NETWORK VULNERABILITY ASSESSMENT

SMART INTERN LONG-TERM INTERNSHIP

CYBERSECURITY

REPORT ON THE PROJECT BY

TSR & TBK DEGREE COLLEGE

Affiliated to Andhra University : Accredited by NAAC : Approved by UGC

Team leader: KARRI MEGHANA

REG NO: 720134205183

Team member: LOKANADHAM KISHOR

REG NO: 720134205159

Team member: GOPAL SOREN

REG NO: 720134205158

Team member: VASIREDDY YAMUNA KIRTHI

REG NO: 720134205191

Team member: KOYYA RAJESH

REG NO: 720134205079

INTRODUCTION

A vulnerability assessment is a process that helps review and analyze endpoint and device networks for security issues. The assessment may detect network flaws and holes in the network that could leave an opportunity for hackers to exploit. A network vulnerability should be performed on an ongoing process as new threats arise and hackers find additional ways to break into systems. Organizations must secure their networks and applications from any kind of hacking threats or cyberattacks. To do this, individual organizations can implement application and network security controls. And one of the critical network security controls/practices is to perform an in-depth network vulnerability assessment.

IMPORTANCE

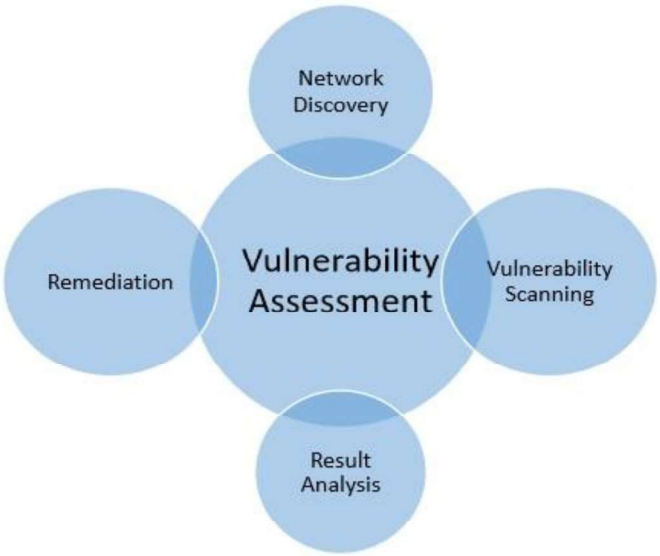
Vulnerability tools are designed to discover which vulnerabilities are present and allow you to resolve these before any hacker takes advantage of them. It is a good process to always document your vulnerability assessment process from start to finish. Your network vulnerability should also generate an assessment report to interpret and track known vulnerabilities and efforts. There are many factors to consider when conducting a network vulnerability assessment, but some of the most important aspects include:

- Identifying all network systems and devices in your network infrastructure

- Determining how these systems and devices are interconnected
- Analyzing system configurations and installed software for known vulnerabilities
- Scanning for open ports and services that could be exploited
- Testing for weak passwords or other authentication issues

OVERVIEW

Network vulnerability assessment is made to scan, investigate, analyze, and report on the level of risk associated with any security vulnerabilities discovered on public, internet-facing devices and provide your organization with appropriate mitigation strategies to address those vulnerabilities.



Common network vulnerabilities

- Insecure wireless networks...
- Removable media devices...
- Outdated software...
- Weak passwords.
- Social engineering attacks...
- Misconfigured firewalls...
- Unpatched software... Etc.

Network vulnerability assessment is a process that should be done regularly in order to identify and address any potential vulnerabilities. By using the steps and tools outlined in this article, you can conduct your own assessment quickly and effectively. Having a secure network is essential for protecting your data and your customers' data, so make sure to take the time to perform a thorough assessment on a regular basis.

Conducting a network vulnerability assessment helps organizations to detect any kind of weaknesses in their system before attackers do. It also provides detailed information to fix those

vulnerabilities with priority.

Not only that but network vulnerability assessment is also required by many compliance standards such as PCI-DSS, HIPAA, SOX, ISO, ETC…..

THERE ARE FURTHER STEPS TO VULNERABILITY ASSESSMEN

1) INFORMATION GATHERING:

Information gathering, or data collection, is a process where you follow a series of steps to conduct research and answer questions or resolve problems you have. Though information gathering isn't bound by cybersecurity, it is an essential skill to have in the field. When you want to understand a specific concept or get more information about one, the first thing you do is research this information. Depending on the importance of your topic, information gathering can take five minutes or several days. In cybersecurity, data gathering helps information security and cybersecurity executives uncover information about a potential target. In this case, information gathering can be carried out for various tasks such as penetration testing, network security monitoring, or something else

It's worth noting that hackers and cybercriminals also gather information about their potential targets.

In other words, information gathering isn't limited to a profession or to one side of the cybersecurity field.

Information Gathering is the act of gathering different kinds of information against the targeted **victim or system**. It is the first step or the beginning stage of Ethical Hacking, where the penetration testers or hackers (both black hat or white hat) performed this stage; this is a necessary and crucial step to be performed. The information gathering collects

information like email footprinting, DNS lookup, whois, open port ,etc.…

a) EMAIL FOOTPRINT ANALYSIS:

Email footprint refers to collecting information from emails by monitoring the email delivery and inspecting the headers.

Information collected through email footprint includes:

- IP address of the recipient
- Geolocation of the recipient
- Delivery information
- Visited links
- Browser and OS information
- Reading time

Email headers contain information about the sender, subject, and recipient. All this information is valuable to hackers when planning to attack their target.

Information contained in email headers includes:

- Sender's name
- IP/Email address of the sender
- Mail server
- Mail server authentication system
- Send and deliver stamp
- Unique number of the message

In this method, a hacker can trace an email and get information from it. Email footprint gives us

information regarding the sender's email, name, location, IP address, etc.

— Domain Profile

Registrar	CSC CORPORATE DOMAINS, INC. CSC Corporate Domains, Inc. IANA ID: 299 URL: www.cscprotectsbrands.com,http://cscdbs.com Whois Server: whois.corporatedomains.com domainabuse@cscglobal.com (p) +1.8887802723
Registrar Status	clientTransferProhibited

```
Domain Name: testfire.net
Registry Domain ID: 8363973_DOMAIN_NET-VRSN
Registrar WHOIS Server: whois.corporatedomains.com
Registrar URL: www.cscprotectsbrands.com
        http://cscdbs.com
Updated Date: 2023-07-19T01:05:02+00:00
        2023-07-19
Creation Date: 1999-07-23T09:52:32+00:00
        1999-07-23
Registrar Registration Expiration Date: 2024-07-
23T13:52:32+00:00
        2024-07-23
Registrar: CSC CORPORATE DOMAINS, INC.
        CSC Corporate Domains, Inc.
Sponsoring Registrar IANA ID: 299
Registrar Abuse Contact Email: domainabuse@cscglobal.com
Registrar Abuse Contact Phone: +1.8887802723
Status:
```

b) DNS GATHERING INFORMATION:

DNS enumeration is the process of locating all the DNS servers and their corresponding records for an organization. A company may have both internal and external DNS servers that can yield information such as usernames, computer names, and IP addresses of potential target systems.

