
Episode 7.01

Injection and Overflow Attacks

Objective 1.7 Given a scenario, implement controls to mitigate attacks and software vulnerabilities.

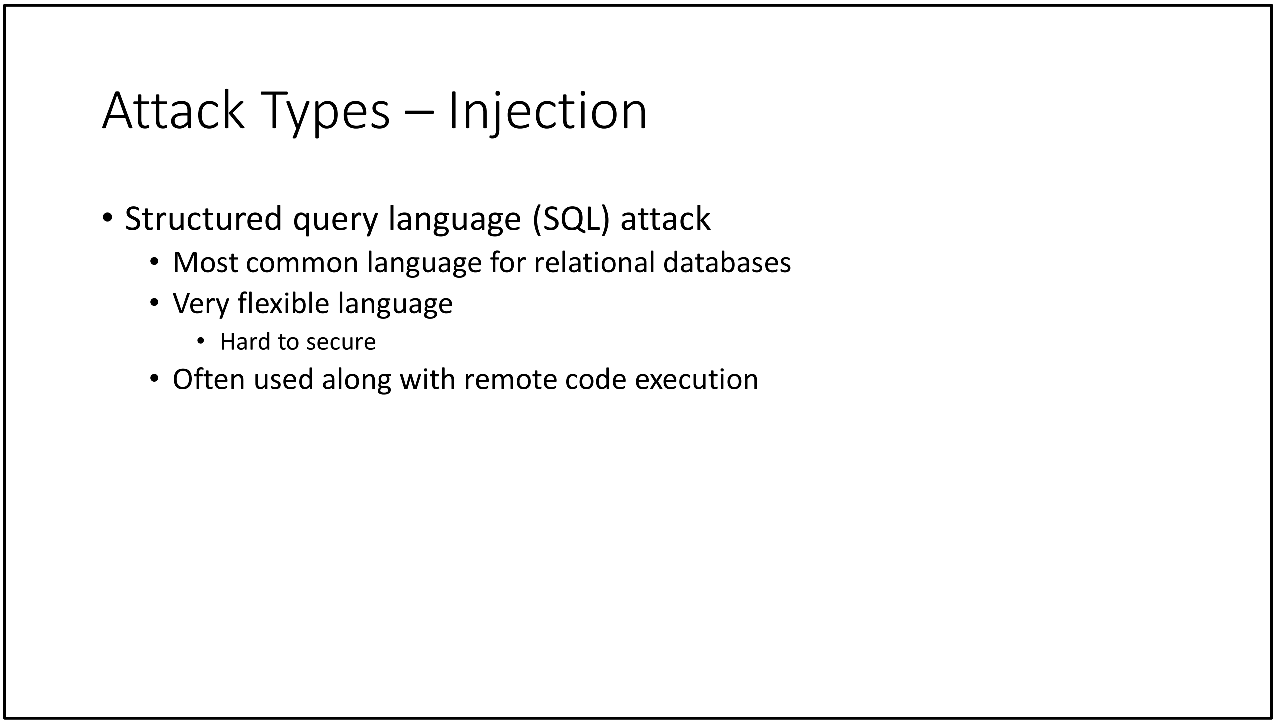
• Attack types

- Extensible markup language (XML) attack
- Structured query language (SQL) injection
- Overflow attack
    - Buffer
    - Integer
    - Heap
- Cross-site scripting
    - Reflected
    - Persistent
    - Document object model (DOM)
- Remote code execution
- Directory traversal



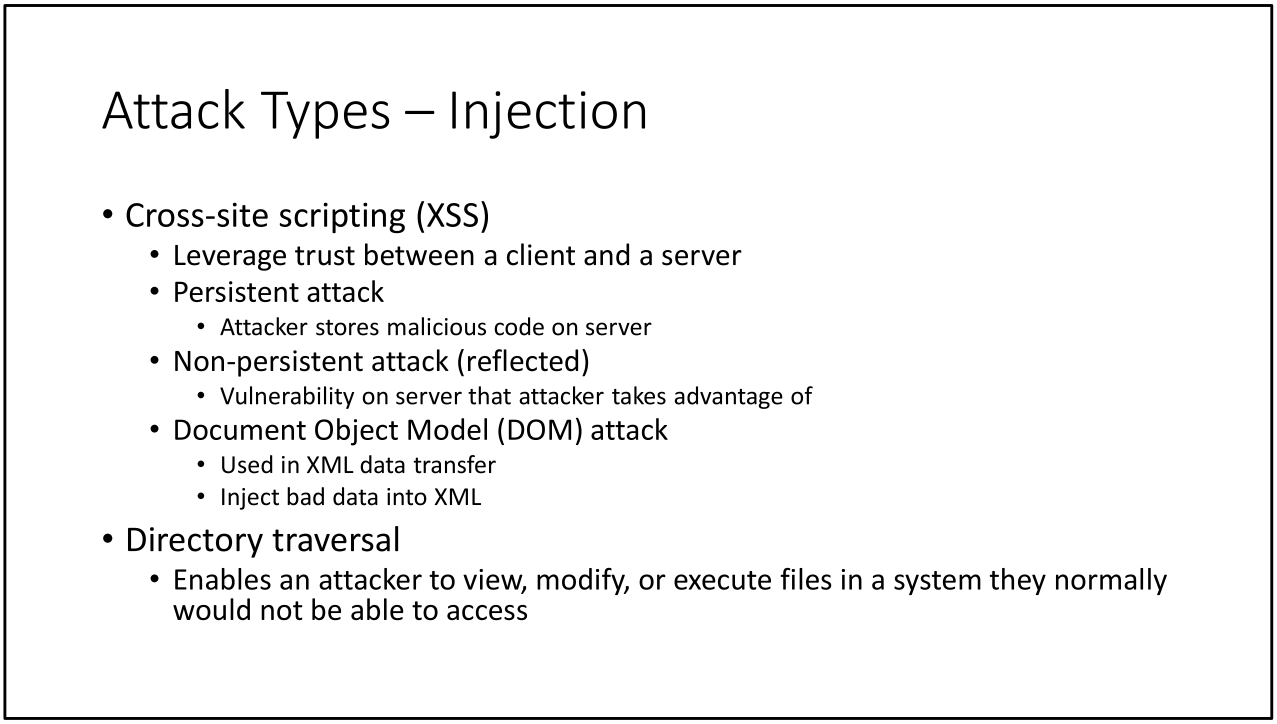
Attack Types – Injection

- Attacker sends input so software does something unintended by designer
- Remote code execution
  - Tells target system to run some unauthorized function
  - Very dangerous
- Extensible markup language (XML) attack
  - Exchange data among different nodes on a network
  - Send embedded bad data
  - Send invalid data that causes XML parser to hang or crash



