**Lecture mapped with CO 2, CO3,CO4**

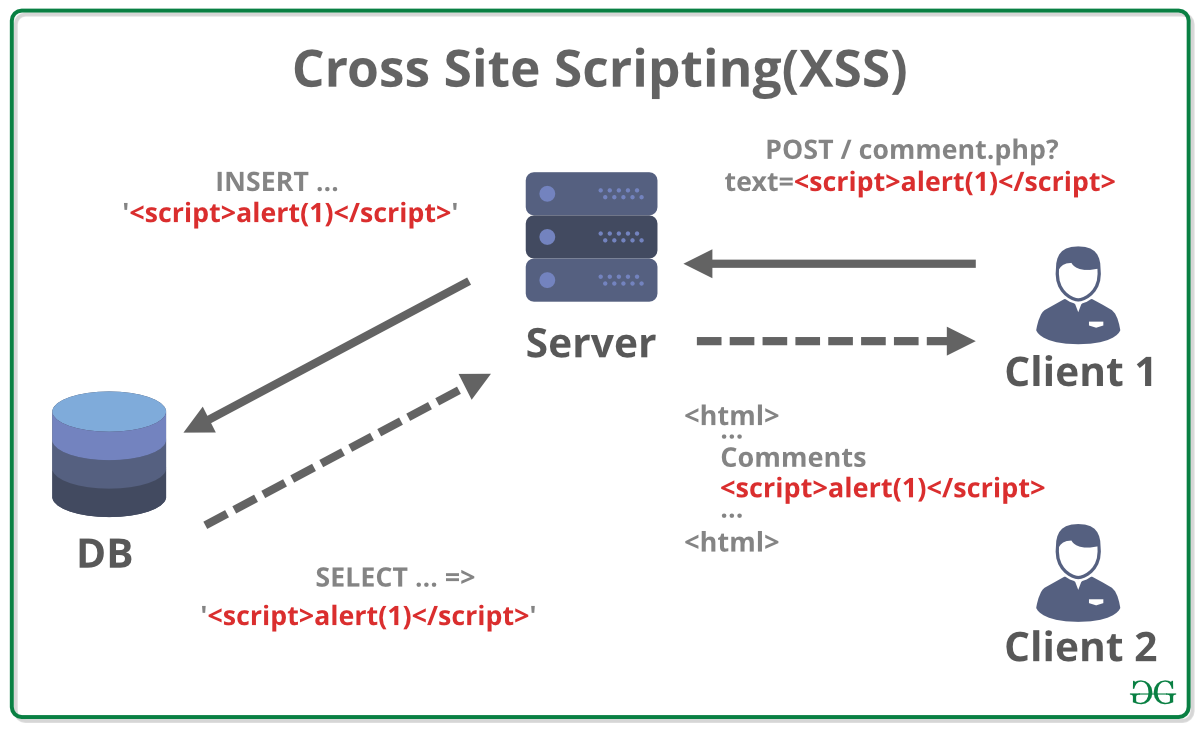
**What is Session Hijacking?**

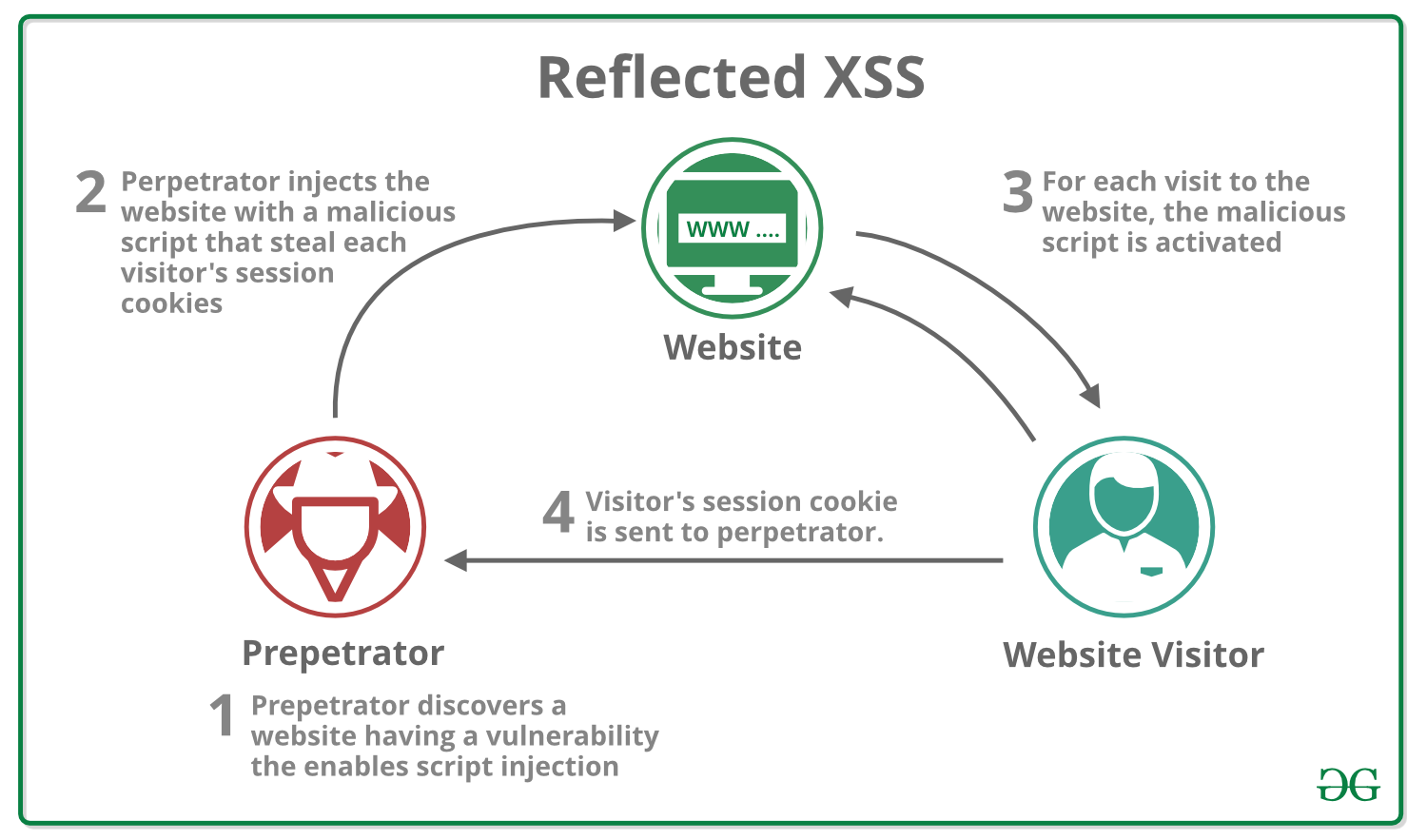
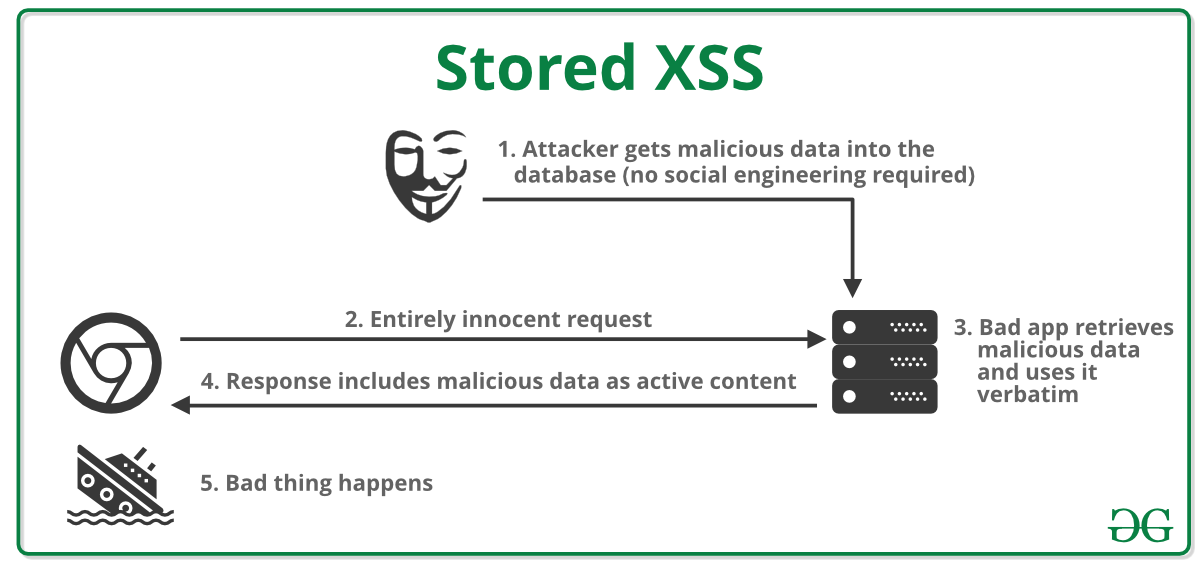
TCP session hijacking is a security attack on a user session over a protected network. The most common method of session hijacking is called IP spoofing, when an attacker uses source-routed IP packets to insert commands into an active communication between two nodes on a network and disguise itself as one of the authenticated users. This type of attack is possible because authentication typically is only done at the start of a TCP session.

