Ethical Hacking(Unit 4), VelTech technical University, Chennai

Presentation · March 2024

DOI: 10.13140/RG.2.2.36635.48164

1 author:

Arun Anoop Mandankandy
Vivekananda College of Engineering & Technology
71 PUBLICATIONS **89** CITATIONS

SEE PROFILE

Ethical Hacking(Unit 4)

Dr.Arun Anoop M CEH CHFI
Associate Professor
Dept. of CSE
SoC, Veltech University, Chennai.

EC-Council

C|EH
Certified Ethical Hacker

Ethical Hacking and Countermeasures v11

PROFESSIONAL SERIES

EC-COUNCIL OFFICIAL CURRICULA

EC-Council

EC-Council New Mexico
101C Sun Ave NE
Albuquerque, NM 87109

NOTICE TO THE READER

Copyright

Every contents copied from EC Council CEH material.

Contents used only for student's study purpose. I didn't modify your contents, instead images collected and pasted.

If any issues, I can remove it from any of my website portal.

Being a certified Ethical Hacker, I have refereed your ebook. After that based on my syllabus ; identified , collected & arranged detailed contents for student's study purpose.

CEH Exam Information

CEH Exam Details	
Exam Title	**Certified Ethical Hacker (CEH)**
Exam Code	**312-50**
Availability	**EC-Council Exam Portal** (please visit **https://www.eccexam.com**) **VUE** (please visit **https://home.pearsonvue.com/eccouncil**)
Duration	**4 Hours**
Questions	**125**
Passing Score	Please refer **https://cert.eccouncil.org/faq.html**

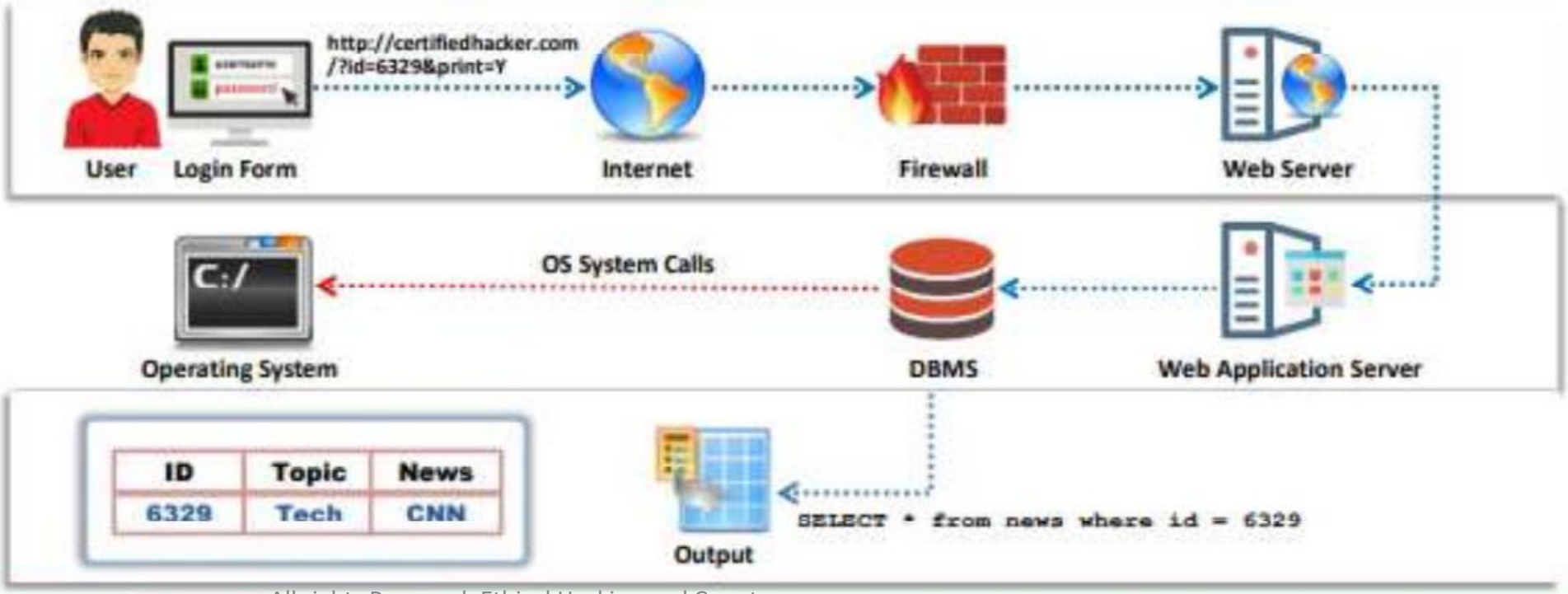
Please visit **https://cert.eccouncil.org/certified-ethical-hacker.html** for more information.

Introduction to Web Applications

Topic 1

- Web applications provide an **interface between end users and web servers** through a set of web pages that are generated at the server end or contain script code to be executed dynamically within the client web browser
- Though web applications enforce certain **security policies**, they are vulnerable to various attacks such as SQL injection, cross-site scripting, and session hijacking

How Web Applications Work



Web applications are software programs that run on web browsers and act as the interface between users and web servers through web pages. They enable the users to request, submit, and retrieve data to/from a database over the Internet by interacting through a user-friendly graphical user interface (GUI). Users can input data via a keyboard, mouse, or touch interface depending on the device they are using to access the web application. Based on browser-supported programming languages such as JavaScript, HTML, and CSS, web applications work in combination with other programming languages such as SQL to access data from the databases.

Web applications are developed as dynamic web pages, and they allow users to communicate with servers using server-side scripts. They allow users to perform specific tasks such as searching, sending emails, connecting with friends, online shopping, and tracking and tracing. Furthermore, there are several desktop applications that provide users with the flexibility to work with the Internet.

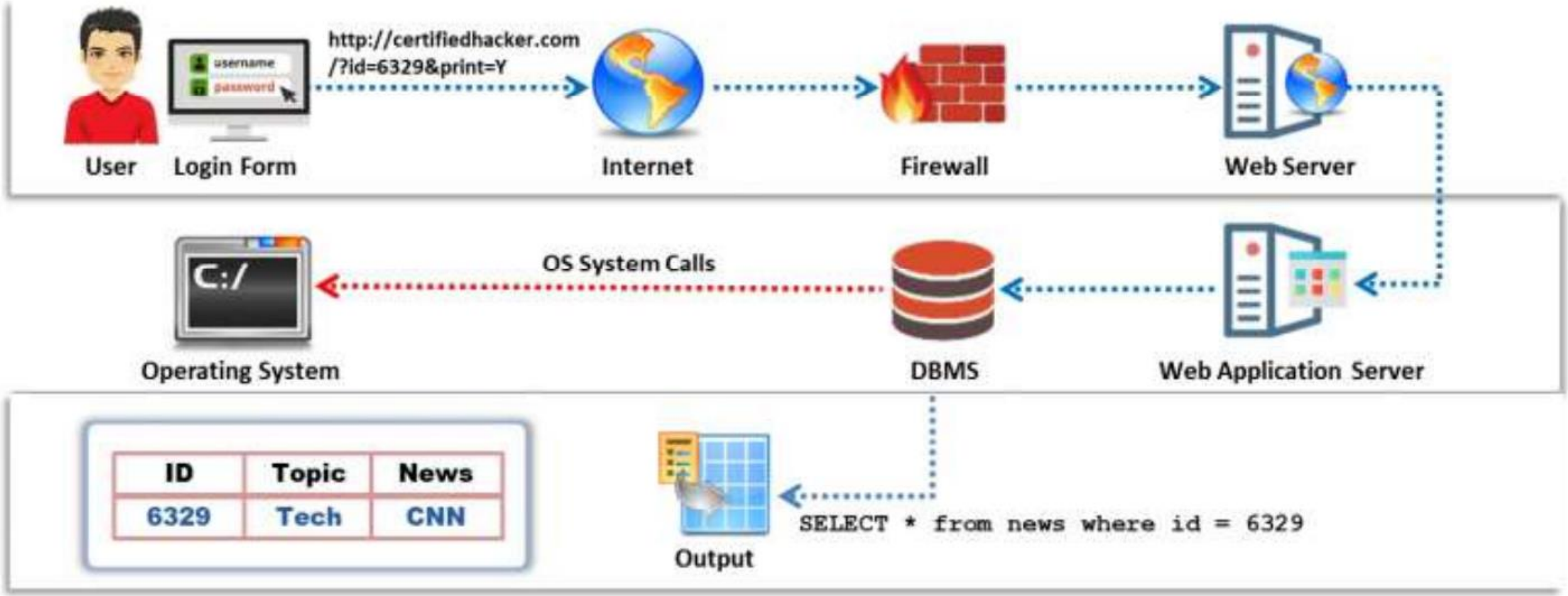


Figure 14.1: Working of web applications

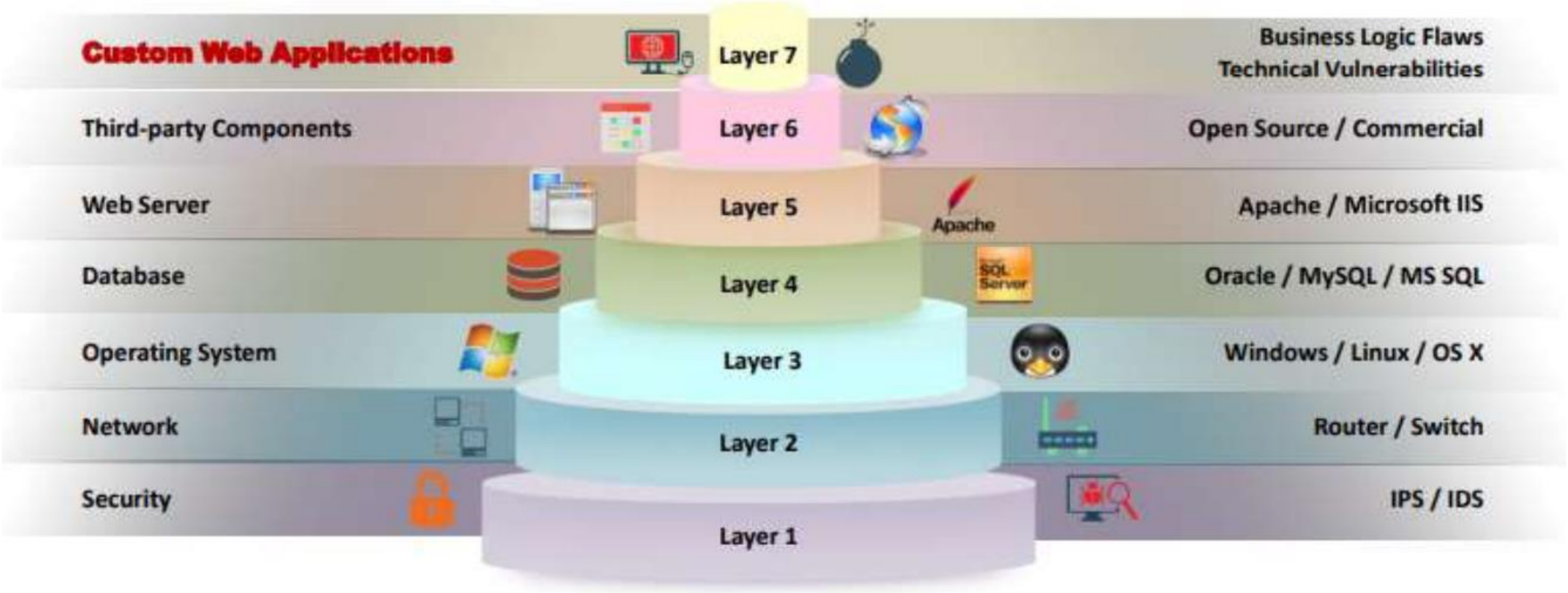
How Web Applications Work

The main function of web applications is to fetch user-requested data from a database. When a user clicks or enters a URL in a browser, the web application immediately displays the requested website content in the browser.

This mechanism involves the following steps:

- First, the user enters the website name or URL in the browser. Then, the user's request is sent to the web server.
- On receiving the request, the web server checks the file extension:
  - If the user requests a simple web page with an HTM or HTML extension, the web server processes the request and sends the file to the user's browser.
  - If the user requests a web page with an extension that needs to be processed at the server side, such as php, asp, and cfm, then the web application server must process the request.
- Therefore, the web server passes the user's request to the web application server, which processes the user's request.
- The web application server then accesses the database to perform the requested task by updating or retrieving the information stored on it.
- After processing the request, the web application server finally sends the results to the web server, which in turn sends the results to the user's browser.

Vulnerability Stack



Vulnerability Stack

One maintains and accesses web applications through various levels that include custom web applications, third-party components, databases, web servers, operating systems, networks, and security. All the mechanisms or services employed at each layer enable the user to access the web application securely. When considering web applications, the organization considers security as a critical component because web applications are major sources of attacks. The vulnerability stack shows various layers and the corresponding elements/mechanisms/services that make web applications vulnerable.

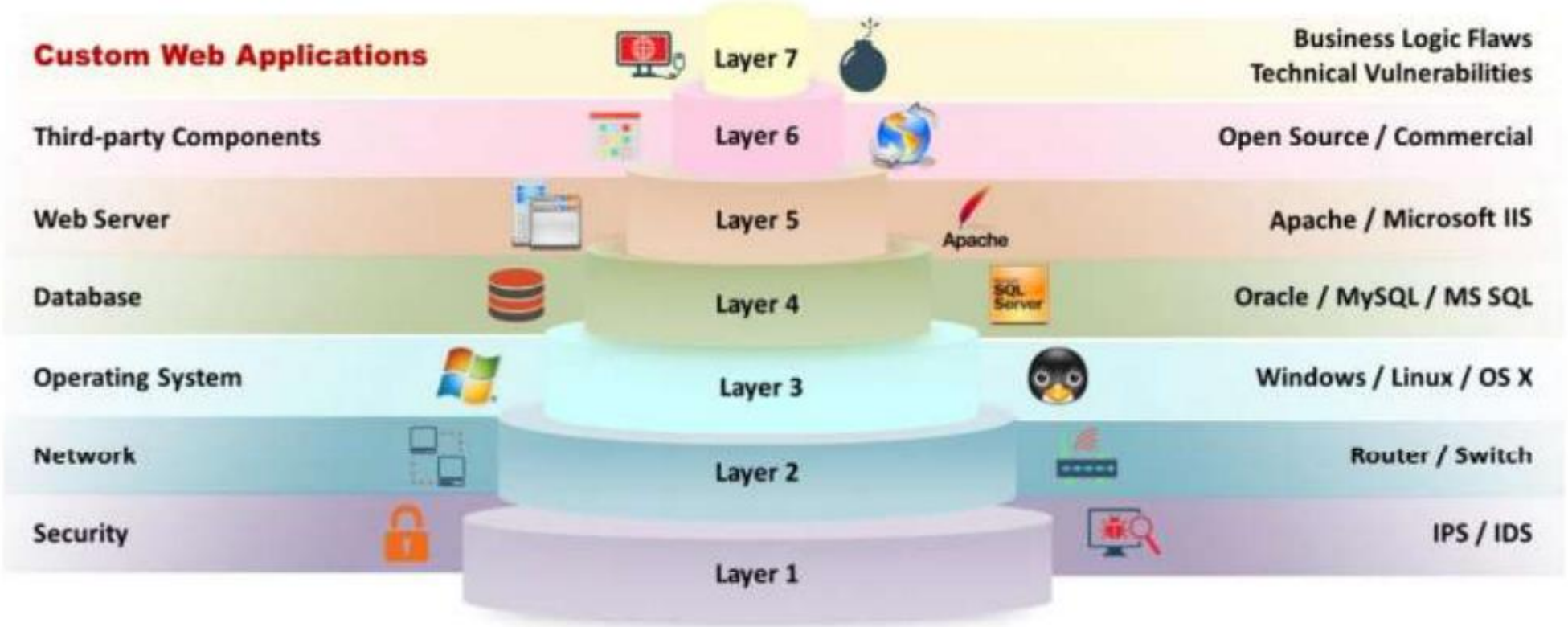


Figure 14.4: Vulnerability Stack

- **Layer 7**

  If an attacker finds vulnerabilities in the business logic (implemented using languages such as .NET and Java), he/she can exploit these vulnerabilities by performing input validation attacks such as XSS.

- **Layer 6**

  Third-party components are services that integrate with the website to achieve certain functionality (e.g., Amazon.com targeted by an attacker is the main website; citrix.com is a third-party website).

  When customers choose a product to buy, they click on the Buy/Checkout button. This redirects them to their online banking account through a payment gateway. Third-party websites such as citrix.com offer such payment gateways. Attackers might exploit such redirection and use it as a medium/pathway to enter Amazon.com and exploit it.

- **Layer 5**

  Web servers are software programs that host websites. When users access a website, they send a URL request to the web server. The server parses this request and responds with a web page that appears in the browser. Attackers can perform footprinting on a web server that hosts the target website and grab banners that contain information such as the web server name and its version. They can also use tools such as Nmap to gather such information. Then, they might start searching for published vulnerabilities in the CVE database for that particular web server or service version number and exploit any that they find.

- **Layer 4**

  Databases store sensitive user information such as user IDs, passwords, phone numbers, and other particulars. There could be vulnerabilities in the database of the target website. These vulnerabilities can be exploited by attackers using tools such as sqlmap to gain control of the target's database.

- **Layer 3**

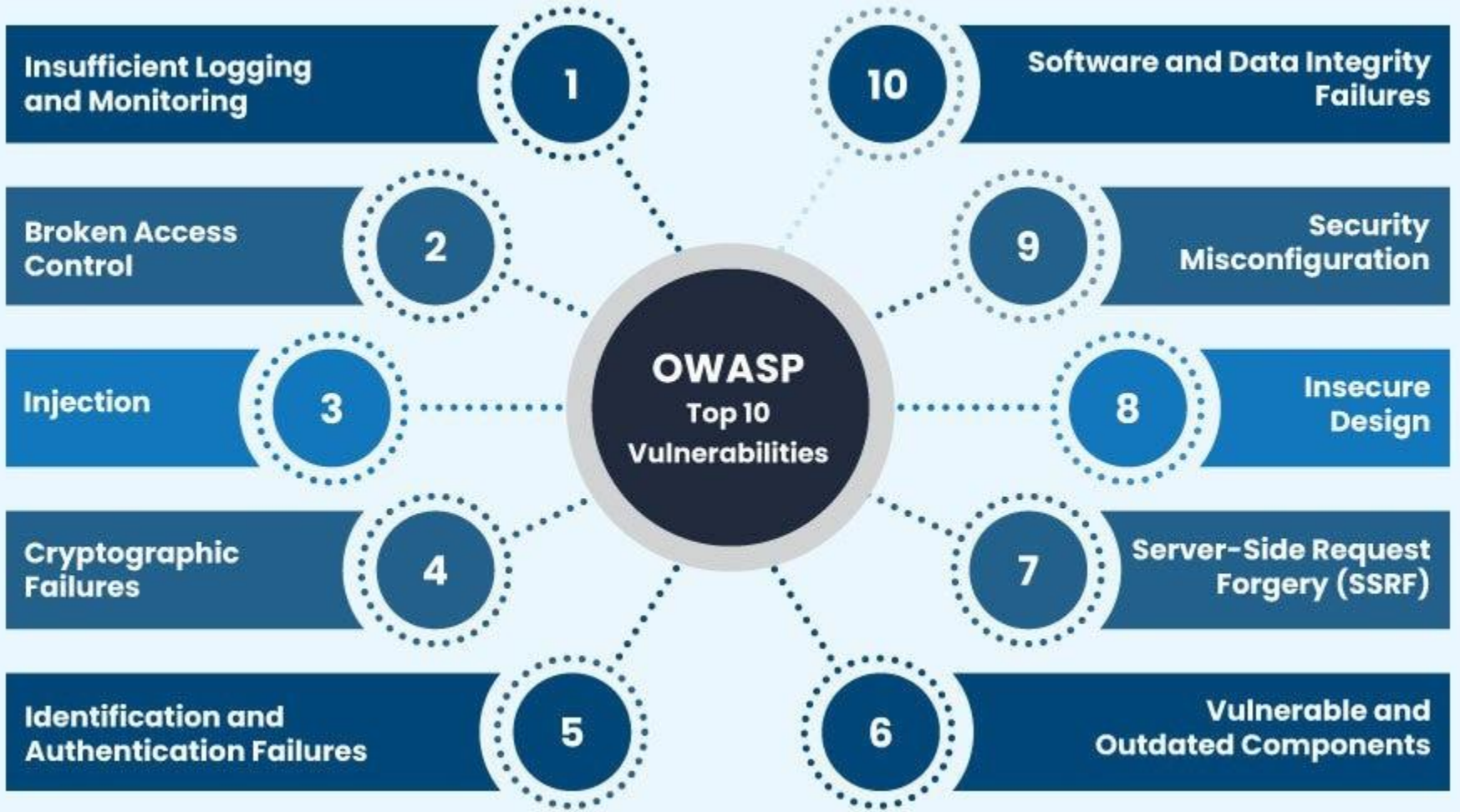
  Attackers scan an operating system to find open ports and vulnerabilities, and they develop viruses/backdoors to exploit them. They send malware through the open ports to the target machine; by running such malware, they can compromise the machine and gain control over it. Later, they try to access the databases of the target website.