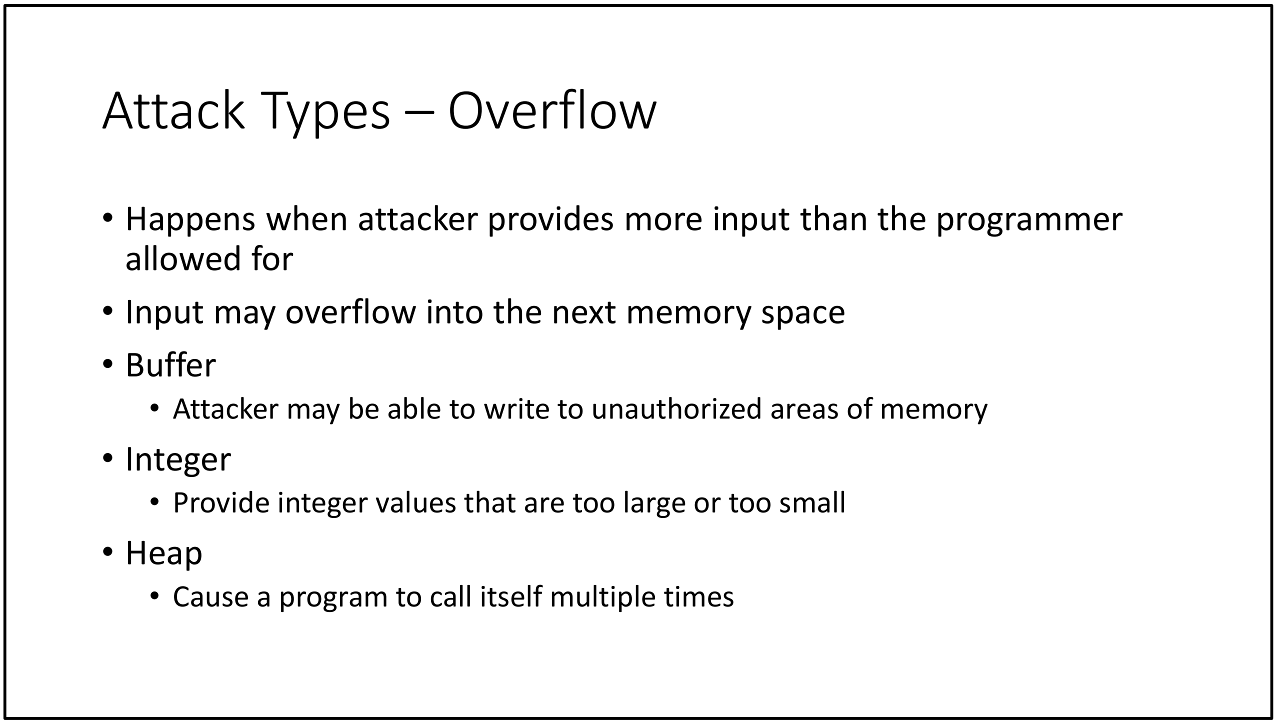
Attack Types – Injection

- Structured query language (SQL) attack
  - Most common language for relational databases
  - Very flexible language
    - Hard to secure
  - Often used along with remote code execution



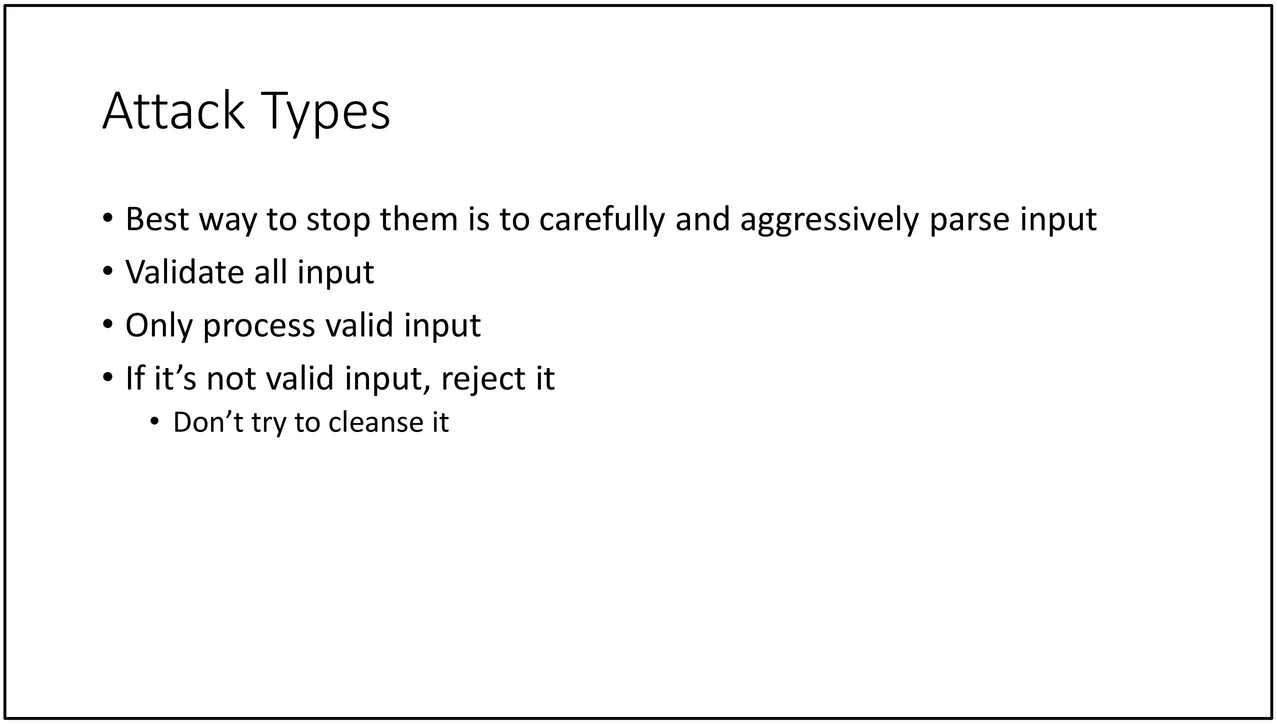
Attack Types – Injection

- Cross-site scripting (XSS)
  - Leverage trust between a client and a server
  - Persistent attack
    - Attacker stores malicious code on server
  - Non-persistent attack (reflected)
    - Vulnerability on server that attacker takes advantage of
  - Document Object Model (DOM) attack
    - Used in XML data transfer
    - Inject bad data into XML
- Directory traversal
  - Enables an attacker to view, modify, or execute files in a system they normally would not be able to access



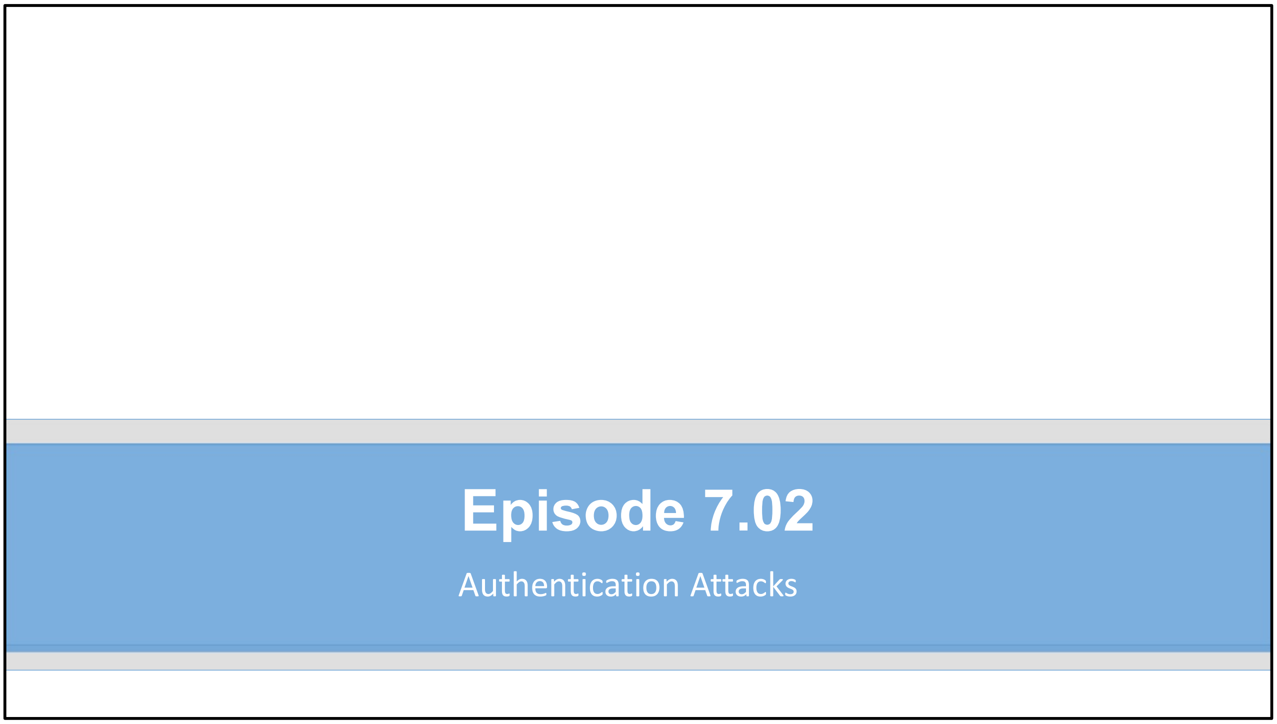
Attack Types – Overflow

- Happens when attacker provides more input than the programmer allowed for
- Input may overflow into the next memory space
- Buffer
  - Attacker may be able to write to unauthorized areas of memory
- Integer
  - Provide integer values that are too large or too small
- Heap
  - Cause a program to call itself multiple times



