Module Objectives

Module Objectives

- Understanding Web Server Concepts
- Understanding Web Server Attacks
- Understanding Web Server Attack Methodology
- Overview of Web Server Attack Tools
- Understanding Different Web Server Attack Countermeasures
- Understanding Patch Management Concepts
- Overview of Web Server Security Tools
- Overview of Web Server Penetration Testing

Module Flow

Web Server Operations

A web server is a computer system that **stores**, **processes** and **delivers web pages** to clients via HTTP

Components of a Web Server

- **Document Root**: Stores critical HTML files related to the web pages of a domain name that will be served in response to the requests
- **Server Root**: Stores server's configuration, error, executable and log files
- **Virtual Document Tree**: Provides storage on a different machine or disk after the original disk is filled up
- **Virtual Hosting**: Technique of hosting multiple domains or websites on the same server
- **Web Proxy**: Proxy server that sits in between the web client and web server to prevent IP blocking and maintain anonymity

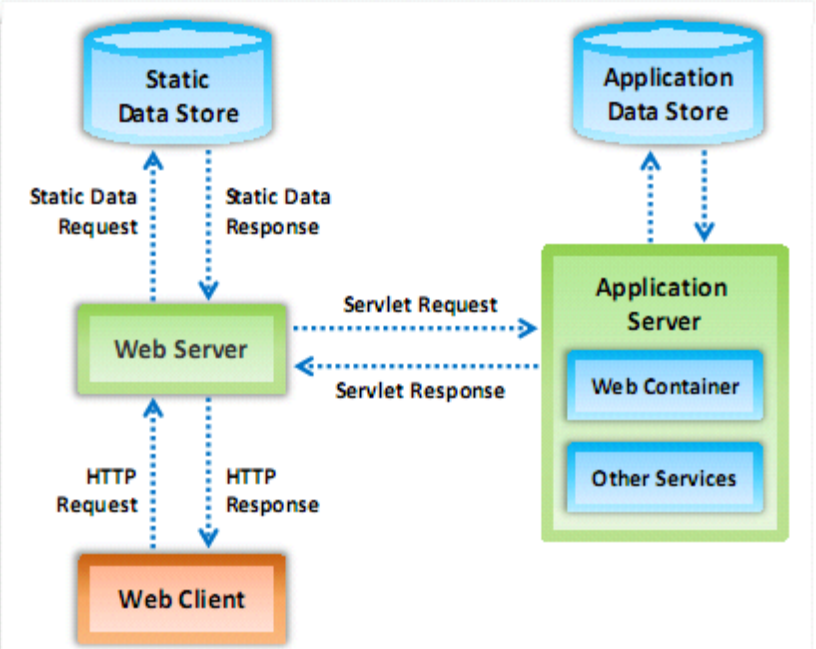
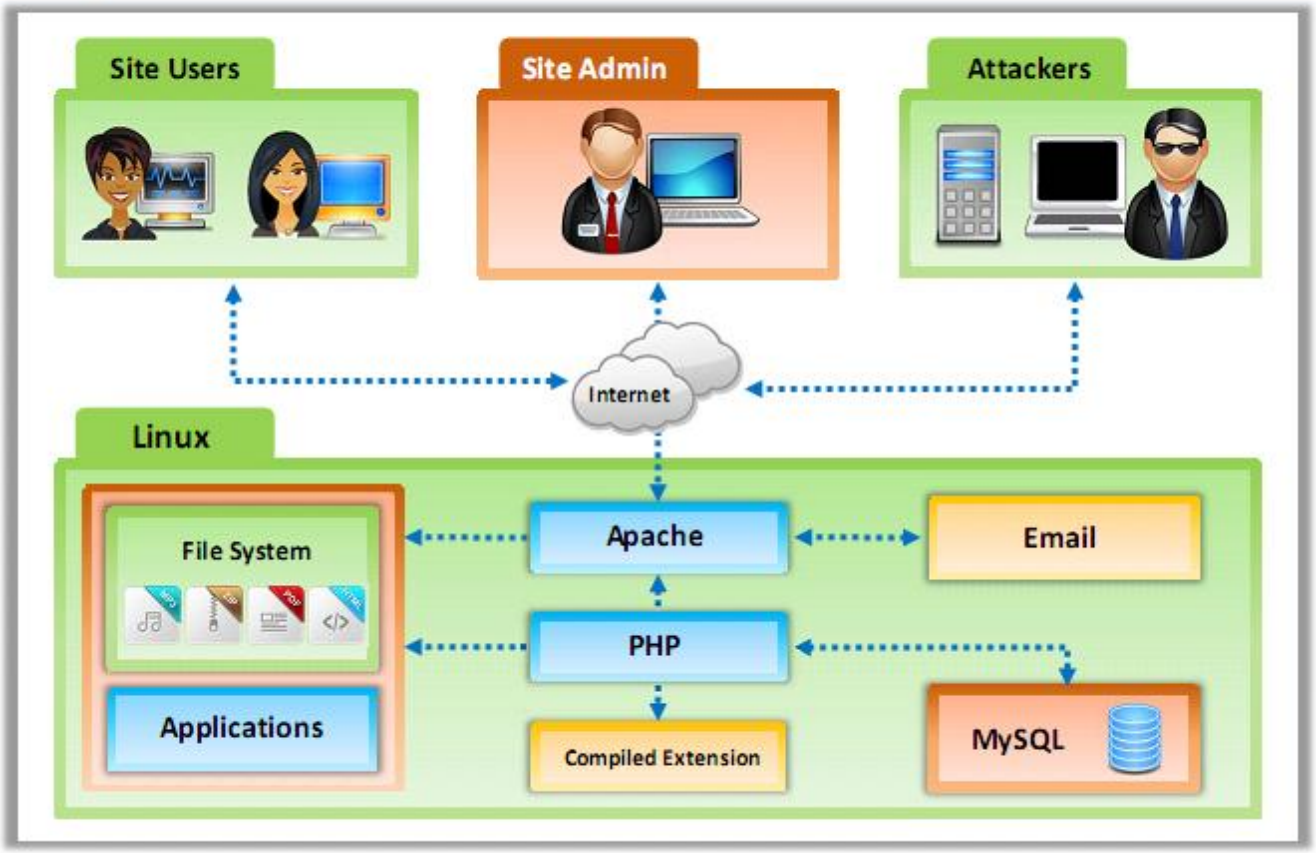
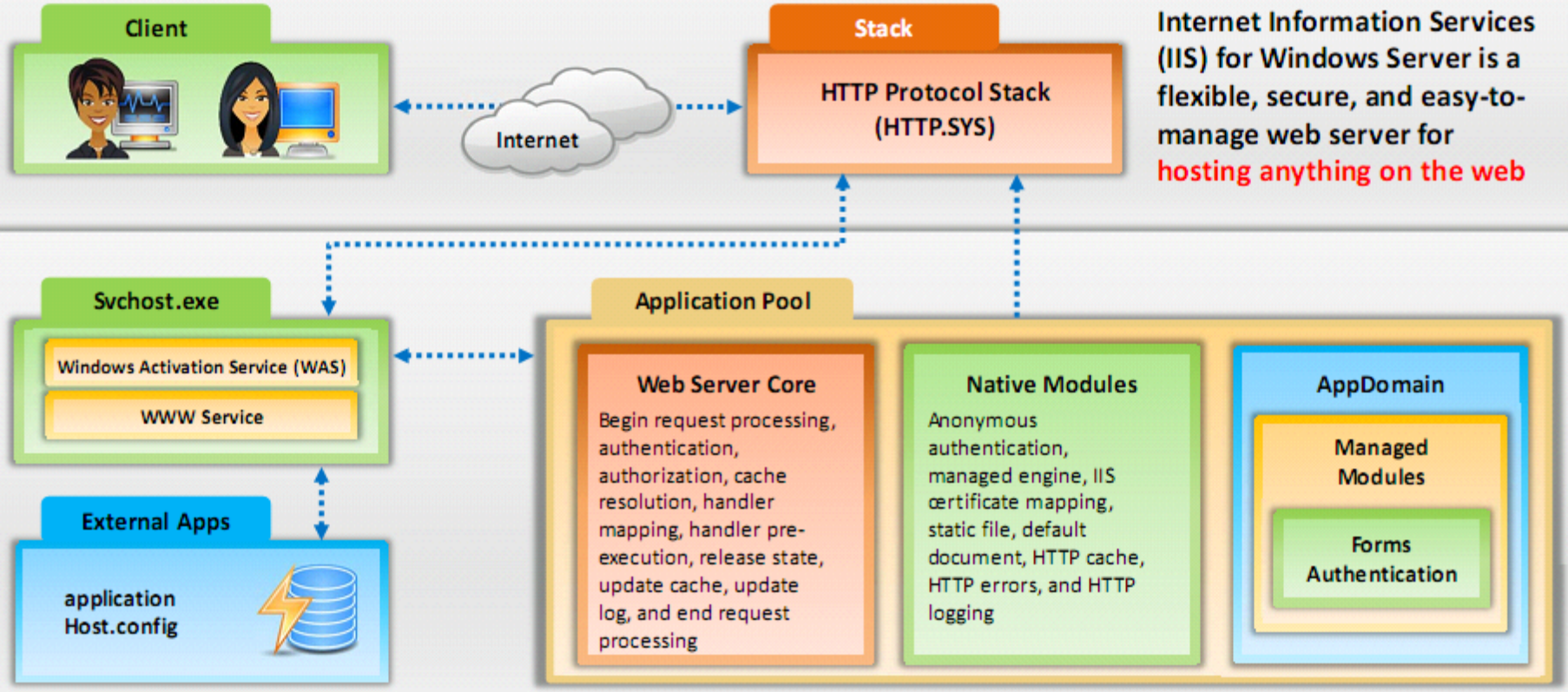