When we talk about DNS enumeration, we generally refer to every technique we use to gather as much data as possible by querying the DNS server of a website or host. DNS is regularly a rewarding hotspot for active data gathering. DNS offers an assortment of information about open organization servers, for example, IP addresses, server names, and server usefulness.

c) WHOIS INFORMATION GATHERING

WHOIS information gathering involves gathering information about the owner of a domain name, IP address, or autonomous system number (ASN). This information can include the owner's name, contact details, and registration dates. This technique can be useful in identifying the owners of malicious or suspicious domains.

Whois is the name of the protocol that is used to interrogate the servers operated by Regional Internet Registries, which hold information about every resource (IP address or domain name) registered on the Internet.

The information that you can obtain about a resource is:

- Name of the owner company
- Address of the owner company
- The IP range that a certain IP belongs to
- Contact phone number
- Contact email
- Administrator's name
- Name servers

Whois Record (last updated on 2023-07-29)

```
Domain Name: testfire.net
Registry Domain ID: 8363973_DOMAIN_NET-VRSN
Registrar WHOIS Server: whois.corporatedomains.com
Registrar URL: www.cscprotectsbrands.com
        http://cscdbs.com
Updated Date: 2023-07-19T01:05:02+00:00
        2023-07-19
Creation Date: 1999-07-23T09:52:32+00:00
        1999-07-23
Registrar Registration Expiration Date: 2024-07-
23T13:52:32+00:00
        2024-07-23
Registrar: CSC CORPORATE DOMAINS, INC.
        CSC Corporate Domains, Inc.
Sponsoring Registrar IANA ID: 299
Registrar Abuse Contact Email: domainabuse@cscglobal.com
Registrar Abuse Contact Phone: +1.8887802723
Status:
        clientTransferProhibited
Registry Registrant ID:
Registrant Name: REDACTED FOR PRIVACY (DT)
Registrant Organization: Not Disclosed
Registrant Street: Not Disclosed
Registrant City: Sunnyvale
Registrant State/Province: CA
Registrant Postal Code: 94085
Registrant Country: US
Registrant Phone: +Not Disclosed
Registrant Phone Ext:
Registrant Fax: +Not Disclosed
Registrant Fax Ext:
Registrant Email: REDACTED FOR PRIVACY (DT)
Registry Admin ID:
Admin Name: REDACTED FOR PRIVACY (DT)
Admin Organization: Not Disclosed
Admin Street: Not Disclosed
Admin City: Sunnyvale
Admin State/Province: CA
Admin Postal Code: 94085
Admin Country: US
Admin Phone: REDACTED FOR PRIVACY (DT)
Admin Phone Ext:
Admin Fax: REDACTED FOR PRIVACY (DT)
Admin Fax Ext:
Admin Email: REDACTED FOR PRIVACY (DT)
Registry Tech ID:
Tech Name: REDACTED FOR PRIVACY (DT)
Tech Organization: Not Disclosed
Tech Street: Not Disclosed
Tech City: Sunnyvale
Tech State/Province: CA
Tech Postal Code: 94085
Tech Country: US
Tech Phone: REDACTED FOR PRIVACY (DT)
```

```
Tech Phone Ext:
Tech Fax: REDACTED FOR PRIVACY (DT)
Tech Fax Ext:
Tech Email: REDACTED FOR PRIVACY (DT)
Registry Billing ID:
Billing Name:
Billing Organization:
Billing Street:
Billing City:
Billing State/Province:
Billing Postal Code:
Billing Country:
Billing Phone:
Billing Phone Ext:
Billing Fax:
Billing Fax Ext:
Billing Email:
Nameservers:
        asia3.akam.net
        eur2.akam.net
        eur5.akam.net
        ns1-206.akam.net
        ns1-99.akam.net
        usc2.akam.net
        usc3.akam.net
        usw2.akam.net
DNSSEC: unsigned
```

d) INFORMATION GATHERING FOR SOCIAL ENGINEERING ATTACKS

Social engineering attacks happen in one or more steps. A perpetrator first investigates the intended victim to gather necessary background information, such as potential points of entry and weak security protocols, needed to proceed with the attack. Then, the attacker uses a form of pretexting such as impersonation to gain the victim's trust and provide stimuli for subsequent actions that break security practices, such as revealing

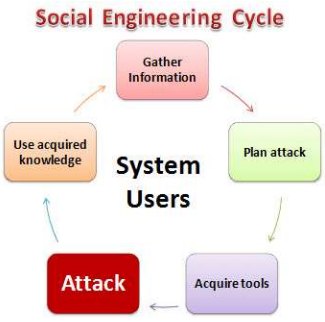
sensitive information or granting access to critical resources.

Social engineering is the tactic of manipulating, influencing, or deceiving a victim in order to gain control over a computer system, or to steal personal and financial information. It uses psychological manipulation to trick users into making security mistakes or giving away sensitive information.

Social engineering attacks happen in one or more steps. A perpetrator first investigates the intended victim to gather necessary background information, such as potential points of entry and weak security protocols, needed to proceed with the attack.

Then, the attacker uses a form of pretexting. such as impersonation to gain the victim's trust and provide stimuli for subsequent actions that break security practices, such as revealing sensitive information or granting access to critical resources.

- PHISHING
- BAITING
- SCAREWARE
- DUMPSTER DIVING



e) EMERGING TRENDS AND TECHNOLOGIES IN INFORMATION GATHERING:

Technological innovations elevate much of the progress in the corporate industry. The cut-throat competition requires companies to stay tuned with technologies and pursue digital transformation. Suppose you consider incorporating a new piece of software or hardware; the question is not if you should implement it: instead, it is how quickly you should do it!

- Adopting new technology is critical for business growth.
- Using technology to its full potential will allow you to meet consumer-changing demands.
- There are 5.6 billion internet users worldwide. An online business presence will open the gate to serving more customers.

Adapting new trends is necessary for firms to deliver quality services, reduce spending, and boost user experience. It is a long-term strategy that asks for time, effort, and expertise. However, it is better to learn about emerging trends in information technology to understand which matches your business needs. This article aims to give an insight into the top technology trends in 2023 and their significance. So read ahead!

- Artificial Intelligence and Machine Learning.
- Advanced Analytics.

- Datafication.
- Robotic Process Automation.
- Augmented and Virtual Reality.

Artificial Intelligence and Machine Learning:

Artificial Intelligence (AI) and Machine Language (ML) have been unquestionably one of the latest advancements. Consequently, its market will reach $267 billion by 2027. Today you can find AI and ML in every field, from finance and healthcare to manufacturing and retail.

Advanced Analytics:

Advanced or predictive analytics generate a predictive model for specific applications, like marketing. It combines Artificial Intelligence and statistical techniques to anticipate outcomes. You can evaluate historical data, identify patterns, and observe trends to determine potential future events before they happen.

Datafication:

Small and large firms across all sectors depend on data. It is safe to say that data underpins modern life; hence, it is vital to secure it. Smartphones, AI devices, industrial machines, etc., use data to communicate with humans and enhance lifestyles.

Robotic Process Automation:

Robotic Process Automation (RPA) has been slowly gaining ground over the past decade. It utilizes various applications and software to automate manual tasks. You can lower expenses by automating data collection, analysis, and customer service.

Augmented and Virtual Reality:

The virtual reality gaming industry and Augmented Reality revealed that humans are open to the idea of escaping the real world. Virtual Reality (VR) and Augmented Reality (AR) are modifying how you use screens and unlock the door to engaging and interactive experiences. Unsurprisingly, the VR and AR market may accumulate revenue worth $31.12 billion with 28.8% user penetration this year.

