

CompTIA Security+

Lecture 03

SY0-701

Vulnerabilities and Attacks

By
Dr. Khaled ELMenshawy

Lecture

03

LEARNING OUTCOMES

- Understanding Hardware Vulnerabilities and methods of mitigation.
- Understanding Bluetooth Vulnerabilities and methods of mitigation.
- Understanding Mobile Vulnerabilities and methods of mitigation.
- Understanding OS Vulnerabilities and methods of mitigation.
- Recognize SQL and XML Injection Attacks.
- Recognize XXE, XSS and DOM Attacks.
- Differentiate between session management, session hijacking, and session prediction.
- Understanding Cross-Site Request Forgery Attacks (CSRF).
- Understanding Buffer Overflow Attack.

What is A vulnerability?

A Vulnerability is a weakness inherent in an asset that leaves it open to a threat.

These weaknesses may allow unauthorized access, data breaches, system disruptions, or other forms of compromise to occur, and that can compromise the security and integrity of the targeted system or network.



What is An Attack?

An Attack is deliberate *actions or activities* that are carried out *by threat actors* with the intent *to exploit vulnerabilities* in a computer system, network, or application.

These actions can take lots of different forms, including *unauthorized access*, *data theft*, *malware infections*, *denial of service attacks*, *social engineering*, or other types of activities aimed at compromising the confidentiality, integrity, or availability of our digital assets.

What is A Hardware Vulnerabilities?

A Hardware Vulnerabilities are security weaknesses that are inherent *in a device's physical components* or design that can be exploited to compromise the integrity, confidentiality or availability of the system and its data.



There are numerous different hardware vulnerabilities includes:

- ***Firmware;*** is a specialized *form of software embedded into hardware devices* that provides basic machine instructions that *allow the hardware to operate effectively* and communicate with other software running on a device.

In Firmware attacks, the attacker can spy on users' devices, intercepting their online activity, stealing their data, and gaining remote control of their machine by *embedding malware into the firmware* using the existing vulnerabilities. To prevent firmware attacks; Constantly *update firmware, Avoid untrusted storage devices, and purchasing hardware with built-in firmware security.*

- ***End of life, Legacy, and unsupported systems*** ; refer to hardware or software products that have *reached the end of their lifecycle* so that their manufacturer or developer is *no longer providing updates, support, or enhancements* for them.

These kinds of systems and applications are vulnerable because any existing security flaws will not be addressed with patches or updates, and this leaves that system exposed to those known vulnerabilities.

- **Unpatched system** ; is a *device, application or piece of software that has not been updated* with the latest security patches.

Not patching the systems in a timely manner though, the organization is going to be exposed to a wide range of security risks because an attacker can easily exploit these known vulnerabilities to gain unauthorized access into your systems, compromise your data, or to disrupt your services.

- **Hardware misconfiguration;** occurs when a device's settings, parameters or options are *not optimally set up*.

These misconfigurations can inadvertently open up security holes that would make the system susceptible to unauthorized access and data breaches.

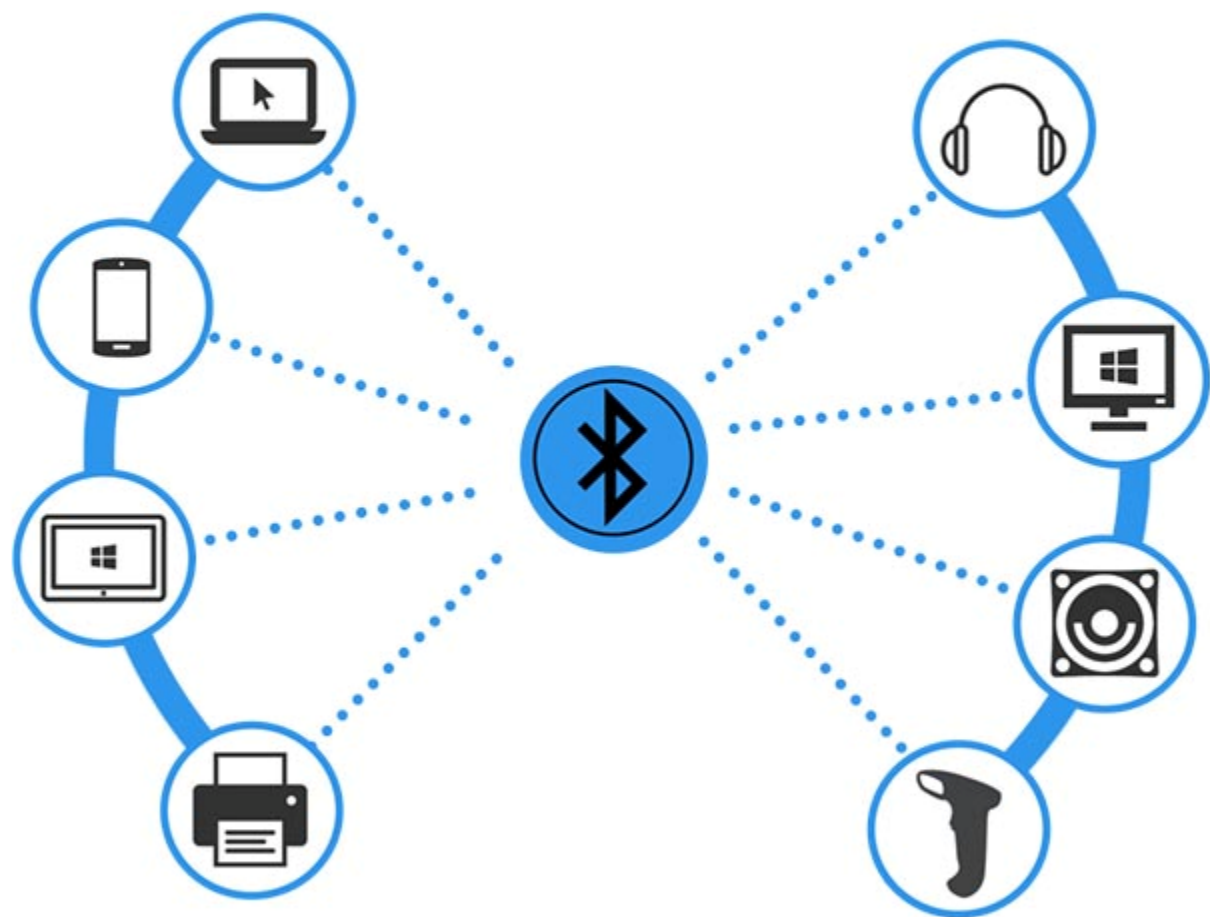
To mitigate the risks associated with hardware misconfigurations; *enforce good configuration management, implement automated tools, providing training to the system administrators.*