Another type of session hijacking is known as a man-in-the-middle attack, where the attacker, using a sniffer, can observe the communication between devices and collect the data that is transmitted.

**Ways:**

**CSS/HTML/SQL Injection attack**

Cross Site Scripting (XSS) is a vulnerability in a web application that allows a third party to execute a script in the user’s browser on behalf of the web application. Cross-site Scripting is one of the most prevalent vulnerabilities present on the web today. The exploitation of XSS against a user can lead to various consequences such as account compromise, account deletion, privilege escalation, malware infection and many more. [](https://media.geeksforgeeks.org/wp-content/uploads/20190516152959/Cross-Site-ScriptingXSS.png) In its initial days, it was called CSS and it was not exactly what it is today. Initially, it was discovered that a malicious website could utilize JavaScript to read data from other website’s responses by embedding them in an iframe, run scripts and modify page contents. It was called CSS (Cross Site Scripting) then. The definition changed when Netscape introduced the Same Origin Policy and cross-site scripting was restricted from enabling cross-origin response reading. Soon it was recommended to call this vulnerability as XSS to avoid confusion with Cascading Style Sheets(CSS). The possibility of getting XSSed arises when a website does not properly handle the input provided to it from a user before inserting it into the response. In such a case, a crafted input can be given that when embedded in the response acts as a JS code block and is executed by the browser. Depending on the context, there are ***two types***of XSS –

1. **Reflected XSS:**If the input has to be provided each time to execute, such XSS is called reflected. These attacks are mostly carried out by delivering a payload directly to the victim. Victim requests a page with a request containing the payload and the payload comes embedded in the response as a script. An example of reflected XSS is XSS in the search field. [](https://media.geeksforgeeks.org/wp-content/uploads/20190516153002/reflectedXSS.png)
2. **Stored XSS:**When the response containing the payload is stored on the server in such a way that the script gets executed on every visit without submission of payload, then it is identified as stored XSS. An example of stored XSS is XSS in the comment thread. 

There is another type of XSS called ***DOM based XSS*** and its instances are either reflected or stored. DOM-based XSS arises when user-supplied data is provided to the DOM objects without proper sanitizing.

SQL injection is a technique used to exploit user data through web page inputs by injecting SQL commands as statements. Basically, these statements can be used to manipulate the application’s web server by malicious users.

* SQL injection is a code injection technique that might destroy your database.
* SQL injection is one of the most common web hacking techniques.
* SQL injection is the placement of malicious code in SQL statements, via web page input.

**Exploitation of SQL Injection in Web Applications**

Web servers communicate with database servers anytime they need to retrieve or store user data. SQL statements by the attacker are designed so that they can be executed while the web-server is fetching content from the application server.It compromises the security of a web application.

**Example of SQL Injection**  
Suppose we have an application based on student records. Any student can view only his or her own records by entering a unique and private student ID. Suppose we have a field like below:  
**Student id:**

**Impact of SQL Injection**  
The hacker can retrieve all the user-data present in the database such as user details, credit card information, social security numbers and can also gain access to protected areas like the administrator portal. It is also possible to delete the user data from the tables.  
Nowadays, all online shopping applications, bank transactions use back-end database servers. So in-case the hacker is able to exploit SQL injection, the entire server is compromised.

