

## TASK – 1

**Date: 23.08.2023**

**Name: Ramar Priya Maha Lakshmi**

**Regno: 21BCE7521**

**1. Explore Top 10 notorious Hackers in the world and summarize into their categories. (White Hat, Grey Hat, Black Hat)**

### **1. Kevin Mitnick**

A seminal figure in American hacking, Kevin Mitnick got his career start as a teen. In 1981, he was charged with stealing computer manuals from Pacific Bell. In 1982, he hacked the North American Defense Command (NORAD), an achievement that inspired the 1983 film *War Games*. In 1989, he hacked Digital Equipment Corporation's (DEC) network and made copies of their software. Because DEC was a leading computer manufacturer at the time, this act put Mitnick on the map. He was later arrested, convicted and sent to prison. During his conditional release, he hacked Pacific Bell's voicemail systems.

Throughout his hacking career, Mitnick never exploited the access and data he obtained. It's widely believed that he once obtained full control of Pacific Bell's network simply to prove it could be done.

Once a notorious **Black Hat hacker**, Mitnick turned into a cybersecurity consultant and author. He now helps organizations improve their security as a **White Hat Hacker**.



## 2. Anonymous

Anonymous got its start in 2003 on 4chan message boards in an unnamed forum. The group exhibits little organization and is loosely focused on the concept of social justice. For example, in 2008 the group took issue with the Church of Scientology and began disabling their websites, thus negatively impacting their search rankings in Google and overwhelming its fax machines with all-black images. In March 2008, a group of "Anons" marched past Scientology centers around the world wearing the now-famous Guy Fawkes mask. As noted by The New Yorker, while the FBI and other law enforcement agencies have tracked down some of the group's more prolific members, the lack of any real hierarchy makes it almost impossible to identify or eliminate Anonymous as a whole.



Anonymous is a decentralized hacking collective involved in various cyber activities, some of which are illegal. They often engage in hacktivism and have no single leader. They are characterized as **Black Hat Collective**.

## 3. Adrian Lamo

In 2001, 20-year-old Adrian Lamo used an unprotected content management tool at Yahoo to modify a Reuters article and add a fake quote attributed to former Attorney General John Ashcroft. Lamo often hacked systems and then notified both the press and his victims. In some cases, he'd help clean up the mess to improve their security. As Wired points out, however, Lamo took things too far in 2002, when he hacked The New York Times' intranet, added himself to the list of expert sources and began conducting research on high-profile public figures. Lamo earned the moniker "The Homeless Hacker" because he preferred to wander the streets with little more than a backpack and often had no fixed address.



Lamo was a **Grey Hat hacker** known for reporting security vulnerabilities to organizations, but his methods sometimes blurred ethical lines.

## 4. Albert Gonzalez

According to the New York Daily News, Gonzalez, dubbed "sounpazi," got his start as the "troubled pack leader of computer nerds" at his Miami high school. He eventually became active on criminal commerce site Shadowcrew.com and was considered one of its best hackers and moderators. At 22, Gonzalez was arrested in New York for debit card fraud related to stealing data from millions of card accounts. To avoid jail time, he became an informant for the Secret Service, ultimately helping indict dozens of Shadowcrew members.

Albert Gonzalez was a notorious Black Hat hacker who led the TJX hacker group. He masterminded one of the largest data breaches in history, stealing millions of credit card records from major retailers. His cybercrimes led to significant financial losses and security concerns for numerous organizations. Gonzalez was eventually apprehended and sentenced to prison for his hacking activities.



## 5. Matthew Bevan and Richard Pryce

Matthew Bevan and Richard Pryce also known as "Kuji" and "DataStream Cowboy" are a team of British **Black Hat hackers** who hacked into multiple military networks in 1996, including Griffiss Air Force Base, the Defense Information System Agency and the Korean Atomic Research Institute (KARI). Bevan (Kuji) and Pryce (Datastream Cowboy) have been accused of nearly starting a third world war after they dumped KARI research onto American military systems. Bevan claims he was looking to prove a UFO conspiracy theory, and according to the [BBC](#), his case bears resemblance to that of Gary McKinnon. Malicious intent or not, Bevan and Pryce demonstrated that even military networks are vulnerable.



## 6. Jeanson James Ancheta

Jeanson James Ancheta often referred to as "botmaster," had no interest in hacking systems for credit card data or crashing networks to deliver social justice. Instead, Ancheta was curious about the use of bots—software-based robots that can infect and ultimately control computer systems. Using a series of large-scale "botnets," he was able to compromise more than 400,000 computers in 2005. According to Ars Technica, he then rented these machines out to advertising companies and was also paid to directly install bots or adware on specific systems. Ancheta was sentenced to 57 months in prison. This was the first time a hacker was sent to jail for the use of botnet technology.



Ancheta's botnets were used to carry out various cybercriminal activities, including distributed denial-of-service (DDoS) attacks, sending spam emails, and distributing malware and is categorized as **Black Hat Hacker**.

## 7. Kevin Poulsen

Kevin Poulsen, also known as the "Dark Dante," is a former **Black Hat hacker** turned journalist. In his earlier years, he engaged in hacking activities, including compromising phone systems and computers. Poulsen gained notoriety for his hacking skills and was even featured on the FBI's wanted list.

However, after serving a prison sentence, Poulsen turned his life around and became a respected journalist. He now covers cybersecurity and technology-related topics. Poulsen's transformation from hacker to journalist is a notable example of how individuals can change their paths and use their knowledge for positive purposes in the field of cybersecurity.



## 8. Jonathan James

Jonathan James, also known as "c0mrade," was a **Black Hat hacker** known for his involvement in high-profile cybercrimes. At the age of 15, he breached various U.S. government systems, including NASA's, causing damage to critical infrastructure. James's actions led to significant concerns about the security of sensitive data and government networks

In 2008, James committed suicide by gunshot. According to the Daily Mail, his suicide note stated, "I have no faith in the 'justice' system. Perhaps my actions today, and this letter, will send a stronger message to the public. Either way, I have lost control over this situation, and this is my only way to regain control."



## 9. Michael Calce

In February 2000, 15-year-old Michael Calce, also known as "Mafiaboy," was a **Black Hat Hacker** discovered how to take over networks of university computers. He used their combined resources to disrupt the number-one search engine at the time: Yahoo. Within one week, he'd also brought down Dell, eBay, CNN and Amazon using a distributed-denial-of-service (DDoS) attack that overwhelmed corporate servers and caused their websites to crash. Calce's wake-up call was perhaps the most jarring for cyber crime investors and internet proponents. If the biggest websites in the world—valued at over \$1 billion—could be so easily sidelined, was any online data truly safe? It's not an exaggeration to say that the development of cyber crime legislation suddenly became a top government priority thanks to Calce's hack.





## 10. ASTRA

This hacker differs from the others on this list in that he has never been publicly identified. However, according to the Daily Mail, some information has been released about ASTRA. Namely that he was apprehended by authorities in 2008, and at that time he was identified as a 58-year-old Greek mathematician. Reportedly, he had been hacking into the Dassault Group, for almost half a decade. During that time, he stole cutting edge weapons technology software and data which he then sold to 250 individuals around the world. His hacking cost the Dassault Group \$360 million in damages. No one knows why his complete identity has never been revealed, but the word 'ASTRA' is a Sanskrit word for 'weapon'.

ASTRA is categorized as **Black Hat Hacker**.



## **TASK - 2**

**Date: 24.08.2023**

**NAME: RAMAR PRIYA MAHA LAKSHMI**

**REGNO: 21BCE7521**

### **1. Determine the vulnerabilities in the open ports**

**Port no's (20,21,22,23,25,53,69,80,110,123,143,443)**

#### **Port 20 (FTP Data) and 21 (FTP Control):**

**Vulnerabilities:** Weak FTP credentials, anonymous access, and outdated FTP servers susceptible to known exploits.

**Effects:** Weak FTP credentials can be easily guessed or brute-forced by attackers, leading to unauthorized access to files on the server. Anonymous access without authentication allows anyone to access and potentially modify or steal files. Outdated FTP servers may have known security vulnerabilities that attackers can exploit to compromise the server's integrity and confidentiality.

**Example:** An FTP server on Port 21 allows anonymous access and uses weak passwords. An attacker exploits this by gaining unauthorized access, potentially uploading malicious files to the server or stealing sensitive data.

#### **Port 22 (SSH):**

**Vulnerabilities:** Weak SSH key pairs, outdated SSH software versions with known vulnerabilities, or exposed SSH services to the public internet.

**Effects:** Weak SSH key pairs can be cracked, allowing attackers to gain unauthorized access to the server. Outdated SSH software versions may have known vulnerabilities that can be exploited to compromise the server's

security. Exposing SSH services to the public internet without proper security measures increases the risk of brute force attacks and unauthorized access.