- **Layer 2**

  Routers/switches route network traffic only to specific machines. Attackers flood these switches with numerous requests that exhaust the CAM table, causing it to behave like a hub. Then, they focus on the target website by sniffing data (in the network), which can include credentials or other personal information.

- **Layer 1**

  IDS and IPS raise alarms if any malicious traffic enters a target machine or server. Attackers adopt evasion techniques to circumvent such systems so that they do not trigger any alarm while exploiting the target.



Topic 3

StrongBox IT
Security | Performance | Automation

OWASP TOP 10 VULNERABILITIES

- INJECTION
- BROKEN AUTHENTICATION
- SENSITIVE DATA EXPOSURE
- XML EXTERNAL ENTITIES (XXE)
- BROKEN ACCESS CONTROL
- SECURITY MISCONFIGURATION
- CROSS-SITE SCRIPTING (XSS)
- INSECURE DESERIALIZATION
- USING COMPONENTS WITH KNOWN VULNERABILITIES
- INSUFFICIENT LOGGING & MONITORING

OWASP Top 10 (2021)

CyberEdx

1 Broken Access Control

does not restrict user access to certain areas

2 Cryptographic Failure

flaw in encryption or decryption mechanism

3 Injection

invalid data into an app is sent by a hacker

4 Insecure Design

flaw in the design of the application

5 Security Misconfiguration

wrong configuration of software or hardware

6 Vulnerable and Outdated Components

software not patched with latest updates

7 Identification and Authentication Failures

identification or authentication not done properly

8 Software & Data Integrity Failures

integrity of data compromised

OWASP Top 10, is a list of 10 most common web app security risks. By writing code & robust testing, we can create secure apps that keep users' confidential data safe from attackers.

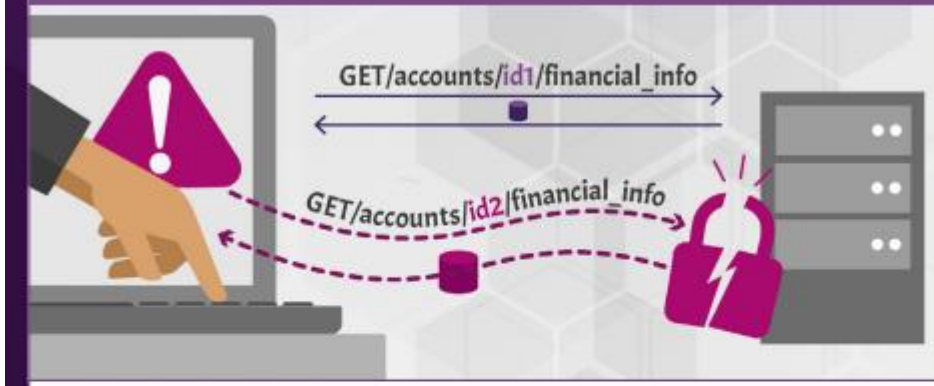
9 Security Logging & Monitoring Failures

failure to log, monitor events like login attempts

10 Server-Side Request Forgery

attacker manipulates app to send unauthorized requests to other systems

A1: BROKEN OBJECT LEVEL AUTHORIZATION



Attacker substitutes ID of their resource in API call with an ID of a resource belonging to another user. Lack of proper authorization checks allows access. This attack is also known as IDOR (Insecure Direct Object Reference).

USE CASES

- API call parameters use IDs of resourced accessed by the API:
  `/api/shop1/financial_details`
- Attackers replace the IDs of their resources with different ones, which they guessed:
  `/api/shop2/financial_details`
- The API does not check permissions and lets the call through
- Problem is aggravated if IDs can be enumerated:
  `/api/123/financial_details`

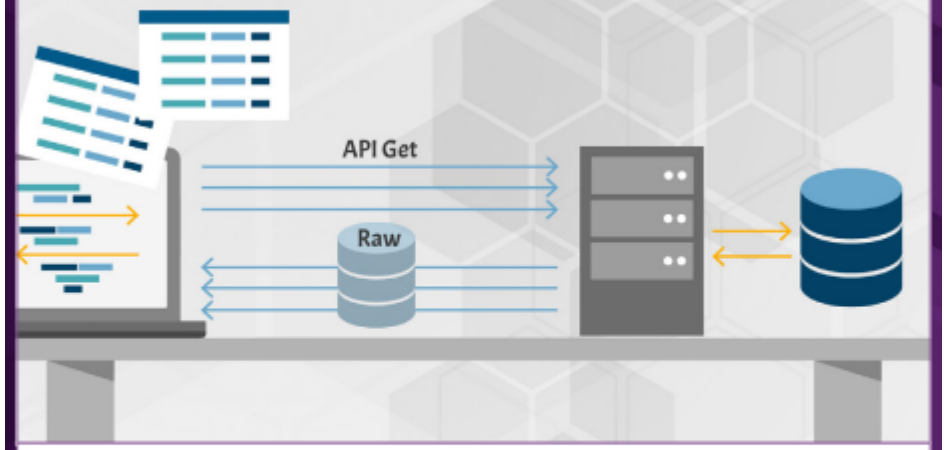
A2: BROKEN AUTHENTICATION

Poorly implemented API authentication allowing attackers to assume other users' identities.

USE CASES

- Unprotected APIs that are considered "internal"
- Weak authentication not following industry best practices
- Weak, not rotating API keys
- Weak, plain text, encrypted, poorly hashed, shared/default passwords
- Susceptible to brute force attacks and credential stuffing
- Credentials and keys in URL
- Lack of access token validation (including JWT validation)
- Unsigned, weakly signed, non-expiring JWTs

A3: EXCESSIVE DATA EXPOSURE



API exposing a lot more data than the client legitimately needs, relying on the client to do the filtering. Attacker goes directly to the API and has it all.

USE CASES

- APIs return full data objects as they are stored by the database
- Client application shows only the data that user needs to see
- Attacker calls the API directly and gets sensitive data

A4: LACK OF RESOURCES & RATE LIMITING

API is not protected against an excessive amount of calls or payload sizes. Attackers use that for DoS and brute force attacks.

USE CASES

- Attacker overloading the API
- Excessive rate of requests
- Request or field sizes
- "Zip bombs"

A5: BROKEN FUNCTION LEVEL AUTHORIZATION

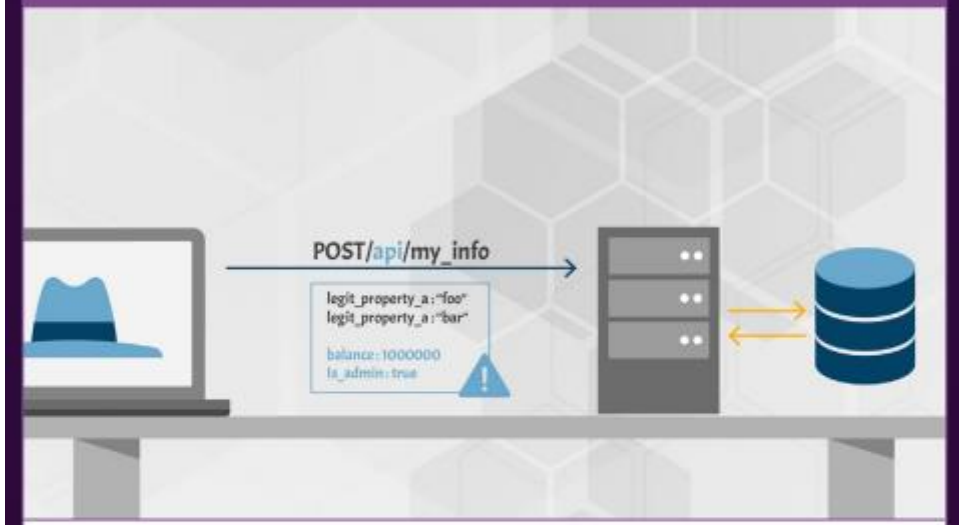
API relies on client to use user level or admin level APIs. Attacker figures out the "hidden" admin API methods and invokes them directly.

USE CASES

- Some administrative functions are exposed as APIs
- Non-privileged users can access these functions if they know how
- Can be a matter of knowing the URL, using a different verb or parameter

```
/api/users/v1/user/myinfo
/api/admins/v1/users/all
```

A6: MASS ASSIGNMENT



USE CASES

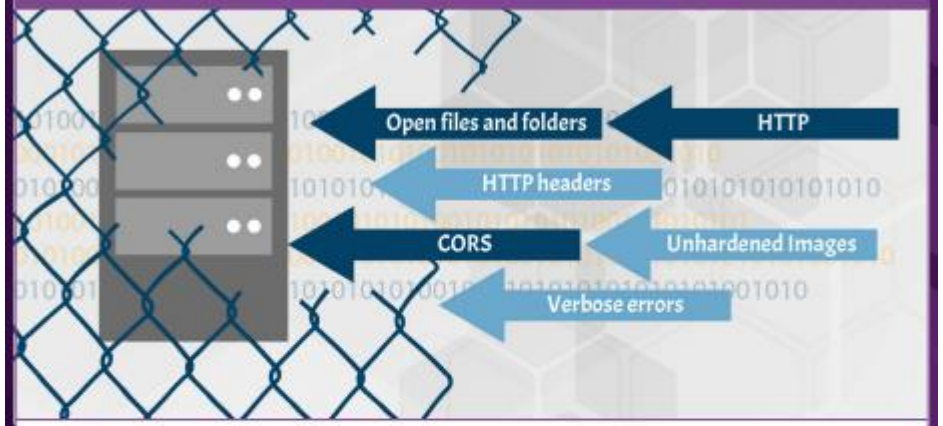
- API working with the data structures
- Received payload is blindly transformed into an object and stored

```
NodeJS:
var user = new User(req.body);
user.save();

Rails:
@user = User.new(params[:user])
```

- Attackers can guess the fields by looking at the GET request data

A7: SECURITY MISCONFIGURATION

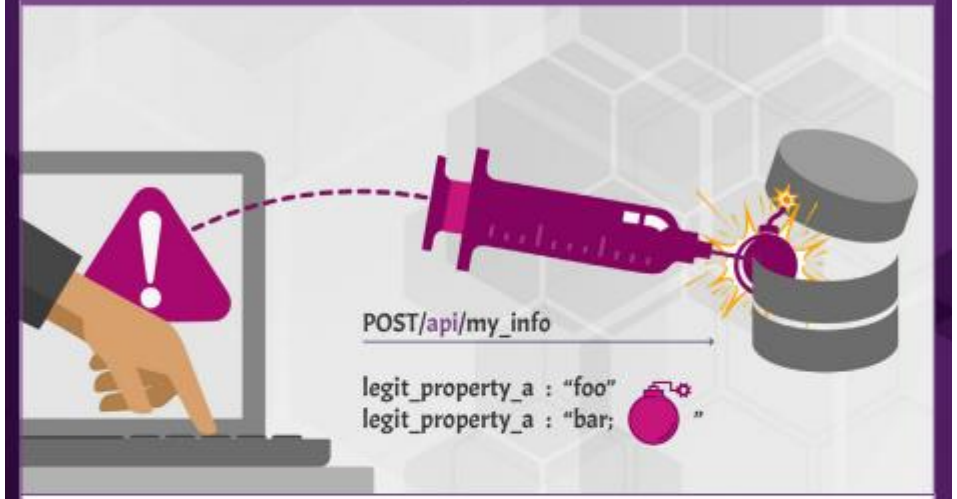


Poor configuration of the API servers allows attackers to exploit them.

USE CASES

- Unpatched systems
- Unprotected files and directories
- Unhardened images
- Missing, outdated, misconfigured TLS
- Exposed storage or server management panels
- Missing CORS policy or security headers
- Error messages with stack traces
- Unnecessary features enabled

A8: INJECTION



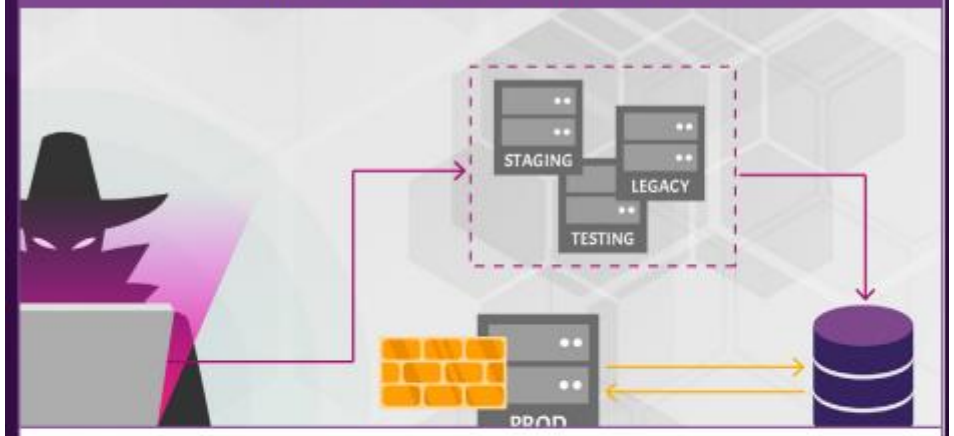
Attacker constructs API calls that include SQL-, NoSQL-, LDAP-, OS- and other commands that the API or backend behind it blindly executes.

USE CASES

Attackers send malicious input to be forwarded to an internal interpreter:

- SQL, NoSQL
- LDAP
- OS commands
- XML parsers
- Object-Relational Mapping (ORM)

A9: IMPROPER ASSETS MANAGEMENT



Attacker finds non-production versions of the API: such as staging, testing, beta or earlier versions - that are not as well protected, and uses those to launch the attack.

USE CASES

- DevOps, cloud, containers, K8S make having multiple deployments easy (Dev, Test, Branches, Staging, Old versions)
- Desire to maintain backward compatibility forces to leave old APIs running
- Old or non-production versions are not properly maintained
- These endpoints still have access to production data
- Once authenticated with one endpoint, attacker may switch to the other

A10: INSUFFICIENT LOGGING & MONITORING

Lack of proper logging, monitoring, and alerting let attacks go unnoticed.

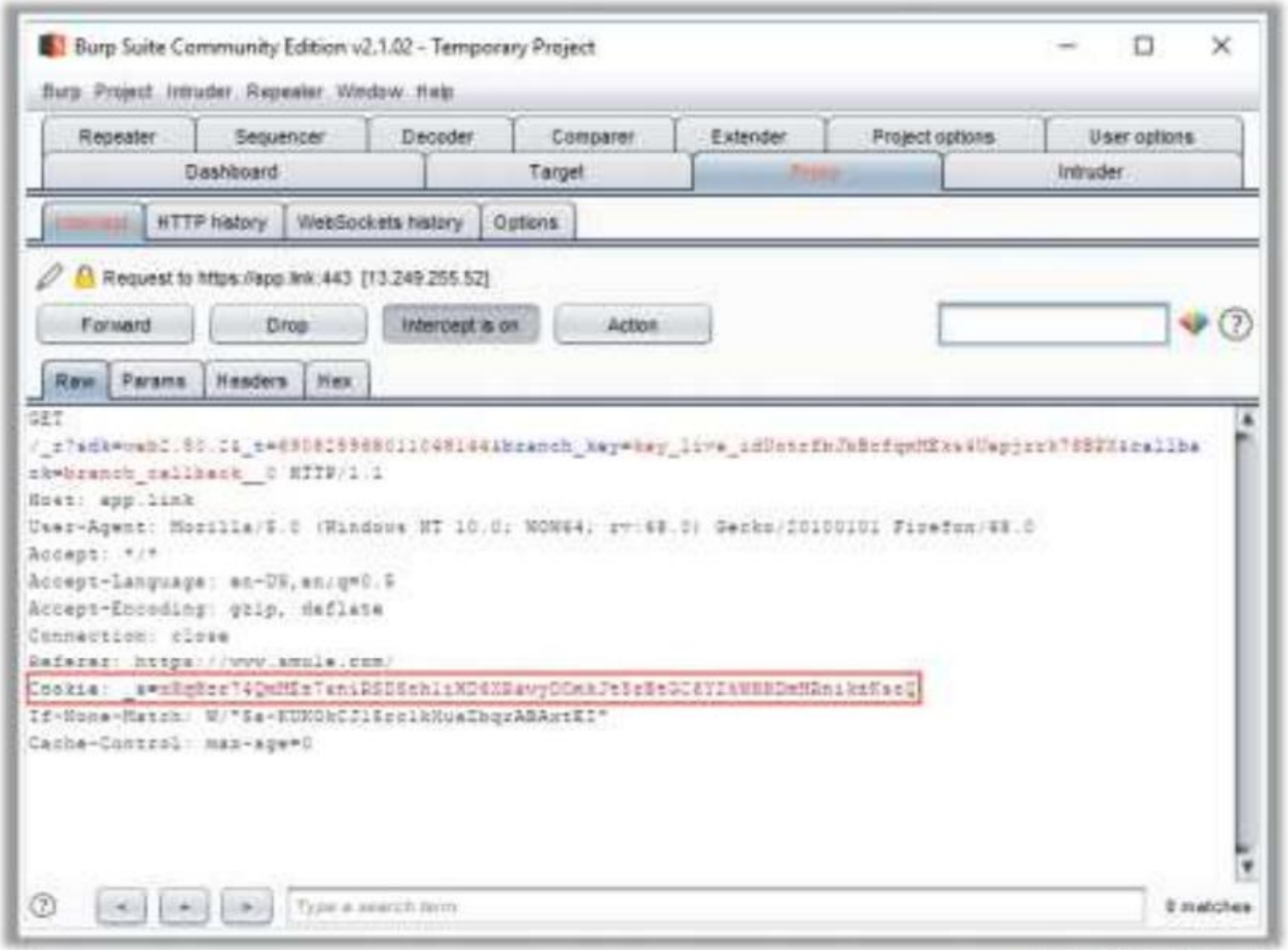
USE CASES

- Lack of logging, monitoring, alerting allow attackers to go unnoticed
- Logs are not protected for integrity
- Logs are not integrated into Security Information and Event Management (SIEM) systems
- Logs and alerts are poorly designed
- Companies rely on manual rather than automated systems