Attack Types

- Best way to stop them is to carefully and aggressively parse input
- Validate all input
- Only process valid input
- If it’s not valid input, reject it
  - Don’t try to cleanse it



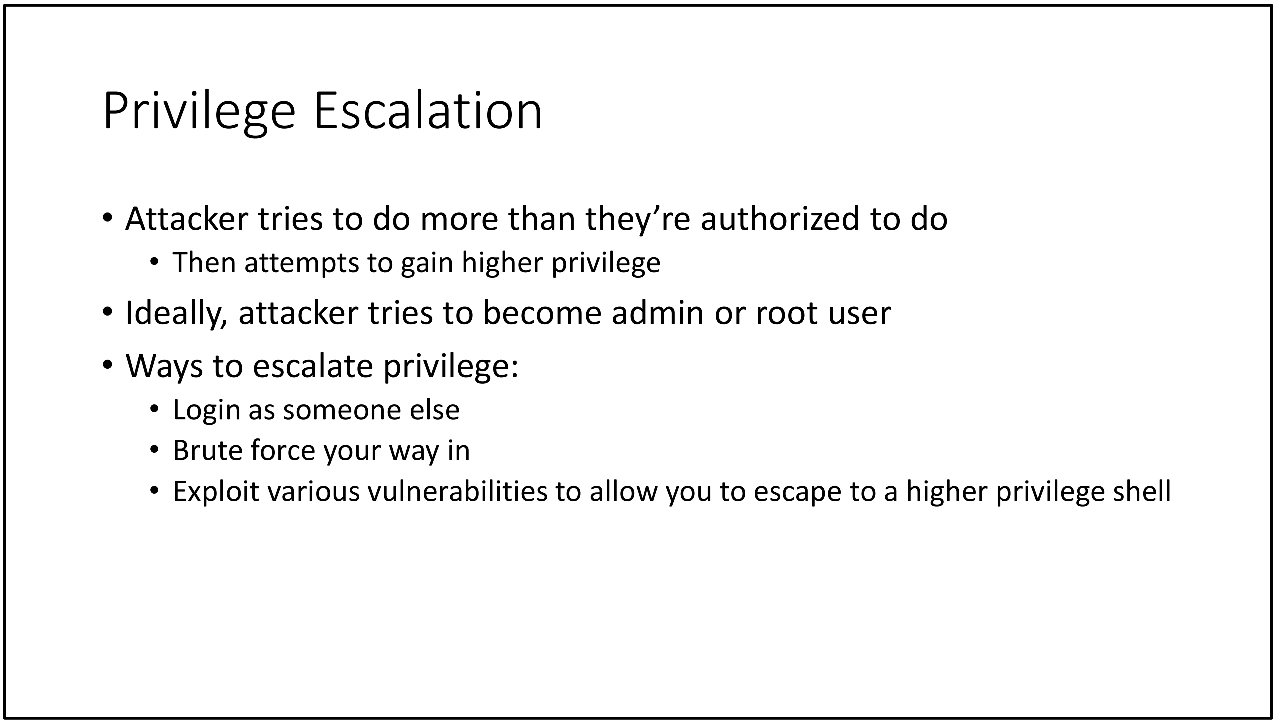
Episode 7.02

Authentication Attacks

Objective 1.7 Given a scenario, implement controls to mitigate attacks and software vulnerabilities.

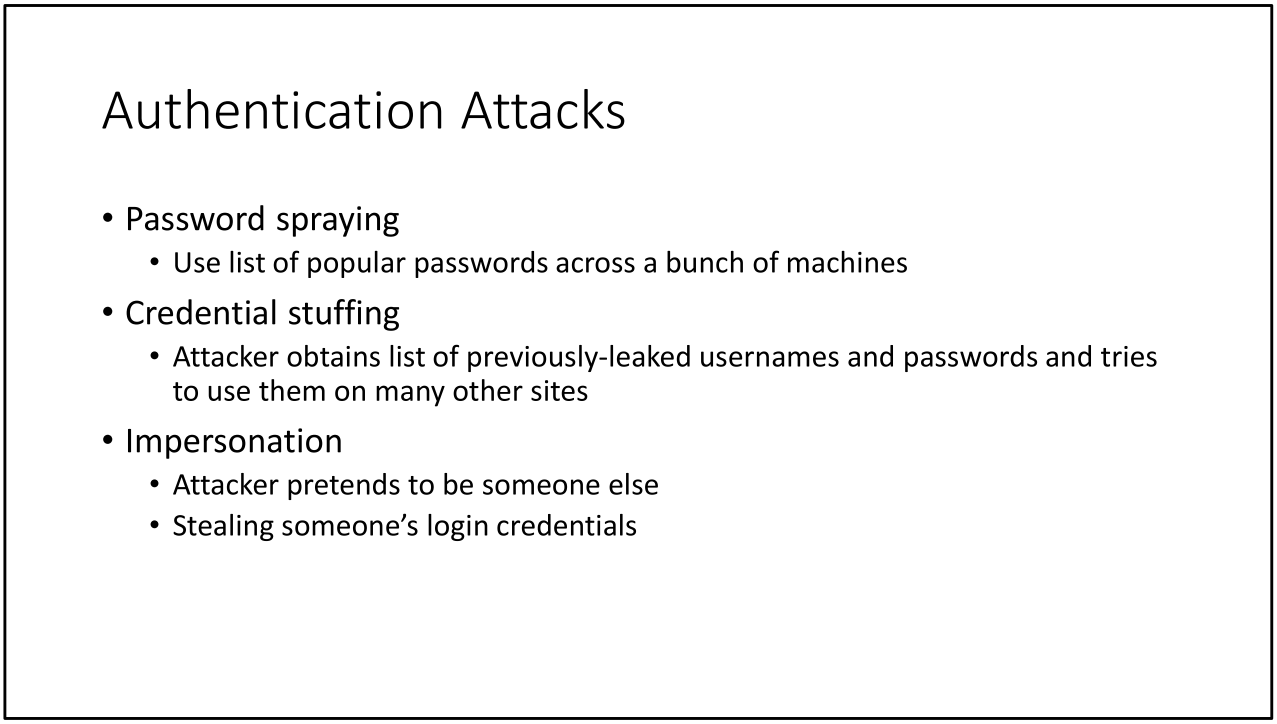
• Attack types

- Privilege escalation
- Password spraying
- Credential stuffing
- Impersonation



Privilege Escalation

- Attacker tries to do more than they're authorized to do
  - Then attempts to gain higher privilege
- Ideally, attacker tries to become admin or root user
- Ways to escalate privilege:
  - Login as someone else
  - Brute force your way in
  - Exploit various vulnerabilities to allow you to escape to a higher privilege shell



Authentication Attacks

- Password spraying
  - Use list of popular passwords across a bunch of machines
- Credential stuffing
  - Attacker obtains list of previously-leaked usernames and passwords and tries to use them on many other sites
- Impersonation
  - Attacker pretends to be someone else
  - Stealing someone’s login credentials