2) VULNERABILITY IDENTIFICATION:

The objective of this step is to draft a comprehensive list of an application's vulnerabilities. Security analysts test the security health of applications, servers or other systems by scanning them with automated tools, or testing and evaluating them manually. Analysts also rely on vulnerability databases, vendor vulnerability announcements, asset management systems and threat intelligence feeds to identify security weaknesses.

a) IDENTIFY AND NAME EACH VULNERABILITY:

A vulnerability is a hole or a weakness in the application, which can be a design flaw or an implementation bug, that allows an attacker to cause harm to the stakeholders of an application. Stakeholders include the application owner, application users, and other entities that rely on the application.

To identify the vulnerability of any website. There were many tools like Nikto, SQL map, burp suite, etc….

TARGET: http://testfire.net/

The following website is tested by nikto.

$nikto -h http://testfire.net/

b) ASSIGN A COMMON WEAKNESS ENUMERATION(CEW) CODE TO EACH VULNERABILITY:

Targeted at both the development community and the community of security practitioners, Common Weakness Enumeration (CWE™) is a formal list or dictionary of common software and hardware weaknesses that can occur in architecture, design, code, or implementation that can lead to exploitable security vulnerabilities. CWE was created to serve as a common language for describing security weaknesses; serve as a standard measuring stick for security tools targeting these weaknesses; and provide a common baseline standard for weakness identification, mitigation, and prevention efforts.

- nmap testphp.vulnweb.com
- Starting Nmap 7.94 (https//nmap.org) at 2023-07-14 13:50 IST
- Nmap scan report for testphp.vulnweb.com
- (44.228.249.3)
- Host is up (0.28s latency).
- CVSSV3.1:
- rDNS record for 44.228.249.3: ec2-44-228-249-3.us-west-2.compute.amazonaws.com
- Not shown: 999 filtered tcp ports (no-response) CVE-ID: CVE-2022-417
- PORT STATE SERVICE
- CWE-ID: CWE-125-Ouch

- 80/tcp open http
- Exploit availability: No
- Nmap done: 1 IP address (1 host up) scanned in 21.04 seconds
- Step -3
- Take ip address of domain and
- nmap -sV 44.228.249.3 -p 80
- nginx 1.0.7-1.23.1
- $ nmap -sV 44.228.249.3 -p 80
- Starting Nmap 7.94 (https://nmap.org) at 2023-07-14 13:51 IST
- ec2-44-228-249-3.us-west-2.compute.amazonaws.com (44.228.249.3)
- Nmap scan report for
- Host is up (0.30s latency).
- PORT STATE SERVICE VERSION
- 80/tcp open http nginx 1.19.0
- che 2.3 an
- External links
- Service detection performed. Please report any incorrect results at https://nmap.org/submit/
- Nmap done: 1 IP address (1 host up) scanned in 23.10 seconds

- Step -4
- Copied the version
- nginx 1.19.0 and
- Pasted it in google

- The results is
- # PHuiP-FPizdaM
- ## What's this
- This is an exploit for a bug in php-fpm (CVE-2019-11043). In certain nginx + php-fpm configurations, the bug is possible to trigger from the outside. This means that a web user may get code execution if you have vulnerable config (see [below](#the-full-list-of-preconditions)).
- ## What's vulnerable
- If a webserver runs nginx + php-fpm and nginx have a configuration like
- location ~ [^/]/.php(/|$) {
- ...
- fastcgi_split_path_info ^(.+?\.php)(/.*)$;
- fastcgi_param PATH_INFO $fastcgi_path_info;
- fastcgi_pass php:9000;
- ...
- }which also lacks any script existence checks (like try_files), then you can probably hack it with this sploit.
- #### The full list of preconditions
- 1. Nginx + php-fpm, location ~ [^/]\.php(/|$) must be forwarded to php-fpm (maybe the regexp can be stricter, see [#1](https://github.com/neex/phuip-fpizdam/issues/1)).

- 2. The fastcgi_split_path_info directive must be there and contain a regexp starting with ^ and ending with $, so we can break it with a newline character.
- 3. There must be a PATH_INFO variable assignment via statement fastcgi_param PATH_INFO $fastcgi_path_info;. At first, we thought it is always present in the fastcgi_params file, but it's not true.
- 4. No file existence checks like try_files $uri =404 or if (-f $uri). If Nginx drops requests to non-existing scripts before FastCGI forwarding, our requests never reach php-fpm. Adding this is also the easiest way to patch.
- 5. This exploit works only for PHP 7+, but the bug itself is present in earlier versions (see [below](#about-php5)).
- ## Isn't this known to be vulnerable for years?
- A long time ago php-fpm didn't restrict the extensions of the scripts, meaning that something like /avatar.png/some-fake-shit.php could execute avatar.png as a PHP script. This issue was fixed around 2010.
-
- The current one doesn't require file upload, works in the most recent versions (until the fix has landed), and, most importantly, the exploit is much cooler.
- ## How to run

- Install it using
- go get github.com/neex/phuip-fpizdam
- If you get strange compilation errors, make sure you're using go >= 1.13. Run the program using phuip-fpizdam [url] (assuming you have the $GOPATH/bin inside your $PATH, otherwise specify the full path to the binary). Good output looks like this:
- 2019/10/01 02:46:15 Base status code is 200
- 2019/10/01 02:46:15 Status code 500 for qsl=1745, adding as a candidate
- 2019/10/01 02:46:15 The target is probably vulnerable. Possible QSLs: [1735 1740 1745]
- 2019/10/01 02:46:16 Attack params found: --qsl 1735 --pisos 126 --skip-detect
- 2019/10/01 02:46:16 Trying to set "session.auto_start=0"...
- 2019/10/01 02:46:16 Detect() returned attack params: --qsl 1735 --pisos 126 --skip-detect <-- REMEMBER THIS
- 2019/10/01 02:46:16 Performing attack using php.ini settings...
- 2019/10/01 02:46:40 Success! Was able to execute a command by appending "?a=/bin/sh+-c+'which+which'&" to URLs
- 2019/10/01 02:46:40 Trying to cleanup /tmp/a...
- 2019/10/01 02:46:40 Done!

- `After this, you can start appending `?a=<your command>` to all PHP scripts (you may need multiple retries).
- ## Playground environment
- If you want to reproduce the issue or play with the exploit locally, do the following:
- 1. Clone this repo and go to the `reproducer` directory.
- 2. Create the docker image using `docker build -t reproduce-cve-2019-11043 .`. It takes a long time as it internally clones the php repository and builds it from the source. However, it will be easier this way if you want to debug the exploit. The revision built is the one right before the fix.
- 2. Run the docker using `docker run --rm -ti -p 8080:80 reproduce-cve-2019-11043`.
- 3. Now you have http://127.0.0.1:8080/script.php, which is an empty file.
- 4. Run the exploit using `phuip-fpizdam http://127.0.0.1:8080/script.php`
- 5. If everything is ok, you'll be able to execute commands by appending `?a=` to the script: http://127.0.0.1:8080/script.php?a=id. Try multiple times as only some of php-fpm workers are infected.
- ## About PHP5

- The buffer underflow in php-fpm is present in PHP version 5. However, this exploit makes use of an optimization used for storing FastCGI variables, [_fcgi_data_seg](https://github.com/php/php-src/blob/5d6e923/main/fastcgi.c#L186). This optimization is present only in php 7, so this particular exploit works only for php 7. There might be another exploitation technique that works in php 5.
- ## Credits

Original anomaly discovered by [d90pwn] (https://twitter.com/d90pwn) during Real World CTF. Root clause found by me (Emil Lerner) as well as the way to set php.inioptions. Final php.ini options set is found by [beched] (https://twitter.com/ahack_ru).