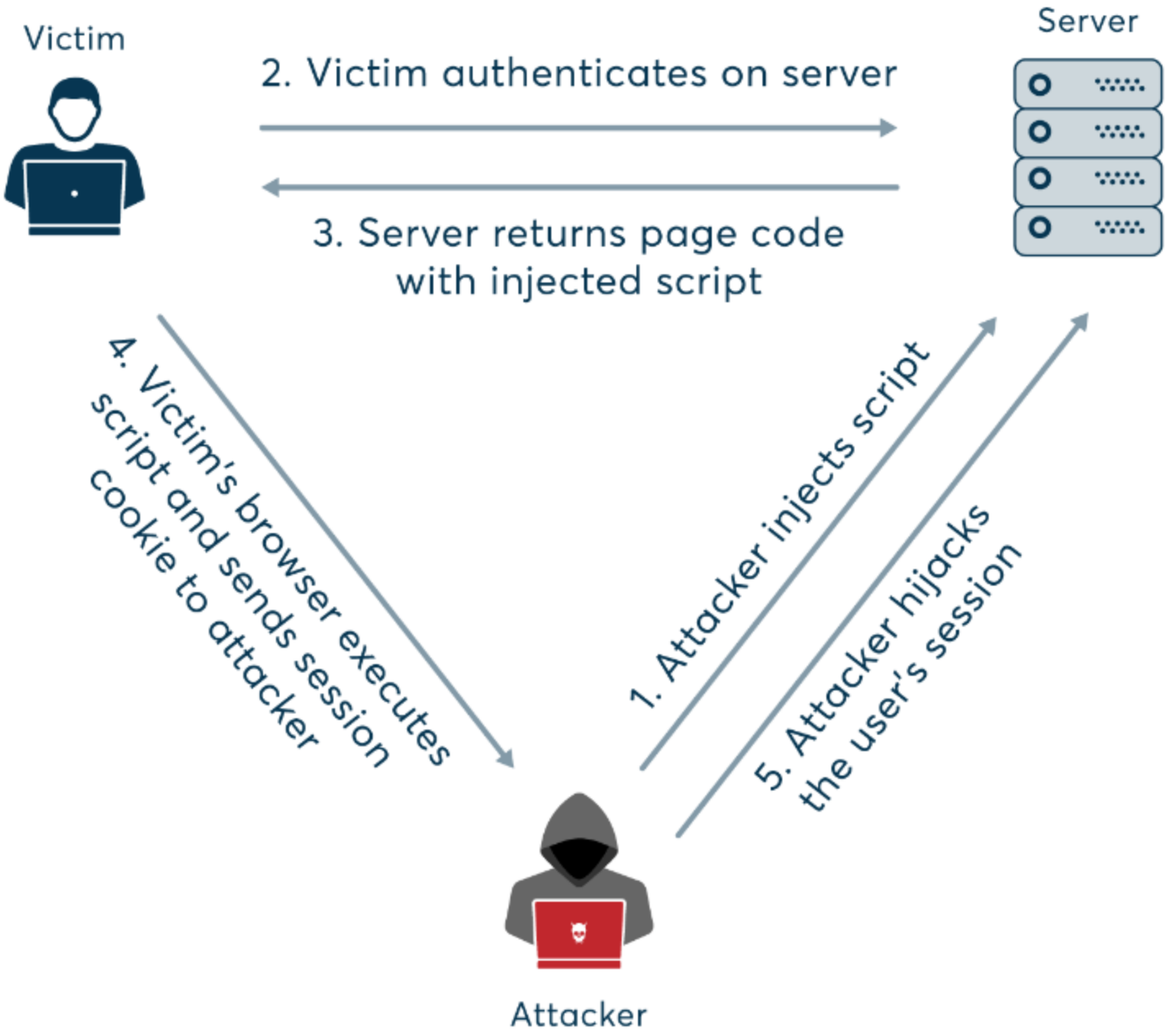
Session Hijacking

- Sniff valid session IDs to **gain unauthorized access** to the web server to snoop data
- Use session hijacking techniques such as session fixation, session sidejacking, Cross-site scripting, etc., to **capture valid session cookies and IDs**
- Use tools such as **Burp Suite**, **JHijack**, **Ettercap**, etc. to automate session hijacking

Note: For complete coverage of Session Hijacking concepts and techniques refer to Module 11: Session Hijacking



https://portswigger.net


Victim
Server
2. Victim authenticates on server
3. Server returns page code with injected script
4. Victim's browser executes script and sends session cookie to attacker
1. Attacker injects script
5. Attacker hijacks the user's session
Attacker

The session token could be compromised in different ways; the most common are:

Predictable session token

- The session ID information for a certain application is normally composed by a string of fixed width. **Randomness is very important** to avoid its prediction.
  - **Example:** Session ID value is "user01", which corresponds to the username. By trying new values for it, like "user02", it could be possible to get inside the application without prior authentication.

Session Sniffing

- Sniffing can be used to hijack a session when there is non-encrypted communication between the web server and the user, and the session ID is being sent in plain text.
  - **Wireshark** and **Kismet** can be used to capture sensitive data packets such as the session ID from the network.

Cross-site scripting (XSS)

- A server can be vulnerable to a cross-site scripting exploit, which enables an attacker to execute malicious code from the user's side, gathering session information. An attacker can target a victim's browser and send a scripted JavaScript link, which upon opening by the user, runs the malicious code in the browser hijacking sessions.

CSRF - Cross-Site Request Forgery

- Forces an end user to execute unwanted actions on a web application in which they're currently authenticated. With a little help of social engineering (such as sending a link via email or chat), an attacker may trick the users of a web application into executing actions of the attacker's choosing;

- CSRF attack can force the user to perform state changing requests like transferring funds, changing their email address, and so forth. If the victim is an administrative account, CSRF can compromise the entire web application.

- **CSRF Scenario:**

  i. Visit your bank's site, log in.
  ii. Then visit the attacker's site (e.g. sponsored ad from an untrusted organization).
  iii. Attacker's page includes form with same fields as the bank's "Transfer Funds" form.
  iv. Form fields are pre-filled to transfer money from your account to attacker's account.
  v. Attacker's page includes Javascript that submits form to your bank.
  vi. When form gets submitted, browser includes your cookies for the bank site, including the session token.
  vii. Bank transfers money to attacker's account.
  viii. The form can be in an iframe that is invisible, so you never know the attack occurred.

Session Fixation

- Session Fixation is an attack that permits an attacker to hijack a valid user session. The attack explores a limitation in the way the web application manages the session ID, more specifically the vulnerable web application.

- **Session fixation Scenario:**

  i. The attacker accesses the web application login page and **receives a session ID** generated by the web application.
  ii. The attacker uses an additional technique such as **CRLF Injection, man-in-the-middle attack, social engineering,** etc., and gets the victim to use the **provided session identifier**.
  iii. The victim accesses the web application login page and logs in to the application. After authenticating, the **web application treats anyone who uses this session ID as if they were this user.**
  iv. The attacker uses the session ID to access the web application, **take over the user session, and impersonate the victim**.

Man-in-the-browser attack

- The Man-in-the-Browser attack is the same approach as Man-in-the-middle attack, but in this case a Trojan Horse is used to intercept and manipulate calls between the main application's executable.

Man-in-the-middle attack

- MITM attack is a general term for when a perpetrator positions himself in a conversation between a user and an application—either to eavesdrop or to impersonate one of the parties, making it appear as if a normal exchange of information is underway.

Other attacks

- **Compression Ratio Info-leak Made Easy (CRIME)**:
  - Is a security exploit against secret web cookies over connections using the HTTPS and SPDY protocols that also use data compression. When used to recover the content of secret authentication cookies, it allows an attacker to perform session hijacking.
- **BREACH**:
  - Is a security exploit against HTTPS when using HTTP compression (SSL/TLS compression). BREACH is built based on the CRIME security exploit.

TCP Hijacking: TCP/IP Hijacking is when an authorized user gains access to a genuine network connection of another user. It is done in order to bypass the password authentication which is normally the start of a session.

- e.g: TELNET Hijacking using Ettercap, Shijack, making a blind hijacking.

github.com/Samsar4/CEH-v10-Study-Guide/blob/master/modules/9-Session-Hijacking.md

VB.NET Menu Contr... | Cracking the DBMS... | [100% working Cod... | How to Code a Neu... | Computer Program... | ER Some Basic P

Files

master

Go to file

- 12-Hacking-Wireless-Networks....
- 13-Hacking-Mobile-Platforms-a...
- 14-Pentesting.md
- 15-Cloud Computing.md
- 16-Cryptography.md
- 2-Scanning-and-Enumeration.md
- 3-System-Hacking.md
- 4-Malware.md
- 5-Sniffing.md
- 6-Social-Engineering.md
- 7-Evading-IDS-Firewalls-and-H...
- 8-Denial-of-Service.md
- 9-Session-Hijacking.md

CEH-v10-Study-Guide / modules / 9-Session-Hijacking.md

Samsar4 M:9 - Content updated

Preview | Code | Blame | 113 lines (90 loc) · 7.83 KB

Session Hijacking

> This chapter has practical labs

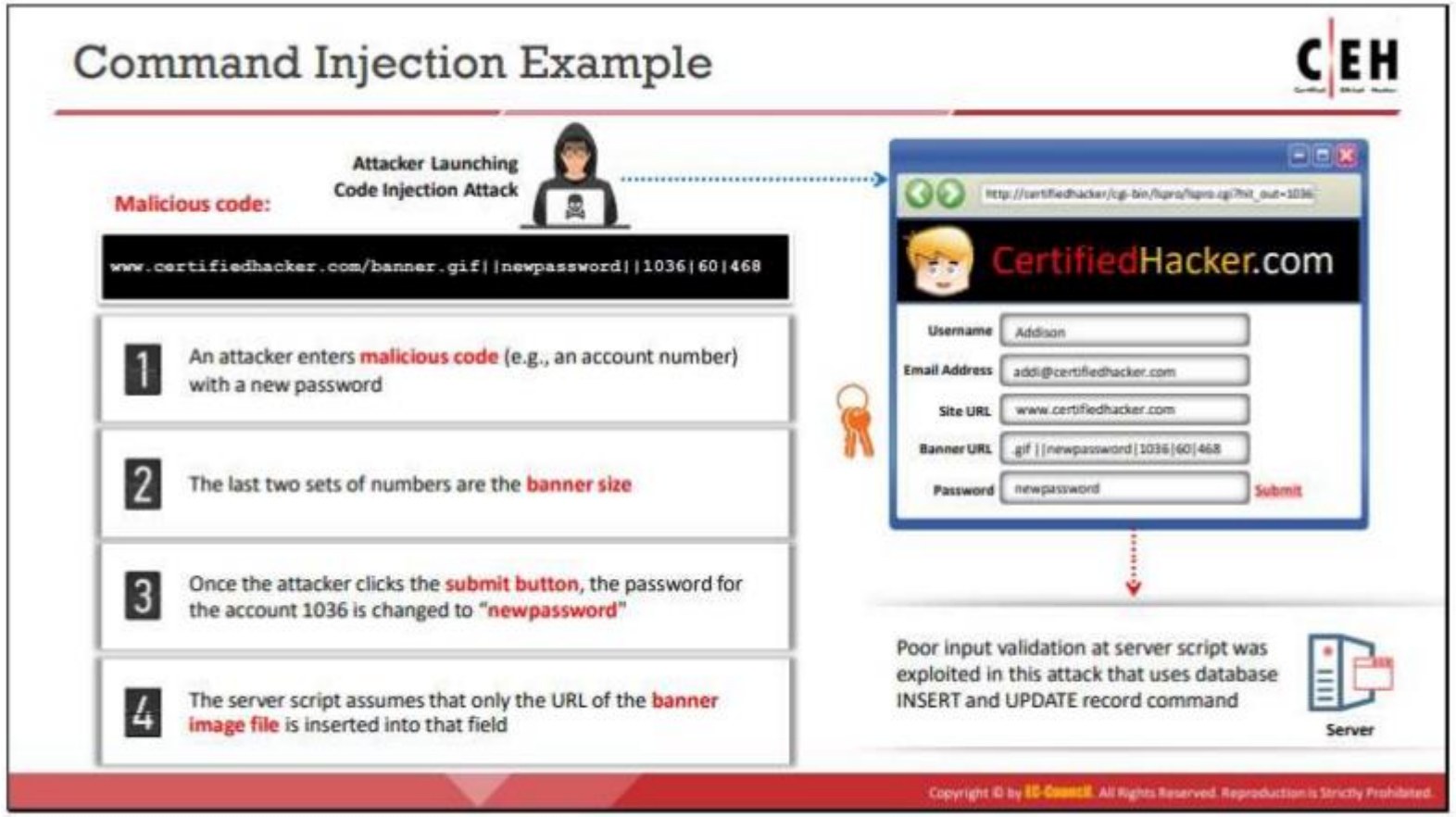
The Session Hijacking attack consists of the exploitation of the w

- HTTP communication uses many different TCP connections
- The most useful method depends on a **token** that the Web
- A **session token** is normally composed of a string of variab
  - like in the URL, in the header of the HTTP requisition a of the HTTP requisition.

The Session Hijacking attack compromises the session token the Web Server.

Topic 5

Code injection



Command Injection Example

An attacker enters the following malicious code (account number) with a new password.

```
www.certifiedhacker.com/banner.gif||newpassword||1036|60|468
```

The last two sets of numbers denote the banner size. Once the attacker clicks the submit button, the password for the account 1036 is changed to "newpassword." The server script assumes that only the URL of the banner image file is inserted into that field.

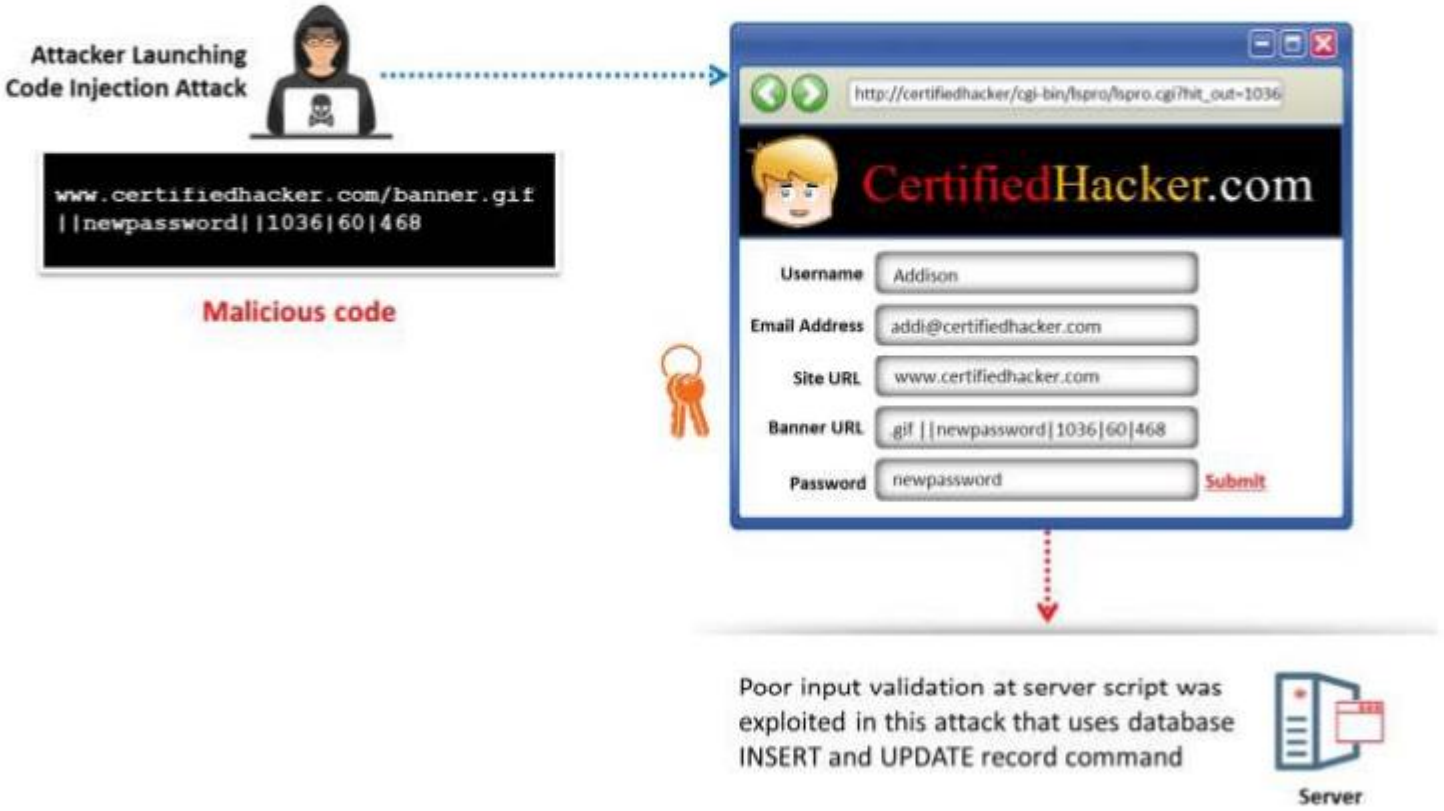
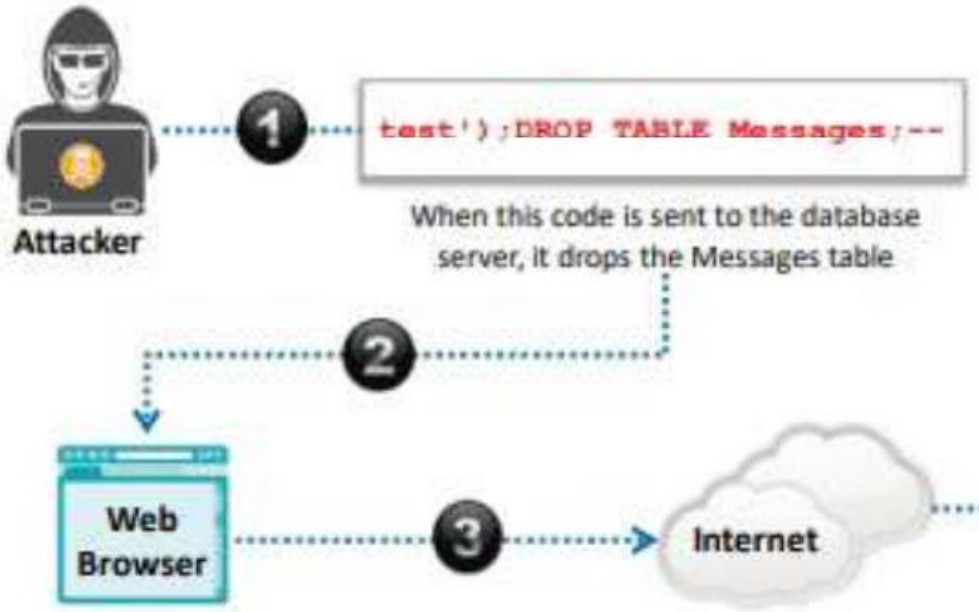


Figure 14.6: Command Injection attack example

SQL Injection Attacks

Topic 6

- SQL injection attacks use a **series of malicious SQL queries** to directly manipulate the database
- An attacker can use a vulnerable web application to **bypass normal security measures** and obtain direct access to valuable data
- SQL injection attacks can often be executed from the **address bar**, from within application fields, and through queries and searches




```
01  <?php
02  function save_email($user, $message)
03  {
04    $sql = "INSERT INTO Messages (
05              user, message
06          ) VALUES (
07          '$user', '$message'
08          )";
09    return mysql_query($sql);
10  }
11  ?>
```

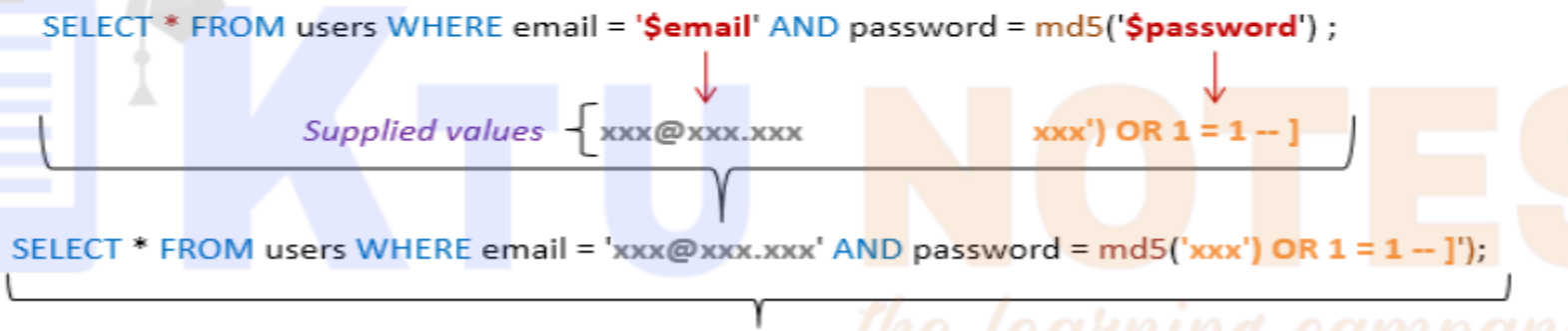
SQL Injection vulnerable server code

Note: For complete coverage of SQL Injection concepts and techniques, refer to Module 15: SQL Injection

Injected SQL code:

SELECT * FROM users WHERE email = 'xxx@xxx.xxx' AND password = md5('xxx') OR 1 = 1 --]');

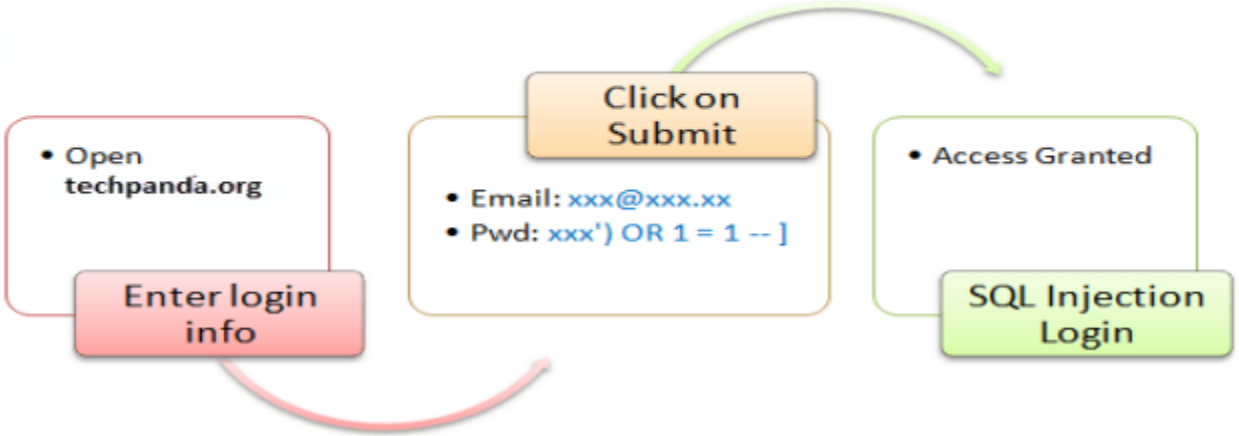
- The diagram below illustrates the statement has been generated.