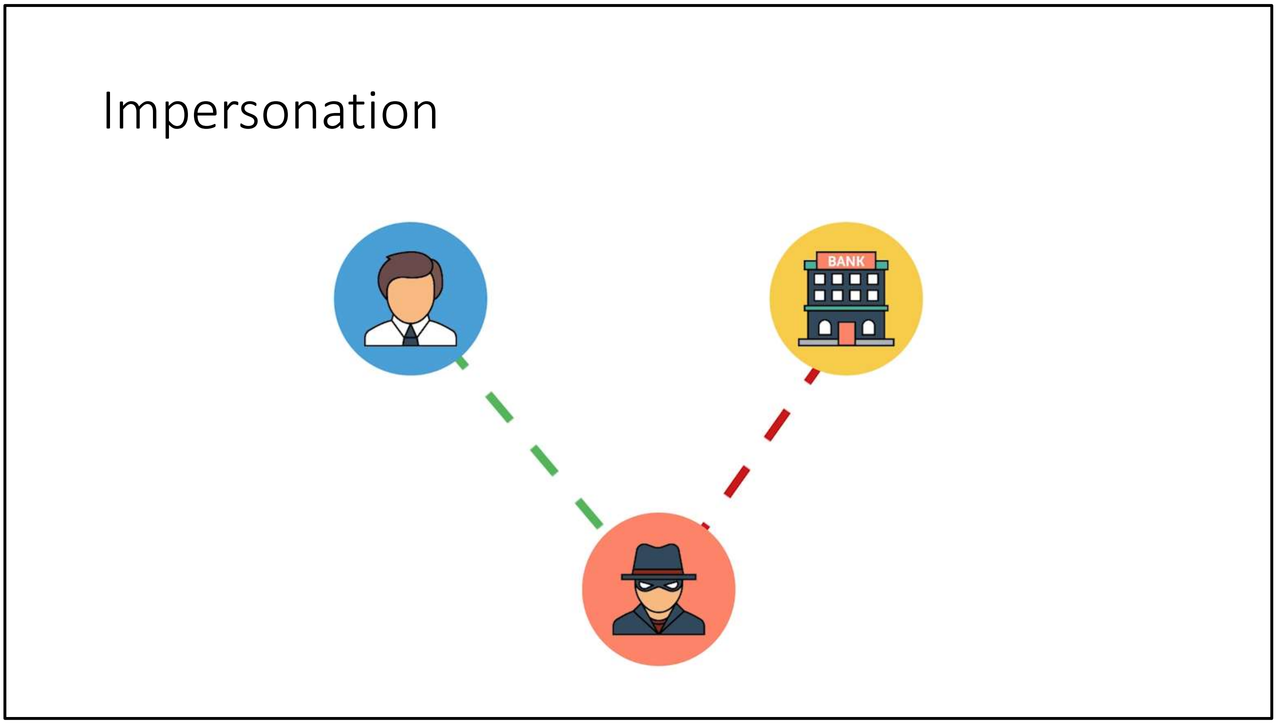
Episode 7.03

Exploits

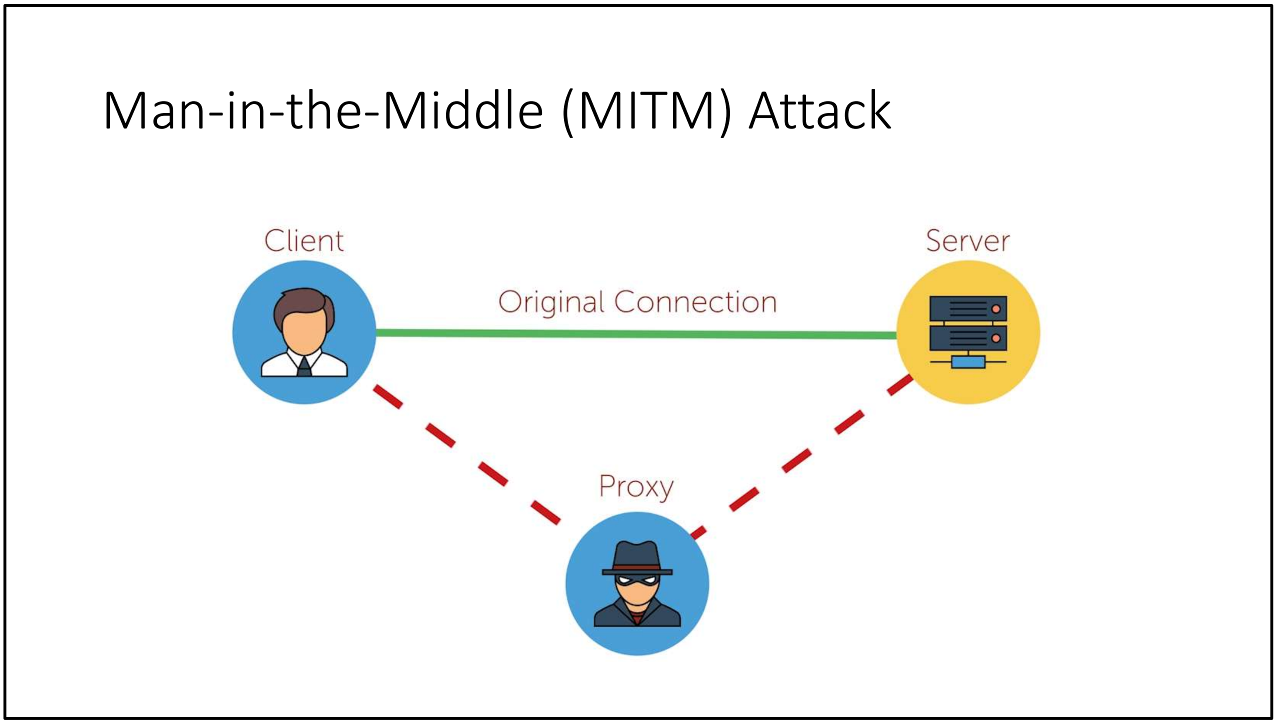
Objective 1.7 Given a scenario, implement controls to mitigate attacks and software vulnerabilities.

• Attack types

- Privilege escalation
- Impersonation
- Man-in-the-middle attack
- Session hijacking
- Rootkit
- Cross-site scripting