Identifying and naming vulnerabilities involves the process of discovering and documenting specific security weaknesses or flaws in an application. This process typically involves using automated tools or manual testing techniques to identify potential vulnerabilities. Once a vulnerability has been identified, it must be given a descriptive name that accurately reflects the nature of the vulnerability.

c) PROVIDE CORRESPONDING Open Web Application Security Project (OWASP) Category And Description For Each Vulnerability:

The Open Web Application Security Project (OWASP) is a nonprofit foundation that provides guidance on how to develop, purchase and maintain trustworthy and secure software applications. OWASP is noted for its popular

Assigning CWE codes to each vulnerability is an essential step in the vulnerability identification process. A CWE code is a unique identifier assigned to a specific type of vulnerability, making it easier to identify and categorize similar types of vulnerabilities. Assigning a CWE code to each vulnerability allows developers and security professionals to more easily track, analyze and remediate potential security issues.

- **A01:2021-Broken Access Control** moves up from the fifth position to the category with the most serious web application security risk; the contributed data indicates that on average, 3.81% of applications tested had one or more Common Weakness Enumerations (CWEs) with more than 318k occurrences of CWEs in this risk category. The 34 CWEs mapped to Broken Access Control had more occurrences in applications than any other category.

- **A02:2021-Cryptographic Failures** shifts up one position to #2, previously known as **A3:2017-Sensitive Data Exposure**, which was broad symptom rather than a root cause. The renewed name focuses on failures related to cryptography as it has been implicitly before. This category often leads to sensitive data exposure or system compromise.
- **A03:2021-Injection** slides down to the third position. 94% of the applications were tested for some form of injection with a max incidence rate of 19%, an average incidence rate of 3.37%, and the 33 CWEs mapped into this category have the second most occurrences in applications with 274k occurrences. Cross-site Scripting is now part of this category in this edition.
- **A04:2021-Insecure Design** is a new category for 2021, with a focus on risks related to design flaws. If we genuinely want to "move left" as an industry, we need more threat modeling, secure design patterns and principles, and reference architectures. An insecure design cannot be fixed by a perfect implementation as by definition, needed security controls were never created to defend against specific attacks.
- **A05:2021-Security Misconfiguration** moves up from #6 in the previous edition; 90% of applications were tested for some form of misconfiguration, with an average incidence rate of 4.5%, and over 208k occurrences of CWEs mapped to this risk category. With more shifts into highly configurable software,

it's not surprising to see this category move up. The former category for **A4:2017-XML External Entities (XXE)** is now part of this risk category.

- **A06:2021-Vulnerable and Outdated Components** was previously titled Using Components with Known Vulnerabilities and is #2 in the Top 10 community survey, but also had enough data to make the Top 10 via data analysis. This category moves up from #9 in 2017 and is a known issue that we struggle to test and assess risk. It is the only category not to have any Common Vulnerability and Exposures (CVEs) mapped to the included CWEs, so a default exploit and impact weights of 5.0 are factored into their scores.
- **A07:2021-Identification and Authentication Failures** was previously Broken Authentication and is sliding down from the second position, and now includes CWEs that are more related to identification failures. This category is still an integral part of the Top 10, but the increased availability of standardized frameworks seems to be helping.
- **A08:2021-Software and Data Integrity Failures** is a new category for 2021, focusing on making assumptions related to software updates, critical data, and CI/CD pipelines without verifying integrity. One of the highest weighted impacts from Common Vulnerability and Exposures/Common Vulnerability Scoring System (CVE/CVSS) data mapped to the 10 CWEs in this category. **A8:2017-Insecure Deserialization** is now a part of this larger category.

- **A09:2021-Security Logging and Monitoring Failures** was previously **A10:2017-Insufficient Logging & Monitoring** and is added from the Top 10 community survey (#3), moving up from #10 previously. This category is expanded to include more types of failures, is challenging to test for, and isn't well represented in the CVE/CVSS data. However, failures in this category can directly impact visibility, incident alerting, and forensics.
- **A10:2021-Server-Side Request Forgery** is added from the Top 10 community survey (#1). The data shows a relatively low incidence rate with above average testing coverage, along with above-average ratings for Exploit and Impact potential. This category represents the scenario where the security community members are telling us this is important, even though it's not illustrated in the data at this time.

d) UNDERSTANDING AND DEFINING VULNERABILITIES:

Understanding and defining vulnerabilities is a critical first step in identifying and mitigating potential risks in an application. A vulnerability can be defined as a flaw or weakness in the system that can be exploited by attackers to compromise the security of the system. Vulnerabilities can exist in different layers of the application, including the network layer, application layer, and database layer. By understanding the different types of vulnerabilities that exist, developers and security professionals can take appropriate measures to mitigate the risks and prevent attacks.

A vulnerability is a weakness in an IT system that can be exploited by an attacker to deliver a successful attack. They can occur through flaws, features, or user error, and attackers will look to exploit any of them, often combining one or more, to achieve their end goal.

A feature is intended functionality which can be misused by an attacker to breach a system. Features may improve the user's experience, help diagnose problems or improve management, but they can also be exploited by an attacker.

When Microsoft introduced macros into their Office suite in the late 1990s, macros soon became the vulnerability of choice with the Melissa worm in 1999 being a prime example. Macros are still exploited today; the Dridex banking Trojan that was spreading in late 2014 relies on spam to

deliver Microsoft Word documents containing malicious macro code, which then downloads Dridex onto the affected system.

JavaScript, widely used in dynamic web content, continues to be used by attackers. This includes diverting the user's browser to a malicious website and silently downloading malware, and hiding malicious code to pass through basic web filtering.

e) IDENTIFYING AND NAMING THE VULNERABILITIES:

Identifying and naming vulnerabilities is the next step in the vulnerability assessment process. This involves conducting a thorough analysis of the application to identify all potential vulnerabilities that could be exploited by attackers. Once identified, each vulnerability should be given a clear and concise name that accurately describes the nature of the vulnerability.

A vulnerability naming scheme is a systematic method for creating and maintaining a standardized dictionary of common names for a set of vulnerabilities in IT systems, such as software flaws in an operating system or security configuration issues in an application. The naming scheme ensures that each vulnerability entered into the dictionary has a unique name. Using standardized vulnerability naming

schemes supports interoperability. Organizations typically have many tools for system security management that reference vulnerabilities—for example, vulnerability and patch management software, vulnerability assessment tools, antivirus software, and intrusion detection systems. If these tools do not use standardized names for vulnerabilities, it may not be clear that multiple tools are referencing the same vulnerabilities in their reports, and it may take extra time and resources to resolve these discrepancies and correlate the information. This lack of interoperability can cause delays and inconsistencies in security assessment, reporting, decision-making, and vulnerability remediation, as well as hampering communications both within organizations and between organizations. Use of standardized names also helps minimize confusion regarding which problem is being addressed, such as which vulnerabilities a new patch mitigates. This helps organizations to quickly identify the information they need, such as remediation information, when a new problem arises. This publication provides information and recommendations related to two commonly used vulnerability naming schemes: Common Vulnerabilities and Exposures (CVE), and Common Configuration Enumeration (CCE). Both are described in detail below. Common Vulnerabilities and Exposures (CVE) The CVE vulnerability naming scheme is for a dictionary of unique, common names for publicly known software flaws. CVE provides the following: < A comprehensive list of publicly known software flaws < A globally unique name to identify each vulnerability < A basis for discussing priorities and risks of vulnerabilities < A way for a user of disparate products and services to integrate vulnerability information A CVE vulnerability entry consists of a unique identifier number.

f) ASSIGNING CWE CODES TO EACH VULNERABILITY:

The main goal of the CWE initiative is to stop vulnerabilities at the source by educating software and hardware acquirers, architects, designers, and programmers on how to eliminate the most common mistakes before a product is delivered. CWE serves as a resource for programmers as they write code, for architects as they design new software, for hardware engineers as they create physical components, and supports educators in teaching security as part of curriculum for software and hardware engineering, computer science, and management information systems; CWE ultimately helps them prevent the kinds of security vulnerabilities that have plagued the software and hardware industries and put enterprises at risk. CWE continues to evolve as a collaborative community effort to populate a publicly available repository of software and hardware errors in code, design, architecture, and implementation for developers and security practitioners that can also be utilized by tool vendors for tagging what their tool's report and claim to cover.