- *There are different methods to mitigate hardware vulnerabilities includes:*
 - ***Hardening***; involves tightening the security of your system by *closing unnecessary ports*, *disabling unnecessary services*, and *setting up proper permissions*.
 - ***Patching***; involves the regular *updating of the software, firmware and applications* with the latest security patches to ensure that known vulnerabilities are being fixed in order to prevent exploitation.
 - ***Configuration enforcement***; is going to be used to ensure that *all devices and systems adhere to a standard secure configuration* to help prevent any misconfiguration and vulnerabilities from occurring.

- **Decommissioning;** is used with *end of life, legacy or unsupported systems*. This means that the system is going to be retired and *removed from the network*, and this is usually done because it reaches the end of its life.
- **Isolation;** is used *when decommissioning is not possible* at the current time, so *isolation of end of life, legacy or unsupported systems from the network* is necessary to avoid security breach.
- **Segmentation;** involves *dividing the network into segments*, to ensure that even if one section of our network is compromised, that breach will not necessarily compromise the entire enterprise network.

What is Bluetooth?

Bluetooth is a wireless technology standard used for exchanging data between fixed and mobile devices over short distances without the need for an Internet connection.



The vulnerabilities associated with Bluetooth includes:

- ***Insecure pairing;*** happens when Bluetooth devices *establish a connection without proper authentication*, or they employ weak authentication methods, which can leave them vulnerable to unauthorized access or interference.
- ***Device spoofing;*** happens when an *attacker can impersonate a legitimate Bluetooth* device by mimicking its address and device name to deceive a user into connecting to a malicious device.
- ***On-path attack;*** happens with a Bluetooth connections, when the attacker can *intercept the communication between two Bluetooth devices* without either party being aware.

The attacks associated with Bluetooth includes:

- **Bluejacking attack;** an attacker *sends unsolicited messages* to a Bluetooth-enabled device to test the vulnerability of the device without causing any kind of serious harm.
- **Bluesnarfing attack;** involves the *unauthorized access* to a device through its Bluetooth connection to *steal information like contacts, call logs, and text messages*.
- **Bluebugging attack;** allowing attackers to *take control of a device's Bluetooth functions* and potentially use the device to make calls, send text messages, or access the Internet.

- **Bluesmack attack;** this attack focuses on a denial of service by sending a large amount of data to cause it to crash or become unresponsive.
- **Blueborne attack;** is going to target devices via Bluetooth, and it can *spread virus through the air to infect numerous devices within seconds* without requiring any intervention from the user. The targeted device does not need to be paired to the attacker's device or even to be set on discoverable mode.

- *There are different methods to mitigate Bluetooth vulnerabilities includes:*
 - *Turned Bluetooth off;* when it's not actively being used.
 - *Set the devices to non-discoverable by default;* prevent the device from being visible to other unknown devices and reduce the risk of any unsolicited connection attempts being made.
 - *Update the device's firmware to the latest version;* updating the firmware will ensure that the device's security remains current and up to date.
 - *Allow pairing only to trusted devices;* only connecting to legitimate devices and not malicious ones.

- ***Using a PIN or a passkey;*** this adds a layer of security during that pairing process.
- ***Cautious of unsolicited connection requests.;*** Unexpected or unsolicited file requests could have contained malware embedded within them.
- ***Using encryption anytime for sending or receiving sensitive data;*** encryption is going to scramble the data being transferred over the Bluetooth connection to ensure that it remains unreadable to any unauthorized parties.