**Preventing SQL Injection**

* User Authentication: Validating input from the user by pre-defining length, type of input, of the input field and authenticating the user.
* Restricting access privileges of users and defining as to how much amount of data any outsider can access from the database. Basically, user should not be granted permission to access everything in the database.
* Do not use system administrator accounts.
* Hypertext Markup Language (HTML) injection is a technique used to take advantage of non-validated input to modify a web page presented by a web application to its users. Attackers take advantage of the fact that the content of a web page is often related to a previous interaction with users. When applications fail to validate user data, an [attacker](https://www.imperva.com/learn/application-security/ethical-hacking/) can send HTML-fomatted text to modify site content that gets presented to other users. A specifically crafted query can lead to inclusion in the web page of attacker-controlled HTML elements which change the way the application content gets exposed to the web.
* **Detailed Description**
* HTML is the language that determines how application data (like a products’ catalog) gets presented to users in their web browser. This language contains visualization commands, like the color of the page’s background and the size of embedded pictures. It also contains links to other web pages, and additional commands intended for the user’s browser. Furthermore, automated tools that collect useful information from the web on behalf of users often do so by systematically accessing and parsing the relevant information in the application’s HTML pages.
* In modern interactive web pages, the content of a web page often reflects the results of processing previous user actions. If the user’s input is not validated and the application is vulnerable, an attacker can craft and send input to the application that lets him inject pieces of his HTML code into the HTML content of the application’s response.
* **Prevention**
* The most common way of detecting HTML injection is by looking for HTML elements in the incoming HTTP stream that contains the user input. A naïve validation of user input simply removes any HTML-syntax substrings (like tags and links) from any user-supplied text. However, there are many instances where the application expects HTML input from the user. For example, this happens when the user submits visually-formatted text or text containing links to legitimate sites with related content. To avoid false positives, the [security](https://www.imperva.com/learn/application-security/application-security/) mechanism that detects possible injections and protects the application should learn in what application context user input is allowed to contain HTML. Also, it should be able to stop HTML input if it learns that such text is pasted as-is in web page generated by vulnerable application components.

**What Does Remote Attack Mean?**

A remote attack is a malicious action that targets one or a network of computers. The remote attack does not affect the computer the attacker is using. Instead, the attacker will find vulnerable points in a computer or network's security software to access the machine or system. The main reasons for remote attacks are to view or steal data illegally, introduce viruses or other malicious software to another computer or network or system, and cause damage to the targeted computer or network.

A remote attack is also known as a remote exploit.

Remote attacks are further classified into the following groups based on the tools and methods the attacker uses to compromise the targeted system.

* Domain Name System (DNS) Poisoning: Tricks the DNS server into accepting falsified data as authentic and originating from the domain owner. The false data are stored for a time, allowing the attacker time to change DNS replies to computers asking for addresses of domains. Users accessing poisoned DNS servers are redirected to websites where they unknowingly download viruses and other malicious content rather than the original content they intended.
* Transmission Control Protocol (TCP) Desynchronization: Triggered when the expected number of packets of data differs from the actual number. The unexpected packets are terminated. A hacker supplies the necessary packets with the exact sequential number. The targeted system accepts the packets, and the hacker is able to interfere with peer-to-peer or server-client communications.
* Denial of Service (DoS) Attacks: A technique that makes a server, computer or network unavailable for its users and clients by flooding it with false client requests that simulate a large usage spike. This obstructs communications between users because the server is preoccupied with large amounts of pending requests to process.
* Internet Control Message Protocol (ICMP) Attacks: An Internet protocol used by networked computers to send error messages. ICMP does not require authentication, which means that an attacker can exploit this weakness and initiate DoS attacks.
* Port Scanning: Computer ports are responsible for allowing data to be sent and received. Port scanners can help identify vulnerable data, exploit vulnerabilities and gain access to take control of computers. If a port is always open so a website can send and receive messages through it, a hacker can disguise himself as that website and gain access through that port.