SELECT * FROM users WHERE **FALSE AND FALSE OR TRUE**

SELECT * FROM users WHERE **FALSE OR TRUE**

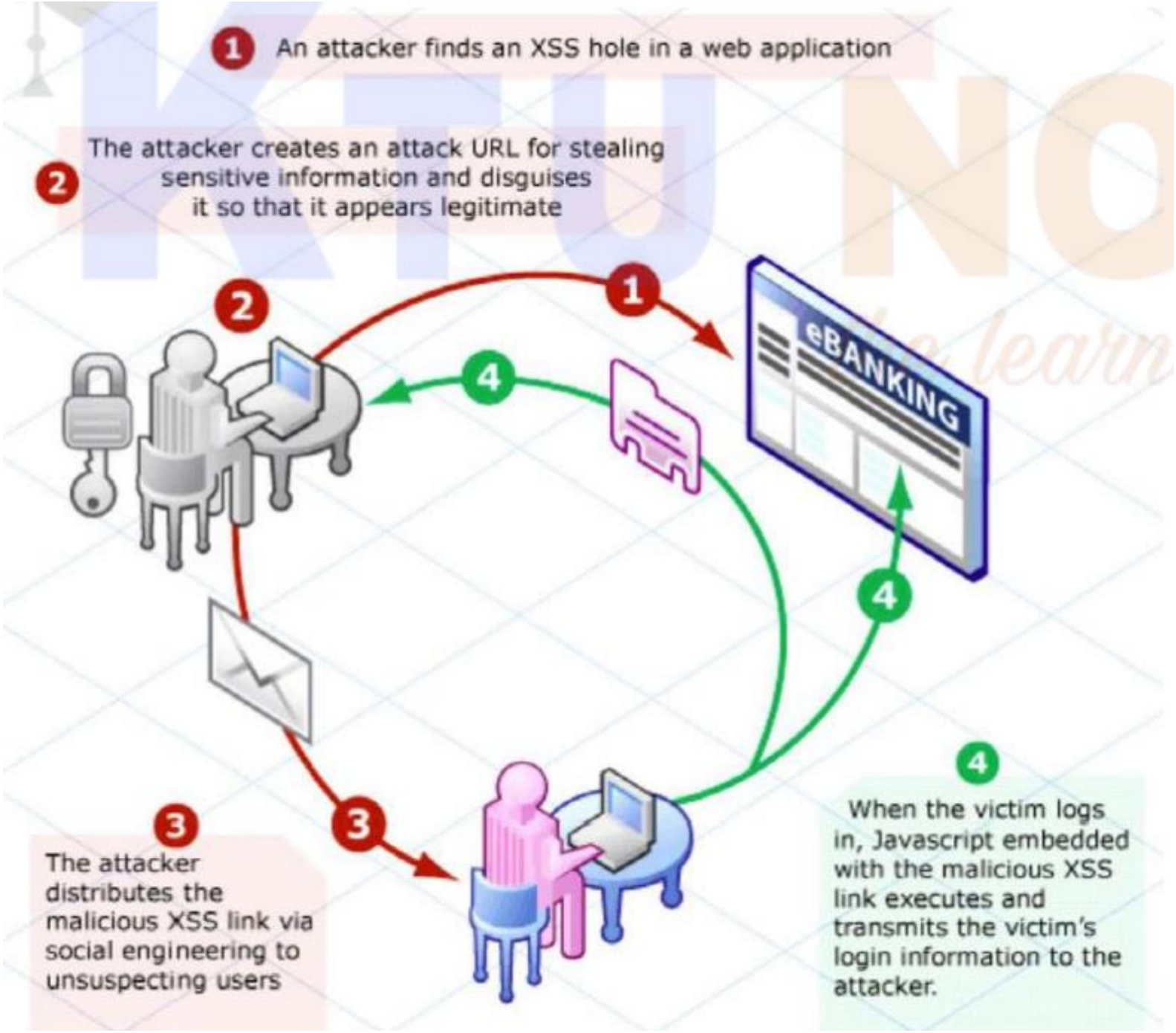
SELECT * FROM users WHERE **TRUE**



Topic 7

Cross Site Scripting (XSS)

- Cross-Site Scripting (referred to as XSS) is a type of web application attack where malicious client-side script is injected into the application output and subsequently executed by the user's browser
- XSS is a vulnerability which when present in websites or web applications, allows malicious users (Hackers) to insert their client-side code (normally JavaScript) in those web pages. When this malicious code along with the original webpage gets displayed in the web client (browsers like IE, Mozilla etc), allows Hackers to gain greater access of that page.
- It can be used to take over a user's browser in a variety of ways
- Cross Site Scripting (XSS) is a type of computer security exploit where information from one context, where it is not trusted, can be inserted into another context, where it is
- The trusted website is used to store, transport, or deliver malicious content to the victim
- The target is to trick the client browser to execute malicious scripting commands
- JavaScript, VBScript, ActiveX, HTML, or Flash
- Caused by insufficient input validation.


1 An attacker finds an XSS hole in a web application
2 The attacker creates an attack URL for stealing sensitive information and disguises it so that it appears legitimate
eBANKING
3 The attacker distributes the malicious XSS link via social engineering to unsuspecting users
4 When the victim logs in, Javascript embedded with the malicious XSS link executes and transmits the victim's login information to the attacker.

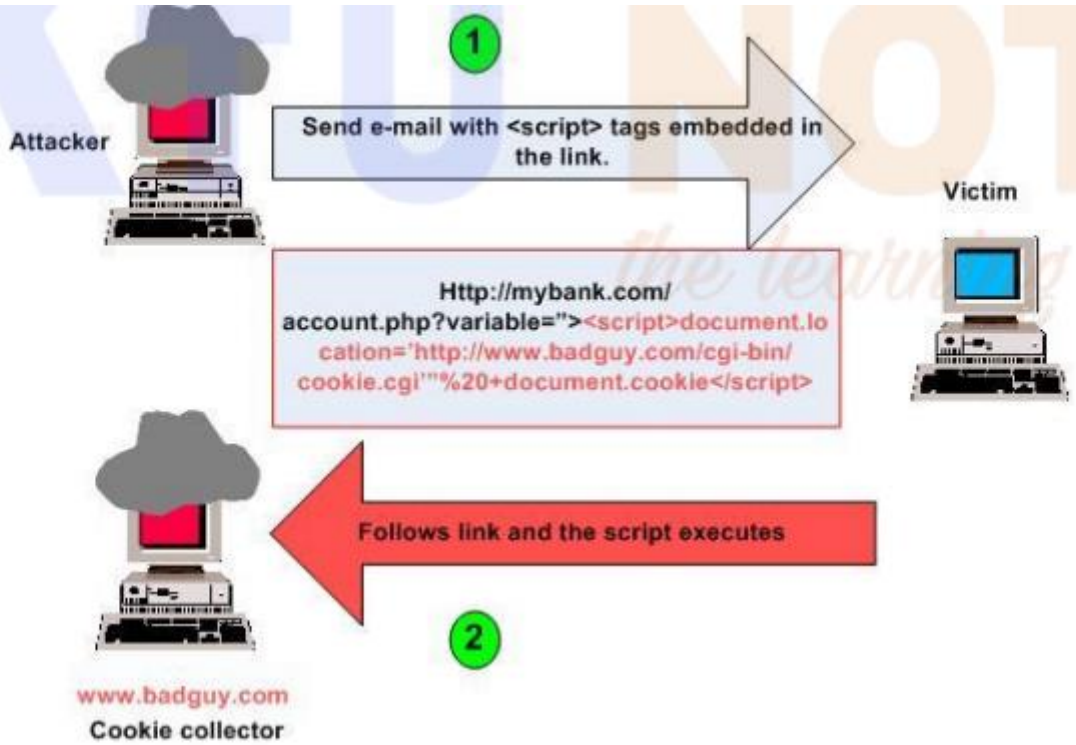
Cross Site Scripting Types

Three known types:

- **Reflected (Non-Persistent)**
  - ✓ Link in other website or email
- **Stored (Persistent)**
  - ✓ Forum, bulletin board, feedback form
- **DOM Based XSS(Local)**
  - ✓ PDF Adobe Reader, FLASH player

1) Reflected (Non-Persistent)

- Reflected cross-site scripting vulnerabilities arise **when data is copied from a request** and **echoed** into the application's immediate response in an unsafe way.
- An attacker can use the vulnerability to construct a request which, if issued by another application user, will cause **JavaScript code supplied by the attacker to execute within the user's browser in the context of that user's session with the application.**
- The attacker-supplied code can perform a wide variety of actions, such as **stealing the victim's session token or login credentials, performing arbitrary actions on the victim's behalf, and logging their keystrokes.**



- **Malicious content does not get stored in the server**
- **The server bounces the original input to the victim without modification**
- Exploits the fact that some servers echo back certain user input back to the client without validating it
- For example, a user may be asked for personal details in an HTML form. Suppose he enters his name as "Prashant". The server then responds with "Hello Prashant"

- Note that the server has echoed back his name
- Now, what would happen if, instead of Prashant, the user enters

<SCRIPT>alert('Fire!')</SCRIPT>

Reflected XSS Example

- Exploit URL:

http://www.nikebiz.com/search/?q=<script>alert('XSS') </script>&x=0&y=0

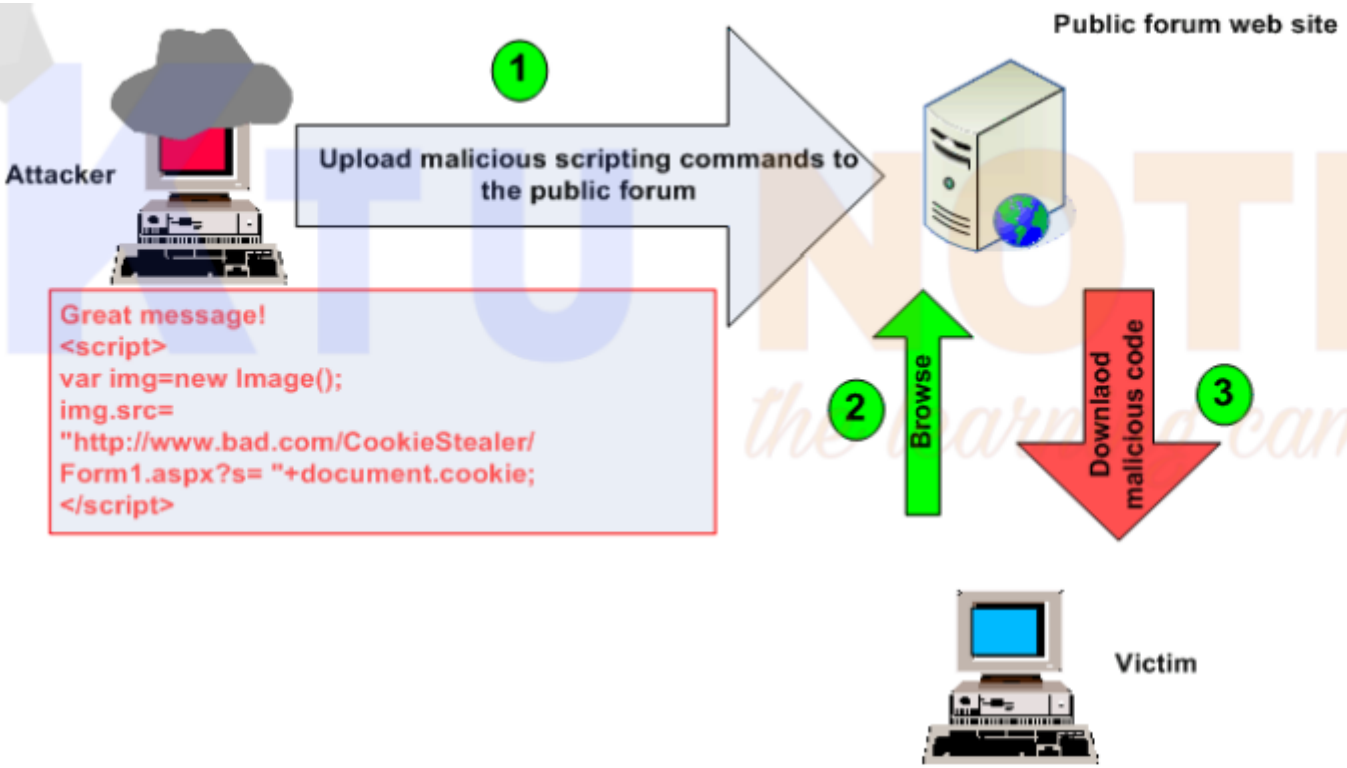
- HTML returned to victim:

<h2>Search Results
 Search: "<script>alert('XSS')

</script>"</h2>

2) Stored XSS

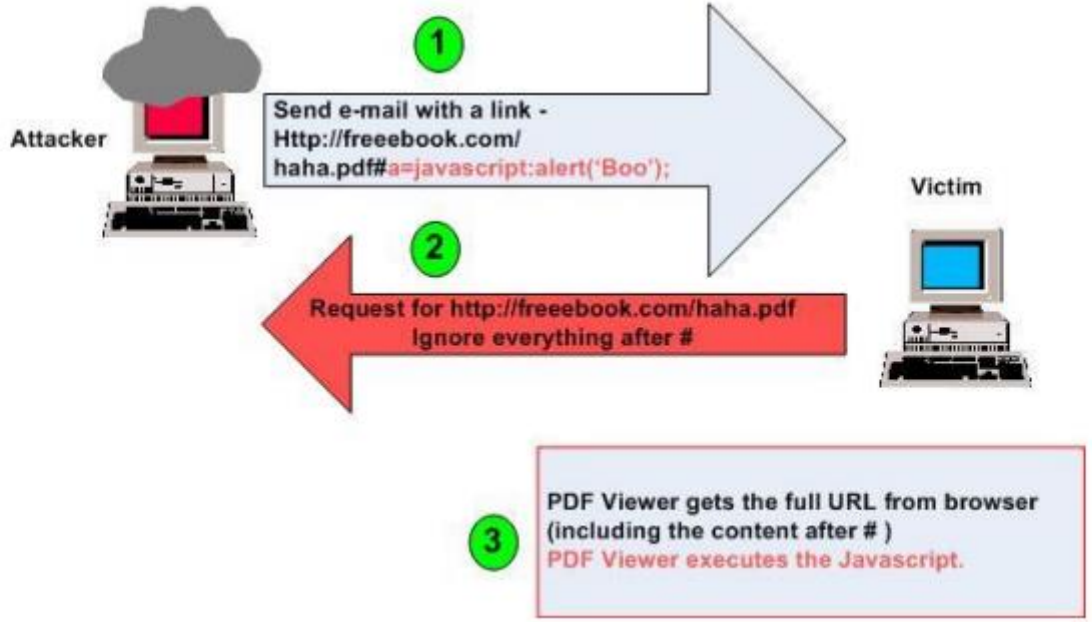
- JavaScript supplied by the attacker is stored by the website (e.g. in a database)
- Doesn't require the victim to supply the JavaScript somehow, just visit the exploited web page
- More dangerous than Reflected XSS
  - Has resulted in many XSS worms on high profile sites like MySpace and Twitter



- The server stores the malicious content
- The server serves the malicious content in its original form
- The malicious code (scripts) on a web page is saved on the web server.
- When an innocent user downloads the web page, the malicious scripts execute on that user's browser.
- Example: Users update their profile on a social networking site. These profiles may be read (downloaded) by other users through their browsers

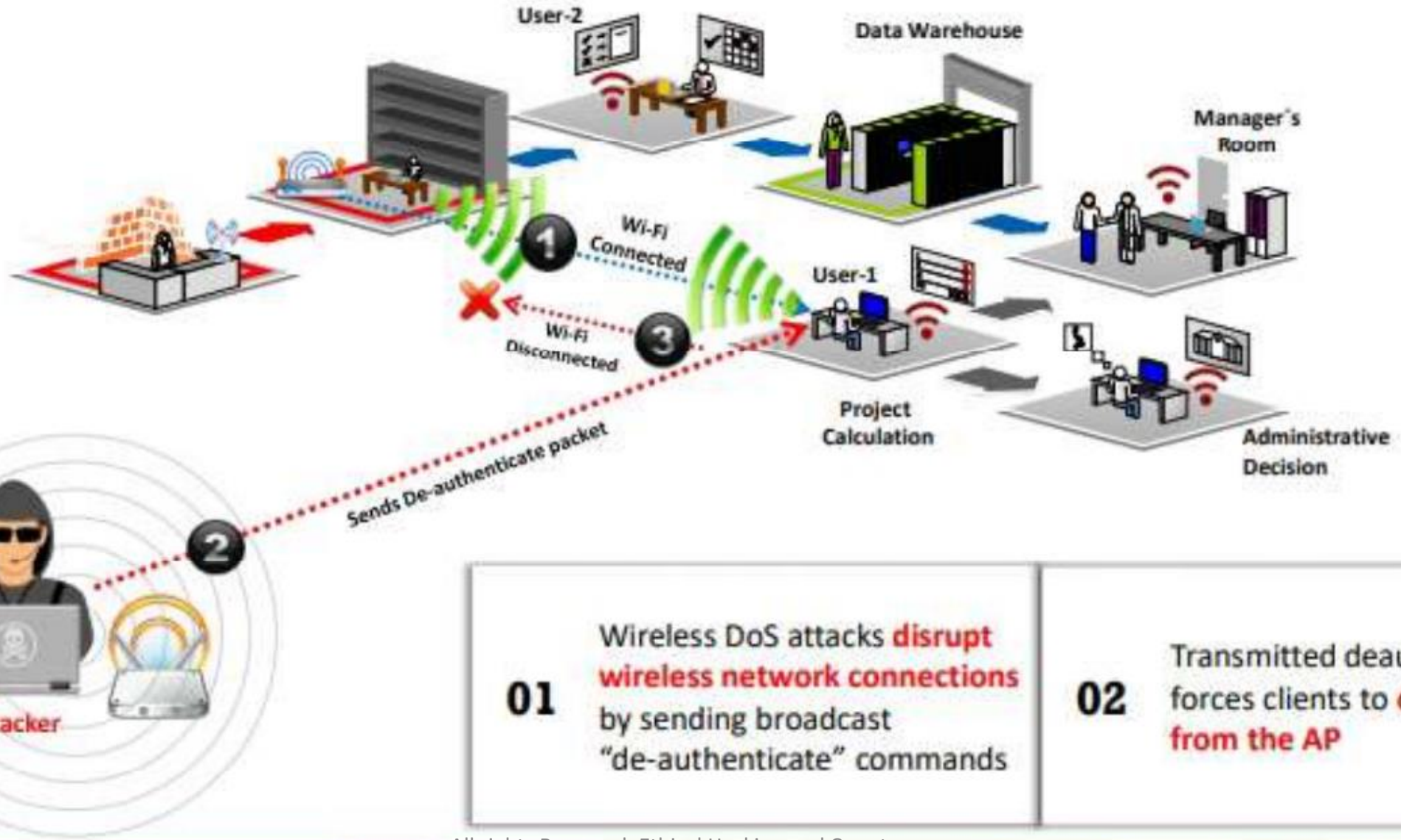
3) DOM Based XSS (Local)

- DOM Based XSS (or as it is called in some texts, "type-0 XSS") is an XSS attack wherein the attack payload is executed as a result of modifying the DOM "environment" in the victim's browser used by the original client-side script, so that the client-side code runs in an "unexpected" manner.
- Occur in the content processing stages performed by the client



Denial-of-Service Attack

Topic 8



01 Wireless DoS attacks **disrupt wireless network connections** by sending broadcast "de-authenticate" commands

02 Transmitted deauthentication forces clients to **disconnect from the AP**

DoS

A Denial of Service (DoS) is a type of attack on a service that disrupts its normal function and prevents other users from accessing it. The most common target for a DoS attack is an online service such as a website, though attacks can also be launched against networks, machines or even a single program.