What is Mobile devices?

Mobile devices, such as *smartphones*, *tablets*, and *wearables*, have become an indispensable part of our daily lives. These devices are used for connections, entertainment, and increasingly secure and manage the personal and professional data.



The vulnerabilities associated with mobile devices includes:

- ***Sideload;*** is *installing applications* on a mobile device from *unofficial sources*, which bypasses the device's default application store. It can *introduce malware or malicious apps onto a device*, because the application files are not scanned or verified by your mobile device's provider's app store submission process.
- ***Jailbreaking and rooting;*** *jailbreaking on iOS* and *Rooting on an Android device* are the process where the user gains access to the superuser (root) of the device, that enable the users to remove the limitations put in place by a device's manufacturer putting the mobile devices to potential security breaches.

- ***Insecure connection methods;*** is *sending sensitive information over unencrypted connection channel*. Always *avoid using open Wi-Fi networks* or pairing with unknown devices over Bluetooth, because both of these can expose your mobile devices to *eavesdropping* (the act of listening to, recording or intercepting private communications), an *on-path attack*, *unauthorized data access*, or other types of malicious attacks.

Instead of relying on open Wi-Fi networks, use the device's own cellular data connection, or use a Wi-Fi network that protected by a robust authentication system.

Mobile Device Management (MDM) to mitigate mobile devices vulnerabilities includes:

- ***Mobile Device Management (MDM)*** ; allows IT to automate, control, and secure administrative policies on laptops, smartphones, tablets, or any other device connected to an organization's network.

IT administrators configure policies through the *MDM server's management console*, and the server then pushes those policies over the air to the *MDM agent on the device*. *The agent applies the policies to the device* by communicating with application programming interfaces (APIs) built directly into the device operating system.

What is Operating Systems?

Operating System, is a *system software* that *functioning and protection of the digital devices* because it bridges the gap between hardware and software to ensure that seamless interactions are going to occur.



The vulnerabilities associated with operating systems includes:

- **Unpatched systems;** are operating systems that *have not been updated with the latest security patches or fixes*. These systems are *highly susceptible to attack* because they contain vulnerabilities that might have been addressed in newer updates.
- **Zero-day vulnerabilities;** is *any vulnerability that's discovered before the vendor can issue a patch for it*, but an attacker was able to figure it out and exploit it to attack the OS.

Implementing a security solutions like a host-based Intrusion Prevention System (IPS) can detect and block suspicious activities that may occur on a given system.

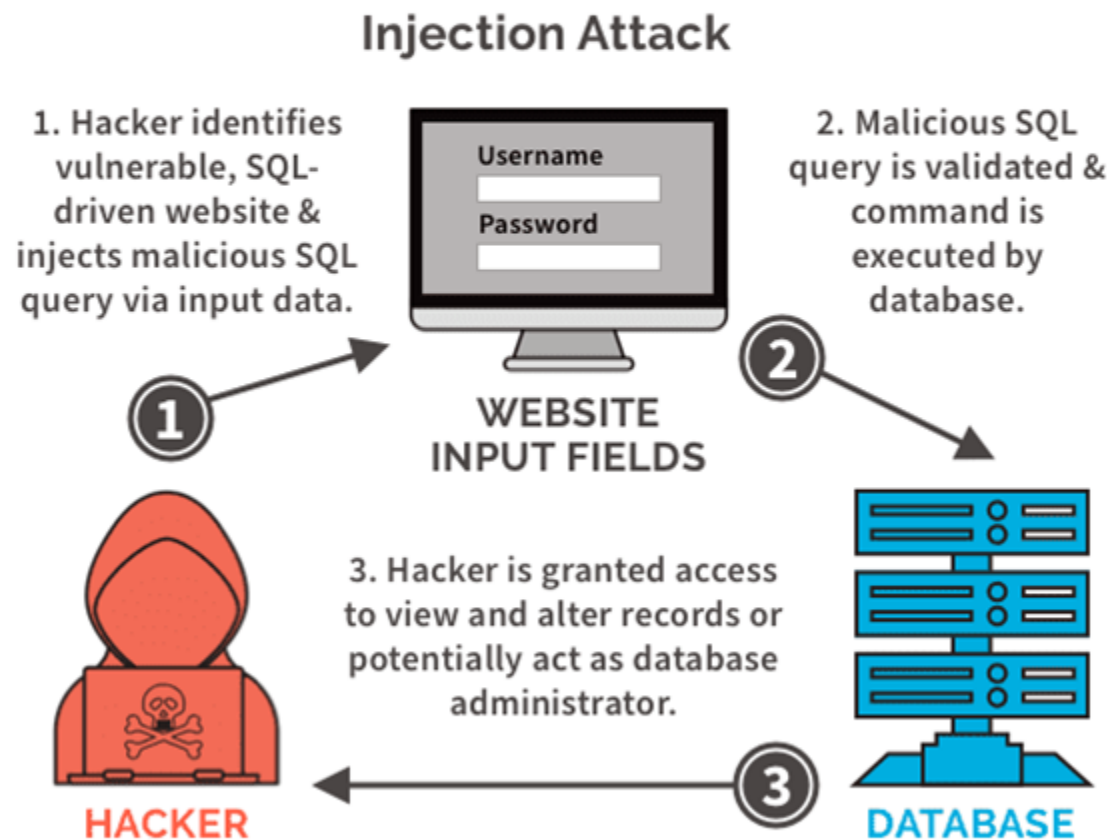
- **Misconfigurations;** occur when the *system settings are not properly configured*, and this leaves the system vulnerable to exploitation.