CWE is industry-endorsed by the CWE Community, which includes representatives from major operating systems vendors, commercial information security tool vendors, academia, government agencies, and research institutions. Community members participate in the development of CWE on the CWE Community Research email list. This means the CWE List, as well as its follow-on CWSS and CWRAF efforts, reflect the insights and combined expertise of the broadest possible collection of

information technology and information security professionals.

Assigning Common Weakness Enumeration (CWE) codes to each vulnerability is an important step in the vulnerability assessment process. CWE is a community-developed list of common software and hardware weaknesses, maintained by the MITRE Corporation, which provides a common language for identifying, understanding, and mitigating software vulnerabilities. By assigning a CWE code to each vulnerability, security professionals and developers can better understand the nature of the vulnerability and take appropriate steps to mitigate the risk.

> Targeted at both the development community and the community of security practitioners, Common Weakness Enumeration (CWE™) is a formal list or dictionary of common software and hardware weaknesses that can occur in architecture, design, code, or implementation that can lead to exploitable security vulnerabilities. CWE was created to serve as a common language for describing security weaknesses; serve as a standard measuring stick for security tools targeting these weaknesses; and to provide a common baseline standard for weakness identification, mitigation, and prevention efforts.
>
> A “weakness” is a condition in a software, firmware, hardware, or service component that, under certain circumstances, could contribute to the introduction of vulnerabilities.

g) PROVIDING OWASP CATEGORY AND DESCRIPTION FOR EACH VULNERABILITY:

Providing OWASP category and description for each vulnerability involves categorizing the vulnerabilities based on the OWASP Top 10, which is a list of the most common web application vulnerabilities. This process involves identifying which OWASP category the vulnerability falls under and providing a detailed description of the vulnerability. This information is important because it helps developers and security professionals prioritize which vulnerabilities to address first, based on their potential impact on the application's security.

- Lack of input validation on user input
- Lack of sufficient logging mechanism
- Fail-open error handling
- Not closing the database connection properly

For a great overview, check out the OWASP Top Ten Project. You can read about the top vulnerabilities and download a paper that covers them in detail. Many organizations and agencies use the Top Ten as a way of creating awareness about application security.

NOTE: Before you add a vulnerability, please search and make sure there isn't an equivalent one already. You may want to consider creating a redirect if the topic is the same. Every vulnerability article has a defined structure.

3) BUSSINESS IMPACT ASSESSMENT:

A business impact analysis (BIA) is the process of determining the criticality of business activities and associated resource requirements to ensure operational resilience and continuity of operations during and after a business disruption. The BIA quantifies the impacts of disruptions on service delivery, risks to service delivery, and recovery time objectives (RTOs) and recovery point objectives (RPOs). These recovery requirements are then used to develop strategies, solutions and plans.

a) CONDUCT A THOROUGH ANALYSIS OF THE POTENTIAL BUSINESS IMPACT OF EACH VULNERABILITY:

securing approval for the BIA from senior management;

gathering trained people who can perform a BIA;

preparing a BIA plan;

gathering information from questionnaires, interviews and documentation that is relevant to the analysis;

evaluating the collected information and interview data

Conducting a business impact assessment is an important step in the vulnerability identification and reporting process. This involves analyzing the potential impact that each vulnerability could have on the organization's operations, reputation, and finances. The assessment should take into account the likelihood of the vulnerability being exploited, the potential damage that could be caused, and

the organization's ability to respond and recover from such an incident. By conducting a thorough business impact assessment, stakeholders can prioritize the vulnerabilities and allocate resources appropriately to mitigate the risks.

b) UNDERSTAND THE POTENTIAL CONSEQUENCES OF EACH VULNERABILITY ON THE BUSINESS:

BIAs have many goals. They are used to determine the most crucial business functions, systems, staff and technology resources needed for operations to run optimally. They are also used to assess the time frame within which the functions must be recovered for the organization to restore operations to a normal working state. The analysis may be manual or computer-assisted. Challenges include determining the revenue impact of a business function and quantifying the long-term impact of losses in market share, business reputation or customers. Impacts to consider include the following:

- delayed sales or income
- increased labor expenses
- regulatory fines
- contractual penalties
- customer dissatisfaction

The business impact analysis report typically includes an executive summary, information on the methodology for data

gathering and analysis, detailed findings on the various business units and functional areas, charts and diagrams to illustrate potential losses, and recommendations for recovery.

c) CONDUCTING A BUSINESS IMPACT ASSESSMENT:

Conducting a business impact assessment involves evaluating the potential impact of vulnerabilities on the business. This involves identifying critical business processes and assessing the impact of the vulnerabilities on these processes. By conducting a business impact assessment, organizations can prioritize vulnerabilities based on their potential impact on the business.

d) UNDERSTANDING POTENTIAL CONSEQUENCES OF VULNERABILITIES:

Understanding potential consequences of vulnerabilities is crucial in determining the level of risk posed by each vulnerability. This involves assessing the likelihood of a vulnerability being exploited, the potential impact of an exploit, and the potential consequences of a successful attack. By understanding the potential consequences of vulnerabilities, organizations can develop appropriate mitigation strategies to minimize the risk to the business.

e) ASSESSING THE RISK TO THE BUSINESS:

Assessing the risk to the business involves evaluating the likelihood of a vulnerability being exploited and the potential impact it could have on the organization. The risk assessment should take into account factors such as the threat landscape, the value of the assets at risk, and the organization's current security posture. By conducting a risk assessment, stakeholders can identify vulnerabilities that pose the greatest risk to the organization and prioritize their remediation efforts. It is important to conduct ongoing risk assessments to ensure that vulnerabilities are identified and addressed in a timely manner.

4) VULNERABILITY PATH AND PARAMETER IDENTIFICATION:

a) METHODS FOR IDENTIFYING VULNERABILITY PATH AND PARAMETERS:

To scan target website vulnerabilities there many tools in kali linux.

- Nikto.
- Nmap.
- Brup suite.
- Dbscan.etc……

According to project to scan vulnerability of target we used tool called Nikto and Nmap.

- Nikto and parameter of nikto.