**Example:** A server on Port 22 uses an outdated SSH version with known vulnerabilities. An attacker exploits one of these vulnerabilities to gain unauthorized access to the server, potentially compromising its security.

### **Port 23 (Telnet):**

**Vulnerabilities:** Unencrypted communication, weak passwords, and potential unauthorized access to the system.

**Effects:** Unencrypted Telnet communication can be intercepted by attackers, exposing login credentials and potentially sensitive data. Weak passwords can be easily guessed or brute-forced, leading to unauthorized access to the system. Unauthorized access may result in data theft, system compromise, or misuse of resources.

**Example:** Telnet communication over Port 23 is unencrypted, allowing an attacker to intercept sensitive information, including login credentials, as it traverses the network.

### **Port 25 (SMTP):**

**Vulnerabilities:** Open relays, email abuse, and email spoofing vulnerabilities.

**Effects:** Open SMTP relays can be exploited by spammers to send massive volumes of unsolicited emails, causing email abuse and potentially leading to the blacklisting of the server's IP address. Email spoofing vulnerabilities allow attackers to impersonate legitimate senders, potentially leading to phishing attacks and email-based scams.

**Example:** An SMTP server on Port 25 is misconfigured as an open relay, enabling spammers to use it to send massive volumes of unsolicited emails, leading to email abuse and potential blacklisting.

### **Port 53 (DNS):**



**Vulnerabilities:** Misconfigured DNS servers, allowing DNS amplification attacks or DNS cache poisoning.

**Effects:** Misconfigured DNS servers can be abused by attackers to conduct DNS amplification attacks, overwhelming the server with traffic and causing downtime. DNS cache poisoning can lead to malicious redirection of users to fraudulent websites, facilitating various cyberattacks, including phishing.

**Example:** A DNS server on Port 53 has a misconfiguration that makes it susceptible to DNS cache poisoning. An attacker exploits this vulnerability to redirect users to malicious websites by poisoning the DNS cache.

### **Port 69 (TFTP):**

**Vulnerabilities:** Limited security features, potential unauthorized access, and file transfer of sensitive files.

**Effects:** TFTP's limited security features make it vulnerable to unauthorized access and file transfers. Attackers can exploit this to access sensitive configuration files, potentially leading to system compromise or unauthorized access to critical network resources.

**Example:** TFTP service on Port 69 allows unauthenticated access and file transfer, leading to unauthorized access and potential exposure of sensitive configuration files.

### **Port 80 (HTTP):**

**Vulnerabilities:** Common web application vulnerabilities like SQL injection, cross-site scripting (XSS), and server misconfigurations.

**Effects:** SQL injection can allow attackers to manipulate databases, steal data, or execute unauthorized actions on a web application. Cross-site scripting (XSS) can enable attackers to inject malicious scripts into web pages viewed by other users, potentially leading to data theft or compromised user accounts. Server misconfigurations may expose sensitive information or grant unauthorized access to system resources.

**Example:** A web application on Port 80 fails to validate user input, allowing an attacker to execute a cross-site scripting (XSS) attack, which can steal user data and compromise the security of the site.

### **Port 110 (POP3):**

**Vulnerabilities:** Weak POP3 credentials, unencrypted communication, and potential email theft.

**Effects:** Weak POP3 credentials can be easily guessed or brute-forced by attackers, leading to unauthorized access to users' email accounts. Unencrypted communication can expose email content to eavesdropping. Unauthorized access or email theft can result in data breaches or the compromise of sensitive information.

**Example:** An email server on Port 110 allows unencrypted access, and weak credentials are used. An attacker can intercept login credentials and potentially access or steal emails.

### **Port 123 (NTP):**

**Vulnerabilities:** Use in DDoS attacks as an amplifier or susceptibility to NTP reflection attacks if misconfigured.

**Effects:** NTP servers can be exploited in Distributed Denial of Service (DDoS) attacks, amplifying the attack traffic and causing service disruption for others. Misconfigured NTP servers can also become part of NTP reflection attacks, redirecting attack traffic to victim targets and overloading their resources.

**Example:** An NTP server on Port 123 is misconfigured to respond to NTP reflection requests from unauthorized sources, allowing an attacker to amplify a DDoS attack using the server's resources.

### **Port 143 (IMAP):**

**Vulnerabilities:** Weak IMAP credentials, unencrypted communication, and potential unauthorized access to emails.

**Effects:** Weak IMAP credentials can be compromised, allowing unauthorized access to users' email accounts. Unencrypted communication can expose email content to interception by attackers. Unauthorized access to emails can lead to data breaches or the misuse of sensitive information.

**Example:** An IMAP server on Port 143 allows unencrypted connections and uses weak authentication, enabling an attacker to intercept login credentials and access users' email accounts.

### **Port 443 (HTTPS):**

**Vulnerabilities:** SSL/TLS vulnerabilities, outdated encryption protocols, and common web application vulnerabilities if running web services.

**Effects:** SSL/TLS vulnerabilities can be exploited to intercept or manipulate encrypted data, compromising confidentiality. Outdated encryption protocols may lack security features, increasing the risk of data exposure. Common web application vulnerabilities, if present, can be exploited to compromise web servers, leading to data breaches or unauthorized access.

**Example:** A web server on Port 443 uses an outdated SSL/TLS protocol, making it susceptible to known vulnerabilities like Heartbleed, which could allow an attacker to steal sensitive data from the server.

### **Mitigation Strategies:**

Mitigation strategies for the vulnerabilities associated with open ports depend on the specific risks and services running on those ports. Here are general mitigation strategies:

- 

- **Strong Authentication and Access Control:**

Implement strong authentication methods to prevent unauthorized access. Use complex, unique passwords or consider multi-factor authentication (MFA).

Employ access control mechanisms to restrict access to only authorized users or IP addresses.

- **Encryption:**

Enable encryption for sensitive data transmission. Use protocols like SSL/TLS for web services (Port 443) and secure variants of other protocols like SSH (Port 22) and IMAPS (Port 143).

- **Regular Patching and Updates:**

Keep all software, including operating systems and applications, up to date with the latest security patches to address known vulnerabilities.

- **Firewalls and Intrusion Detection/Prevention Systems (IDPS):**

Deploy firewalls to filter incoming and outgoing traffic, allowing only necessary services and blocking potential threats.

Use IDPS to detect and block malicious activities, including intrusion attempts.

- **Network Segmentation:**

Isolate critical systems and networks to limit the impact of a security breach. Segmenting networks can prevent lateral movement by attackers.

- 

### **Security Audits and Vulnerability Scanning:**

Regularly conduct security audits and vulnerability assessments to identify and remediate weaknesses in system configurations and applications.

- **Security Best Practices:**

Adhere to security best practices for each service running on open ports. For example, configure DNS servers (Port 53) to prevent open recursion and cache poisoning.

- **Application Security:**

Implement secure coding practices for web applications (Port 80) to prevent common vulnerabilities like SQL injection and cross-site scripting (XSS).

- **Logging and Monitoring:**

Set up logging and monitoring systems to detect and respond to suspicious activities on open ports promptly.

- **Access Controls for File Transfers (TFTP - Port 69):**

Restrict TFTP access to authorized users or trusted hosts.

Limit file transfer permissions and ensure sensitive files are not accessible via TFTP.

- **DDoS Mitigation (NTP - Port 123):**

Implement DDoS mitigation strategies, including rate limiting and traffic filtering, to prevent NTP reflection attacks.

- 

### **Regular Backup and Recovery:**

Maintain regular backups of critical data and configurations to facilitate recovery in case of a security incident.

- **Penetration Testing and Red Teaming:**

Conduct penetration tests and red team exercises to simulate attacks and identify vulnerabilities proactively.

Mitigation strategies should be tailored to the specific services and vulnerabilities associated with each open port. Regular security assessments, monitoring, and a proactive approach to security are key to reducing risks and ensuring a robust defense against potential threats.



# TASK – 3

Date: 25.08.2023

NAME: RAMAR PRIYA MAHA LAKSHMI

REGNO: 21BCE7521

## 1. Top 5 OWASP CWE description with Business Impact.

### 1. CWE: CWE-284: Improper Access Control

**OWASP CATEGORY: A01 2021 Broken Access Control**

**DESCRIPTION:** The product does not restrict or incorrectly restricts access to a resource from an unauthorized actor.

**BUSINESS IMPACT:** can harm businesses by enabling unauthorized users to access sensitive data, causing data breaches, financial loss, reputation damage, legal issues, operational disruptions, intellectual property theft, and increasing insider threat risks.

Web Security Academy > Access control > Lab

Lab: Unprotected admin functionality with unpredictable URL

APPRENTICE

LAB

Not solved

This lab has an unprotected admin panel. It's located at an unpredictable location, but the location is disclosed somewhere in the application.

Solve the lab by accessing the admin panel, and using it to delete the user `carlos`.