These configurations could be due to *default accounts / passwords are enabled*, *Secure password policy is not implemented*, or unnecessary services being left enabled such as Remote desktop, NetBIOS over TCP/IP and NetBIOS name resolution, and Bluetooth Support Service. *Configuration management tools* should be employed across your organizational network such as (Ansible, Puppet, CFEngine,...etc.).

- **Data exfiltration;** refers to *unauthorized data transfers* from within an organization to an external location. Protecting against data exfiltration involves multiple layers of defense including the *use of encryption* and *endpoint protection tools*.
- **Malicious updates;** where an attacker's been able to *patch a malicious update to a well-known and trusted program* in order to compromise the systems of the programs and users. To guard against the malicious update, it's crucial to *verify the source the updates* from trusted vendors and official channels. Also *verify the authenticity of updates* through a digital signature or a hash before installing the update on the system.

What is SQL and XML Injection Attacks?

SQL and XML Injection Attacks, occurs when an attacker sends malicious data to a system in such a way that it can be interpreted and run as a valid SQL query to perform a database operation with malicious intent.



SQL Injection Attacks

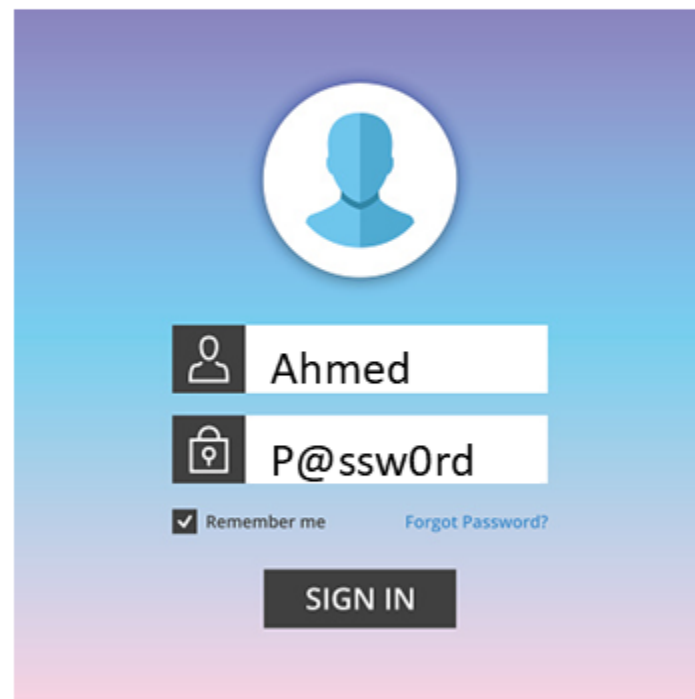
- Dealing with a database in order to *read*, *write*, or *search for data inside* is done by using statements in a language known as the *Structured Query Language, or SQL*.
- There are four main actions inside of **SQL** that you can use to manipulate the data held inside of a database. These are *select*, *insert*, *delete*, and *update*.
- When the user log into a website, he may encounter a basic login form asking for his username and password match what is being stored inside of the database, to do this, it's going to use an SQL statement.

SQL Injection Attacks

➤ ***For example**, for the statement as follows:*

```
Select * from Users where user_id =  
'Ahmed' and password = 'P@ssw0rd';
```

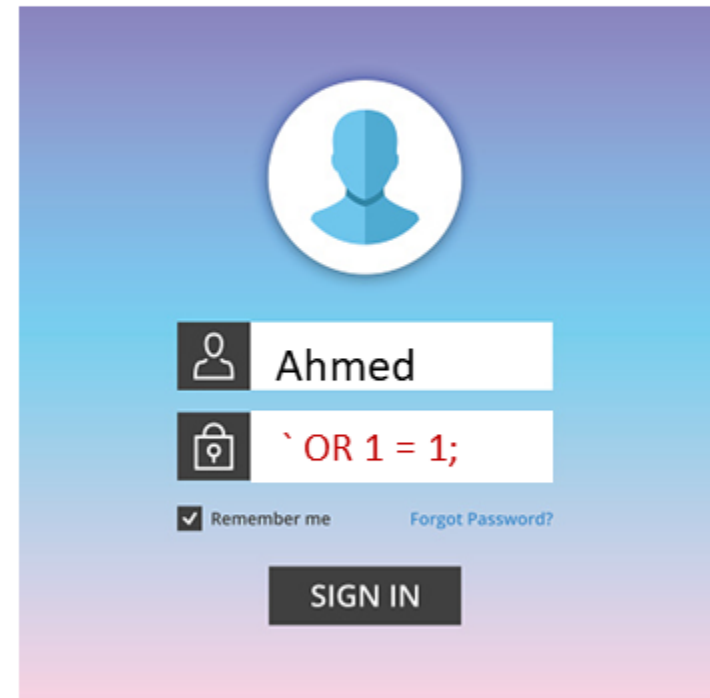
This statement goes into the database, and then will determine if there is a perfect match. If it is, the user is going to have access granted to the website, if it is not, the website will deny access.