Figure: Typical Client-Server Web Server Operation

Open Source Web Server Architecture

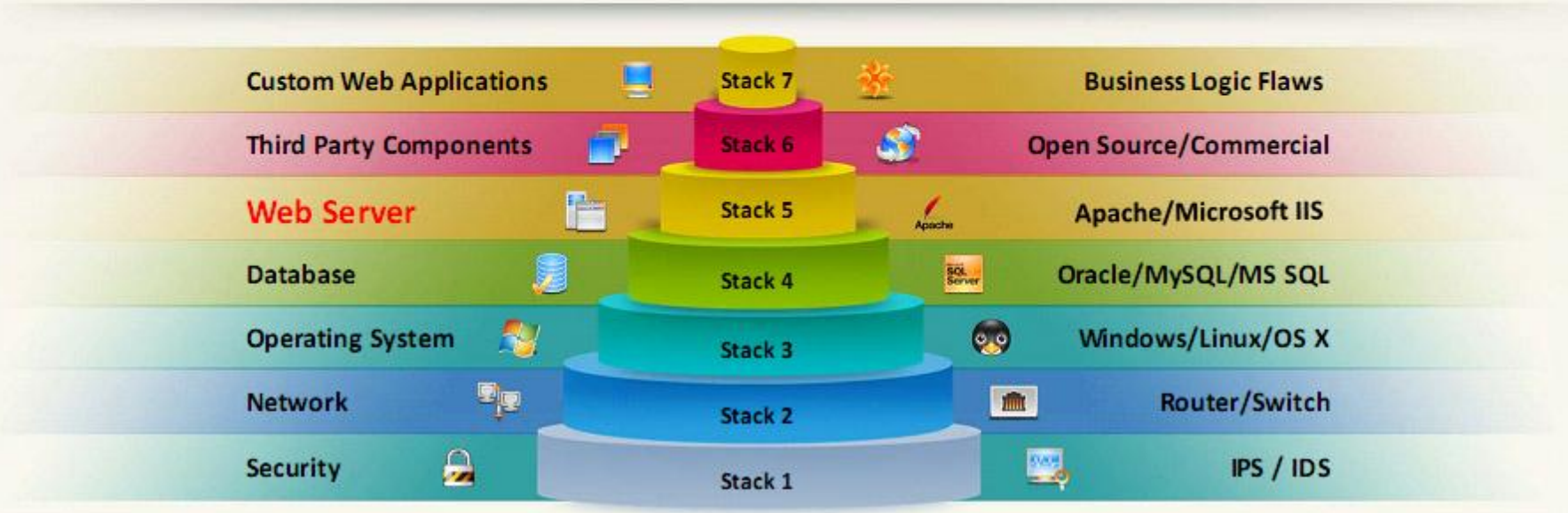


IIS Web Server Architecture



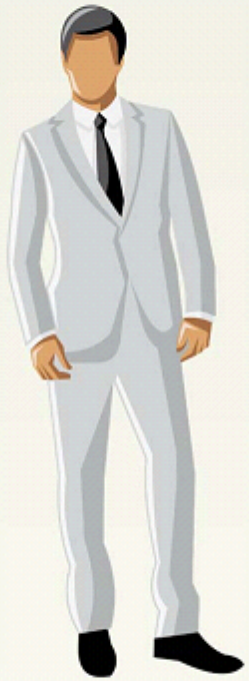
Web Server Security Issue

- Attackers usually target **software vulnerabilities** and configuration errors to compromise web servers
- **Network** and **OS level attacks** can be well defended using proper network security measures such as firewalls, IDS, etc. However, web servers are accessible from anywhere on the web, which makes them **more vulnerable** to attacks



Why Web Servers Are Compromised?

- **Improper** file and directory **permissions**
- Installing the server with **default settings**
- **Unnecessary services** enabled, including content management and remote administration
- **Security conflicts** with business ease-of-use case
- **Lack of proper security policy**, procedures, and maintenance
- **Improper authentication** with external systems
- **Default accounts** with their default passwords, or no passwords
- **Unnecessary** default, backup, or sample **files**
- **Misconfigurations** in web server, operating systems, and networks
- **Bugs** in server software, OS, and web applications
- **Misconfigured SSL certificates** and encryption settings
- Administrative or **debugging functions** that are **enabled** or accessible on web servers
- Use of **self-signed certificates** and default certificates



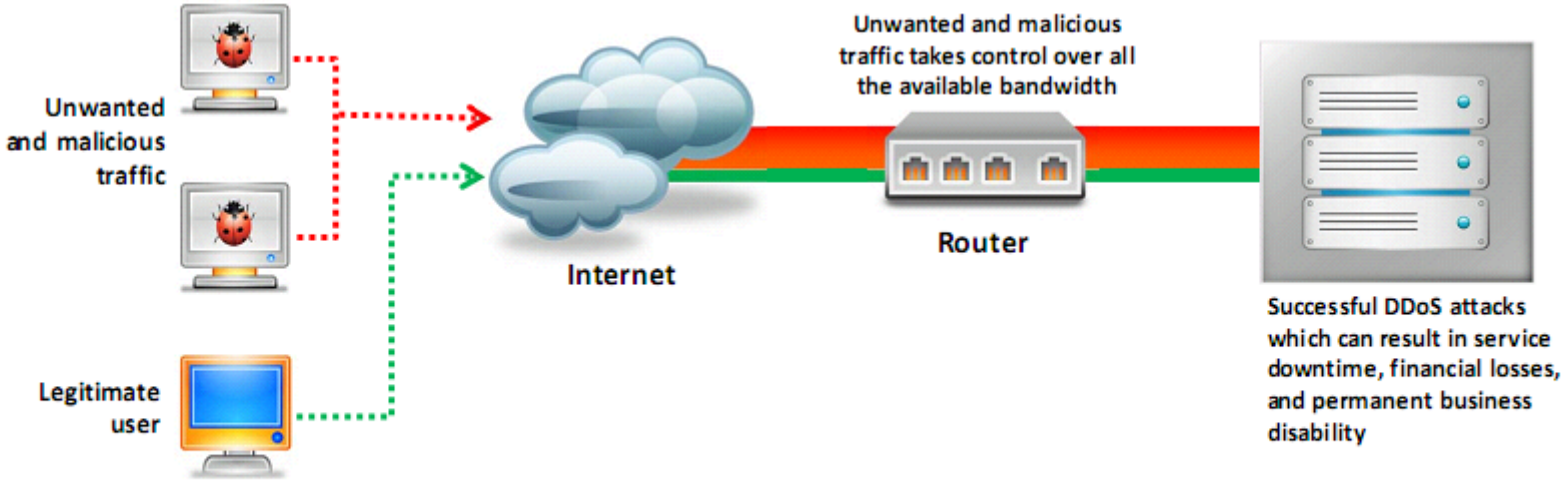
Impact of Web Server Attacks

1. Compromise of user account
2. Website defacement
3. Secondary attacks from the website
4. Root access to other applications or servers
5. Data tampering and data theft