Nikto is an Open Source software written in Perl language that is used to scan a web-server for the vulnerability that can be exploited and can compromise the server. It can also check for outdated version details of 1200 server and can detect problems with specific version details of over 200 servers

Nmap and parameter of nmap:

b) TYPES OF VULNERABILITY PATHS AND PARAMETERS:

In order to understand vulnerability, let us look at some of the main reasons behind it. These have come to light through research experiences and empirical evidence from different parts of the world. Some of the key reasons for ever-increasing levels of vulnerabilily are:Rapid population growth where disaster events can claim more lives. Environmental degradation due to poor land use, deforestation, overcultivation and overgrazing. These render the land more prone to floods and landslides. Increased rate of industrialization and rapid urbanization without the necessary safety measures. Impoverished conditions. Blind adherence to cultural practices Gender inequalities. War and civil strife. Lack of public awareness and information.

c) COMMON TOOLS AND TECHNIQUES FOR IDENTIFYING VULNERABILITY PATHS AND PARAMETERS:

Nikto: an application that scans web-based applications and web servers for known bad files that could potentially be dangerous. Other things that it

can detect include outdated configs, port scanning, username enumeration and more.

Skipfish: Skipfish is an automated tool that performs reconnaissance tasks on web servers. It generates a sitemap and then recursively probes the site with penetration tests to identify vulnerabilities.
Wapiti: Wapiti is another penetration testing tool that manages to probe common such SQL injection, cross-site scripting and it uses GET and POST methods as part of its attacking capabilities.

OWASP-ZAP: The Zed Attack Proxy scanner is a pentesting app that allows you to test web apps while still in the dev stage. This lets you design tests to find problems before they get released into production environments.

XSSPY: As the name suggests, this is a Python tool that tests for cross-site scripting vulnerabilities in websites and web applications. After an initial scan that maps out the entire site, it then begins the detailed task of scanning every element that it uncovered in search of XSS vulnerabilities.

W3af: This is a web application framework that lets you attack and audit web apps and uncovers and exploits web application vulnerabilities as part of your vulnerabilities assessment. It is available as a GUI and console application, and it has over 130 different plugins for different tasks.

The most common software security vulnerabilities include:

- Missing data encryption
- OS command injection
- SQL injection
- Buffer overflow
- Missing authentication for critical function

d) BEST PRACTICES FOR VULNERABILITY PATH AND PARAMETER IDENTIFICATION:

To ensure that vulnerability paths and parameters are identified accurately and comprehensively, it is important to use a combination of manual and automated testing methods. It is also important to test applications in different environments and with different user roles to identify all possible attack vectors. Finally, thorough documentation and reporting of identified vulnerabilities and their associated paths and parameters is crucial for developers to be able to address the vulnerabilities effectively.

e) CHALLENGES AND LIMITATIONS OF VULNERABILITY PATH AND PARAMETER IDENTIFICATION:

One of the biggest challenges in identifying vulnerability paths and parameters is the constantly evolving nature of vulnerabilities and attack methods. Additionally, some vulnerabilities may be difficult to identify and require specialized knowledge and skills to detect. Another limitation is the potential for false positives or false negatives in vulnerability scanning and testing, which can lead to wasted time and resources.

5) DETAILED INSTRUCTION FOR VULNERABILITY REPRODUCTION:

This policy applies to the following systems and services:

- Websites operated by us.
- Software applications made available by us for use on or through computers and mobile devices.
- Our social media pages and applications.

Any system or service not expressly listed above, such as any connected systems or services, is excluded from scope and is not authorized for testing. Additionally, vulnerabilities found in systems owned or controlled by third parties fall outside the scope of this policy. They should be reported directly to the third party according to their policies.

Though we develop and maintain other Internet-accessible systems or services, we ask that active research and testing be conducted only on the systems and services covered by the scope of this policy.

a) IMPORTANCE OF PROVIDING DETAILED INSTRUCTIONS:

This cheat sheet is intended to provide guidance on the vulnerability disclosure process for both security researchers and organisations. This is an area where collaboration is extremely important, but that can often result in conflict between the two parties.

Researchers should:

Ensure that any testing is legal and authorised.

- Respect the privacy of others.
- Make reasonable efforts to contact the security team of the organisation.
- Provide sufficient details to allow the vulnerabilities to be verified and reproduced.
- Not demand payment or rewards for reporting vulnerabilities outside of an established bug bounty program.

Organisations should:

- Provide a clear method for researchers to securely report vulnerabilities.

- Clearly establish the scope and terms of any bug bounty programs.
- Respond to reports in a reasonable timeline.
- Communicate openly with researchers.
- Not threaten legal action against researchers.
- Request CVEs where appropriate.
- Publish clear security advisories and changelogs.
- Offer rewards and credit.

b) COMPONENTS OF A WELL-WRITTEN VULNERABILITY REPRODUCTION INSTRUCTION:

This section is intended to provide guidance for security researchers on how to report vulnerabilities to organizations.Warnings and Legality Before carrying out any security research or reporting vulnerabilities, ensure that you know and understand the laws in your jurisdiction. This cheat sheet does not constitute legal advice, and should not be taken as such..The following points highlight a number of areas that should be considered:

- If you are carrying out testing under a bug bounty or similar program, the organisation may have established safe harbor policies, that allow you to legally carry out testing, as long as you stay within the scope and rules of their program. Make sure that you read the scope carefully - stepping outside of the scope and rules may be a criminal offence.
- Some countries have laws restricting reverse engineering, so testing against locally installed software may not be permitted.
- Do not demand payment or other rewards as a condition of providing information on security vulnerabilities, or in exchange for not publishing the details or reporting them to industry regulators, as this may constitute blackmail.
- If you receive bug bounty payments, these are generally considered as income, meaning that they may be taxable. Reporting this income and ensuring

that you pay the appropriate tax on it is your responsibility.

- If you find vulnerabilities as part of your work, or on equipment owned by your employer, your employer may prevent you from reporting these or claiming a bug bounty. Read your contract carefully and consider taking legal advice before doing so.

c) STEPS FOR REPRODUCING VULNERABILITIES:

Step 1: Finding Contact Details.

The first step in reporting a vulnerability is finding the appropriate person to report it to. Although some organisations have clearly published disclosure policies, many do not, so it can be difficult to find the correct place to report the issue.

Where there is no clear disclosure policy, the following areas may provide contact details:

- Bug bounty programs such as BugCrowd, HackerOne, huntr.dev, Open Bug Bounty or Standoff.
- A security.txt file on the website at /.well-known/security.txt as per RFC 9116.
- An existing issue tracking system.
- Generic email addresses such as security@ or abuse@.
- The generic "Contact Us" page on the website.
- Social media platforms.
- Phoning the organisation.
- Reaching out to the community.

When reaching out to people who are not dedicated security contacts, request the details for a relevant member of staff, rather than disclosing the vulnerability details to whoever accepts the initial contact (especially over social media).

If it is not possible to contact the organisation directly, a national or sector-based CERT may be able to assist.

Step 2: Initial Report.

Once a security contact has been identified, an initial report should be made of the details of the vulnerability. Ideally this

should be done over an encrypted channel (such as the use of PGP keys), although many organisations do not support this.

The initial report should include:

- Sufficient details of the vulnerability to allow it to be understood and reproduced.
- HTTP requests and responses, HTML snippets, screenshots or any other supporting evidence.
- Redact any personal data before reporting.
- Some organisations may try and claim vulnerabilities never existed, so ensure you have sufficient evidence to prove that they did.
- Proof of concept code (if available).
- The impact of the vulnerability.
- Any references or further reading that may be appropriate.

In many cases, especially in smaller organisations, the security reports may be handled by developers or IT staff who do not have a security background. This means that they may not be familiar with many security concepts or terminology, so reports should be written in clear and simple terms.