SQL Injection Attacks

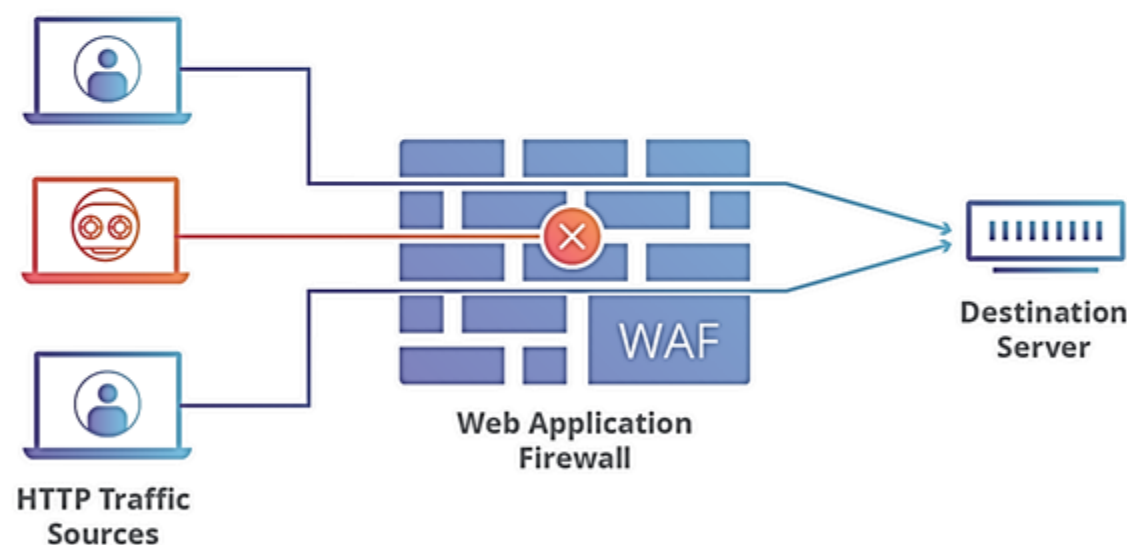
- A code injection is the insertion of additional information or code through a data input form from a client to an application.
- To do that the attacker inputs the username “Ahmed” then input the following parameters instead of password ``OR 1 = 1;`
- The data from the form is going to be inserted back into the SQL statement as:

```
Select * from Users where user_id = 'Ahmed' and password = '  
`OR 1=1;'
```



There are different methods to mitigate SQL Injection Attacks include:

- **Input validation;** validate the inputs of the user before executing SQL Statement.
- **Web application firewall;** is an application layer protocol filtering and monitoring HTTP traffic between a web application and the Internet to protect web applications from attacks such as SQL Injection.



XML Injection Attacks

- ***Extensible Markup Language (XML)***; used by web applications for authentication, authorization, or other types of data exchange.
- XML data is submitted from the client to the server, or from one server to another.
- To protect XML data in transit, it should always be *sent inside of an encrypted tunnel such as by using a TLS tunnel*.
- To protect the server when it receives XML data, performing *input validation (checking whether user input meets certain criteria)* and *input sanitization (removing any potentially harmful or unwanted characters or elements from user input)* of the data before you process it.

XML Injection Attacks

➤ *XML format;*

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<note>
```

```
  <to>Ahmed</to>
```

```
  <from>Mona</from>
```

```
  <heading>Reminder</heading>
```

```
  <body>Don't forget me this weekend!</body>
```

```
</note>
```

```
</xml>
```

XML Injection Attacks

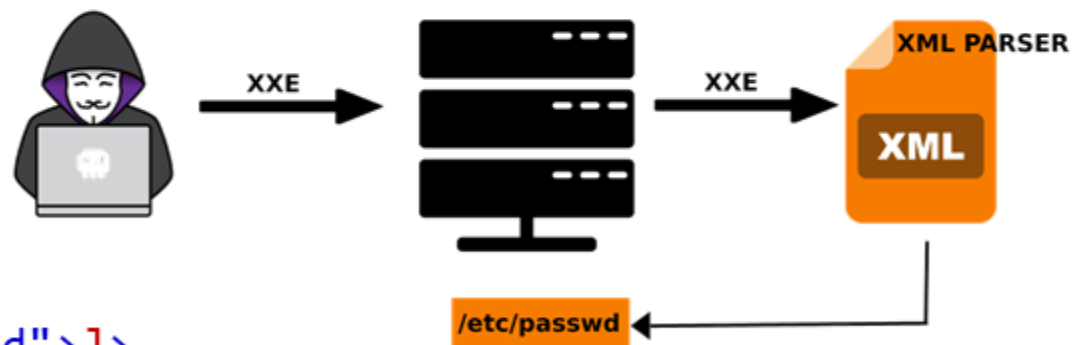
- ***XML bomb (A Billion Laughs Attack)***; is a *Denial-of-Service attack (DoS)* which is a result of XML injection vulnerability. It is designed in such a way that it keeps expanding leading to a billion “*lol*” being called at once which will consume memory on the host, and potentially crash it.