Module Flow

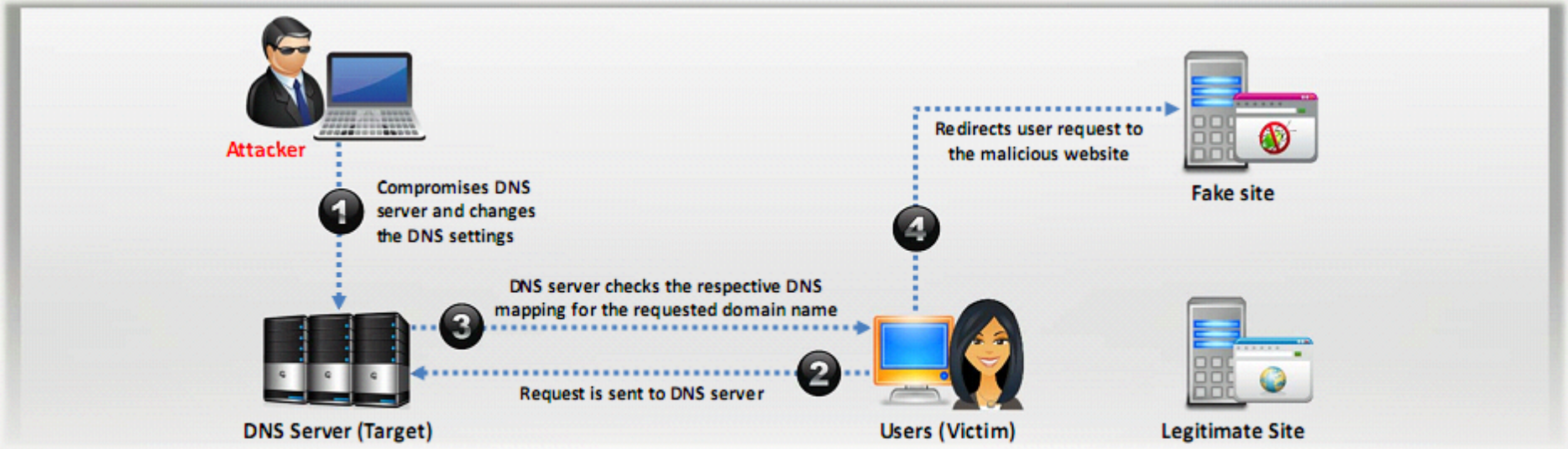
DoS/DDoS Attacks

- Attackers may send numerous **fake requests** to the web server which results in the **web server crashing** or becoming unavailable to the legitimate users
- Attackers may target **high profile web servers** such as banks, credit card payment gateways, government owned services, etc. to **steal user credentials**



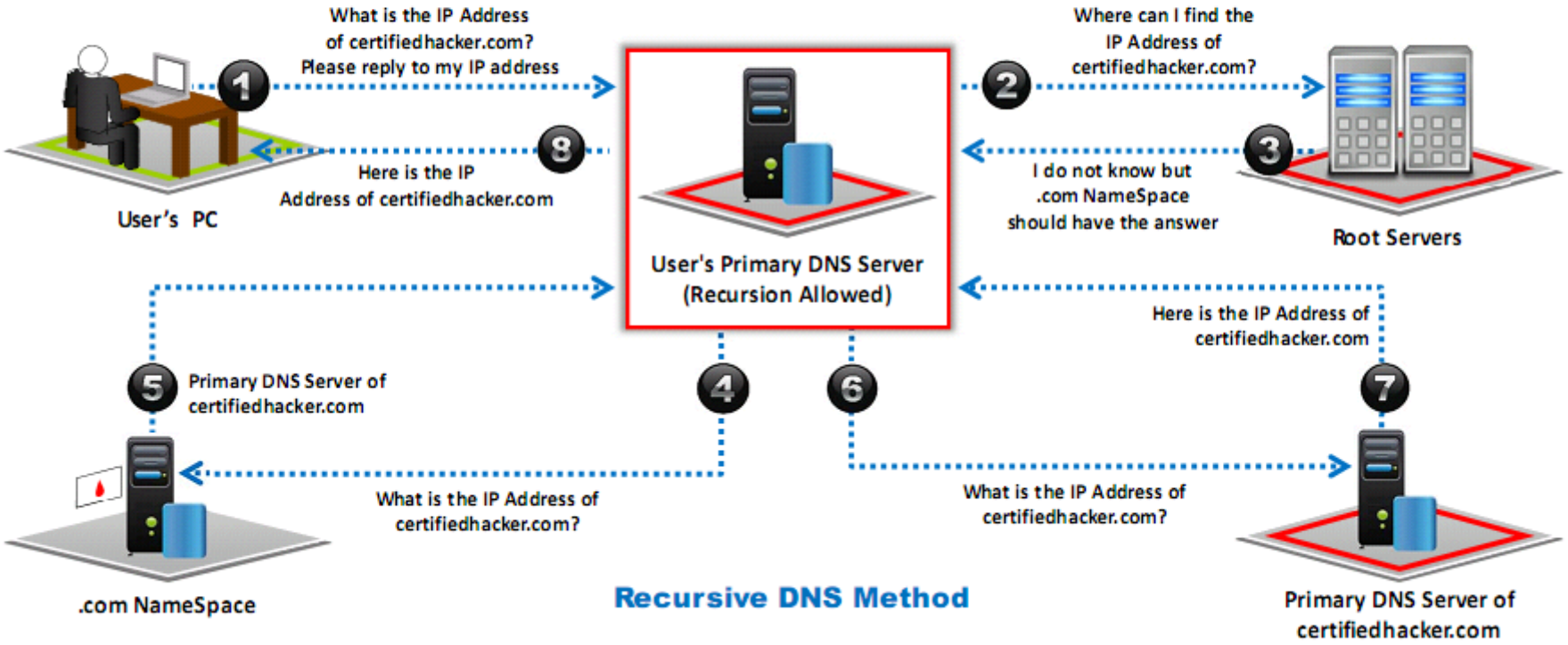
DNS Server Hijacking

Attacker compromises DNS server and **changes the DNS settings** so that all the requests coming towards the target web server are redirected to his/her own malicious server



DNS Amplification Attack

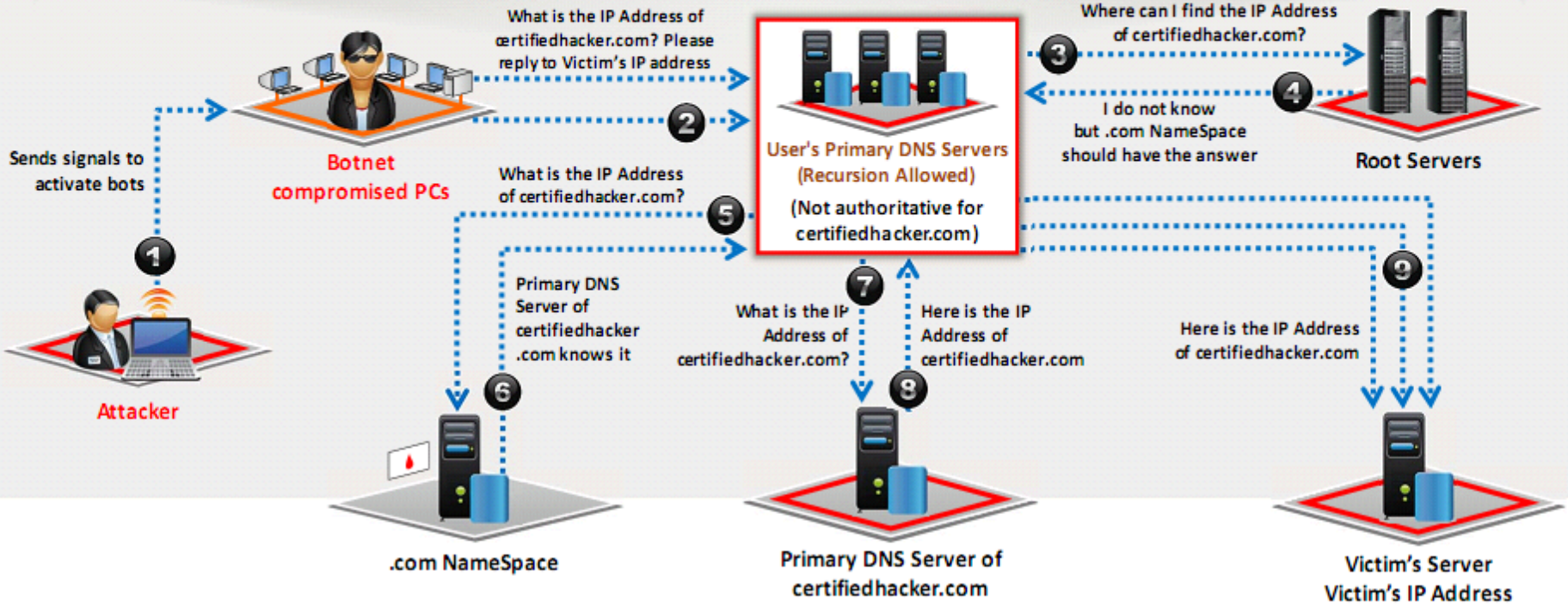
- Attacker takes advantage of **DNS recursive method** of DNS redirection to perform DNS amplification attack