ACCESS THE LAB

Solution


Community solutions

n functionality with unpredictable URL

LAB Not solved

Home | My account

E LIKE TO SHOP



```

var isAdmin = false;
if (isAdmin) {
  var topLinksTag = document.getElementsByClassName("top-
Links")[0];
  var adminPanelTag = document.createElement('a');
  adminPanelTag.setAttribute('href', "/admin-52sbgl");
  adminPanelTag.innerText = 'Admin panel';
  topLinksTag.append(adminPanelTag);
  var pTag = document.createElement('p');
  pTag.innerText = 'I';
  topLinksTag.appendChild(pTag);
}

```

admin

admin

Back to lab description >>

## Users

wiener - [Delete](#)

carlos - [Delete](#)

User deleted successfully!

## Users

wiener - [Delete](#)

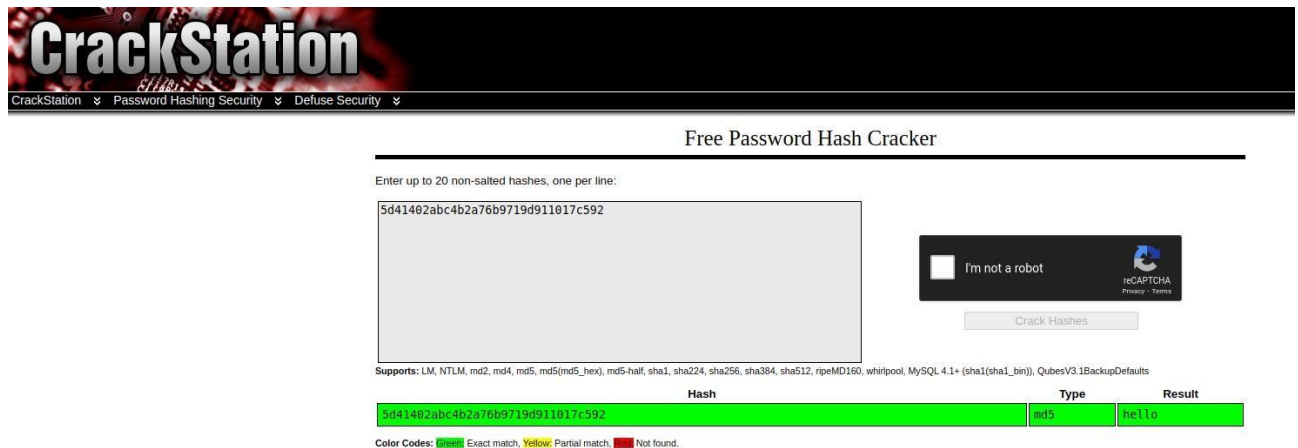
- Improper access controls can help attacker in altering or permanently deleting an existing user.

## 2. CWE: CWE-327: Use of a Broken or Risky Cryptographic Algorithm

### OWASP CATEGORY: A02 2021 Cryptographic Failures

**DESCRIPTION:** The product uses a broken or risky cryptographic algorithm or protocol.

**BUSINESS IMPACT:** can seriously impact businesses by compromising data security, leading to potential breaches, loss of trust, regulatory penalties, and operational disruptions.



The screenshot shows the CrackStation website's 'Free Password Hash Cracker' interface. At the top, there's a navigation bar with 'CrackStation', 'Password Hashing Security', and 'Defuse Security'. The main heading is 'Free Password Hash Cracker'. Below it, a text input field contains the hash '5d41402abc4b2a76b9719d911017c592'. To the right of the input field is a reCAPTCHA widget with the text 'I'm not a robot' and a 'Crack Hashes' button. Below the input field, a list of supported hash types is shown: LM, NTLM, md2, md4, md5, md5(md5\_hex), md5-half, sha1, sha224, sha256, sha384, sha512, rpeMD160, whirlpool, MySQL 4.1+ (sha1 sha1\_bin), QubesV3.1BackupDefaults. A table below this shows the results of the hash cracking process. The table has three columns: Hash, Type, and Result. The first row shows the hash '5d41402abc4b2a76b9719d911017c592' with Type 'md5' and Result 'hello'. Below the table, a color code legend indicates: Green for Exact match, Yellow for Partial match, and Red for Not found.

| Hash                             | Type | Result |
|----------------------------------|------|--------|
| 5d41402abc4b2a76b9719d911017c592 | md5  | hello  |

### 3. CWE: CWE-564: SQL Injection: Hibernate

#### OWASP CATEGORY: A03 2021 Injection

**DESCRIPTION:** Using Hibernate to execute a dynamic SQL statement built with user-controlled input can allow an attacker to modify the statement's meaning or to execute arbitrary SQL commands.

**BUSINESS IMPACT:** allows attackers to manipulate database queries through vulnerabilities in Hibernate, potentially leading to unauthorized data access, breaches, financial losses, legal issues, and reputation damage.

#### Lab: SQL injection vulnerability allowing login bypass

APPRENTICE

LAB

Not solved

This lab contains a [SQL injection](#) vulnerability in the login function.

To solve the lab, perform a SQL injection attack that logs in to the application as the `administrator` user.

ACCESS THE LAB

Solution

Community solutions

## Login

Username

administrator'--

Password

\*\*\*\*\*

Log in

## My Account

Your username is: administrator

Email

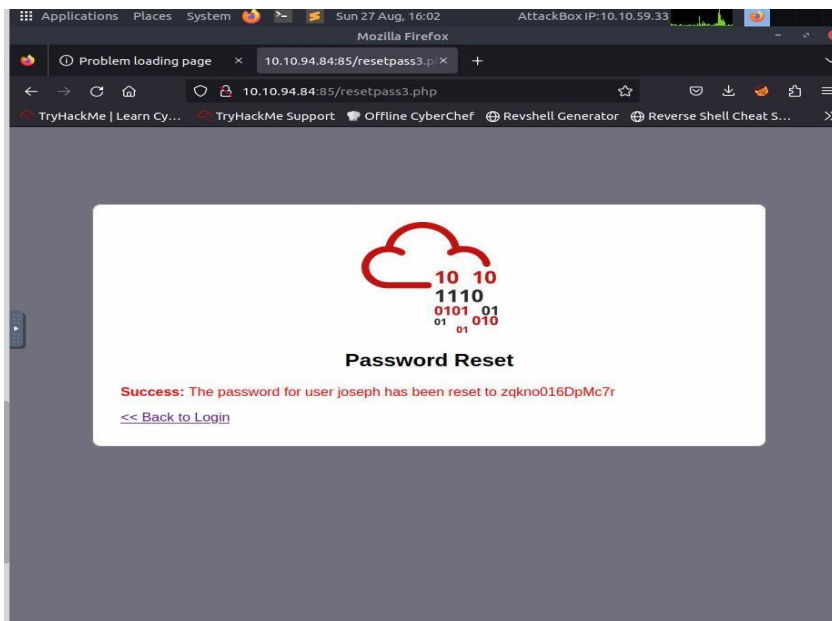
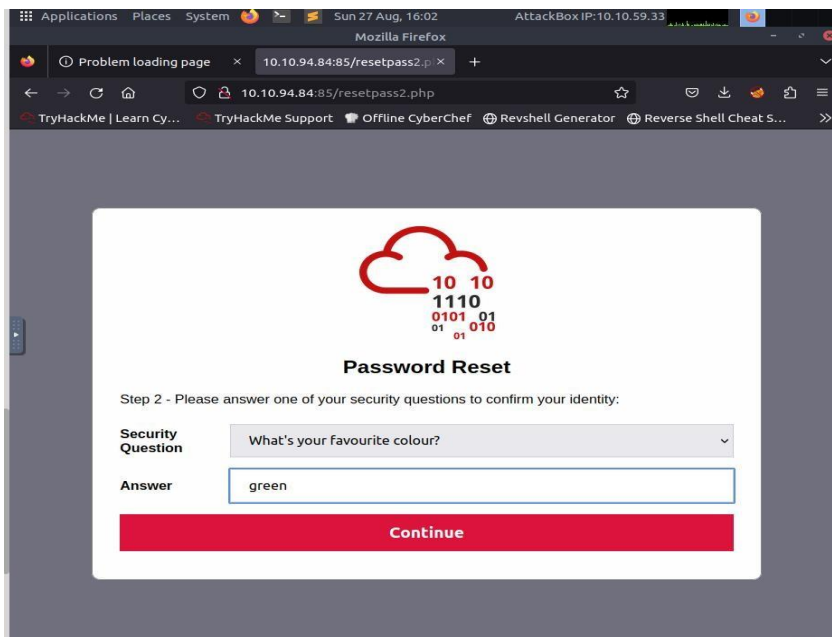
Update email

- SQL Statements can be used to modify the table and gain access to users' data.

## 4. CWE: CWE-657: Violation of Secure Design Principles OWASP CATEGORY: A04 2021 Insecure Design

**DESCRIPTION:** The product violates well-established principles for secure design.

**BUSINESS IMPACT:** has substantial business implications due to its potential to result in vulnerabilities and weak security structures. This can lead to data breaches, financial losses, reputational damage, and regulatory penalties.