It may also be beneficial to provide a recommendation on how the issue could be mitigated or resolved. However, unless the details of the system or application are known, or you are very confident in the recommendation then it may be better to point the developers to some more general guidance (such as an OWASP cheat sheet).

If you are planning to publish the details of the vulnerability after a period of time (as per some responsible disclosure policies), then this should be clearly communicated in the initial email - but try to do so in a tone that doesn't sound threatening to the recipient.

If the organisation does not have an established bug bounty program, then avoid asking about payments or rewards in the initial contact - leave it until the issue has been acknowledged (or ideally fixed). In particular, do not demand payment before revealing the details of the vulnerability. At best this will look like an attempt to scam the company, at worst it may constitute blackmail.

Step 3:Ongoing Communication.

While simpler vulnerabilities might be resolved solely from the initial report, in many cases there will be a number of emails back and forth between the researcher and the organisation. Especially for more complex vulnerabilities, the

developers or administrators may ask for additional information or recommendations on how to resolve the issue. They may also ask for assistance in retesting the issue once a fix has been implemented. Although there is no obligation to carry out this retesting, as long as the request is reasonable then and providing feedback on the fixes is very beneficial.

It may also be necessary to chase up the organisation if they become unresponsive, or if the established deadline for publicly disclosing the vulnerability is approaching. Ensure that this communication stays professional and positive - if the disclosure process becomes hostile then neither party will benefit.

Be patient if it's taking a while for the issue to be resolved. The developers may be under significant pressure from different people within the organisation, and may not be able to be fully open in their communication. Triaging, developing, reviewing, testing and deploying a fix within in an enterprise environment takes significantly more time than most researchers expect, and being constantly hassled for updates just adds another level of pressure on the developers.

Step 4:When to Give Up.

Despite every effort that you make, some organisations are not interested in security, are impossible to contact, or may be actively hostile to researchers disclosing vulnerabilities. In some cases they may even threaten to take legal action against researchers. When this happens it is very disheartening for the researcher - it is important not to take this personally. When this happens, there are a number of options that can be taken.

- Publicly disclose the vulnerability, and deal with any negative reaction and potentially even a lawsuit. Whether or not they have a strong legal case is irrelevant - they have expensive lawyers and fighting any kind of legal action is expensive and time consuming. Before going down this route, ask yourself is it really worth it?
- Anonymously disclose the vulnerability. However, if you've already made contact with the organisation and tried to report the vulnerability to them, it may be pretty obvious who's responsible behind the disclosure. If you are going to take this approach, ensure that you have taken sufficient operational security measures to protect yourself.
- Report the vulnerability to a third party, such as an industry regulator or data protection authority.
- Move on.

There are many organisations who have a genuine interest in security, and are very open and co-operative with security researchers. Unless the vulnerability is extremely serious, it is

not worth burning yourself out, or risking your career and livelihood over an organisation who doesn't care.

Step 5 : Publishing.
Once a vulnerability has been patched (or not), then a decision needs to be made about publishing the details. This should ideally be done through discussion with the vendor, and at a minimum the vendor should be notified that you intend to publish, and provided with a link to the published details. The disclosure would typically include:

- A high level summary of the vulnerability and its impact.
- Details of which version(s) are vulnerable, and which are fixed.
- Technical details or potentially proof of concept code.
- Mitigations or workarounds.
- Links to the vendor's published advisory.
- The timeline for the discovery, vendor communication and release.

Some organisations may request that you do not publish the details at all, or that you delay publication to allow more time to their users to install security patches. In the interest of maintaining a positive relationship with the organisation, it is worth trying to find a compromise position on this.

Whether to publish working proof of concept (or functional exploit code) is a subject of debate. Some people will view this as a "blackhat" move, and will argue that by doing so you are directly helping criminals compromise their users. On the other hand, the code can be used to both system administrators and penetration testers to test their systems, and attackers will be able to develop or reverse engineering working exploit code if the vulnerability is sufficiently valuable.

If you are publishing the details in hostile circumstances (such as an unresponsive organisation, or after a stated period of time has elapsed) then you may face threats and even legal action. Whether there is any legal basis for this will depend on your jurisdiction, and whether you signed any form of non-disclosure agreement with the organisation. Make sure you understand your legal position before doing so.

Note that many bug bounty programs forbid researchers from publishing the details without the agreement of the organisation. If you choose to do so, you may forfeit the bounty or be banned from the platform - so read the rules of the program before publishing.

Step 6:Receiving Vulnerability Reports.

This section is intended to provide guidance for organisations on how to accept and receive vulnerability reports.

Step 7:Bug Bounty Programs.
Bug bounty programs incentivise researchers to identify and report vulnerabilities to organisations by offering rewards. These are usually monetary, but can also be physical items (swag). The process is often managed through a third party such as BugCrowd or HackerOne, who provide mediation between researchers and organisations.

When implementing a bug bounty program, the following areas need to be clearly defined:

- Which systems and applications are in scope.
  Live systems or a staging/UAT environment?
  Excluding systems managed or owned by third parties.
- Which types of vulnerabilities are eligible for bounties (SSL/TLS issues? Missing HTTP security headers? Version disclosure?)
- Legal provisions such as safe harbor policies.
  The disclose.io project provides some example policies.
  Take legal advice from lawyers, not from this cheat sheet.
- How much to offer for bounties, and how is the decision made.
- The program could get very expensive if a large number of vulnerabilities are identified.
- Too little and researchers may not bother with the program.
- The timmeline for the initial response, confirmation, payout and issue resolution.

d) BEST PRACTICES FOR WRITING EFFECTIVE VULNERABILITY REPRODUCTION INSTRUCTIONS:

Effective vulnerability reproduction instructions should be clear, concise, and easy to understand. Instructions should be written in plain language and avoid technical jargon. Screenshots or videos can be used to supplement written instructions and provide
visual aids for developers.

There are five main stages in the vulnerability management cycle include:

- Step 1. Assess
- Step 2. Prioritize
- Step 3. Act
- Step 4. Reassess
- Step 5. Improve

e) **TOOLS AND TECHNIQUES FOR VERIFYING VULNERABILITY FIXES:**

Vulnerability testing is a process of evaluating and identifying security weaknesses in a computer system, network, or software application. It involves systematically scanning, probing, and analyzing systems and applications to uncover potential vulnerabilities, such as coding errors, configuration flaws, or outdated software components.

The main goal of vulnerability testing is to discover and address these security gaps before they can be exploited by attackers, ultimately improving the overall security and resilience of the system.

Vulnerability Testing Methods :

Vulnerability testing methods can be broadly categorized based on the approach taken to identify vulnerabilities. Here's an overview of active testing, passive testing, network testing, and distributed testing:

- **Active Testing**

  Active testing is a vulnerability testing method in which testers interact directly with the target system, network, or application to identify potential security weaknesses. It typically involves sending inputs, requests, or packets to the target and analyzing the responses to discover vulnerabilities.

Active testing can be intrusive and may cause disruptions or performance issues in the target system, but it is usually more effective in finding vulnerabilities than passive testing. Examples of active testing include:

- Port scanning to identify open ports and services running on a network.
- Fuzz testing, which involves sending malformed or unexpected inputs to applications to discover

vulnerabilities related to input validation and error handling.

- **Passive Testing:**

  Passive testing is a non-intrusive vulnerability testing method that involves observing and analyzing the target system, network, or application without directly interacting with it. Passive testing focuses on gathering information about the target, such as network traffic, configuration settings, or application behavior, to identify potential vulnerabilities.