Recursive DNS Method

DNS Amplification Attack (Cont'd)

Attacker uses compromised PCs with **spoofed IP addresses** to amplify the DDoS attacks on victims' DNS server by exploiting DNS recursive method



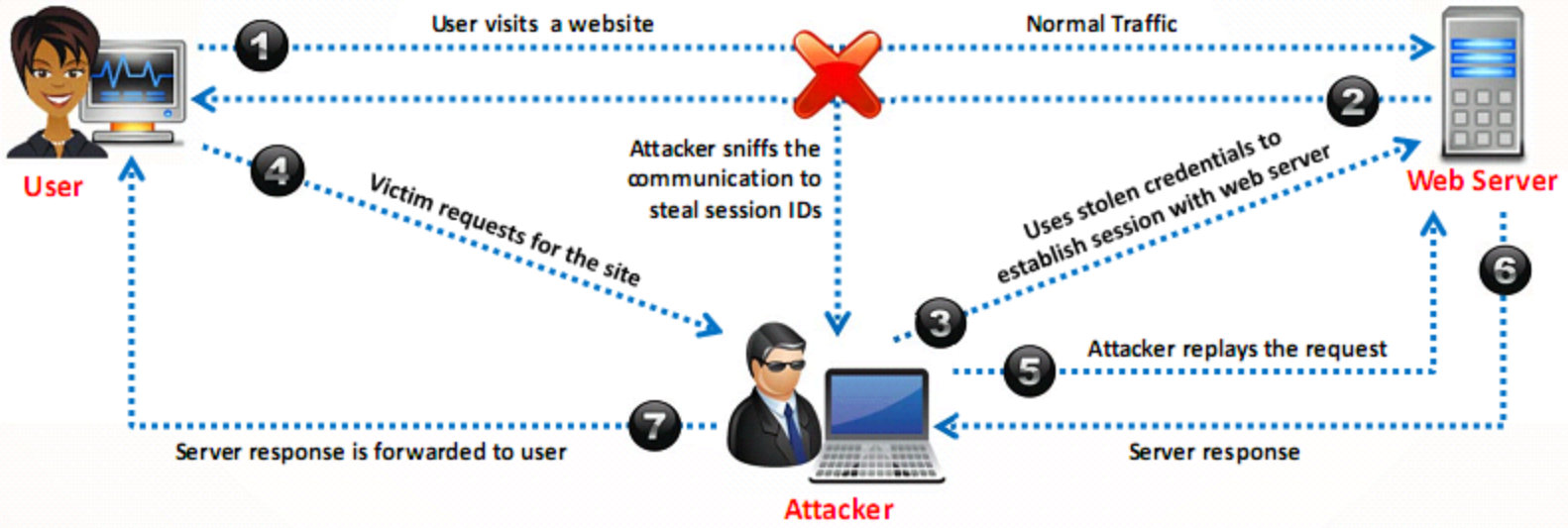
Directory Traversal Attacks

- In directory traversal attacks, attackers use **../ (dot-dot-slash)** sequence to access restricted directories outside of the web server root directory
- Attackers can use **trial and error method** to navigate outside of the root directory and access sensitive information in the system

Man-in-the-Middle/Sniffing Attack

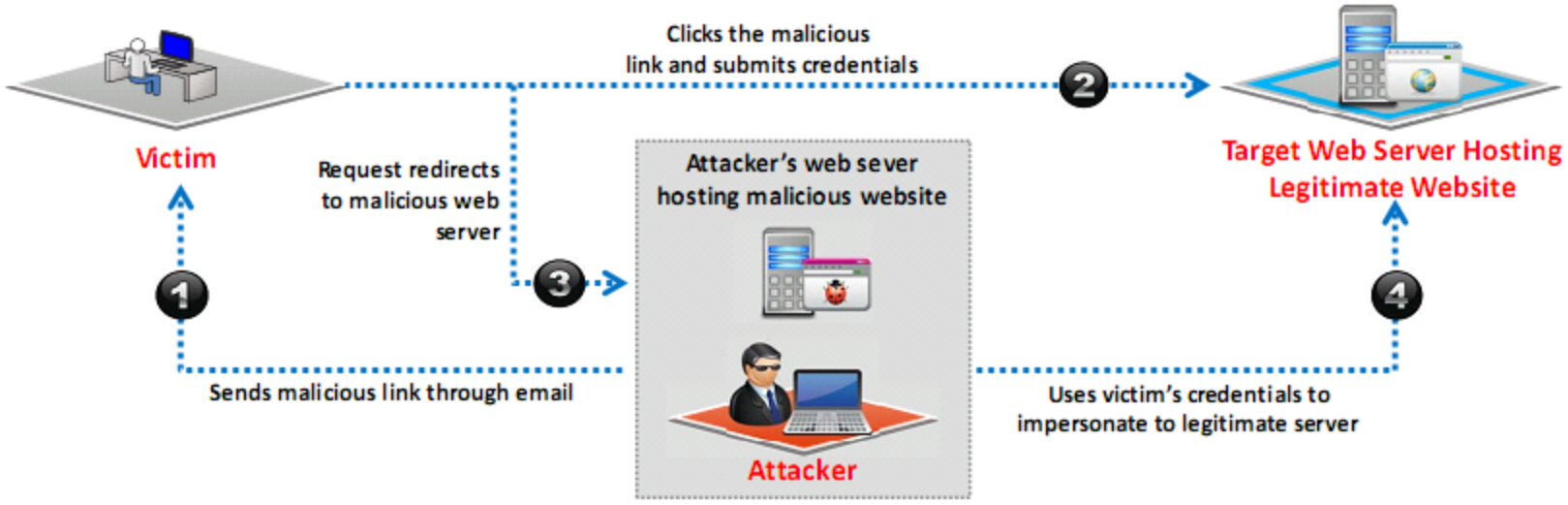
01 Man-in-the-Middle (MITM) attacks allow an attacker to access sensitive information by **intercepting and altering communications** between an end-user and web servers

02 Attacker **acts as a proxy** such that all the communication between the user and web server passes through him



Phishing Attacks

- Attacker tricks user to submit **login details** for website that looks legitimate, but it redirect to the malicious website hosted on attacker web server
- Attacker **steals the credentials** entered and uses it to impersonate the user with the website hosted on the legitimate target server
- Attacker then can perform **unauthorized** or **malicious operations** with the website target server



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 **MYSQL 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

Web Server Misconfigurations

- Verbose Debug/Error Messages
- Anonymous or Default Users/Passwords
- Sample Configuration and Script Files
- Remote Administration Functions
- Unnecessary Services Enabled
- Misconfigured/Default SSL Certificates

Web Server Misconfiguration Examples

- This configuration allows anyone to view the **server status** page, which contains detailed information about the current use of the web server, including information about the **current hosts** and requests being processed

httpd.conf file on an **Apache** server

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

- This configuration gives **verbose error messages**

php.ini file

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

HTTP Response Splitting Attack

HTTP response splitting attack involves **adding header response data into the input field** so that the server splits the response into two responses

The attacker can **control the first response to redirect the user to a malicious website** whereas the other responses will be discarded by the web browser

Server Code

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

Input = Jason

HTTP/1.1 200 OK
...
Set-Cookie: author=Jason
...