- Violation of security design principles includes poor security designing.
- The chances of an attacker stealing passwords is higher when poor security related questions are set.

## 5. CWE: CWE-319: Cleartext Transmission of Sensitive Information

**OWASP CATEGORY: A05 2021 Security Misconfiguration**  
**DESCRIPTION:** The product transmits sensitive or securitycritical

data in cleartext in a communication channel that can be sniffed by unauthorized actors.

**BUSINESS IMPACT:** can harm businesses by exposing confidential data during transmission, leading to data breaches, compromised customer trust, regulatory violations, and potential legal consequences.



- Websites with http allows attackers to steal sensitive information.
- In the absence of an SSL certificate, all your communications are not encrypted at all. Meaning attackers have access to all the data.
- Sensitive Information must only be channelled through HTTPS communications.



# **TASK**

**Date: 28.08.2023**

**NAME: RAMAR PRIYA MAHA LAKSHMI**

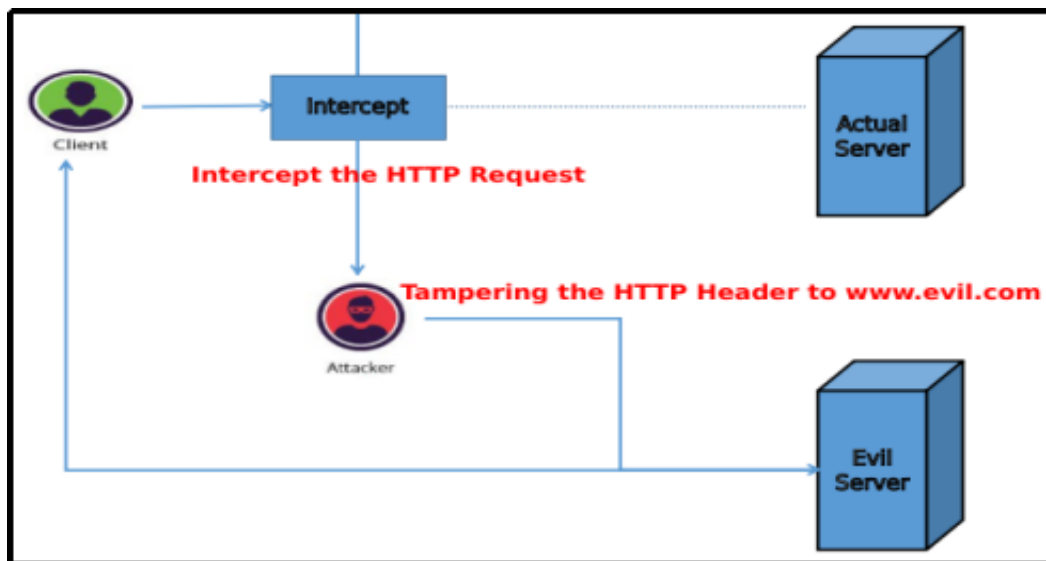
**REGNO: 21BCE7521**

**1. Understanding any Top 10 web applications Vulnerabilities (other than Top 10 OWASP) write a paragraph about that and add an image to the respective vulnerability**

## **a. HTTP Header Injection:**

HTTP Header Injection is a web vulnerability that allows attackers to manipulate HTTP response headers sent by a web application to a user's browser. By injecting malicious content into these headers, attackers can set cookies, redirect users to harmful websites, inject malware, or bypass security mechanisms. For example, they might forge headers to impersonate a trusted source, leading to potential data breaches or phishing attacks. Proper input validation and output encoding are essential to prevent this type of injection, ensuring the integrity and security of HTTP headers and the overall web application.

**Mitigation:** Mitigate HTTP Header Injection by sanitizing and validating user inputs and using web security libraries or frameworks that automatically encode or escape header content. Regularly update and patch web servers and applications to address known vulnerabilities related to headers.



## b. Content Spoofing:

Content spoofing, also known as content injection or text injection, is a web security vulnerability that allows attackers to manipulate the content displayed to users on a website. This can involve injecting malicious or misleading information, such as fake messages or phishing links, into web pages. Content spoofing attacks can erode trust, mislead users, and lead to various malicious actions. To mitigate this vulnerability, web developers must implement strict input validation, output encoding, and security headers to ensure that content is accurately and securely rendered, helping to protect users from deceptive or harmful content. Regular security testing and code reviews are crucial to identifying and addressing content spoofing risks effectively.

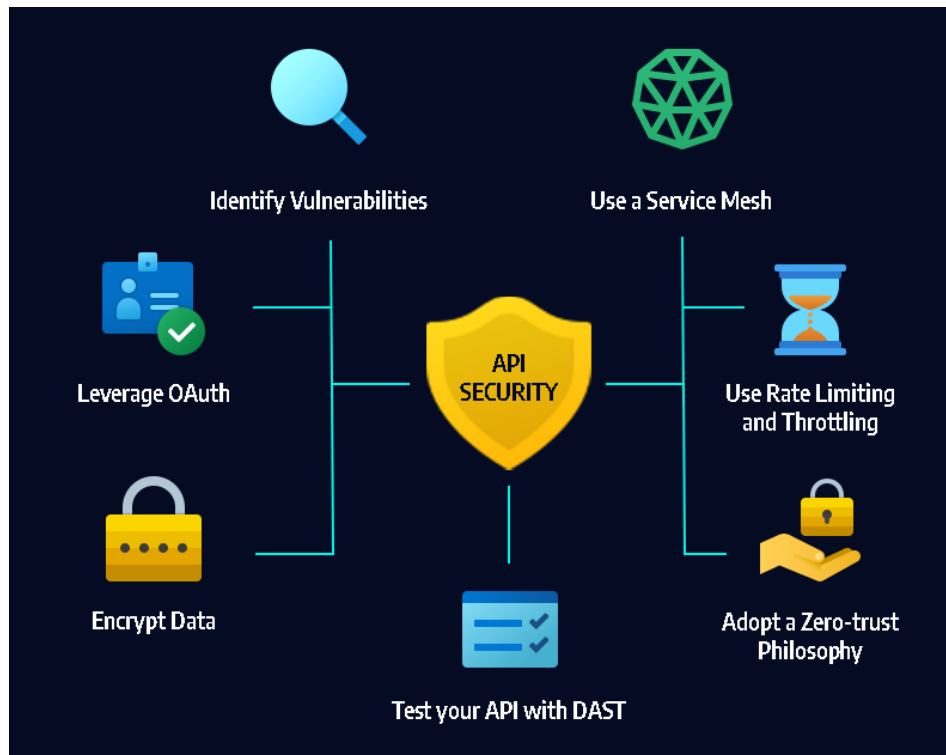
**Mitigation:** Mitigate content spoofing by implementing rigorous input validation and encoding of user-generated content and by employing strict security headers to prevent malicious injection. Regular security testing and code reviews are essential to identify and address vulnerabilities proactively.



### c. API Security Vulnerabilities:

API Security Vulnerabilities refer to weaknesses and risks associated with the use of Application Programming Interfaces (APIs) in web and mobile applications. These vulnerabilities can expose sensitive data, compromise user accounts, or enable malicious activities. Common API security issues include insufficient authentication, excessive data exposure, and lack of rate limiting, among others. To mitigate API security vulnerabilities, developers should implement strong authentication and authorization mechanisms, validate user inputs, monitor and limit API usage, and conduct thorough security testing and code reviews to identify and address potential weaknesses proactively. Additionally, using API security tools and frameworks can help bolster the security of APIs and prevent exploitation by malicious actors.

**Mitigation:** Mitigating API security vulnerabilities includes implementing strong authentication, input validation, rate limiting, conducting security testing, and maintaining documentation for ongoing security awareness.