  This method is less likely to cause disruptions or performance issues but may be less effective in finding vulnerabilities compared to active testing. Examples of passive testing include:

  - Traffic monitoring to identify patterns or anomalies that may indicate security weaknesses.
  - Configuration reviews to assess security settings and identify misconfigurations.

- **Network Testing :**

  Network testing is a vulnerability testing method focused on identifying security weaknesses in network infrastructure, including devices, protocols, and configurations. It aims to discover vulnerabilities that could allow unauthorized access, eavesdropping, or Denial of Service (DoS) attacks on the network.

  Network testing typically involves both active and passive testing techniques to evaluate the network's security posture comprehensively. Examples of network testing include:

  - Scanning for open ports and services on network devices.
  - Analyzing network protocols and configurations for security flaws.

- **Distributed Testing:**

  Distributed testing is a vulnerability testing method that involves using multiple testing tools or systems, often deployed across different locations, to scan and analyze the target system, network, or application for vulnerabilities.

This approach can help provide a more comprehensive view of the target's security posture, as it helps identify vulnerabilities that may be visible only from specific locations or under specific conditions. Distributed testing can also help distribute the load of vulnerability testing, reducing the impact on the target system and increasing the efficiency of the testing process.

Examples of distributed testing include:

- Using multiple vulnerability scanners from different locations to scan a web application for potential security flaws.
- Coordinating a team of testers in different geographical locations to perform simultaneous network vulnerability testing.

f) CHALLENGES AND LIMITATIONS OF VULNERABILITY REPRODUCTION INSTRUCTION:

Challenges and limitations of vulnerability reproduction instruction may include differences in system configurations or environments, difficulty in replicating complex vulnerabilities, and the need for access to source code or proprietary systems. It is important to address these challenges to ensure that vulnerabilities are accurately identified and addressed.

Cyber threats and attacks are on the rise for industrial, manufacturing and critical infrastructure organizations. Many of these threats are the result of vulnerabilities present in the organization's OT systems. Targeted threat actors or untargeted ransomware attacks can exploit these vulnerabilities to gain access into industrial networks for financial gain or to interrupt operations.

The rise of cyber threats are due to factors such as the acceleration of IIoT technology and digital transformation such as

Industry 4.0 in manufacturing, an increase in remote work and remote access amid the COVID-19 pandemic, and the prevalence of ransomware.

6) COMPREHENSIVE AND DETAILED REPORTING:

By definition, a comprehensive report is intended to explore a topic or an idea in great detail. In business, comprehensive reports are often used to evaluate and discuss a company's financial situation. Comprehensive reports may be used for other purposes as well, such as summarizing a new business trend or describing a new target market. Learning exactly how to write a comprehensive report can be a useful business skill for employees at any level.

a) IMPORTANCE OF COMPREHENSIVE AND DETAILED REPORTING:

You may be asking yourself how important a comprehensive report is, as well as how it can be helpful. A comprehensive report is often used as an evaluation. An evaluation and a way to discuss the issue. Comprehensive reports are more common in the business world than most. These types of reports are used for discussing, analyzing and evaluating situations. As well as used for the purpose of summarizing things in a specific manner. Regardless of this report mostly used in business, a comprehensive report can still be used in other fields.

Tips for Writing Comprehensive Narrative Reports

When you begin with your comprehensive report, there are things you need to consider. Especially when you have found the difference between comprehensive reports and narrative reports. To be able to write one would also be a good experience and idea. To start, try and check out the following tips.

- Make a draft
- Choose a topic
- Talk about chosen topic
- Explain in full detail
- Write the final output
- Revise and report it

b) KEY COMPONENTS OF COMPREHENSIVE AND DETAILED REPORTING:

- **components of comprehensive and detailed reporting:**

Common elements of reports

- Title:
  Your title should be brief, topic-specific, and informative, clearly indicating the purpose and scope of your study. Include key words in your title so that search engines can easily access your work. For example: *Measurement of water around Station Pier.*
- Abstract:
  An abstract is a concise summarythat helps readers to quickly assess the content and direction of your paper. It should be brief, written in a single paragraph and cover: the scope and purpose of your report; an overview of methodology; a summary of the

main findings or results; principal conclusions or significance of the findings; and recommendations made.

The information in the abstract must be presented in the same order as it is in your report. The abstract is usually written last when you have developed your arguments and synthesised the results.

- Introduction:

The introduction creates the context for your research. It should provide sufficient background to allow the reader to understand and evaluate your study without needing to refer to previous publications. After reading the introduction your reader should understand exactly what your research is about, what you plan to do, why you are undertaking this research and which methods you have used. Introductions generally include:

  - The rationale for the present study. Why are you interested in this topic? Why is this topic worth investigating?
  - Key terms and definitions.
  - An outline of the research questions and hypotheses; the assumptions or propositions that your research will test.

- Literature Review:

Not all research reports have a separate literature review section. In shorter research reports, the review is usually part of the Introduction.

A literature review is a critical survey of recent relevant research in a particular field. The review should be a selection of carefully organised, focused and relevant literature that develops a narrative 'story' about your topic. Your review should answer key questions about the literature:

  - What is the current state of knowledge on the topic?
  - What differences in approaches / methodologies are there?
  - Where are the strengths and weaknesses of the research?
  - What further research is needed? The review may identify a gap in the literature which

provides a rationale for your study and supports your research questions and methodology.

The review is not just a summary of all you have read. Rather, it must develop an argument or a point of view that supports your chosen methodology and research questions.

c) STRATEGIES FOR EFFECTIVE REPORTING:

Strategies for effective reporting include identifying the purpose and scope of the report, understanding the audience's needs, selecting appropriate data sources and analysis techniques, and using clear and concise language to present findings. The report should Be well-organized, visually appealing, and use data visualization tools to help the audience better understand the data.

D) CHALLENGES IN IMPLEMENTING COMPREHENSIVE AND DETAILED REPORTING:

Challenges in implementing comprehensive and detailed reporting include data quality issues, data silos, lack of resources,
and difficulty in identifying the right metrics to measure. Organizations also face challenges in presenting data in a way that is easily
digestible for different stakeholders, such as executives, managers, and frontline employees.

e) **IMPACT OF COMPREHENSIVE AND DETAILED REPORTING ON DECISION MAKING:**

Comprehensive and detailed reporting can have a significant impact on decision-making by providing stakeholders with the information they need to make informed decisions. It can help identify areas for improvement, highlight potential risks, and guide resource allocation. By providing a comprehensive view of an organization's operations, financial performance, and overall health, stakeholders can make more informed decisions that align with their strategic goals.

f) BEST PRACTICES FOR CREATING COMPREHENSIVE AND DETAILED REPORTS:

- Be Strategist With Your Reporting
- Be Consistent In Your Reports
- Simplify The Data Being Collected
- Coordinate With Team Members To Ensure Consistency of Data
- Set Milestones
- Keep Database Records
- Reassess and Reevaluate Internal Reporting Practices

CONCLUSION

In conclusion, Network Vulnerability Assessments (NVA) is how you determine whether your network is secure against malicious attacks. It involves identifying possible weaknesses in your network and then taking steps to fix them. The goal of any Vulnerability is to ensure that no single point of failure exists on your network.

You can conduct an Network Vulnerability using many different approaches, and each process has its strengths and weaknesses. Therefore, it's important to choose the right method based on your needs.