DoS attacks can cause the following problems:

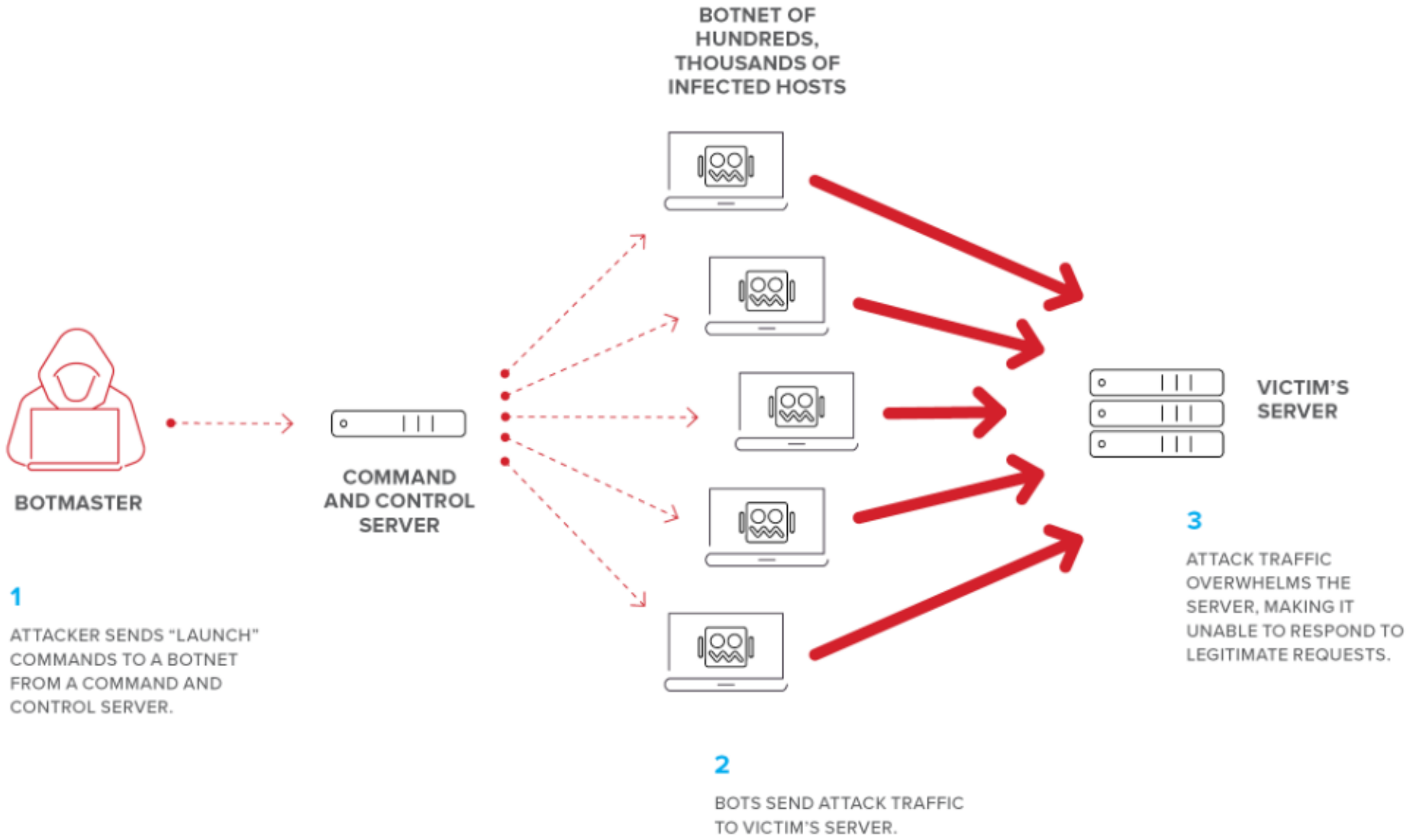
- Ineffective services
- Inaccessible services
- Interruption of network traffic
- Connection interference

DDoS

A distributed denial of service (DDoS) attack is launched from numerous compromised devices, often distributed globally in what is referred to as a ***botnet****.*

Network of zombie computers a hacker uses to start a distributed attack.

- Botnets can be designed to do malicious tasks including sending **spam, stealing data, ransomware, fraudulently clicking on ads or distributed denial-of-service (DDoS) attacks.**
- Can be controlled over HTTP, HTTPS, IRC, or ICQ



- **Botnet Scanning Methods:**
  - **Random** - Randomly looks for vulnerable devices
  - **Hitlist** - Given a list of devices to scan for vulnerabilities
  - **Topological** - Scan hosts discovered by currently exploited devices
  - **Local subnet** - Scans local network for vulnerable devices
  - **Permutation** - Scan list of devices created through pseudorandom permutation algorithm

Three Types of DoS / DDoS

1. Volumetric attacks

- Consumes the bandwidth of target network or service.
- Send a massive amount of traffic to the target network with the goal of consuming **so much bandwidth** that users are denied access.
- Bandwitdh depletion attack: Flood Attack and Amplification attack.
  - **Attacks:**
    - UDP flood attack
    - ICMP flood attack
    - Ping of Death attack
    - Smurf attack (IP)
    - Fraggle (UDP)
    - Malformed IP packet flood attack
    - Spoofed IP packet flood attack
  - ⚠ **Volumetric attacks is measured in Bits per second (Bps).**

2. Protocol Attacks

- Consume other types of resources like **connection state tables** present in the network infrastructure components such as **load balancers, firewalls, and application servers.**
    - **Attacks:**
        - SYN flood attack
        - Fragmentation attack
        - ACK flood attack
        - TCP state exhaustion attack
        - TCP connection flood attack
        - RST attack
    - > ⚠️ **Protocol attacks is measured in Packets per second (Pps).**

3. Application Layer Attacks

- Includes low-and-slow attacks, GET/POST floods, attacks that target Apache, Windows or OpenBSD vulnerabilities and more.
- Consume the resources necessary for the application to run.
- Target web servers, web application and specific web-based apps.
- Abuse higher-layer (7) protocols like HTTP/HTTPS and SNMP.
    - **Attacks:**
        - HTTP GET/POST attack
        - Slowloris attack
    - > ⚠️ **Application layer attacks is measured in Requests per second (Rps).**
    - > ⚠️ **Application level attacks are against weak code.**

Attacks explanation

IP Fragmentation attacks

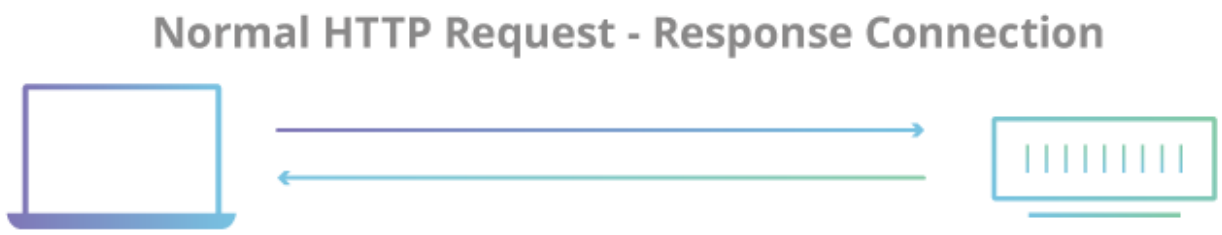
- IP / ICMP fragmentation attack is a common form of volumetric DoS. In such an attack, datagram fragmentation mechanisms are used to overwhelm the network.
- Bombard the destination with fragmented packets, causing it to use memory to reassemble all those fragments and overwhelm a targeted network.
- **Can manifest in different ways:**
  - **UDP Flooding** - attacker sends large volumes of fragments from numerous sources.
  - **UDP and ICMP** fragmentation attack - only parts of the packets is sent to the target; Since the packets are fake and can't be reassembled, the server's resources are quickly consumed.
  - **TCP fragmentation attack** - also know as a Teardrop attack, targets TCP/IP reassembly mechanisms; Fragmented packets are prevented from being reassembled. The result is that data packets overlap and the targeted server becomes completely overwhelmed.

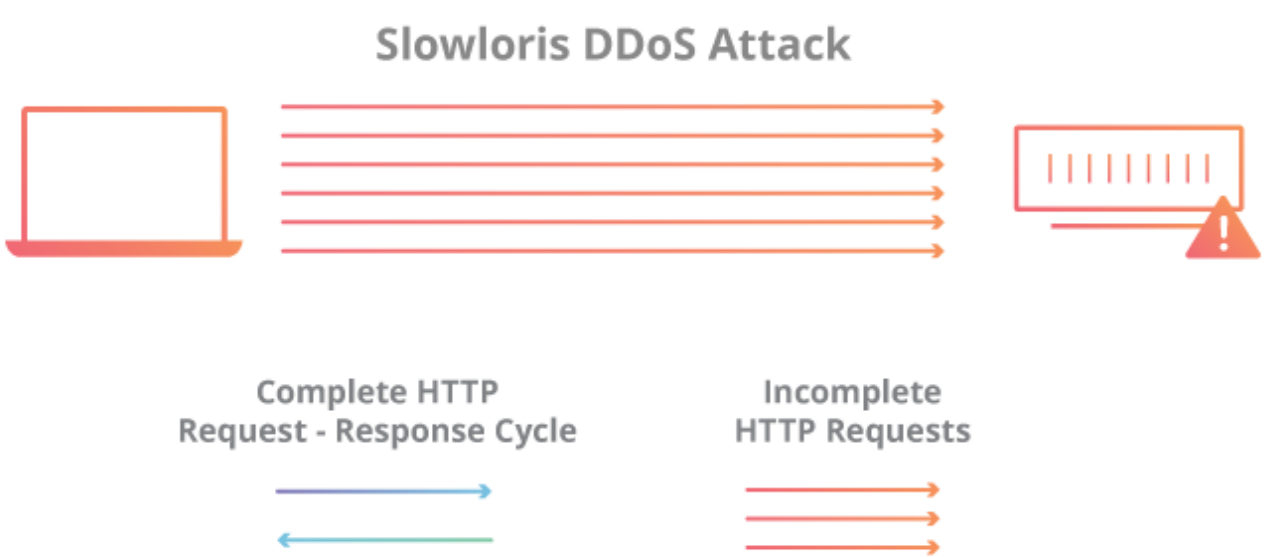
TCP state-exhaustion attack

- Attempt to consume connection state tables like: **Load balancers, firewalls and application servers.**

Slowloris attack

Is an application layer attack which operates by utilizing partial HTTP requests. The attack functions by opening connections to a targeted Web server and then keeping those connections open as long as it can.



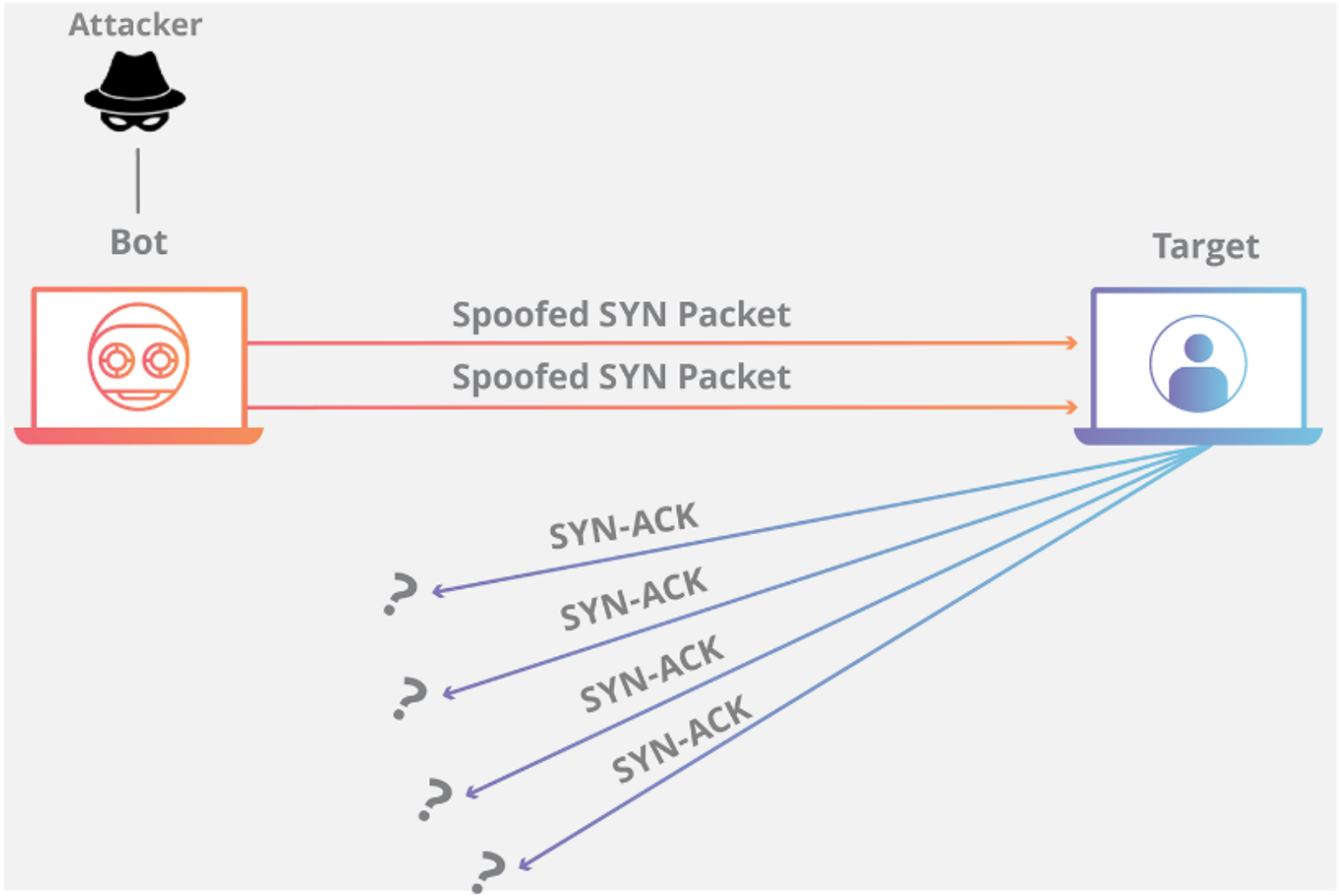


- The attacker first opens multiple connections to the targeted server by sending multiple partial HTTP request headers.
- The target opens a thread for each incoming request
- To prevent the target from timing out the connections, the attacker periodically sends partial request headers to the target in order to keep the request alive. In essence saying, "I'm still here! I'm just slow, please wait for me."
- The targeted server is never able to release any of the open partial connections while waiting for the termination of the request.
- Once all available threads are in use, the server will be unable to respond to additional requests made from regular traffic, resulting in denial-of-service.

SYN attack

- Sends thousands of SYN packets
- Uses a **false source address** / spoofed IP address.
- The server then responds to each one of the connection requests and leaves an open port ready to receive the response.
- Eventually engages all resources and exhausts the machine

SYN flood (half-open attack)

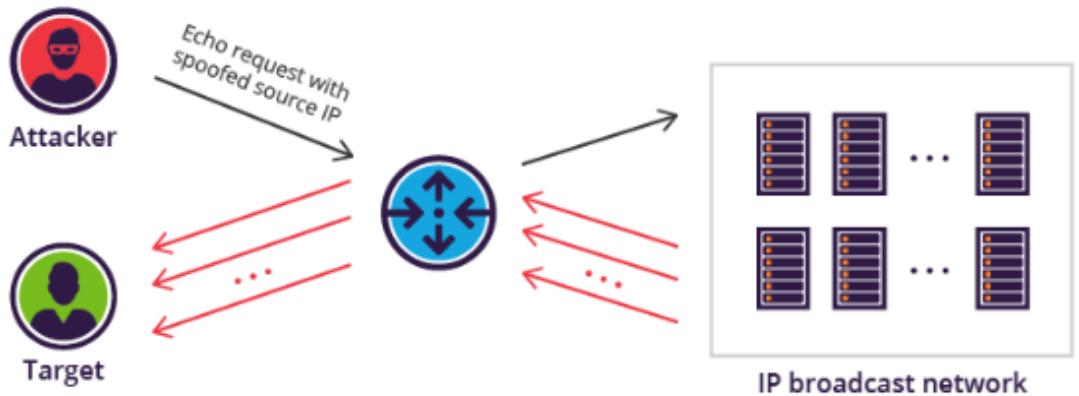
- Sends thousands of SYN packets
- While the **server waits for the final ACK packet**, **which never arrives**, the attacker continues to send more SYN packets. The arrival of each new SYN packet causes the server to temporarily maintain a new open port connection for a certain length of time, and once all the available ports have been utilized the server is unable to function normally.
- Eventually bogs down the computer, runs out of resources.
- 

ICMP flood

- Sends ICMP Echo packets with a spoofed address; eventually reaches limit of packets per second sent
  - Is possible to use `hping3` to perform ICMP flood:
    - `hping -1 --flood --rand-source <target>`

Smurf attack

- The Smurf attack is a **distributed denial-of-service** attack in which large numbers of ICMP packets with the intended victim's **spoofed source IP are broadcast to a computer network using an IP broadcast address.**
  - Is possible to use `hping3` to perform this attack and bash script to loop through the subnet.
    - `hping3 -1 -c 1000 10.0.0.$i --fast -a <spoofed target>`



  -

Fraggle

- Same concept as Smurf attack but with **UDP packets** (UDP flood attack).
  - Is possible to use `hping3` to perform Fraggle attack/ UDP flood
    - `hping3 --flood --rand-source --udp -p <target>`

Ping of Death

- Fragments ICMP messages; after reassembled, the ICMP packet is larger than the maximum size and crashes the system
  - Performs by sending an IP packet larger than the 65,536 bytes allowed by the IP protocol.
  - Old technique that can be acceptable to old systems.