Input = JasonTheHacker\r\nHTTP/1.1 200 OK\r\n

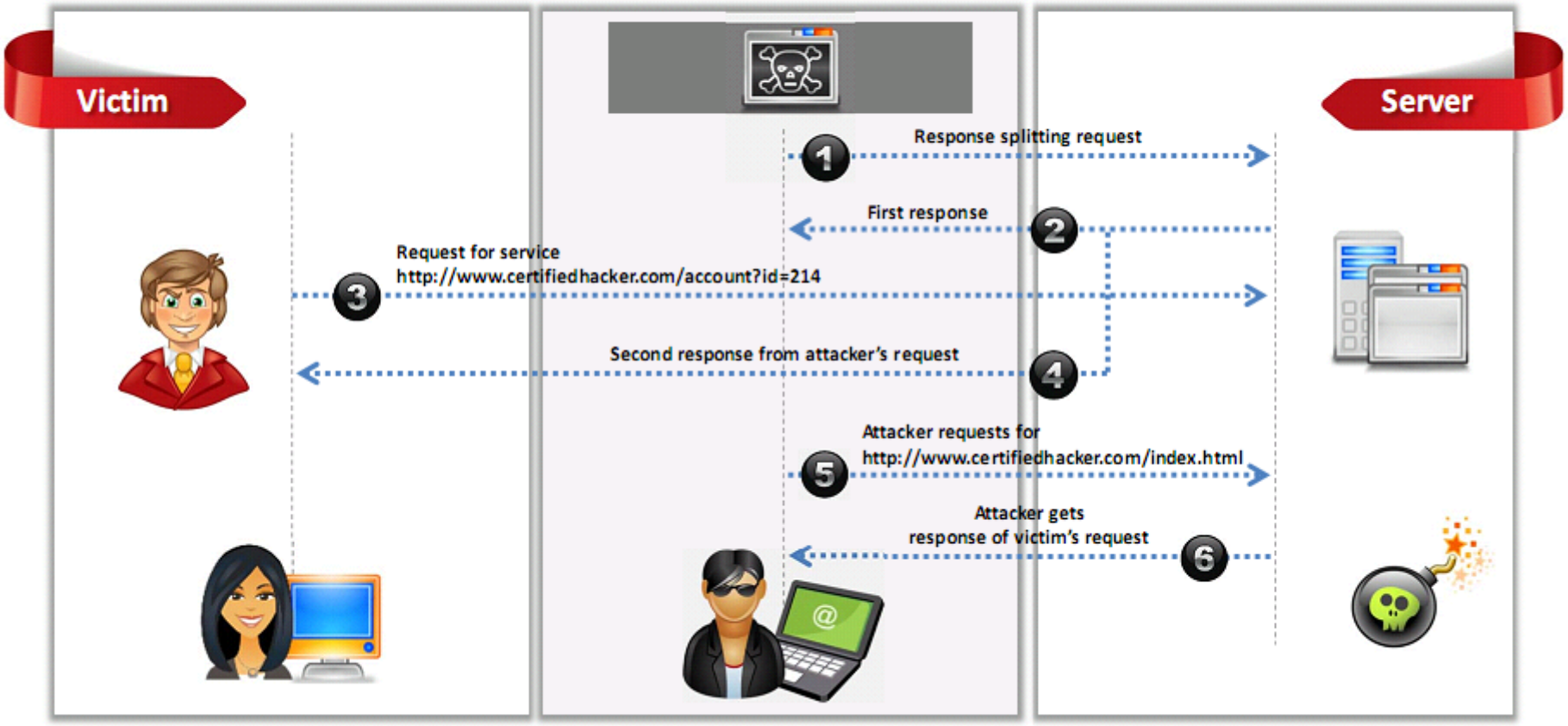
First Response (Controlled by Attacker)

Set-Cookie: author=JasonTheHacker
HTTP/1.1 200 OK
...

Second Response

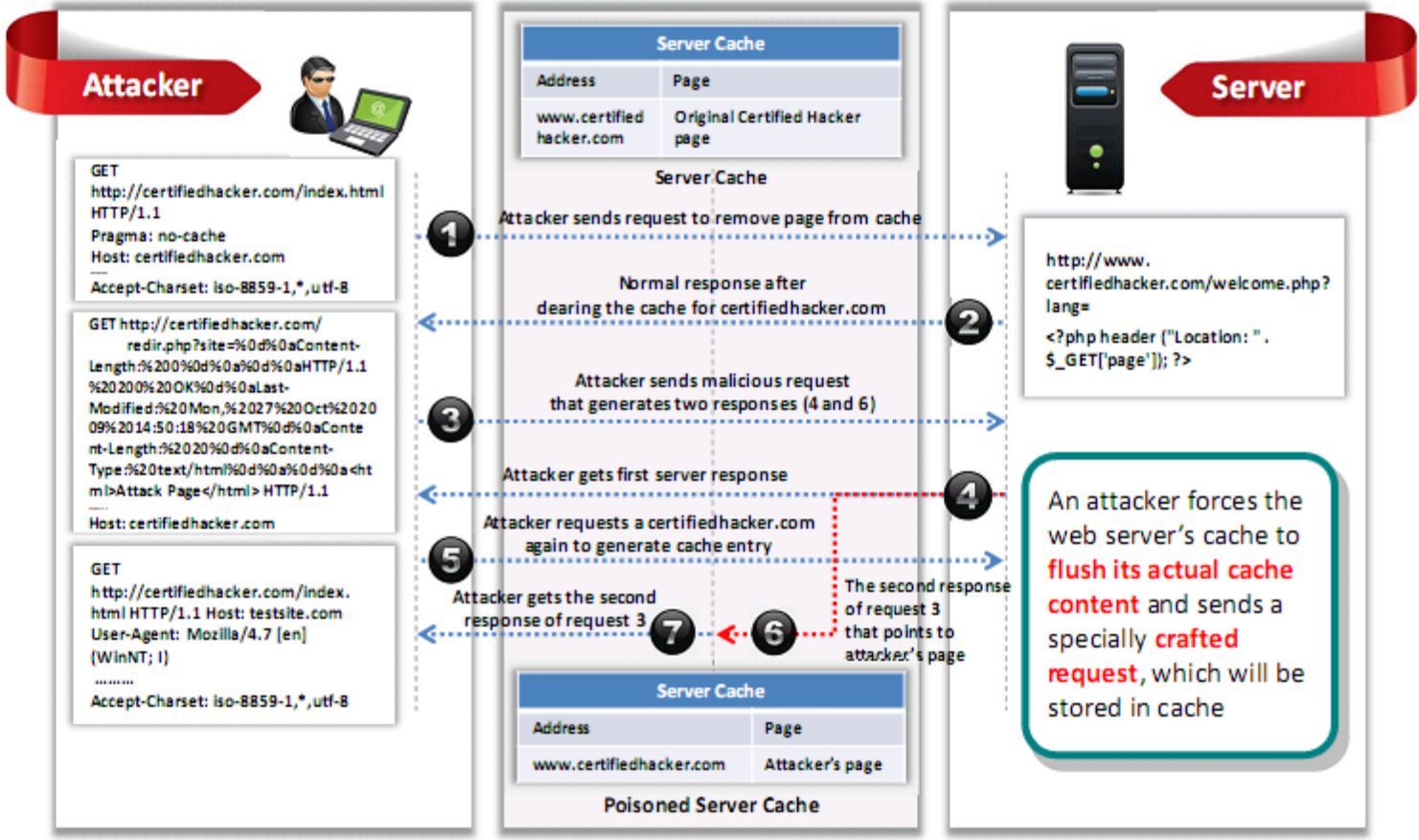
HTTP/1.1 200 OK
...

HTTP Response Splitting Attack (Cont'd)



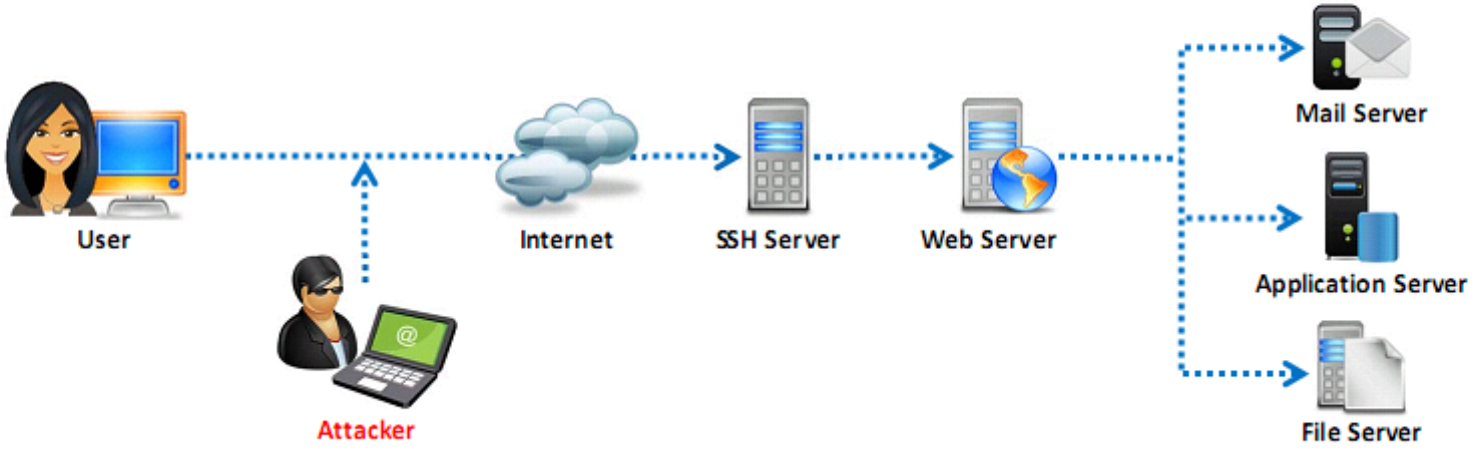
Web Cache Poisoning Attack

- Web cache poisoning attacks the **reliability of an intermediate web cache source**
- In this attack, the attackers **swap cached content** for a random URL with infected content
- Users of the web cache source can **unknowingly use the poisoned content** instead of the true and secured content when requesting the required URL through the web cache