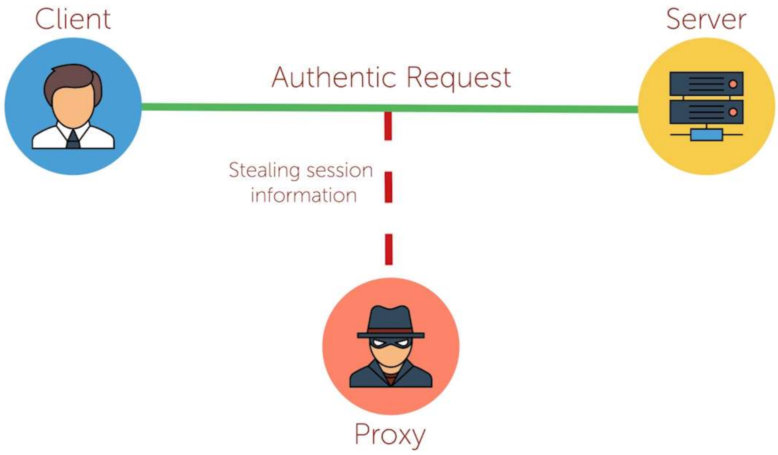


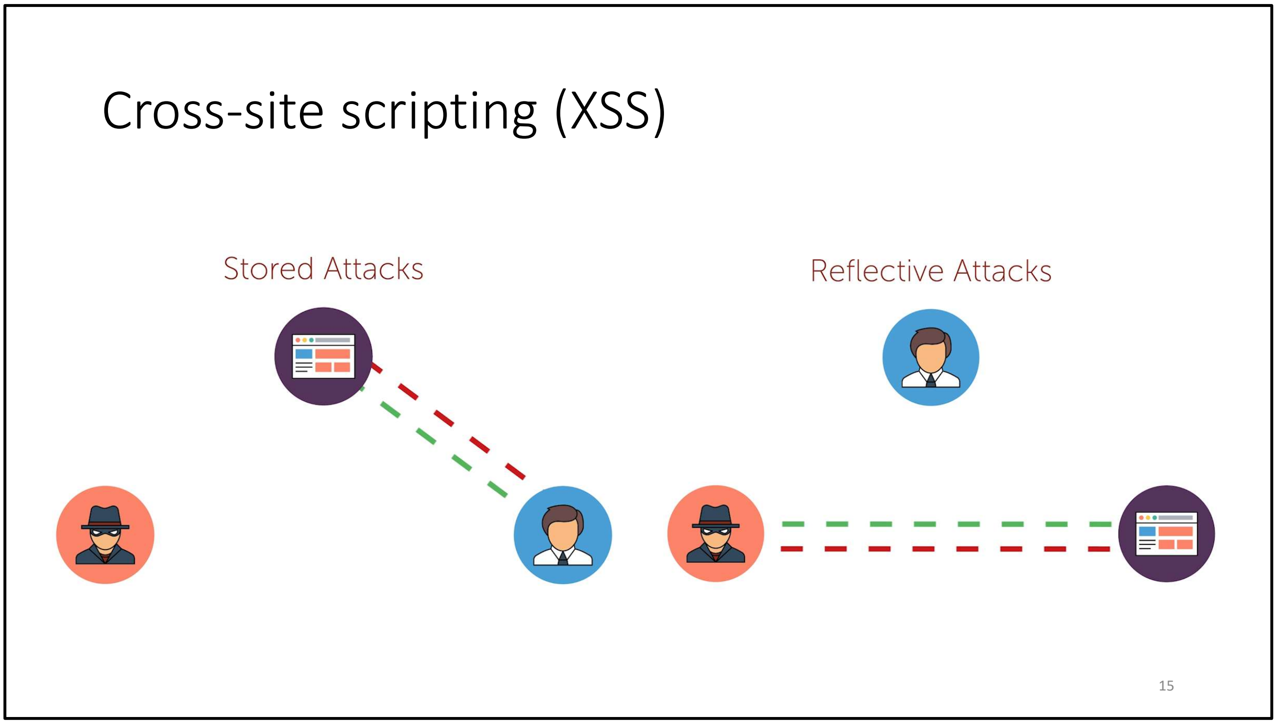
Impersonation



Man-in-the-Middle (MITM) Attack

Session Hijack

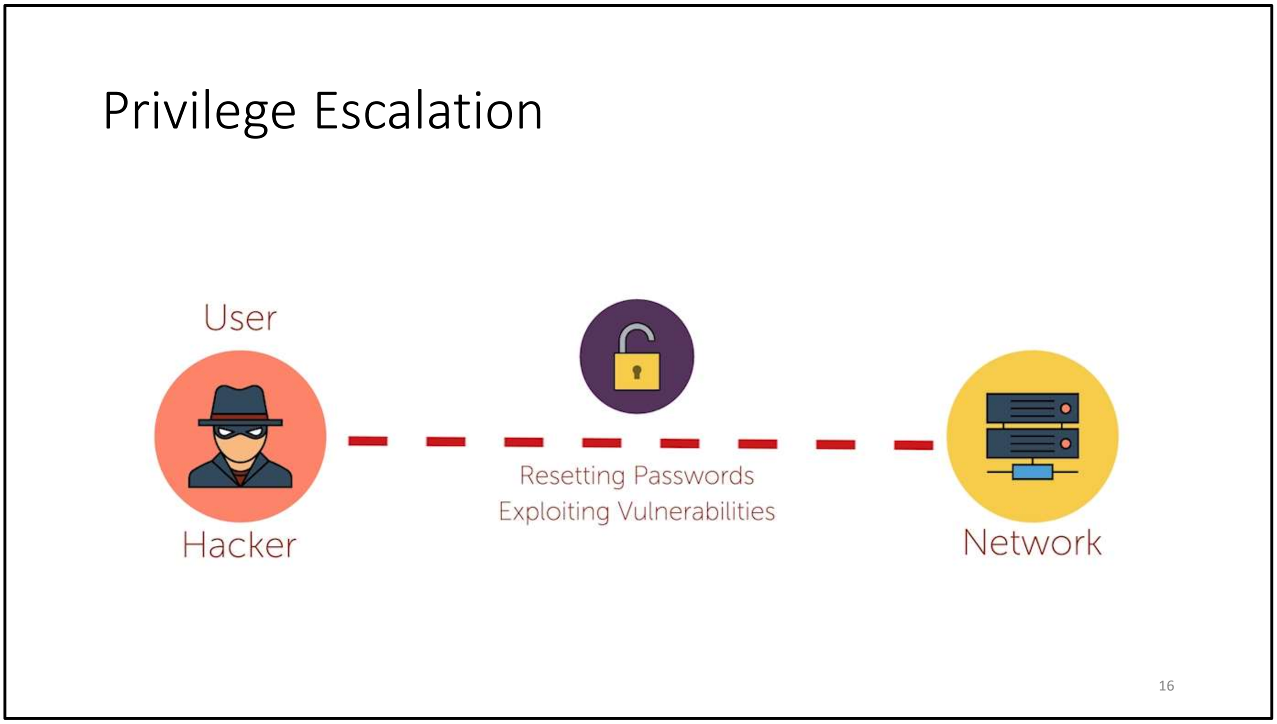




Cross-site scripting (XSS)