Teardrop

- Overlaps a large number of garbled IP fragments with oversized payloads; causes older systems to crash due to fragment reassembly

Peer to peer

- Clients of peer-to-peer file-sharing hub are disconnected and directed to connect to the target system

Multi-vector attack

- Is a combination of **Volumetric, protocol, and application-layer attacks.**

Phlashing / Permanent DoS

- A DoS attack that causes permanent damage to a system.
- Modifies the firmware and can also cause a **system to brick.**
- *e.g: Send fraudulent hardware update to victim; crashing BIOS.*

LAND attack

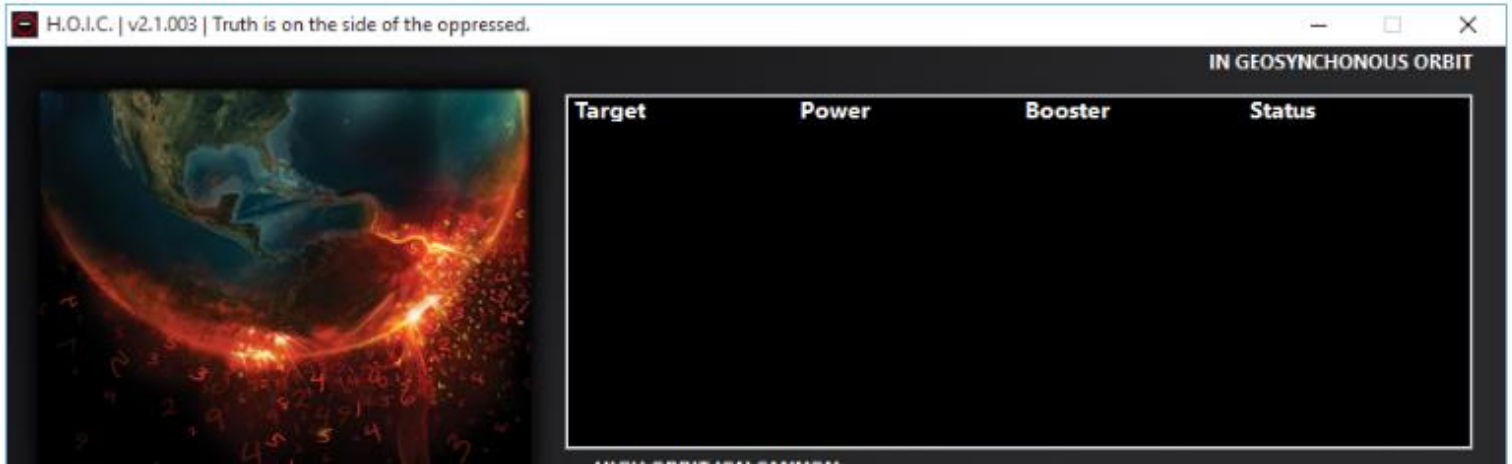
- Sends a SYN packet to the target with a spoofed IP the same as the target; if vulnerable, target loops endlessly and crashes

DoS/DDoS Attack Tools:

- **Low Orbit Ion Cannon** (LOIC) - DDoS tool that floods a target with TCP, UDP or HTTP requests

- 

High Orbit Ion Cannon (HOIC) - More powerful version of LOIC; Targets TCP and UDP; The application can open up to 256 simultaneous attack sessions at once, bringing down a target system by sending a continuous stream of junk traffic until legitimate requests are no longer able to be processed;



github.com/Samsar4/CEH-v10-Study-Guide/blob/master/modules/8-Denial-of-Service.md

VB.NET Menu Contr... | Cracking the DBMS... | [100% working Cod... | How to Code a Neu... | Computer Program... | Some Basic Progra... | :: CUK University :: | Eduso

Product | Solutions | Open Source | Pricing | Search

Samsar4 / CEH-v10-Study-Guide Public

Code | Issues 2 | Pull requests | Actions | Projects | Security | Insights

Files

master

Go to file

- 12-Hacking-Wireless-Networks....
- 13-Hacking-Mobile-Platforms-a...
- 14-Pentesting.md
- 15-Cloud Computing.md
- 16-Cryptography.md
- 2-Scanning-and-Enumeration.md
- 3-System-Hacking.md
- 4-Malware.md
- 5-Sniffing.md
- 6-Social-Engineering.md
- 7-Evading-IDS-Firewalls-and-H...
- 8-Denial-of-Service.md

CEH-v10-Study-Guide / modules / 8-Denial-of-Service.md

Samsar4 Readme, M:8 - Content updated

Preview | Code | Blame 185 lines (138 loc) · 9.39 KB

Denial of Service

> This chapter has practical labs

DoS

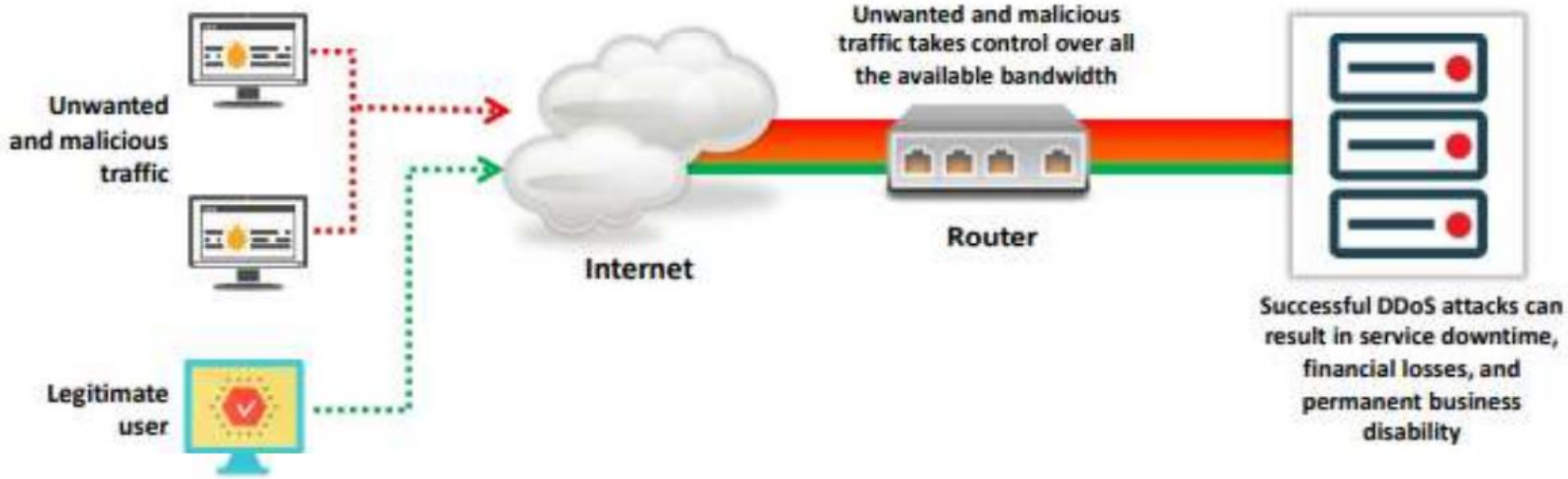
A Denial of Service (DoS) is a type of attack on a service that disrupts its normal function and preve common target for a DoS attack is an online service such as a website, though attacks can also be l single program.

DoS attacks can cause the following problems:

- Ineffective services

DoS/DDoS Attacks

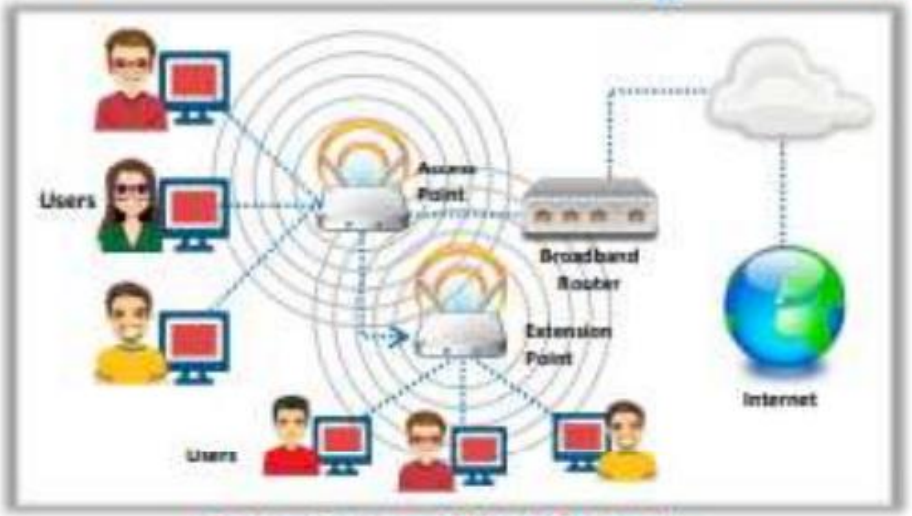
- Attackers may send numerous **fake requests** to the web server, which causes **web server crashing** or makes it unavailable to the legitimate users
- Attackers may target **high profile web servers** such as banks, credit card payment gateways, and government owned services to **steal user credentials**



Wireless Networks

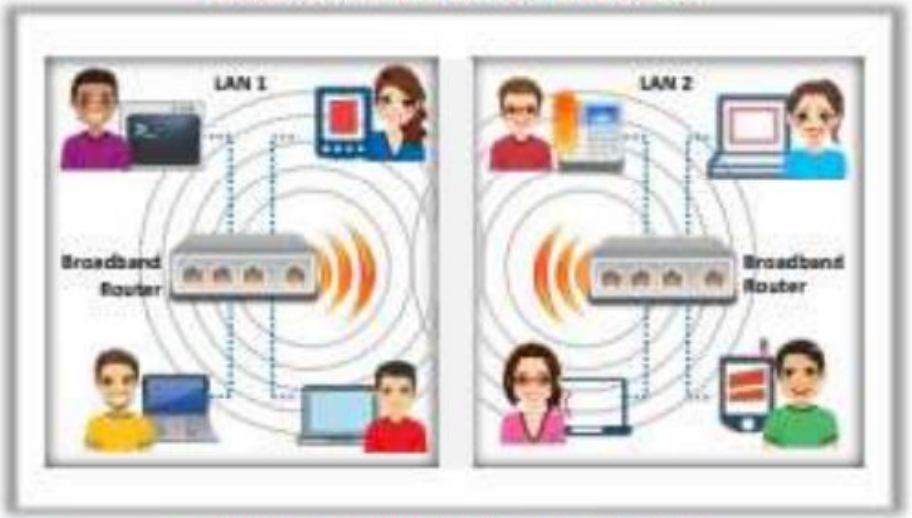
- Wireless network (Wi-Fi) refers to WLANs based on **IEEE 802.11 standard**, which allows the device to access the network from anywhere within an **AP range**
- Devices, such as a personal computer, video-game console, and smartphone, use Wi-Fi to connect to a **network resource**, such as the Internet, via a **wireless network AP**