#### d. Predictable Resource Location:

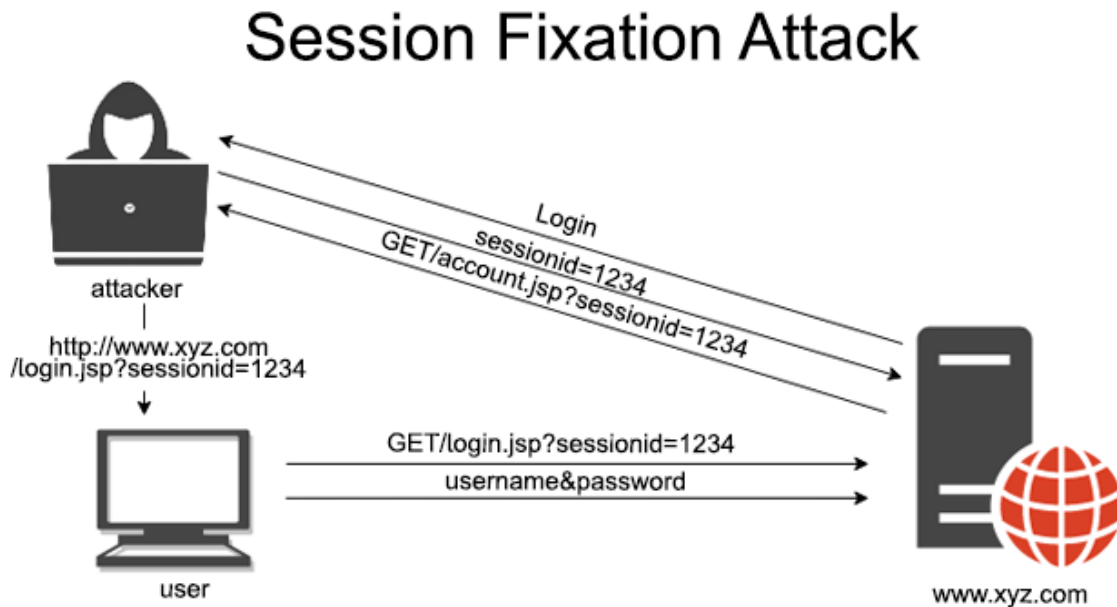
Predictable Resource Location is a web application vulnerability where URLs or resource identifiers follow a predictable pattern or are easily guessable. Attackers can exploit this by predicting or brute-forcing resource locations, gaining unauthorized access to sensitive data or functionality.

**Mitigation:** Mitigation involves implementing strong access controls, employing unpredictable resource naming conventions, and using proper authentication and authorization mechanisms to prevent this type of attack.

#### e. Session Fixation:

Session Fixation is a web application vulnerability where an attacker sets a user's session identifier (e.g., a session cookie) before the user logs in, essentially forcing the victim to use a predetermined session. Once the victim logs in, the attacker can hijack their session, potentially gaining unauthorized access to the victim's account.

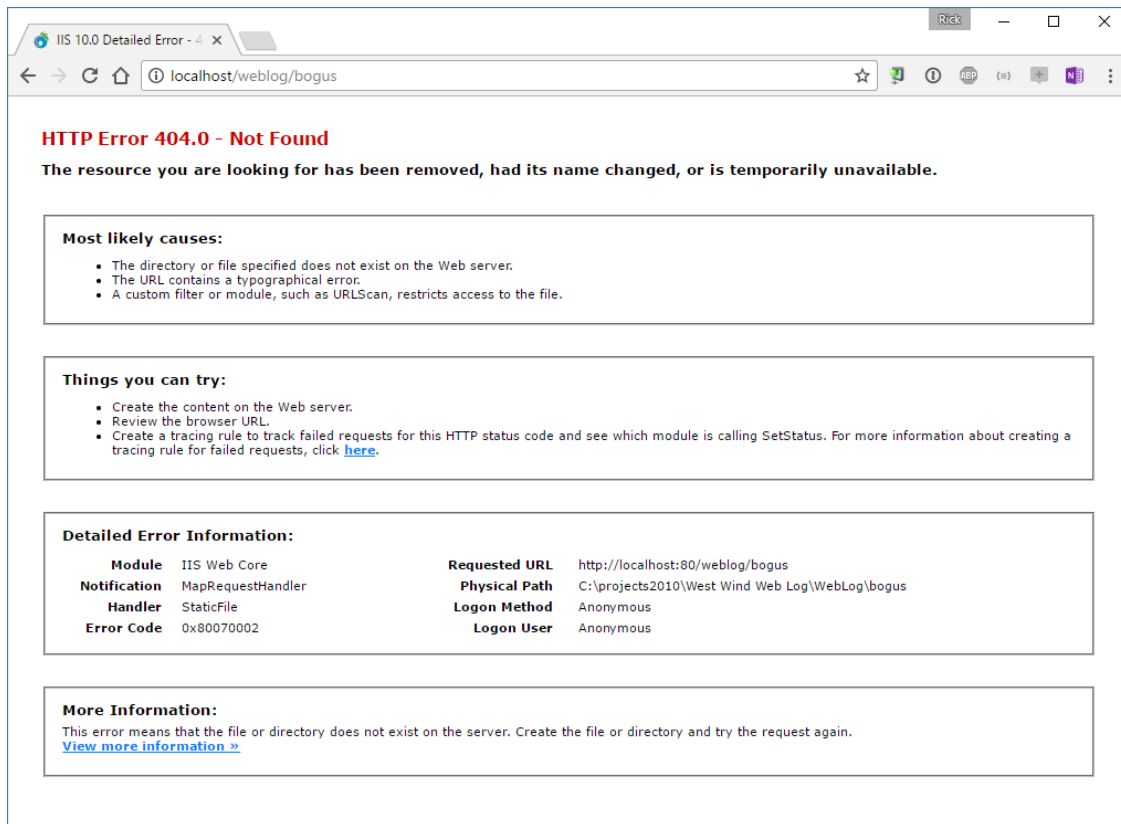
**Mitigation:** Mitigation involves regenerating session identifiers upon login and employing strong session management practices to prevent this type of attack.



### f. Inadequate Error Handling

Inadequate Error Handling is a web application vulnerability where error messages or responses reveal sensitive information or internal details about the application's structure and functionality. Attackers can exploit this information to understand the system's weaknesses and potentially launch targeted attacks.

**Mitigation:** Mitigation involves implementing custom error handling to provide generic error messages to users while logging detailed error information securely for system administrators to review. This prevents the exposure of sensitive data and enhances overall security.

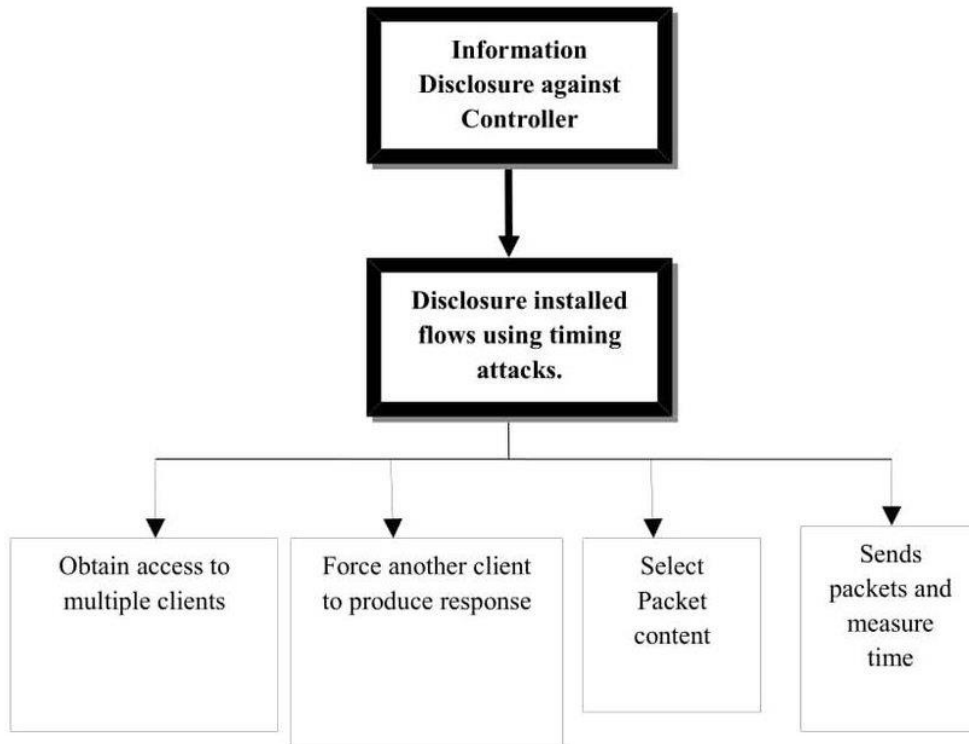


## g. Information Disclosure:

Information Disclosure is a web application vulnerability where sensitive data or system details are inadvertently revealed to users or attackers. This may include exposing database error messages, internal file paths, or confidential user information. Attackers can exploit such disclosures to gather intelligence for targeted attacks.

**Mitigation:** Mitigation strategies involve implementing robust access controls, error handling, and input validation to prevent unauthorized access to sensitive information and to ensure that error messages and responses reveal minimal details about the application's internal workings.





## h. Missing Function-Level Access:

Missing Function-Level Access Control is a web application vulnerability where users can access functions or features they shouldn't have permission to use. Attackers can exploit this to gain unauthorized access to sensitive areas of the application or perform actions reserved for privileged users.

**Mitigation:** Mitigation involves implementing proper access controls and authorization checks at both the front-end and back-end levels to ensure that users can only access functions or resources they are authorized to use, preventing unauthorized actions and data exposure.