Stored Attacks

Reflective Attacks



Privilege Escalation

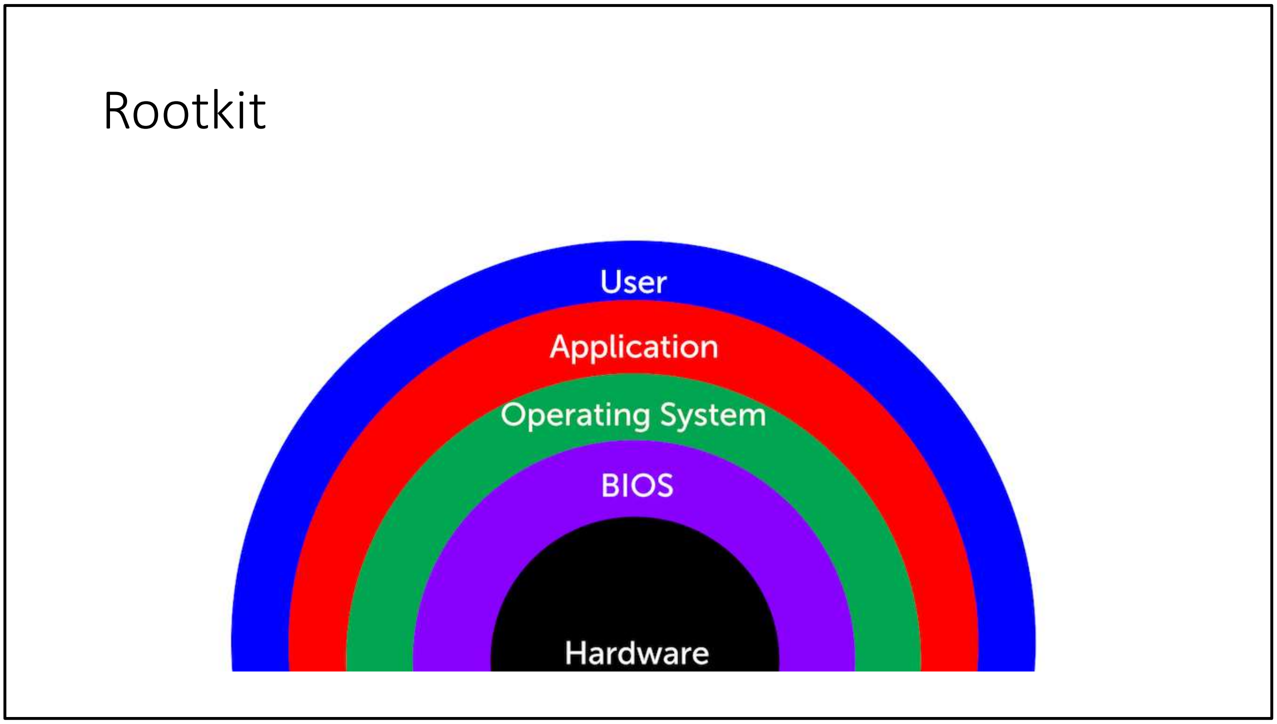
User

Hacker

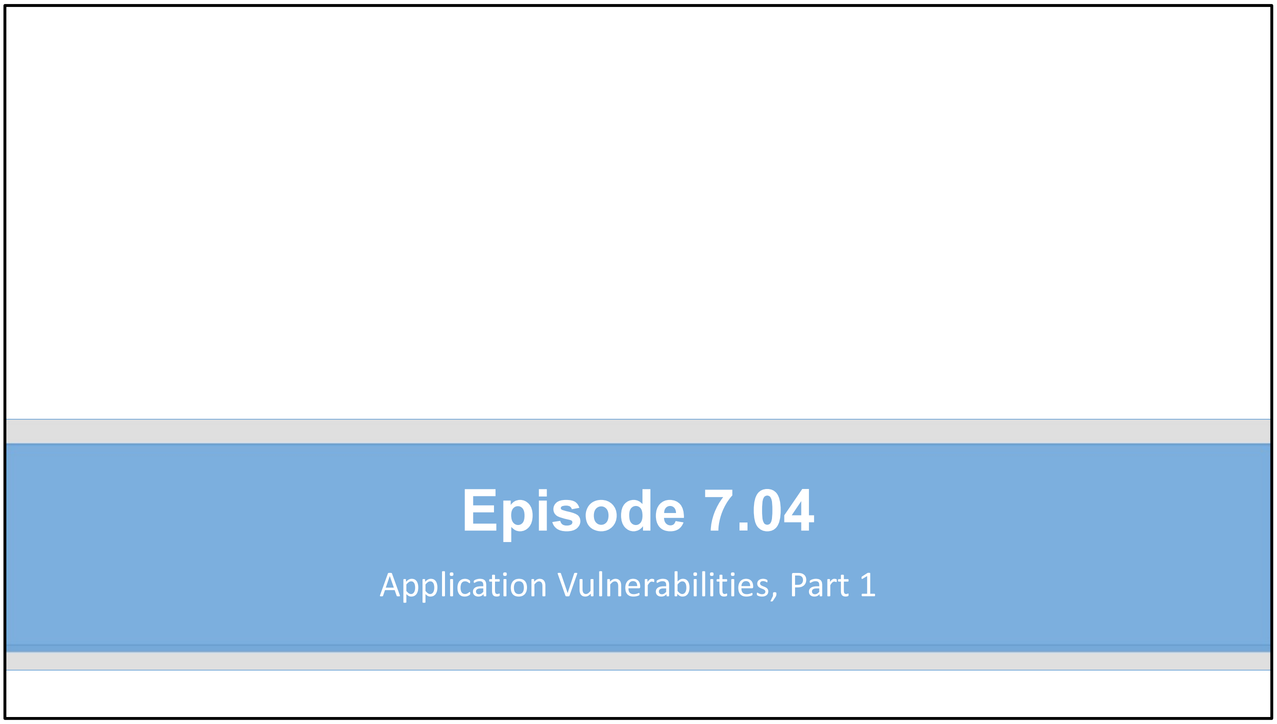
Resetting Passwords

Exploiting Vulnerabilities

Network



Rootkit



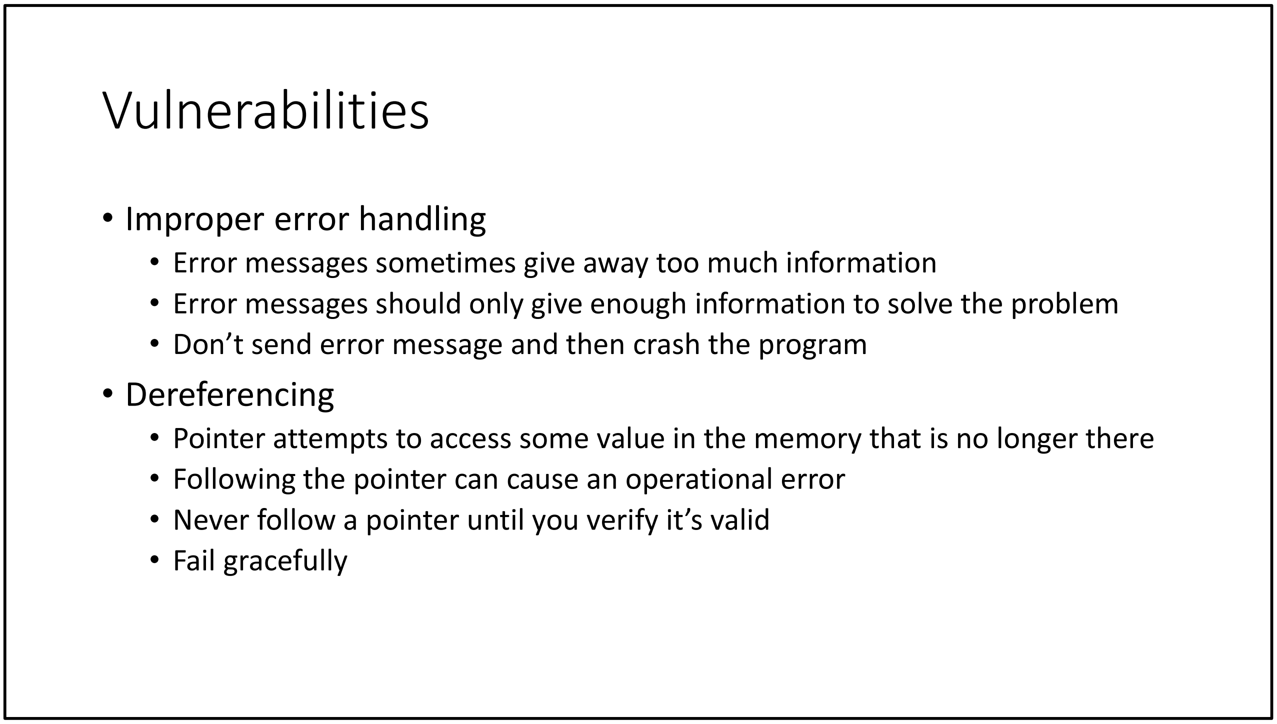
Episode 7.04

Application Vulnerabilities, Part 1

Objective 1.7 Given a scenario, implement controls to mitigate attacks and software vulnerabilities.