```
<?xml version="1.0"?>
<!DOCTYPE lolz [
<!ENTITY lol "lol">
<!ELEMENT lolz (#PCDATA)>
<!ENTITY lol1 "&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;">
<!ENTITY lol2 "&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;">
<!ENTITY lol3 "&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;">
<!ENTITY lol4 "&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;">
<!ENTITY lol5 "&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;">
<!ENTITY lol6 "&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;">
<!ENTITY lol7 "&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;">
<!ENTITY lol8 "&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;">
<!ENTITY lol9 "&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;">
]>
<lolz>&lol9;</lolz>
```

XXE Attacks

- **External Entity Attack (XXE)**; allows hackers to *view and retrieve sensitive files* on the application server, and they can even be used to perform SSRF (Server-Side Request Forgery).

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE layer [
  <!ELEMENT layer ANY>
  <!ENTITY external SYSTEM "file:///etc/password">]>
<layer>&xxe;</layer>
```



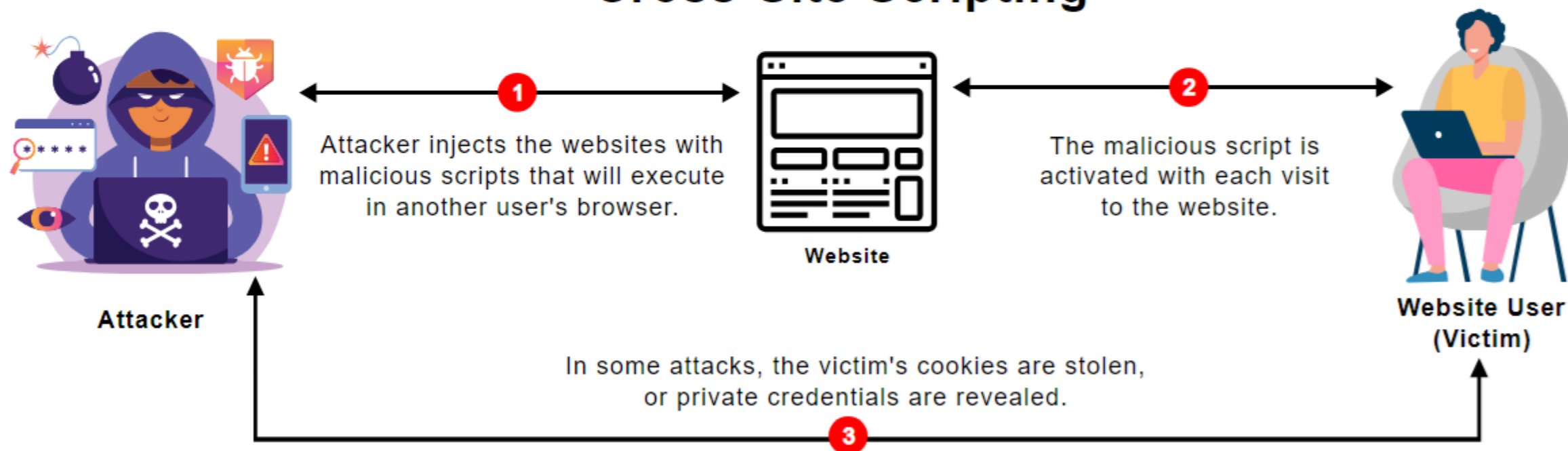
XSS Attacks

- **Cross-Site scripting (XSS);** is a server-side attack occurs when hackers *execute malicious code within a victim's browser*.
- **First**, the attacker *identifies an input validation vulnerability* within a trusted website.
- **Second**, the attacker *crafts a URL to perform the code injection against that trusted website* and embed it in an email message using a phishing campaign or post it to an online forum.
- **Third**, the *trusted site will return a page, containing the malicious code injected*, as well as the regular site code.

XSS Attacks

- *Fourth*, the *malicious code is now run in the client's browser* finalizing the cross-site scripting attack.

Cross-Site Scripting



`https://www.newegg.com/p/pl?d=rapoo<script> alert (1) </script>`

XSS Attacks

- ***Non-persistent cross-site scripting attack;*** *occurs when it's launched* because the user clicking on that link or entering it into the web browser and pressing enter.
- ***Persistent cross-site scripting attack;*** allows *an attacker to insert code into the backend database* used by that trusted website. By doing this, the attacker doesn't have to wait for somebody to click on a link anymore because *the malicious code is now embedded into that website's database.*

DOM Attacks

- ***Document Object Model (DOM) cross-site scripting attack***; is a client-side attack where the *attacker manipulates client's browser environment* (Document Object Model) and places payload into page content *to modify the content and layout of the webpage*.