In a role-based access control scheme, a role represents a set of access permissions and privileges. A user can be assigned one or more roles. A role-based access control scheme normally consists of two parts: role permission management and role assignment. A broken role-based access control scheme might allow a user to perform accesses that are not allowed by his/her assigned roles, or somehow allow privilege escalation to an unauthorized role.

**General Goal(s):**

Each user is a member of a role that is allowed to access only certain resources. Your goal is to explore the access control rules that govern this site. Only the [Admin] group should have access to the 'Account Manager' resource.

**Congratulations. You have successfully completed this lesson.**

\* User Larry [User, Manager] was allowed to access resource Account Manager

Change user: Larry ▼

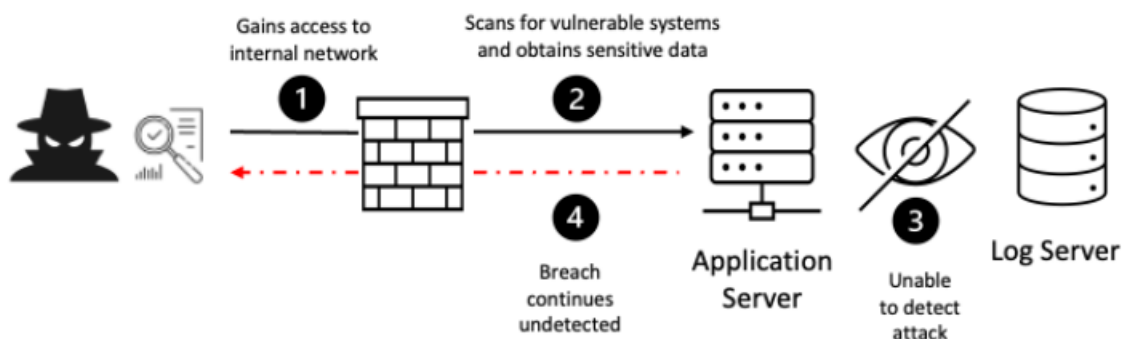
Select resource: Account Manager ▼

Check Access

## i. Insufficient Logging and Monitoring:

Insufficient Logging and Monitoring is a web application vulnerability where an application fails to adequately record and monitor security events and user activities. This lack of visibility makes it difficult to detect and respond to security incidents and suspicious activities in real-time. Attackers can exploit this by operating under the radar and carrying out malicious actions undetected. Mitigation requires implementing comprehensive logging of relevant security events, setting up real-time monitoring, and establishing incident response processes to swiftly identify and respond to security threats, enhancing overall application security.

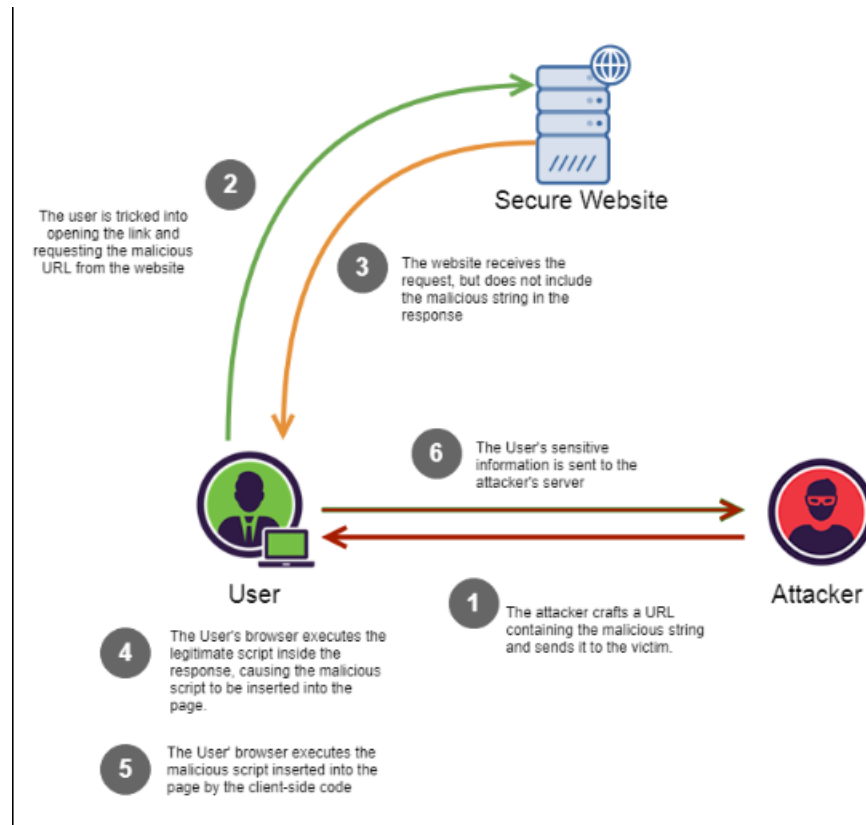
**Mitigations:** Mitigate Insufficient Logging and Monitoring by implementing comprehensive logging of security events, including authentication, access, and error logs. Set up real-time monitoring alerts to detect and respond to suspicious activities promptly, and establish incident response procedures to address security incidents effectively. Regularly review and refine logging and monitoring practices to stay ahead of evolving threats.



## j. DOM-Based Cross Site Scripting (DOM-XSS)

DOM-Based Cross-Site Scripting (DOM XSS) is a web application vulnerability where client-side JavaScript code manipulates the Document Object Model (DOM) to inject malicious scripts. Unlike traditional XSS attacks, DOM XSS occurs when the browser interprets user-supplied data in the DOM, making it challenging to detect and prevent. Attackers exploit this to execute malicious scripts within a user's browser, potentially stealing data or hijacking user sessions.

**Mitigation:** Mitigation involves thorough input validation and output encoding, using security libraries, and educating developers about secure DOM manipulation to prevent these types of attacks.



## **TASK - 5**

**DATE: 29.08.2023**

**NAME: RAMAR PRIYA MAHA LAKSHMI**

**REGNO: 21BCE7521**

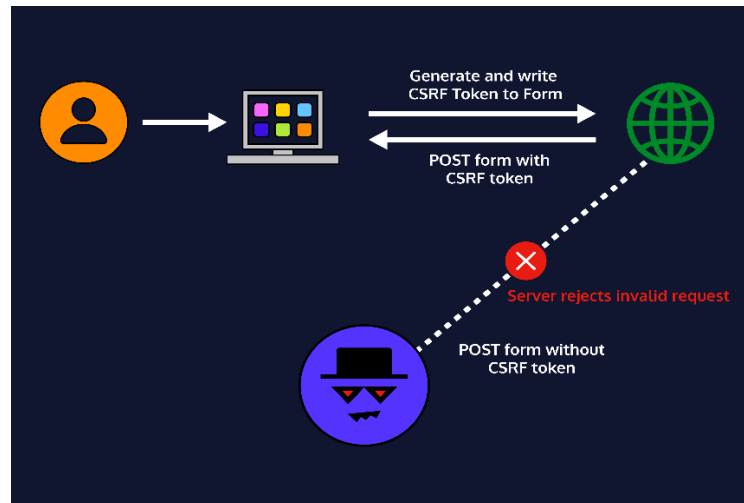
### **1. Web Server Attacks**

#### **What is Web Server attack?**

A Web Server attack refers to the malicious action directed towards a web server with the intent to exploit vulnerabilities, gain unauthorized access, disrupt services, or compromise the security of the server and the application it hosts.

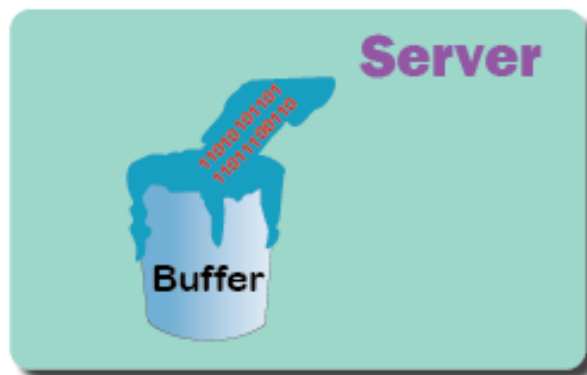
**1. Cross-Site Request Forgery (CSRF):** Attackers trick users into unknowingly performing actions on a different site, often causing unintended changes on the victim's account.