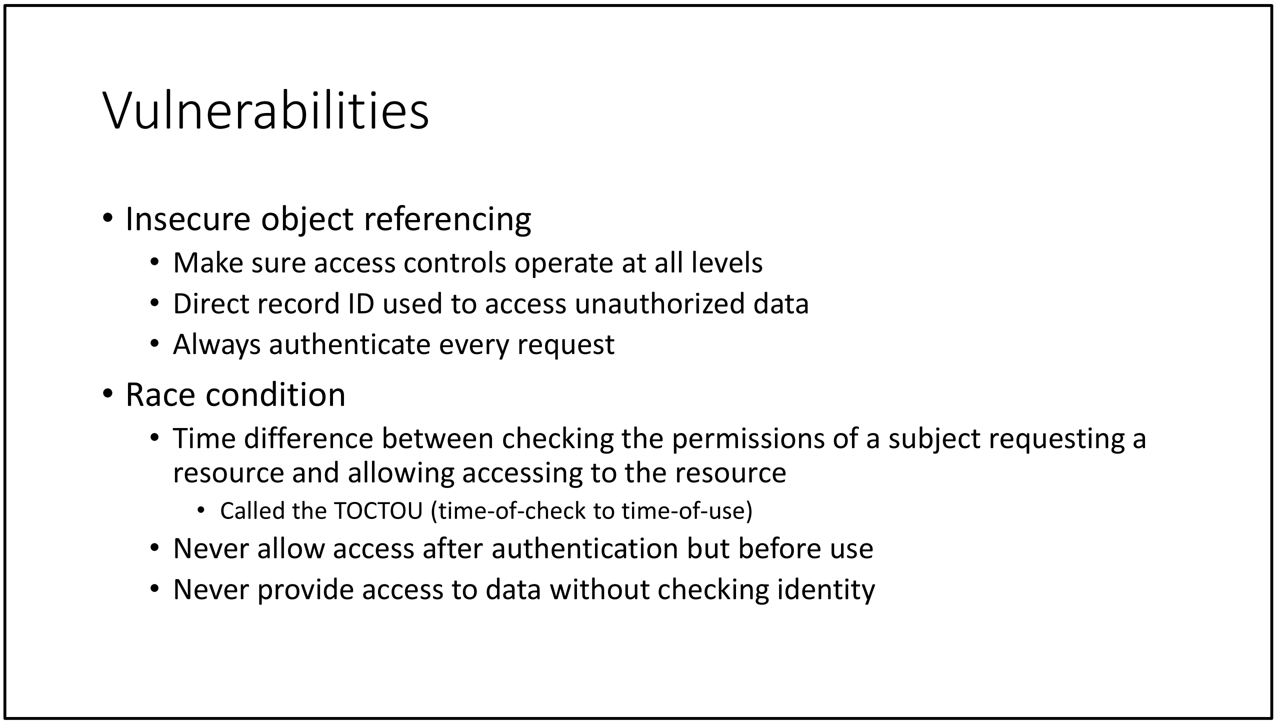
• Vulnerabilities

- Improper error handling
- Dereferencing
- Insecure object reference
- Race condition
- Broken authentication
- Sensitive data exposure



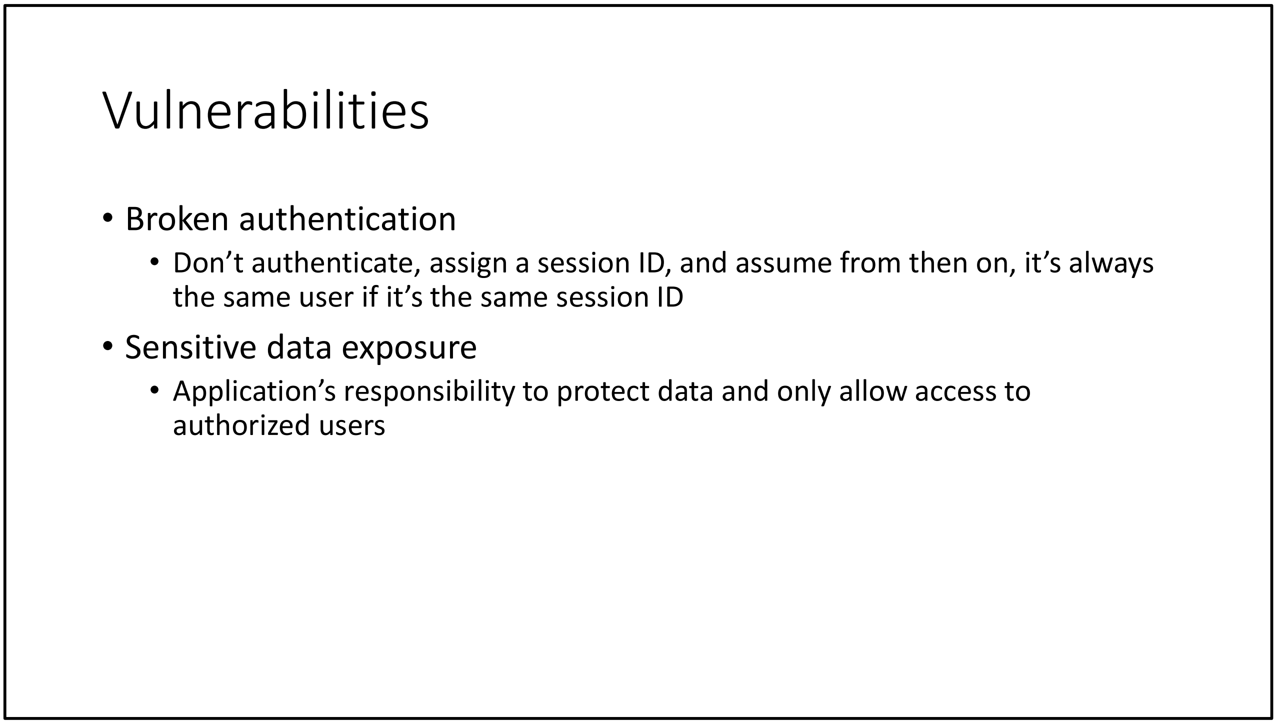
Vulnerabilities

- Improper error handling
  - Error messages sometimes give away too much information
  - Error messages should only give enough information to solve the problem
  - Don’t send error message and then crash the program
- Dereferencing
  - Pointer attempts to access some value in the memory that is no longer there
  - Following the pointer can cause an operational error
  - Never follow a pointer until you verify it’s valid
  - Fail gracefully



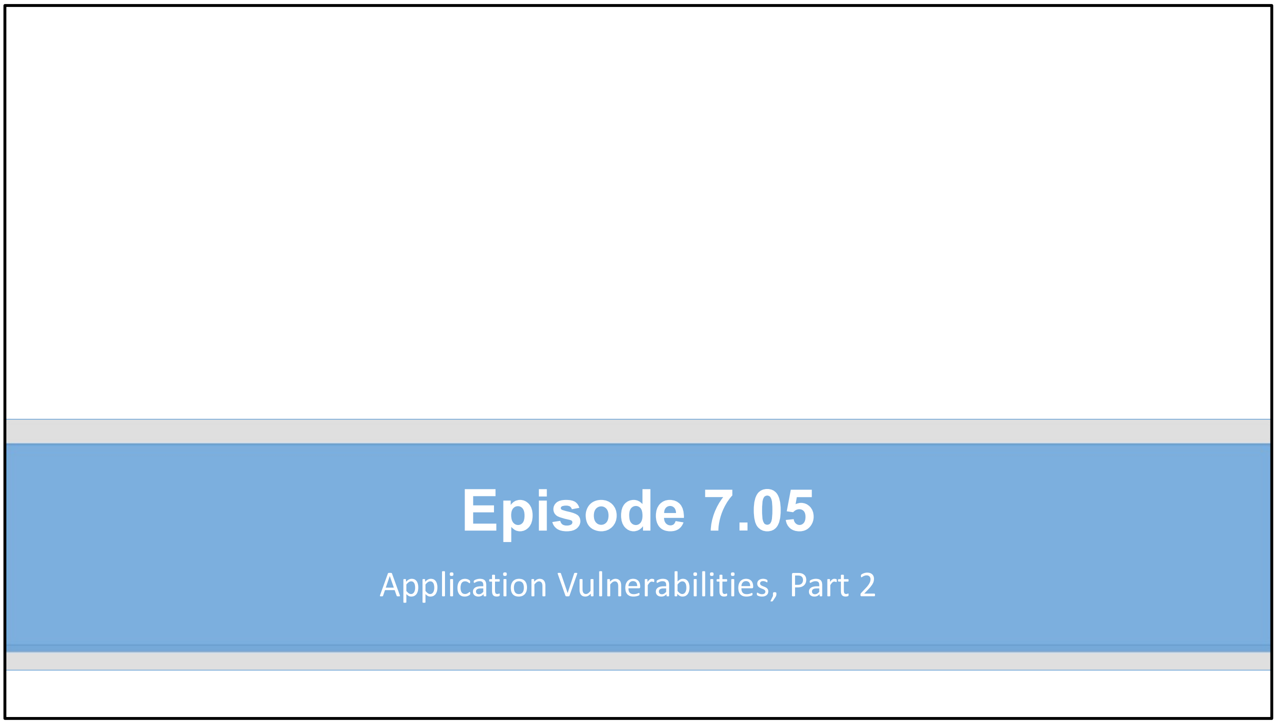
Vulnerabilities

- Insecure object referencing
  - Make sure access controls operate at all levels
  - Direct record ID used to access unauthorized data
  - Always authenticate every request
- Race condition
  - Time difference between checking the permissions of a subject requesting a resource and allowing accessing to the resource
    - Called the TOCTOU (time-of-check to time-of-use)
  - Never allow access after authentication but before use
  - Never provide access to data without checking identity