Types of Wireless Networks

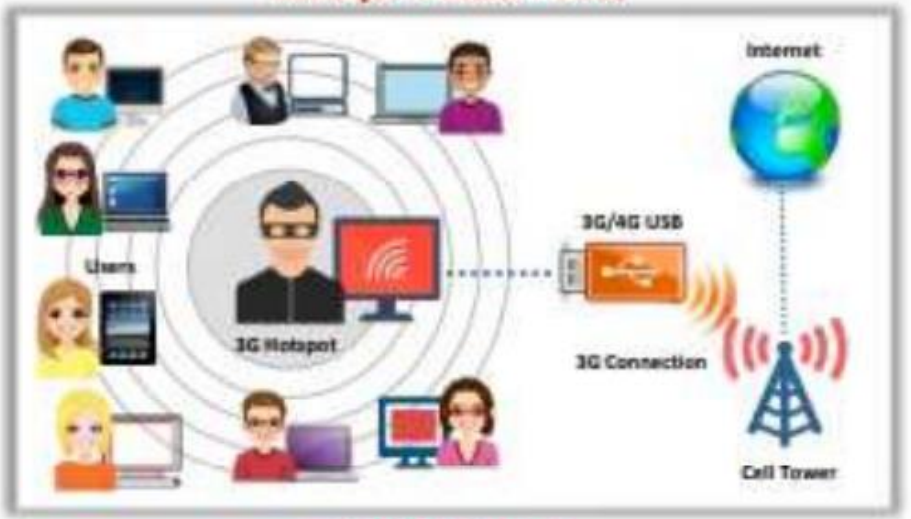


Extension to a Wired Network

Multiple Access Points



LAN-to-LAN Wireless Network



3G/4G Hotspot

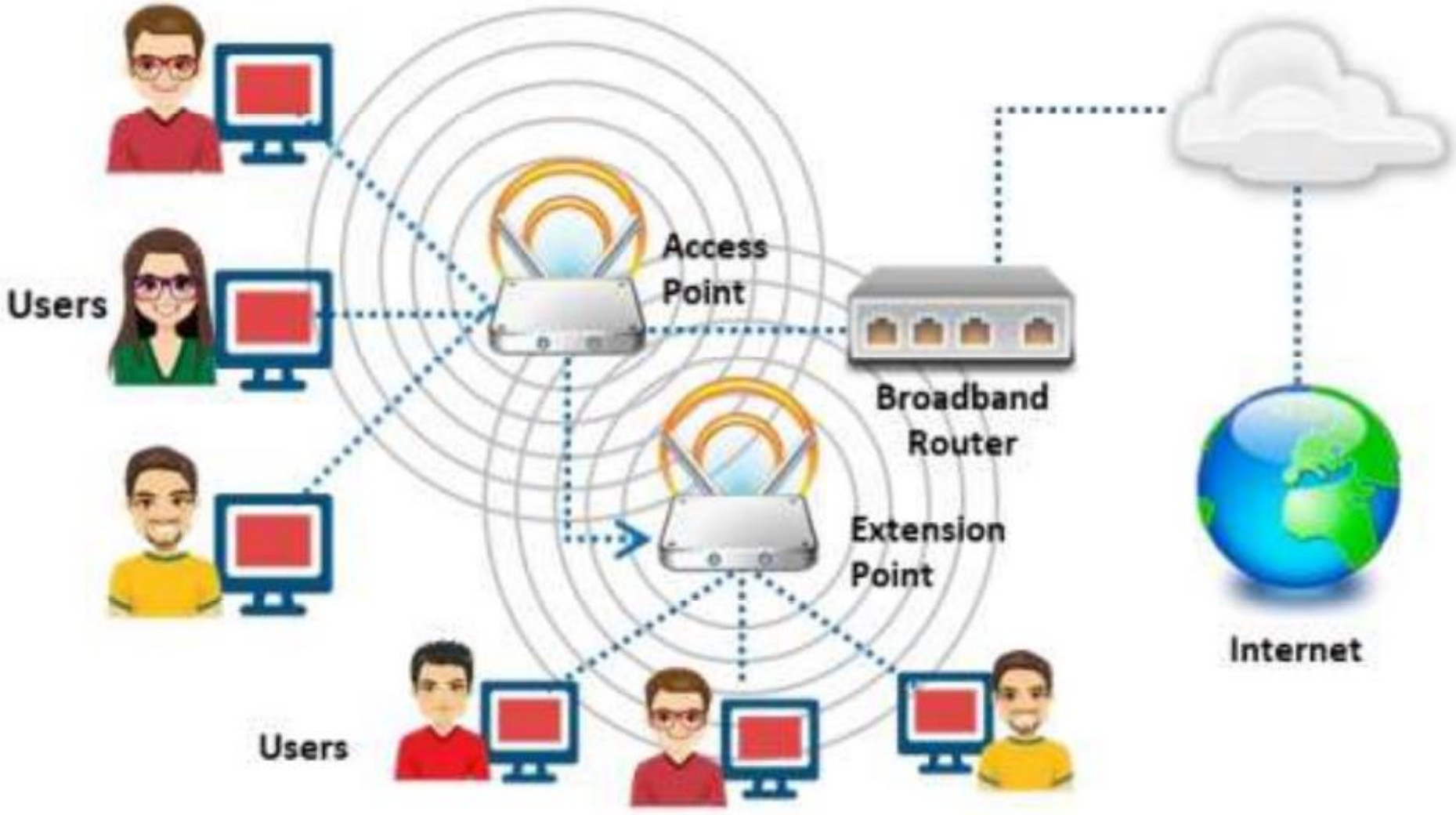


Figure 16.1: Extension to a wired network

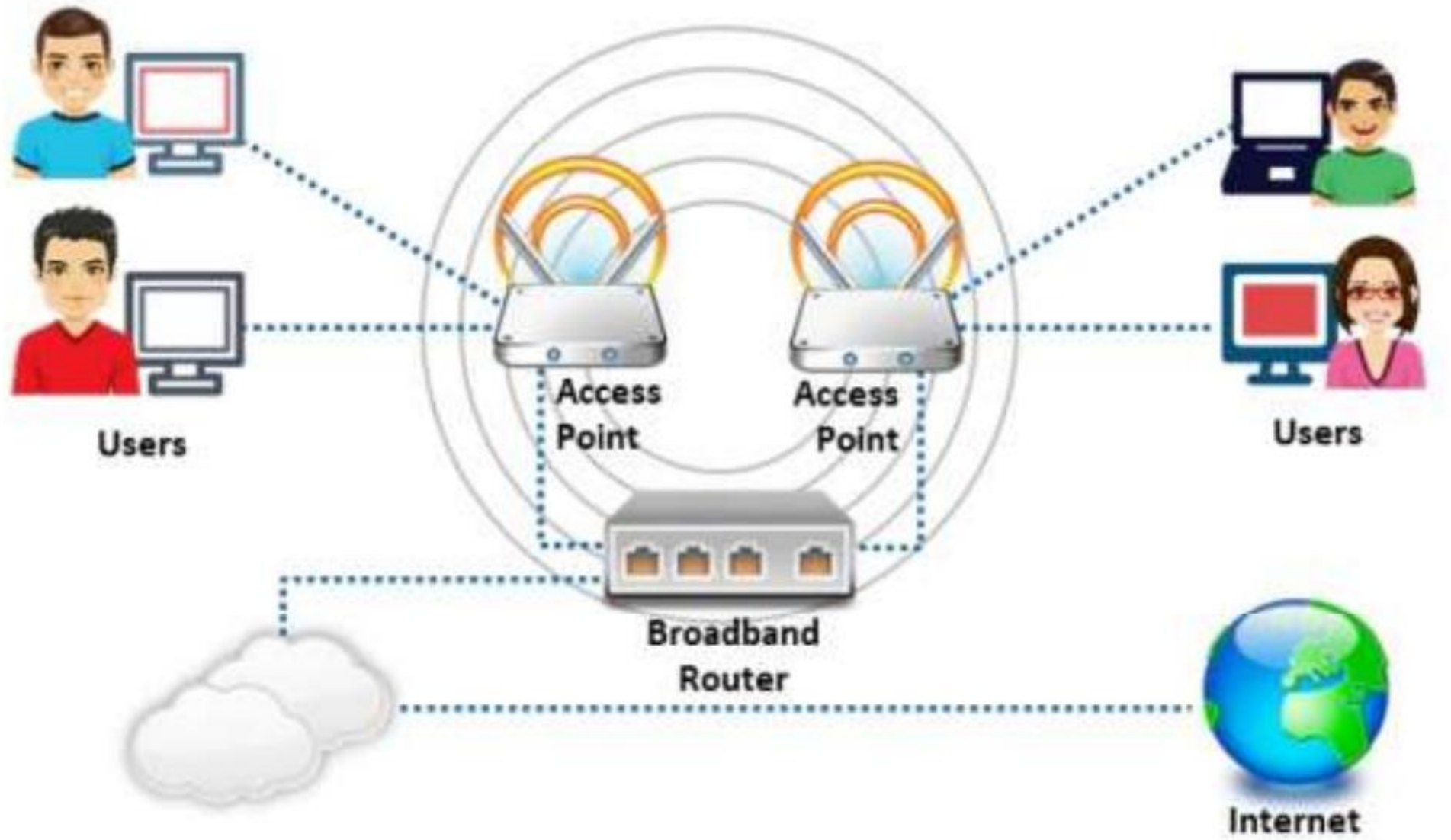


Figure 16.2: Multiple access points

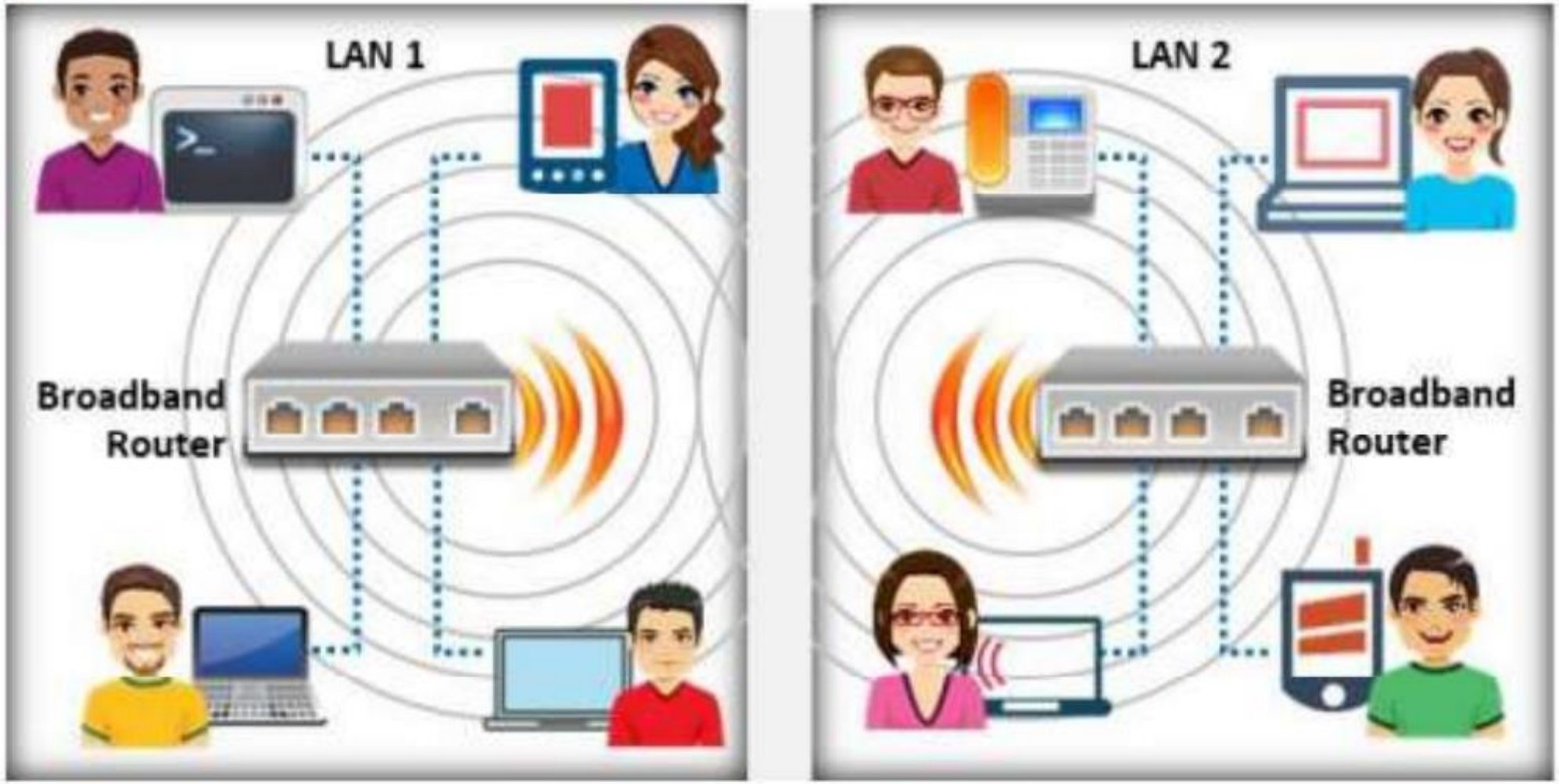


Figure 16.3: LAN-to-LAN wireless network

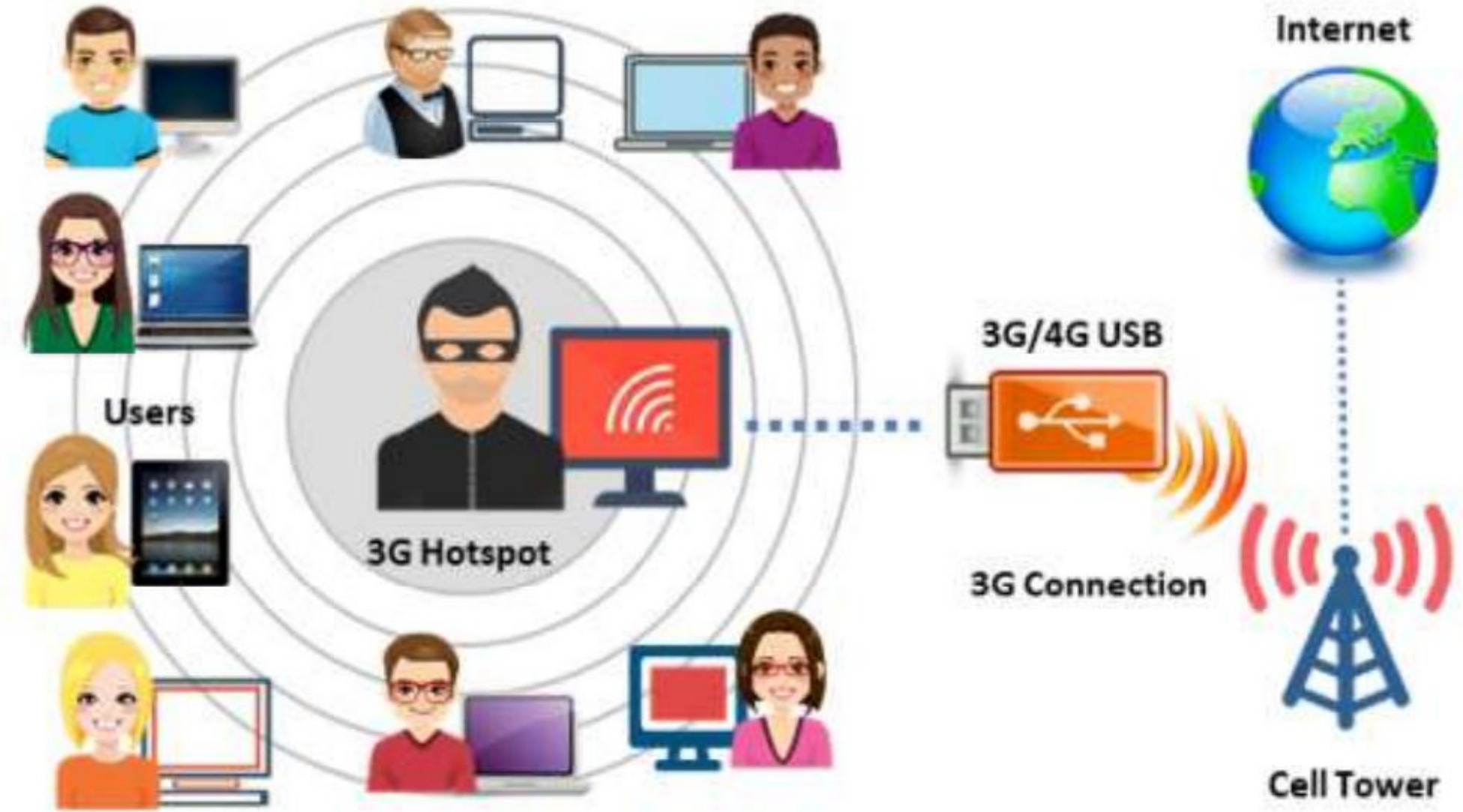


Figure 16.4: 3G/4G hotspot

Wireless Threats

Access Control Attacks

- Wireless access control attacks aim to penetrate a network by **evading WLAN access control measures**, such as AP MAC filters and Wi-Fi port access controls
  - WarDriving
  - Rogue Access Points
  - MAC Spoofing
  - AP Misconfiguration
  - Ad Hoc Associations
  - Promiscuous Client
  - Client Mis-association
  - Unauthorized Association

Integrity Attacks

- In integrity attacks, attackers **send forged control, management, or data frames over a wireless network** to misdirect the wireless devices to perform another type of attacks (e.g., DoS)
  - Data Frame Injection
  - WEP Injection
  - Bit-Flipping Attacks
  - Extensible AP Replay
  - Data Replay
  - Initialization Vector Replay Attacks
  - RADIUS Replay
  - Wireless Network Viruses

Confidentiality Attacks

- These attacks attempt to **intercept confidential information sent over wireless associations**, regardless of whether they were sent in clear text or encrypted by Wi-Fi protocols
  - Eavesdropping
  - Traffic Analysis
  - Cracking WEP Key
  - Evil Twin AP
  - Honeypot AP
  - Session Hijacking
  - Masquerading
  - Man-in-the-Middle Attack

Wireless Threats (Cont'd)

Availability Attacks

- Availability attacks aim at **obstructing the delivery of wireless services to legitimate users**, either by crippling those resources or by denying them access to WLAN resources

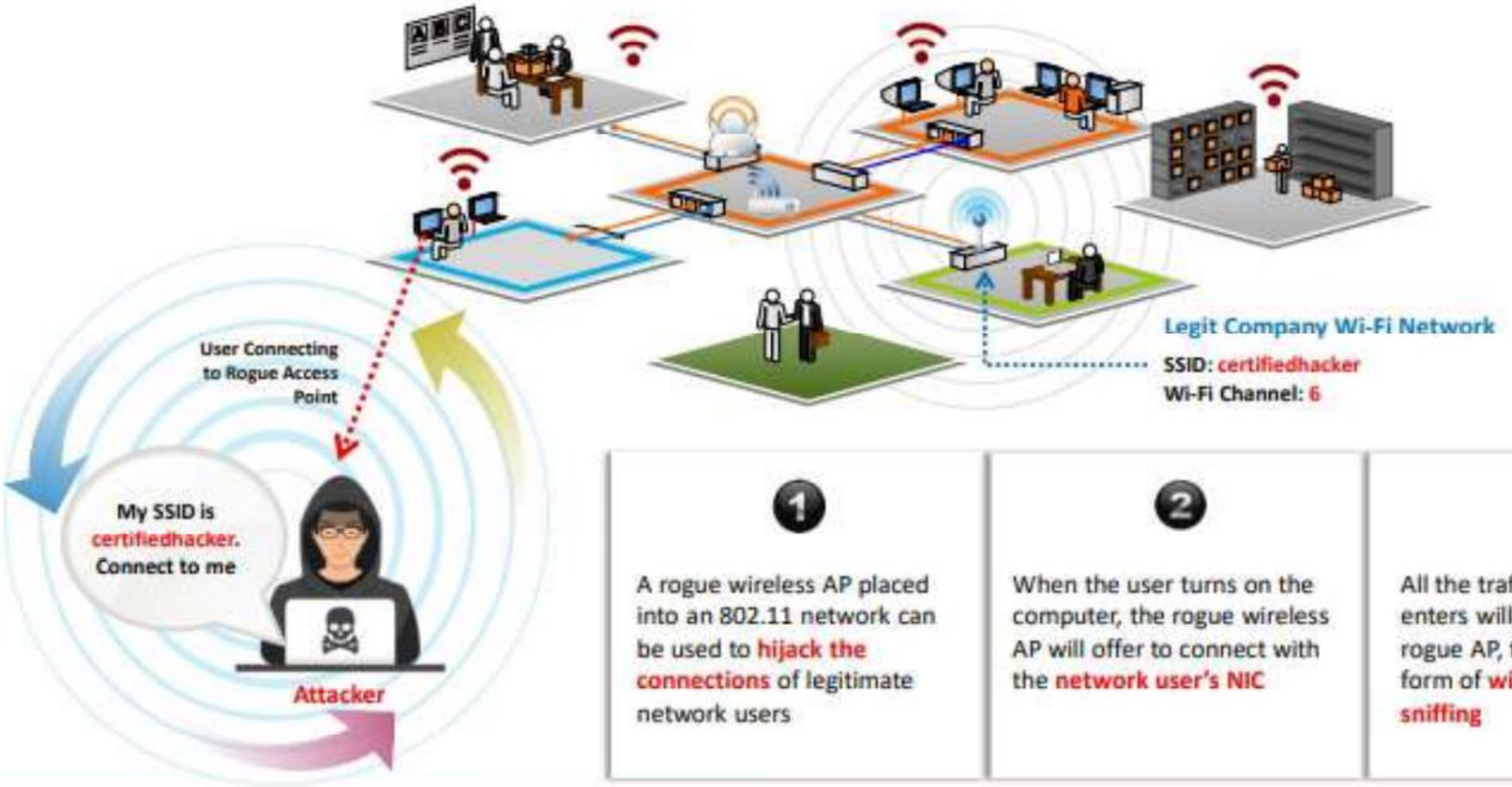
Access Point Theft	Denial-of-Service	Authenticate Flood
Disassociation Attacks	De-authenticate Flood	ARP Cache Poisoning Attack
EAP-Failure	Routing Attacks	Power Saving Attacks
Beacon Flood		TKIP MIC Exploit

Authentication Attacks

- The objective of authentication attacks is to **steal the identity of Wi-Fi clients**, their personal information, login credentials, etc. to gain unauthorized access to network resources

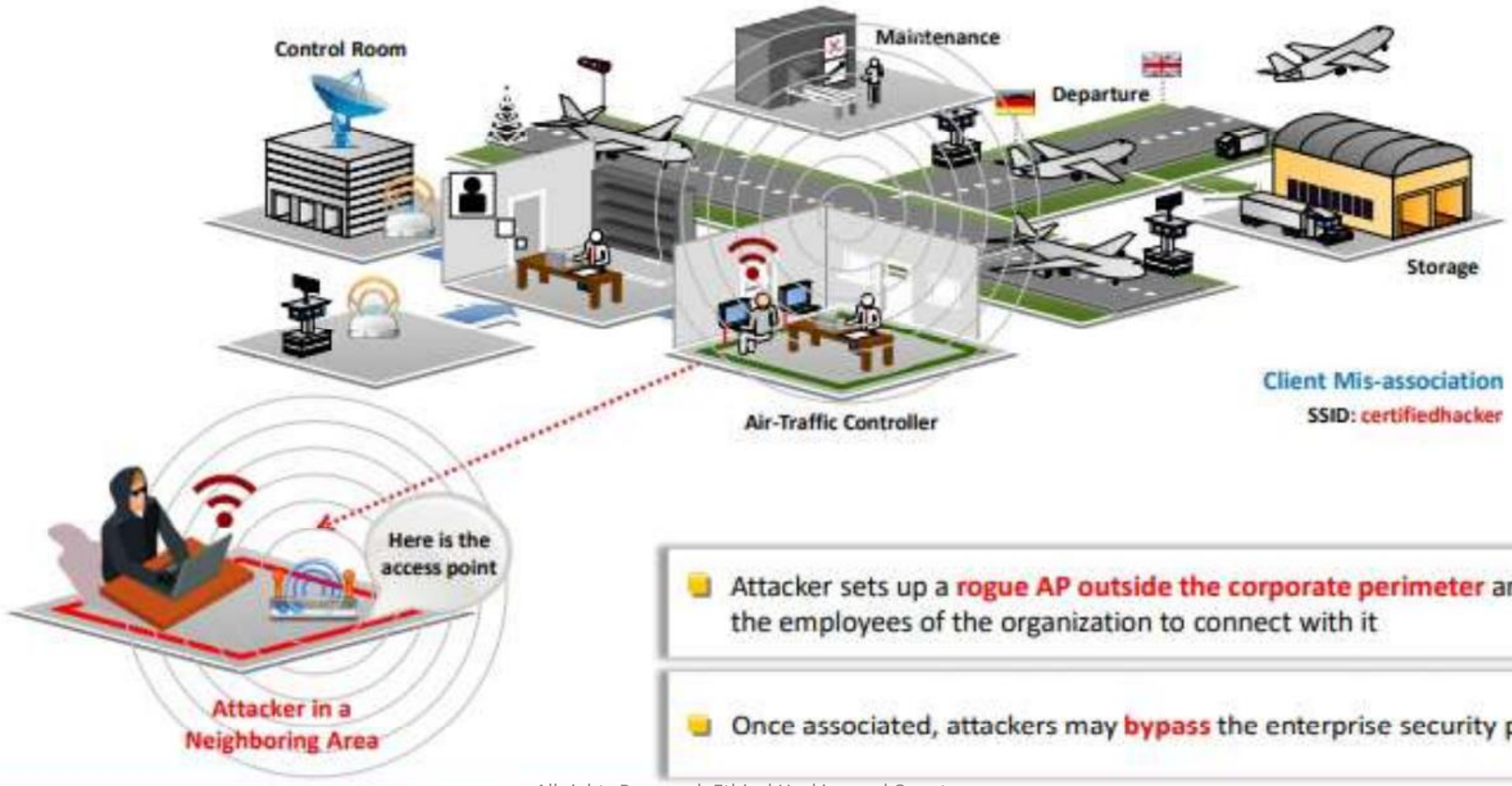
PSK Cracking	Key Reinstallation Attack	Identity Theft
LEAP Cracking		Shared Key Guessing
VPN Login Cracking		Password Speculation
Domain Login Cracking		Application Login Theft

Rogue AP Attack



1. A rogue wireless AP placed into an 802.11 network can be used to **hijack the connections** of legitimate network users
2. When the user turns on the computer, the rogue wireless AP will offer to connect with the **network user's NIC**
3. All the traffic the user enters will pass through the rogue AP, thus enabling a form of **wireless packet sniffing**

Client Mis-association



- Attacker sets up a **rogue AP outside the corporate perimeter** and lures the employees of the organization to connect with it
- Once associated, attackers may **bypass** the enterprise security policies