SSH Brute Force Attack

1. SSH protocols are used to create an encrypted SSH tunnel between two hosts in order to transfer unencrypted data over an insecure network
2. Attackers can brute force SSH login credentials to gain unauthorized access to a SSH tunnel
3. SSH tunnels can be used to transmit malwares and other exploits to victims without being detected



Web Server Password Cracking

- An attacker tries to exploit weaknesses to hack **well-chosen passwords**
- The most **common passwords** found are password, root, administrator, admin, demo, test, guest, qwerty, pet names, etc.

Attacker targets mainly for:

- SMTP servers
- Web shares
- SSH Tunnels
- Web form authentication cracking
- FTP servers

- Attackers use different methods such as **social engineering**, **spoofing**, **phishing**, using a Trojan Horse or virus, wiretapping, keystroke logging, etc.

- Many hacking attempts start with **cracking passwords** and prove to the web server that they are a valid user

- Passwords can be cracked **manually** by guessing or by performing dictionary, brute force, hybrid attacks using **automated tools** such as Cain & Abel, THC Hydra, etc.

Web Application Attacks

- Vulnerabilities in **web applications** running on a web server provide a broad attack path for web server compromise

Parameter/Form Tampering	Cookie Tampering	Unvalidated Input and File Injection Attacks
Session Hijacking	SQL Injection Attacks	Directory Traversal
Denial-of-Service (DoS) Attack	Cross Site Scripting (XSS) Attacks	Buffer Overflow Attacks
Cross Site Request Forgery (CSRF) Attack	Command Injection Attacks	Source Code Disclosure

Note: For complete coverage of web application attacks refer to Module 14: Hacking Web Applications

Module Flow

1. Web Server Concepts
2. Web Server Attacks
3. Web Server Attack Methodology
4. Web Server Attack Tools
5. Counter-measures
6. Patch Management
7. Web Server Security Tools
8. Web Server Pen Testing

Web Server Attack Methodology

01 Information Gathering

02 Web Server Footprinting

03 Website Mirroring

04 Vulnerability Scanning

05 Session Hijacking

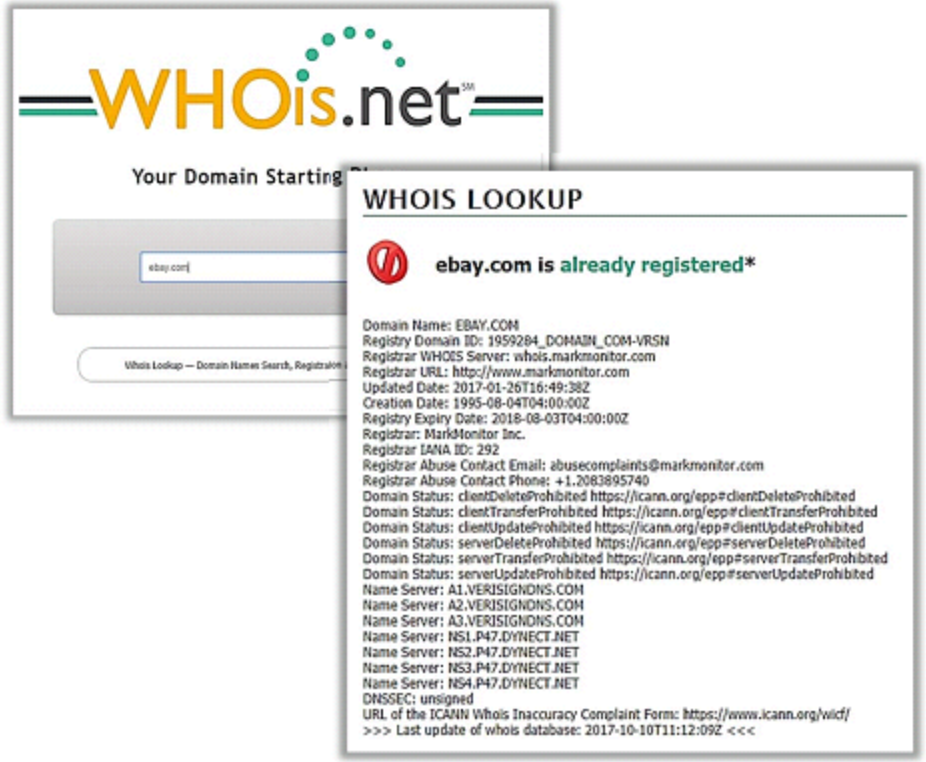
06 Web Server Passwords Hacking

Information Gathering

1. Information gathering involves collecting information about the **targeted company**

2. Attackers search the **Internet**, **newsgroups**, **bulletin boards**, etc. for information about the company

3. Attackers use **Whois.net**, **Whois Lookup**, etc. tools and query the Whois databases to get details such as a domain name, an IP address, or an autonomous system number



https://www.whois.net

Note: For complete coverage of information gathering techniques refer to Module 02: Footprinting and Reconnaissance

Information Gathering from Robots.txt File

- The robots.txt file contains the **list of the web server directories and files** that the web site owner wants to hide from web crawlers
- An attacker can simply request Robots.txt file from the URL and retrieve sensitive information such as **root directory structure**, **content management system information**, etc., about the target website

robots.txt - Notepad

File Edit Format View Help

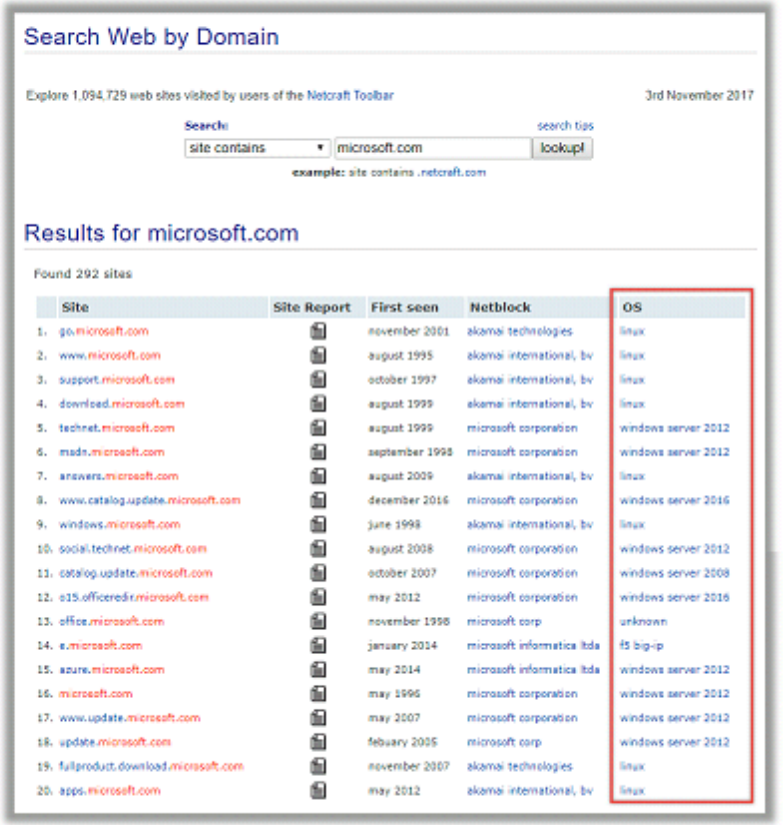
```
User-agent: *
Disallow: /en-us/windows/si/matrix.html
Disallow: /en-us/windows/si/matrix.html
Disallow: /*/security/search-results.aspx?
Disallow: /*/music/*/search/
Disallow: /*/search/
Disallow: /*/music/*/Search/
Disallow: /*/Search/
Disallow: /*/newsearch/
Disallow: *action=catalogsearch&
Allow: /*/store/*/search/
Allow: /*/store/*/layout/
Allow: /*/store/music/groove-music-pass/*
Allow: *action=catalogsearch&catalog_mode=grid&page=2$
Allow: *action=catalogsearch&catalog_mode=grid&page=3$
Allow: *action=catalogsearch&catalog_mode=grid&page=4$
Allow: *action=catalogsearch&catalog_mode=grid&page=5$
Allow: *action=catalogsearch&catalog_mode=grid&page=6$
Allow: *action=catalogsearch&catalog_mode=grid&page=7$
Allow: *action=catalogsearch&catalog_mode=grid&page=8$
Allow: *action=catalogsearch&catalog_mode=list&page=2$
Allow: *action=catalogsearch&catalog_mode=list&page=3$
Allow: *action=catalogsearch&catalog_mode=list&page=4$
Allow: *action=catalogsearch&catalog_mode=list&page=5$
Allow: *action=catalogsearch&catalog_mode=list&page=6$
Allow: *action=catalogsearch&catalog_mode=list&page=7$
Allow: *action=catalogsearch&catalog_mode=list&page=8$
Disallow: *action=accessorysearch&product=*&*
Allow: *action=accessorysearch&product=*$
Disallow: *action=accessorysearch&
```