Vulnerabilities

- Broken authentication
  - Don’t authenticate, assign a session ID, and assume from then on, it’s always the same user if it’s the same session ID
- Sensitive data exposure
  - Application’s responsibility to protect data and only allow access to authorized users



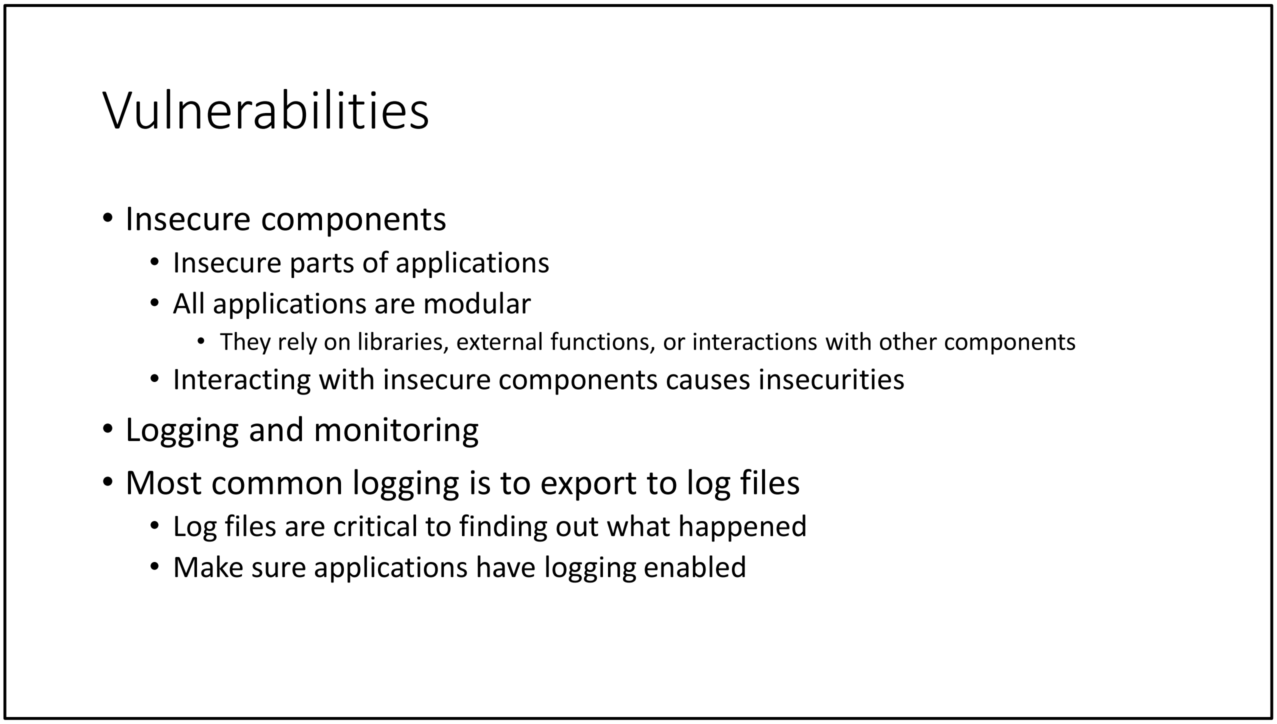
Episode 7.05

Application Vulnerabilities, Part 2

Objective 1.7 Given a scenario, implement controls to mitigate attacks and software vulnerabilities.

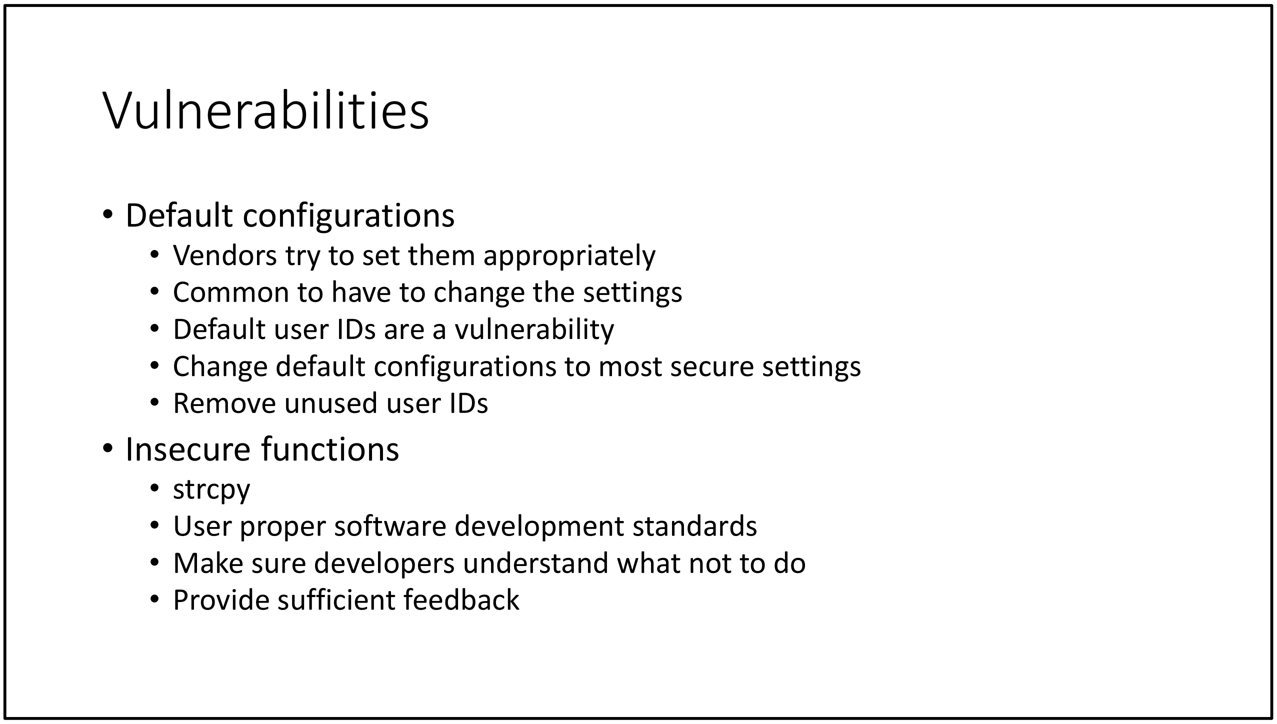
- Vulnerabilities

- Insecure components
- Insufficient logging and monitoring
- Weak or default configurations
- Use of insecure functions
    - strcpy



Vulnerabilities

- Insecure components
  - Insecure parts of applications
  - All applications are modular
    - They rely on libraries, external functions, or interactions with other components
  - Interacting with insecure components causes insecurities
- Logging and monitoring
- Most common logging is to export to log files
  - Log files are critical to finding out what happened
  - Make sure applications have logging enabled



Vulnerabilities

- Default configurations
  - Vendors try to set them appropriately
  - Common to have to change the settings
  - Default user IDs are a vulnerability
  - Change default configurations to most secure settings
  - Remove unused user IDs
- Insecure functions
  - strcpy
  - User proper software development standards
  - Make sure developers understand what not to do
  - Provide sufficient feedback