`https://www.newegg.com/index.html#default<script>alert(document.cookie) </script>`

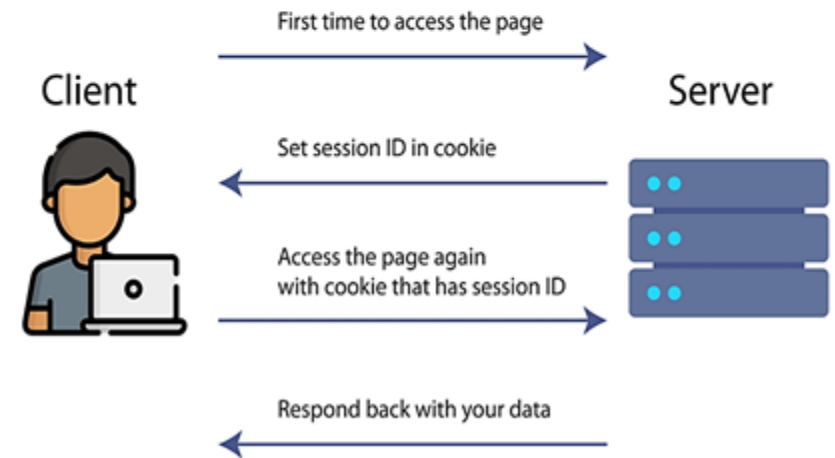


`document.write`

`document.location`

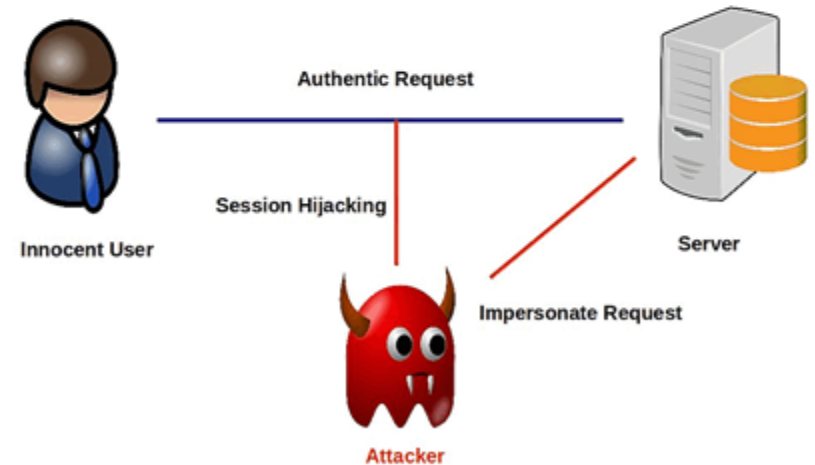
Session Management

- **Session Management;** is *handling multiple requests and responses from a user or entity on a website or web application* that enables web applications to *uniquely identify a user across a number of different actions and requests* while keeping the state of the data generated by that user and ensuring it remains assigned to that particular user.
- The cookie (*persistent or non-persistent*) allows the server to identify the user and retrieve the user session from the session database, so that the user session is maintained.



Session Hijacking

- **Session Hijacking;** refers to the *malicious act of taking control of a user's web session*. where the attacker disconnects a host and then replaces it with his or her own machine by spoofing the original host IP.
- A lot of session hijacking occurs by *taking cookie theft or modification of those session cookies*. Now, if the attacker is able to steal the session cookie, they can take over the session as the already authenticating user.