Web Server Footprinting/Banner Grabbing

1. Gather **valuable system-level data** such as account details, operating system, software versions, server names, and database schema details

2. **Telnet** a web server to footprint a web server and gather information such as server name, server type, operating systems, applications running, etc.

3. Use tool such as **Netcraft**, **httprecon**, and **ID Serve** to perform footprinting



https://www.netcraft.com

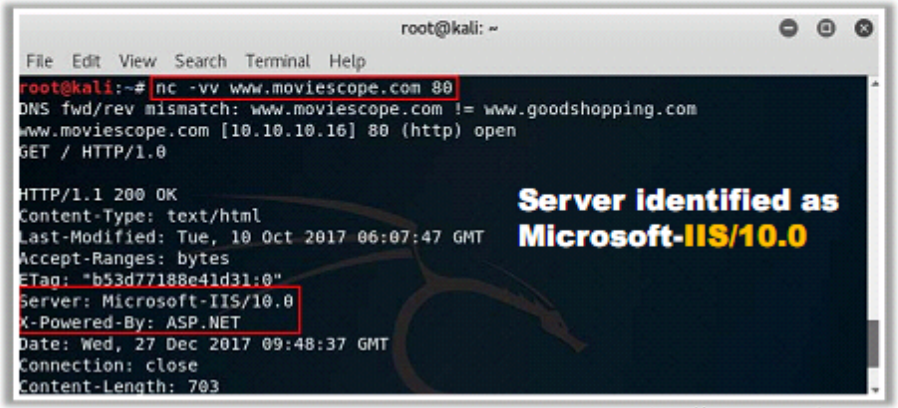
Web Server Footprinting Tools

Netcat

This utility **reads and writes data across network connections**, using the TCP/IP protocol

`# nc -vv www.moviescope.com 80` – press **[Enter]**

`GET / HTTP/1.0` - Press **[Enter]** twice



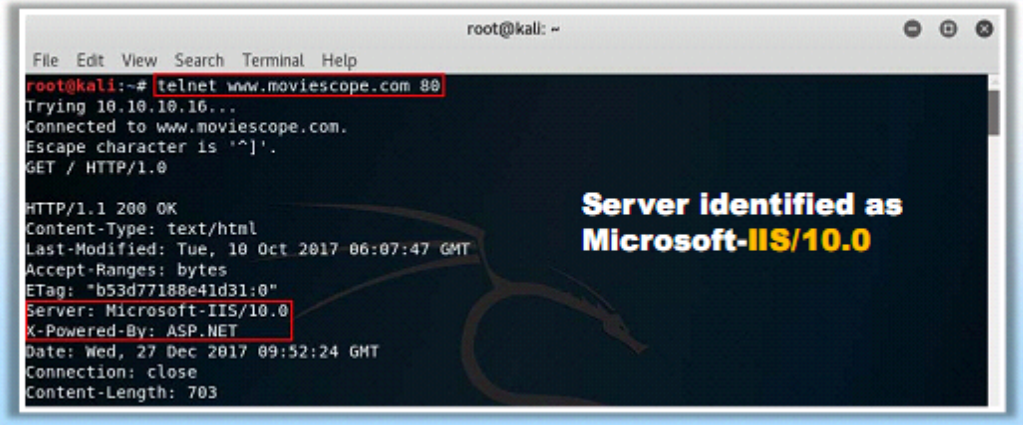
http://netcat.sourceforge.net

Telnet

This technique probes **HTTP servers** to determine the **Server field** in the HTTP response header

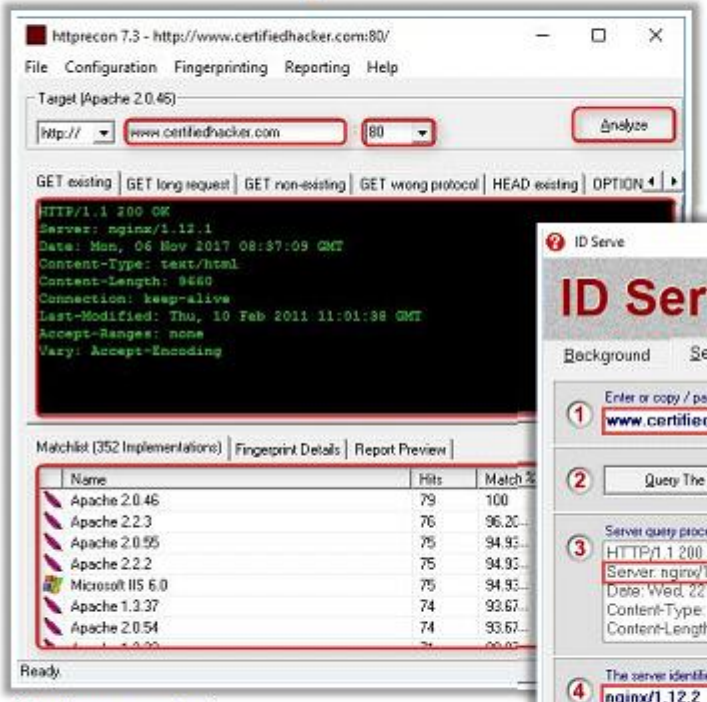
`telnet www.moviescope.com 80` – press **[Enter]**

`GET / HTTP/1.0` - Press **[Enter]** twice



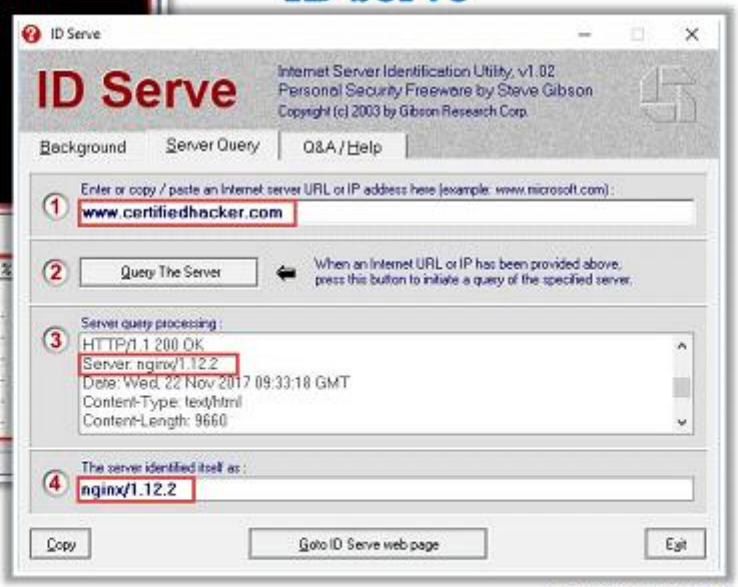
Web Server Footprinting Tools (Cont'd)

httpprecon



http://www.computec.ch

ID Serve



https://www.grc.com

Web Server Footprinting Tools

- Recon-ng (*https://bitbucket.org*)
- Uniscan (*https://sourceforge.net*)
- SpiderFoot (*http://www.spiderfoot.net*)
- httprint (*http://www.net-square.com*)
- Nmap (*https://nmap.org*)
- ScanLine (*https://www.mcafee.com*)
- X probe (*https://sourceforge.net*)
- POf (*https://github.com*)
- Satori (*http://chatteronthewire.org*)