Misconfigured AP Attack

SSID Broadcast

APs are configured to **broadcast SSIDs** to authorized users

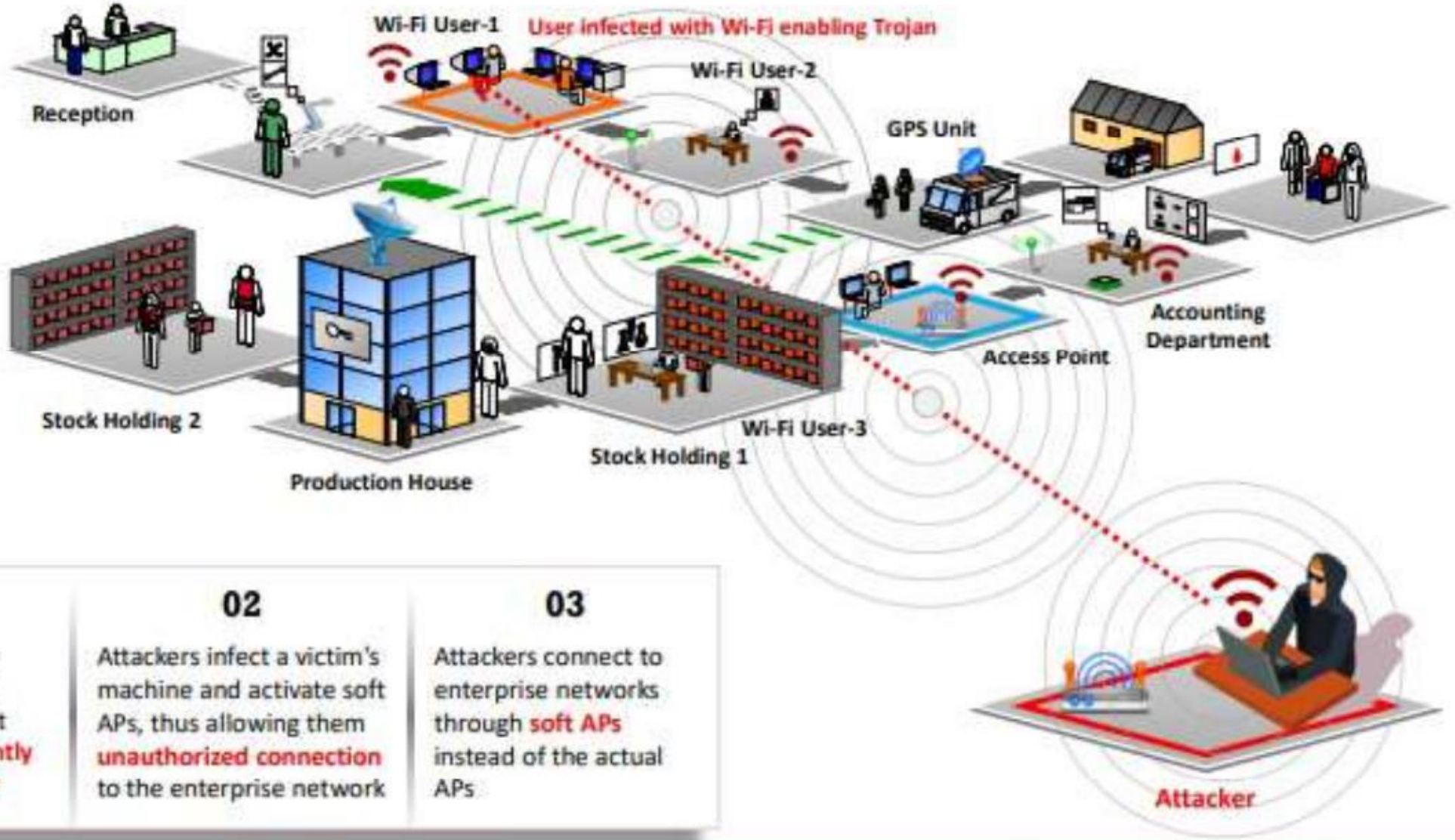
Weak Password

To verify authorized users, network administrators **incorrectly use the SSIDs as passwords**

Configuration Error

SSID broadcasting is a configuration error that enables intruders to **steal an SSID** and cause the AP assume they are allowed to connect

Unauthorized Association



01

Soft APs are client cards or embedded WLAN radios in some PDAs and laptops that can be launched inadvertently or through a virus program

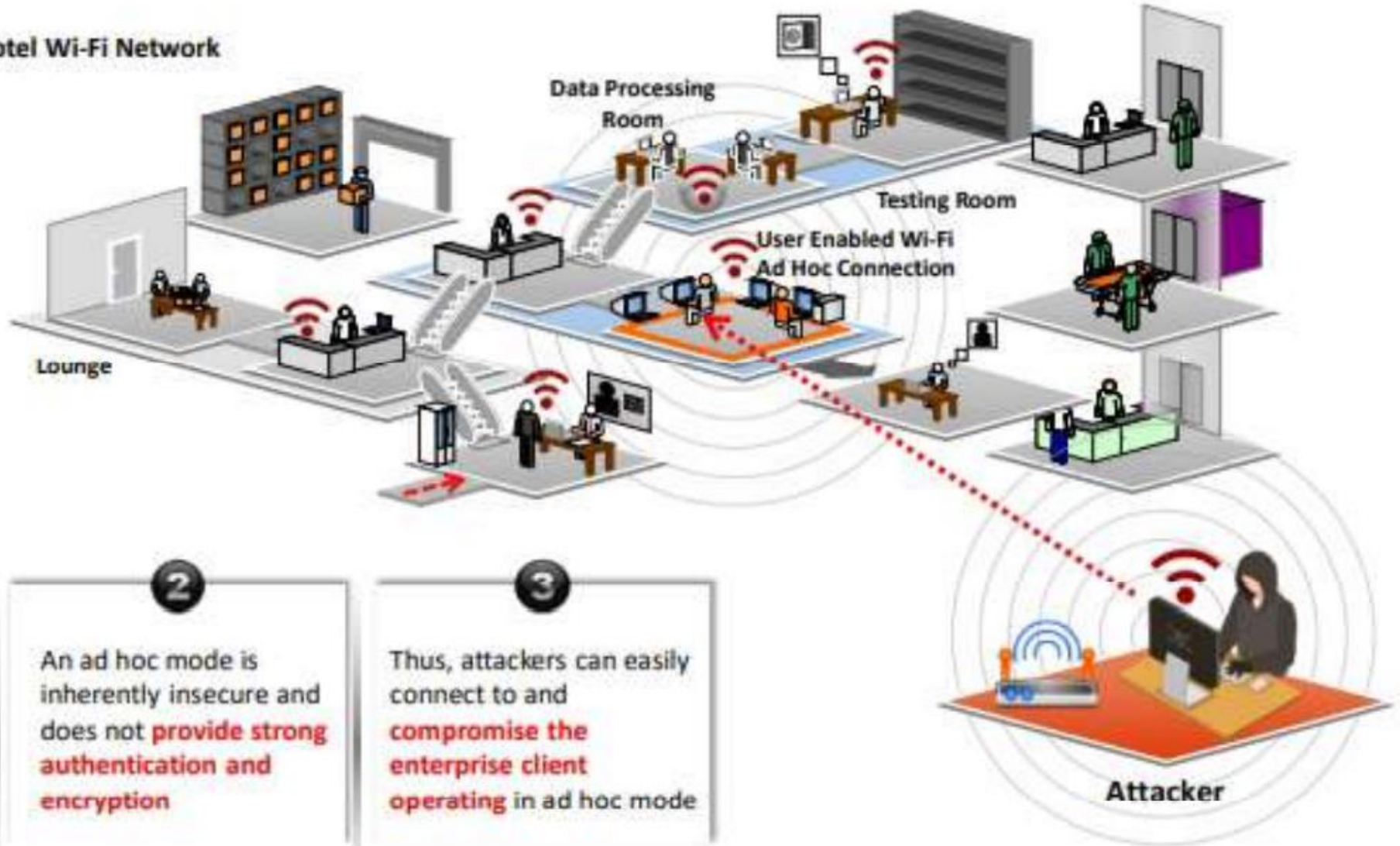
02

Attackers infect a victim's machine and activate soft APs, thus allowing them unauthorized connection to the enterprise network

03

Attackers connect to enterprise networks through soft APs instead of the actual APs

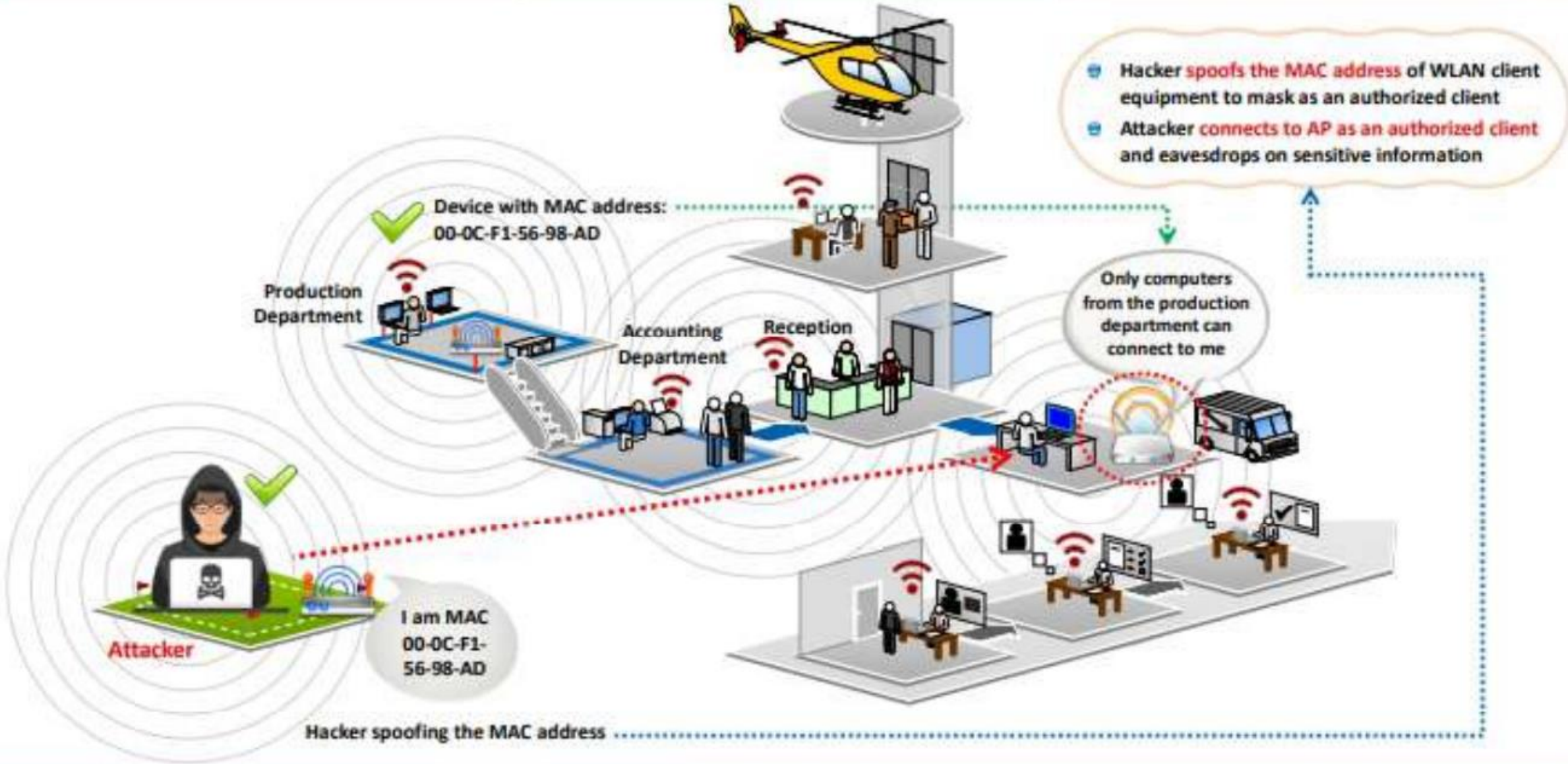
Ad-Hoc Connection Attack



1. Wi-Fi clients communicate directly via an **ad hoc mode** that does not require an AP to relay packets
2. An ad hoc mode is inherently insecure and does not **provide strong authentication and encryption**
3. Thus, attackers can easily connect to and **compromise the enterprise client operating** in ad hoc mode

Honeypot AP Attack

AP MAC Spoofing



Security+ Guide to Network Security Fundamentals, Fifth Edition

Chapter 9
Wireless Network Security

Wireless Attacks

- Several attacks can be directed against wireless data systems
- Attacks can be directed against:
  - Bluetooth systems
  - Near field communication devices
  - Wireless local area networks

Bluejacking

- **Bluejacking** - Attack that sends unsolicited messages to Bluetooth-enabled devices
- Can be text messages, images, or sounds
- Considered more annoying than harmful
- No data is stolen

Bluesnarfing

- **Bluesnarfing** - Unauthorized access to wireless information through Bluetooth connection
- Often between cell phones and laptops
- Attacker copies e-mails, contacts, or other data by connecting to Bluetooth device without owner's knowledge

Near Field Communication (NFC)

- **Near field communication (NFC)** –Low speed and low power technology for smartphones and smart cards
- Used to establish communication between devices in close proximity
- Once devices tapped together or brought within several centimeters each other two-way communication established
- NFC's ease of use opened door for wide range of practical short-range communications

NFC Contactless Payment

- NFC devices increasingly used in *contactless payment systems* so consumer can pay for purchase by tapping store's payment terminal with smartphone
- Users store credit card and/or store loyalty card information in "virtual wallet" the smartphone to pay for purchases at NFC-enabled point-of-sale (PoS) checkout device
- NFC contactless payment systems has risks because of the nature of this technology

Contactless Payment System (Figure 9-3)

Figure 9-3 Contactless payment system

© scyther5/Shutterstock.com

NFC risks and defenses (Table 9-2)

Vulnerability	Explanation	Defense
Eavesdropping	The NFC communication between device and terminal can be intercepted and viewed.	Because an attacker must be extremely close to pick up the signal, users should be aware of this. Also, some NFC applications can perform encryption.
Data manipulation	Attackers can jam an NFC signal so transmission cannot occur.	Some NFC devices can monitor for data manipulation attacks.
Man-in-the-middle attack	An attacker can intercept the NFC communications between devices and forge a fictitious response.	Devices can be configured in *active-passive* pairing so one device only sends while the other can only receive.
Device theft	The theft or loss of a smartphone could allow an attacker to use that phone for purchases.	Smartphones should be protected with passwords or PINs.

Table 9-2 NFC risks and defenses

Additional WLAN Enterprise Attacks

- In addition to creating multiple entry points, several different wireless attacks can be directed at enterprise:
  - Rogue access points
  - Evil twins
  - Intercepting wireless data
  - Wireless replay attacks
  - Wireless denial of service attacks

Rogue Access Points

- **Rogue access point** - Unauthorized AP allows attacker to bypass network security configurations and opens network and users to attacks
- Attacker who can access network through rogue access point is behind firewall and network protections
- Rogue APs can be hardware or software

Intercepting Wireless Data

- One of most common wireless attacks is intercepting and reading data (*packet sniffing*) being transmitted
- Attacker can pick up RF signal from open or misconfigured AP and read any confidential wireless transmissions
- If attacker manages to connect to enterprise wired network through rogue AP, also could read broadcast and multicast wired network traffic that leaks from wired network to wireless network

Wireless Replay Attack

- Wireless attack can “hijacking” wireless connection to perform wireless man-in-the-middle attack
- Makes it appear that wireless device and network computers are communicating with each other, when actually they sending and receiving data through evil twin AP (“man-in-the-middle”)
- **Wireless replay** - Attacker captures data being transmitted, records, and then sends to original recipient without attacker’s presence being detected

Wireless Denial of Service Attack

- **RF jamming** - Using intentional RF interference to flood RF spectrum with enough interference to prevent device from effectively communicating with AP
- Another wireless DoS attack takes advantage of an IEEE 802.11 design weakness
- Different types of frames can be “spoofed” by an attacker to prevent client from being able to remain connected to WLAN

Wireless Home Attacks

- Home users face several risks from attacks on their insecure wireless networks:
  - Data theft
  - Read wireless transmissions
  - Inject malware
  - Download harmful content

- **War driving** - Searching for wireless signals from automobile or on foot using portable computing device
- **War chalking** - Documenting and advertising location of wireless LANs for others

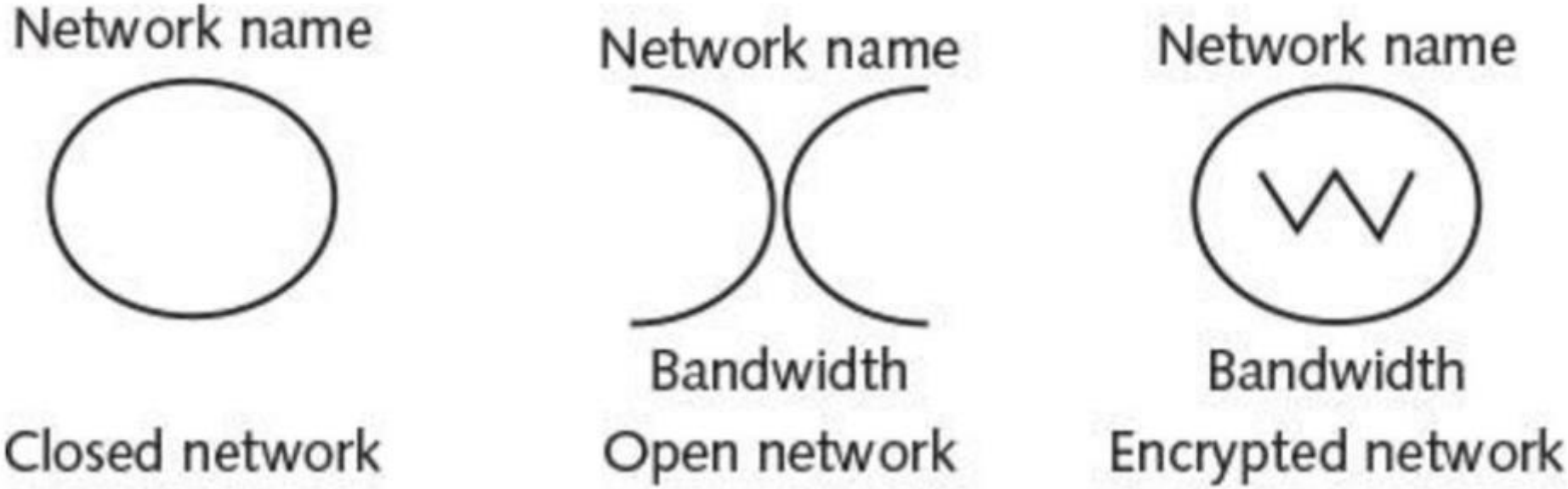


Figure 9-8 War chalking symbols

Tool	Purpose
Mobile computing device	A mobile computing device with a wireless NIC can be used for war driving. This includes a standard portable computer, a pad computer, or a smartphone.
Wireless NIC adapter	Many war drivers prefer an external wireless NIC adapter that connects into a USB or other port and has an external antenna jack.
Antenna(s)	Although all wireless NIC adapters have embedded antennas, attaching an external antenna will significantly increase the ability to detect a wireless signal.
Software	Client utilities and integrated operating system tools provide limited information about a discovered WLAN. Serious war drivers use more specialized software.
Global positioning system (GPS) receiver	Although this is not required, it does help to pinpoint the location more precisely if this information will be recorded or shared with others.

Table 9-4 War driving tools

WEP Vulnerabilities

- WEP security vulnerabilities:
  - WEP limited by length of IV of only 24 bits
  - WEP creates detectable pattern that can provide attacker with valuable information to break encryption

WPA Vulnerabilities

- Vulnerabilities in WPA:
  - Key management
    - Key sharing done manually without security protection
    - Keys must be changed on regular basis
    - Key must be disclosed to guest users
  - Passphrases
    - PSK passphrases fewer than 20 characters subject to cracking

Rogue AP Detection

- Several methods to detect rogue AP:
  - *Wireless device probe* - Standard wireless device (portable laptop computer) can be configured as wireless probe
  - *Desktop probe* – Desktop computer used as probe
  - *Access point probe* – APs can detect neighboring APs
  - *Dedicated probe* – Exclusively monitor RF frequency for transmissions

WPA(WiFi protected Access) vulnerabilities:
Key Management;
KRACK(Key Reinstallation Attack).
WPA3, new authentication method helps thwart KRACK & offline dictionary attack.
Using TKIP for strong authentication.

WEP(Wired Equivalent Privacy) encryption.
Security protocol defined by 802.11b standard.
Uses 24 bit IV(Weak authentication) to form stream cipher RC4 for confidentiality & CRC-32 checksum for integrity.

Wireless Hacking Methodology

Topic 12

- The objective of the wireless hacking methodology is to **compromise a Wi-Fi network** to gain unauthorized access to network resources

1. Wi-Fi Discovery
2. GPS Mapping
3. Wireless Traffic Analysis
4. Launch of Wireless Attacks
5. Wi-Fi Encryption Cracking
6. Compromise the Wi-Fi Network

EC-Council

C|EH
Certified Ethical Hacker

Ethical Hacking and Countermeasures v11

PROFESSIONAL SERIES

EC-Council

EC-Council New Mexico
101C Sun Ave NE
Albuquerque, NM 87109

NOTICE TO THE READER

Ethical Hacking and Countermeasures Copyright © by EC-Council
All Rights Reserved. Reproduction is Strictly Prohibited.

Colourful Thank You Slide Design

THANK YOU