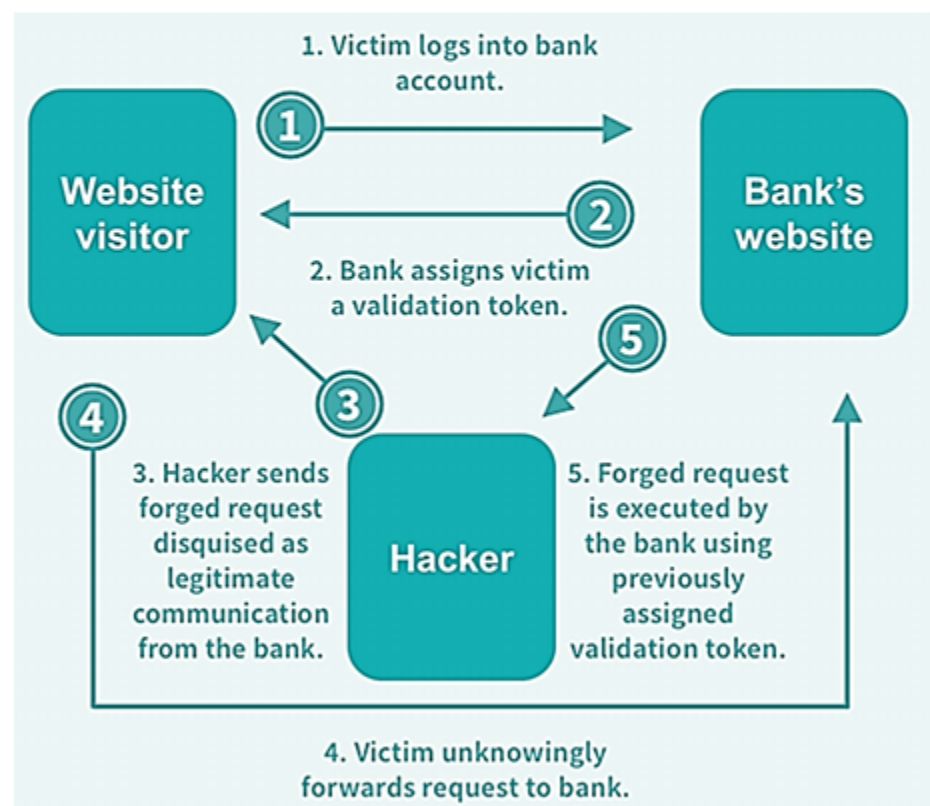


Session Prediction

- ***Session Prediction;*** is an attack technique used to hijack or impersonate a web or application user. *The attack focuses on predicting session ID values that permit an attacker to bypass the authentication method of an application.* By analyzing and understanding the session ID generation process, an attacker can predict a valid session ID value and get access to the application.
- ***To prevent session prediction attacks;*** *the session tokens need to be generated using a non-predictable algorithm,* and they need to be truly randomized. That way, the session tokens won't be able to be easily guessed.

Cross-Site Request Forgery Attacks

➤ **Cross-Site Request Forgery Attacks (CSRF)** ; is an attack that *forces authenticated users to submit a request to a Web application against which they are currently authenticated. For this to work*, the attacker needs to convince the victim to start a session with the targeted website. Once that occurs, the attacker can then pass an HTTP request to the victim's browser and spoof this as an action on the target site.



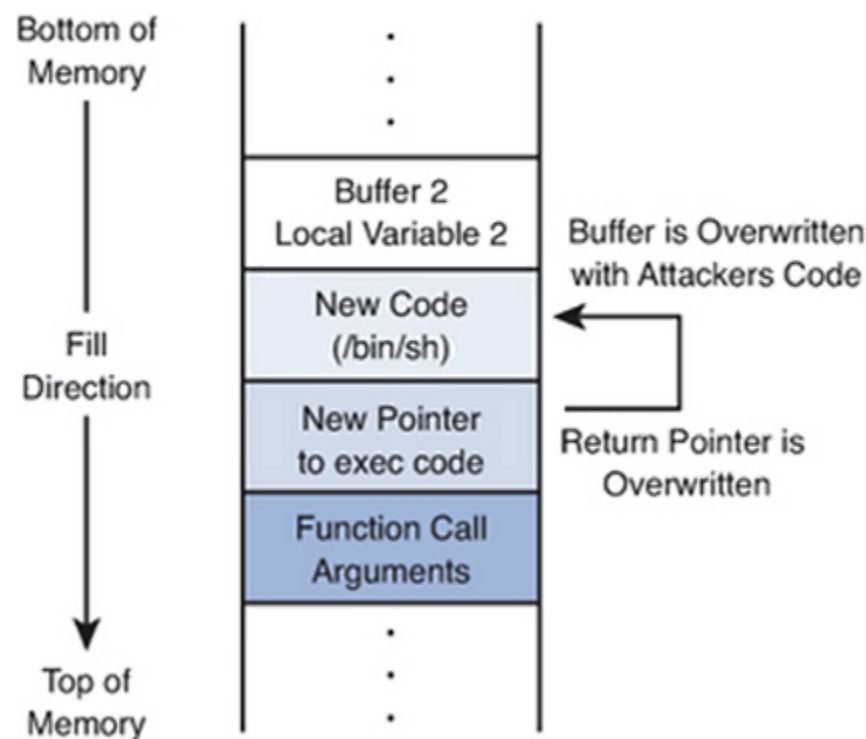
Cross-Site Request Forgery Attacks

There are different methods to mitigate CSRF Attacks include:

- Logging off web applications when not in use.
- Securing usernames and passwords.
- Not allowing browsers to remember passwords.
- Avoiding simultaneously browsing while logged into an application.

What is Buffer Overflow Attack?

A Buffer Overflow Attacks; A buffer is a temporary area for data storage. When more data (than was originally allocated to be stored) gets placed by a program or system process, the extra data overflows. It causes some of that data to leak out into other buffers, which can corrupt or overwrite.



There are different methods to mitigate buffer overflow Attacks include:

- ***The address space layout randomization (ASLR);*** moves at random around locations of data regions to *randomize address spaces*, which makes overflow attacks almost impossible.
- ***Data execution prevention;*** This method *prevents an attack from being able to run code in non-executable regions* by flagging areas of memory as executable or non-executable.