**Example:** Forcing a logged-in user to unknowingly change their password by clicking a link.



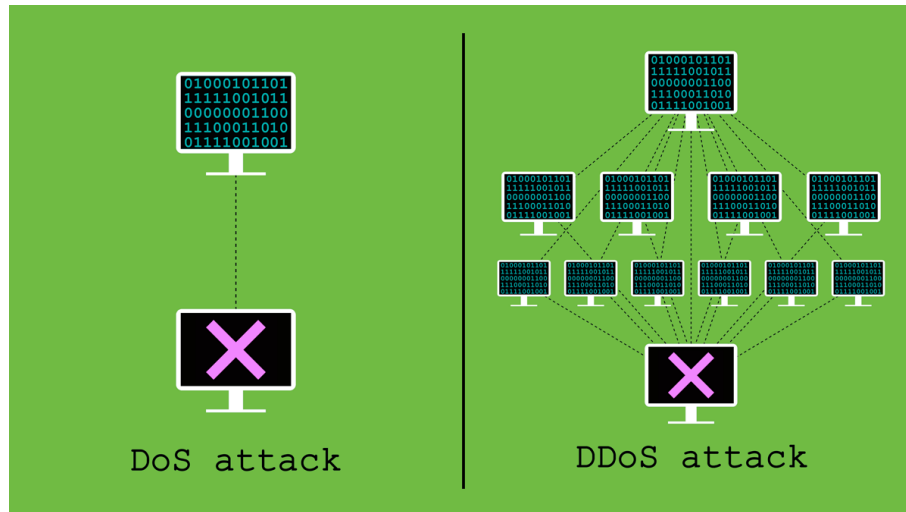
**2. Buffer Overflow Attacks:** Attackers send more data than a server's buffer can handle, causing it to overwrite adjacent memory and potentially execute malicious code.

**Example:** Sending excessive data to a vulnerable application, causing it to crash or execute malicious code.



**3. Denial of Service (DoS) and Distributed DoS (DDoS):** Attackers flood a server with excessive traffic or requests, overwhelming its resources and rendering it unavailable to legitimate users. In DDoS attacks, multiple compromised systems are used.

**Example:** Flooding the server with an excessive amount of traffic or request.



**4. SQL Injection (SQLi):** Attackers exploit improper input validation by injecting malicious SQL queries to user input fields. This can lead to unauthorized access, data manipulation, or even complete control over a database.

**Example:** Inputting `'OR '1'='1` into login form to bypass authentication.

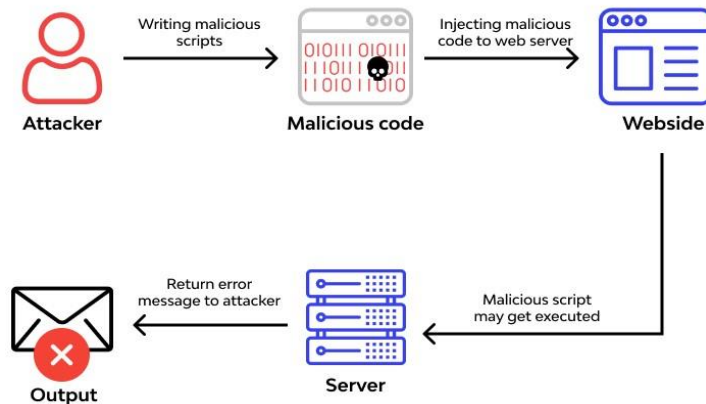


**5. Remote Code Execution (RCE):** Attackers exploit vulnerabilities to run arbitrary code on a server,



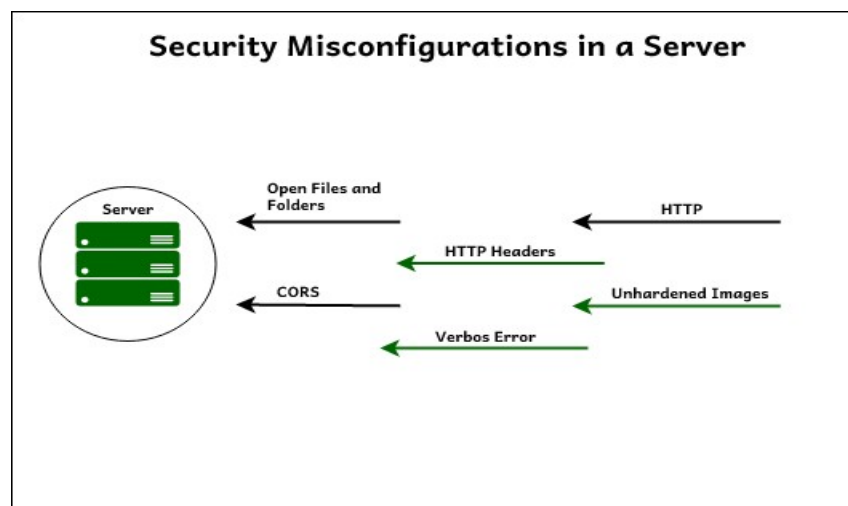
potentially gaining control over the entire system. This is especially dangerous when combined with poor server security.

**Example:** Uploading a file containing malicious code and getting it executed on the server.



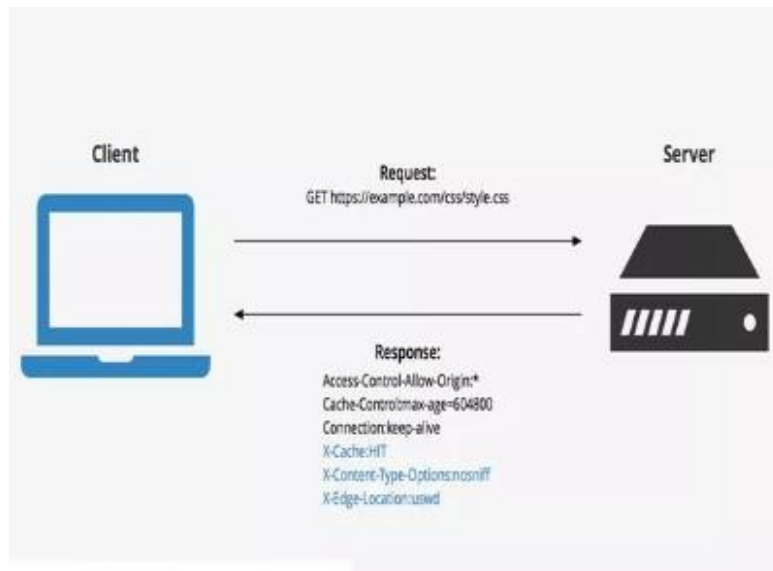
**6. Server Misconfiguration:** Attackers identify improperly configured servers to exploit security gaps, often resulting from oversight or lack of security awareness.

**Example:** Accessing sensitive files because directory listing is enabled.



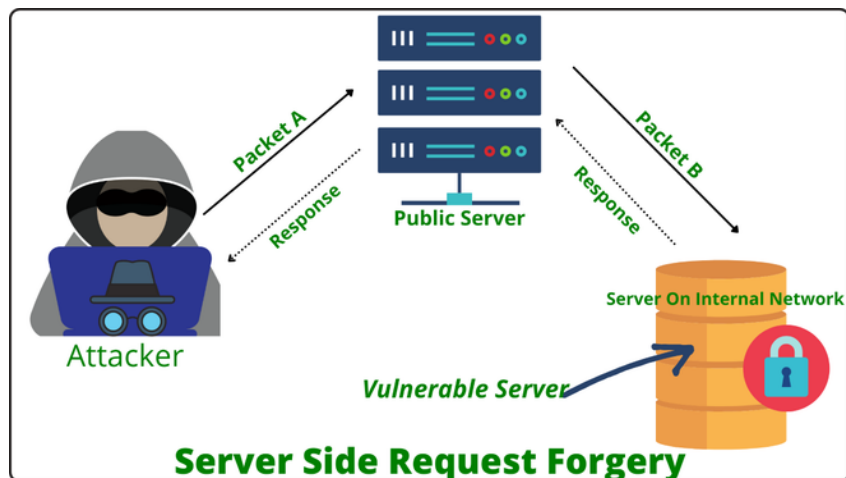
**7. HTTP Header Injection:** Attackers manipulate HTTP headers to trick servers into processing unexpected or malicious instructions, potentially leading to unauthorized access or data leakage.

**Example:** Modifying headers to redirect users to a phishing site.



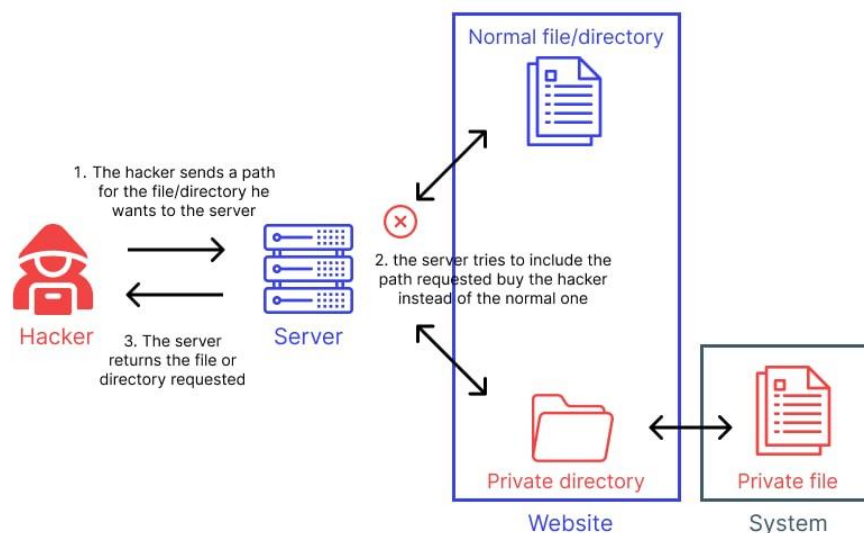
**8. Server-Side Request Forgery (SSRF):** Attackers manipulate a web application into sending requests to internal resources that should not be accessible externally. This can expose sensitive data or internal services.

**Example:** Forcing the server to make requests to internal databases or services.



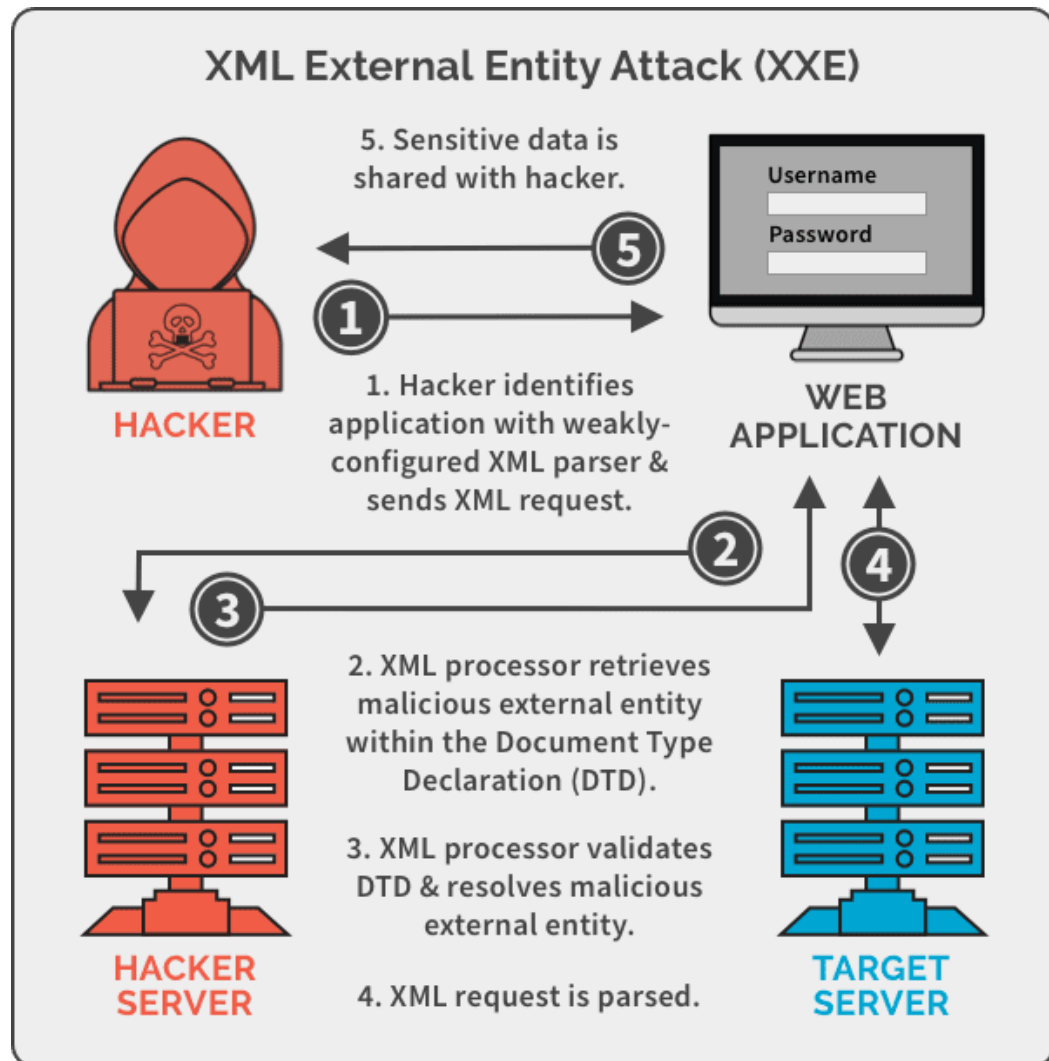
**9. Directive Traversal:** Attackers exploit inadequate input validation to navigate through directories and access unauthorized files. This can expose sensitive configuration files or even compromise the entire server.

**Example:** Modifying a URL to access files outside the intended directory



**10. XML External Entity (XXE) Attacks:** Attackers exploit weak XML parsers to include external entities that can disclose sensitive information.

**Example:** Uploading an XML file that retrieves confidential data from the server.



## **TASK - 6**

**Date: 30.08.2023**

**NAME: RAMAR PRIYA MAHA LAKSHMI**

**REGNO: 21BCE7521**

### **1. Understanding CIS Critical Security Controls**

#### **Control 1: Inventory and Control of Hardware Assets –**

Manage all the hardware devices on the network. Ensure that only authorized devices have access, and measures are taken to identify and block unauthorized or unmanaged devices.

#### **Control 2: Inventory and Control of Software Assets –**

Manage all software on the network. Ensure that only authorized software is in use. Prevent the installation or execution of unmanaged software.

#### **Control 3: Continuous Vulnerability Management –**

Regularly monitor to identify for any vulnerability in the system. Take necessary actions to fix the vulnerability if found to minimize the time frame in which attackers could exploit them.

#### **Control 4: Controlled Use of Administrative Privileges –**

Apply proper administrative principles and privileges on computers, networks, and applications.

**Control 5: Secure Configuration for Hardware and Software on Mobile Devices, Laptops, Workstations and Servers –**

Implement strong security configurations for mobile devices, laptops, servers and workstations. Use a thorough configuration management process to prevent attackers from exploiting vulnerabilities in services and settings.

**Control 6: Maintenance, Monitoring and Analysis of Audit**

**Logs** - Collect, manage, and analyze audit logs of events that can provide insights into potential attacks. These logs can help detect, understand, and recover from security breaches.

**Control 7: Email and Web Browser Protections** – Reduce the opportunities for attackers to manipulate user behavior through web browsers and email systems. Minimize the potential attack surface in these critical communication tools.

**Control 8: Malware Defenses** – Establish measures to control the introduction, spread, and execution of malicious software within your environment. Employ automation to keep defense mechanisms up to date.

**Control 9: Limitation and Control of Network Ports, Protocols, and Services** – Reduce the number of open network ports and protocols to minimize potential attack vectors. This control involves assessing and restricting network communication to only essential services.

**Control 10: Data Recovery Capabilities** – Establish and maintain robust data recovery capabilities. This control involves implementing strategies, procedures, and technologies that enable the rapid recovery of data in the event of data loss, data corruption, or other forms of data compromise. By ensuring that data can be reliably and quickly restored, you can minimize downtime and data loss in the face of unexpected incidents or cyberattacks.

**Control 11: Secure Configuration for Network Devices, such as Firewalls, Routers and Switches** – By applying robust security configurations and settings to network devices such as firewalls, routers and switches one can maintain the integrity and security of the network infrastructure.

**Control 12: Boundary Defense** – Protect the work boundaries from external threats. This includes implementing firewalls, intrusion detection systems and other security measures to monitor and control incoming and outgoing network traffic.

**Control 13: Data Protection** – Implementing measures to secure from internal/external threats. This control includes access controls, data loss prevention strategies, and user monitoring to mitigate the risk of threats.

**Control 14: Controlled Access Based on the Need to Know** – Implement access controls that grant users access only to the

information and resources necessary to perform their job functions. This principle, often referred to as “need-to-know”, ensures that individuals can only access data and systems that are relevant to their roles and responsibilities.

**Control 15: Wireless Access Control** – Secure the wireless networks to prevent unauthorized access. Proper configuration, authentication, and encryption are critical to protecting wireless infrastructure.

**Control 16: Account Monitoring and Control** – Monitor user accounts and privileges for signs for suspicious activity. Regularly review and audit user accounts to prevent unauthorized access and privilege escalation.

**Control 17: Implement a Security Awareness and Training Program** – Educate the workspace about security best practices and threats. An informed workspace is better equipped to recognize and respond to security incidents.

**Control 18: Application Software Security** – Ensure the security of the software application throughout their lifecycle. This involves integrating security practices at every stage of application development process, from design and coding to testing and deployment. By identifying and mitigating vulnerabilities and weaknesses early in the early in the development process, one can reduce the risk of security breaches and protect sensitive data.



**Control 19: Incident Response and Management** - Establish a robust incident response plan to detect, respond to, and recover from security incidents. Timely and effective incident response is essential to minimizing the impact of security breaches.

**Control 20: Penetration Tests and Red Team Exercises** - Regularly test your security controls by conducting penetration testing and red team exercises. These activities help identify vulnerabilities and weaknesses that might be exploited by real attackers.