Enumerating Web Server Information Using Nmap

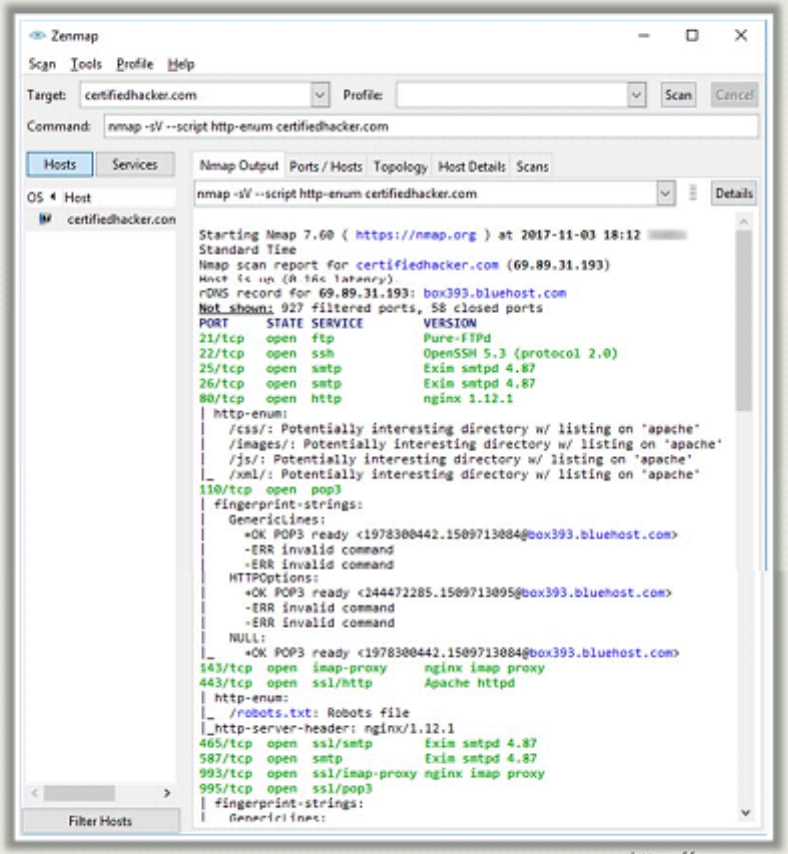
1. Attackers can use advanced **Nmap commands** and **Nmap Scripting Engine (NSE) scripts** to enumerate information about the target website

2. `nmap sV -O -p target IP address`

3. `nmap -sV --script=http-enum target IP address`

4. `nmap target IP address -p 80 --script = http-frontpage-login`

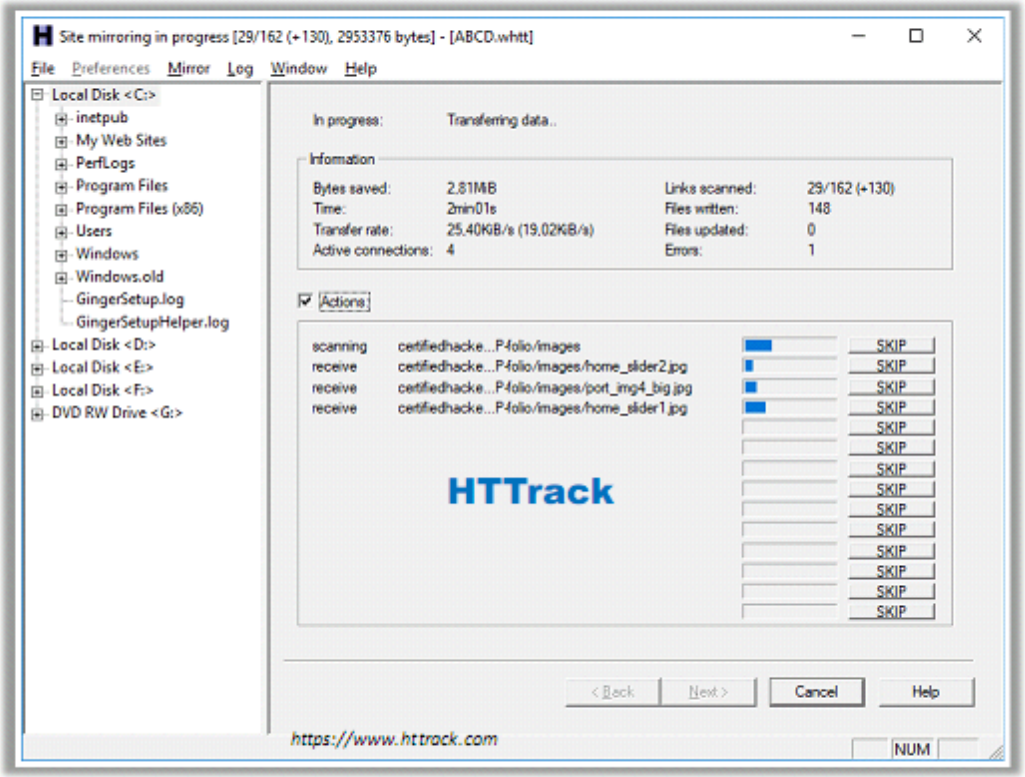
5. `nmap --script http-passwd --script-args http-passwd.root =/ target IP address`



https://nmap.org

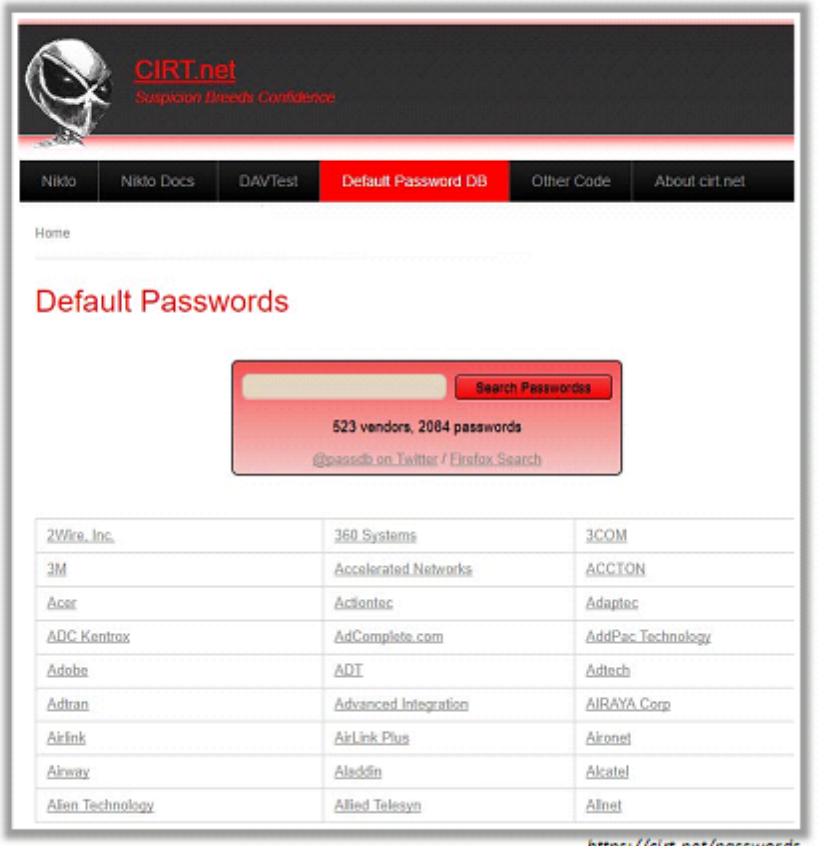
Website Mirroring

- Mirror a website to create a complete profile of the site's **directory structure**, **files structure**, **external links**, etc.
- Search for comments and other items in the **HTML source code** to make footprinting activities more efficient
- Use tools such as **HTTrack**, **WebCopier Pro**, etc. to mirror a website



Finding Default Credentials of Web Server

- Many web server administrative interfaces are **publically accessible** and are located in the **web root** directory
- Often these administrative interface credentials are **not properly configured** and remain **set to default**
- Attackers attempt to **identify the running application interface** and performs following techniques to identify the default login credentials:
  - Consult the **administrative interface documentation** and identify the default passwords
  - Use **Metasploit's built-in database** to scan the server
  - Use online resources like **Open Sez Me** (*http://open-sez.me*), **cirt.net** (https://cirt.net/passwords), etc.
  - Attempt **password guessing** and **brute-forcing** attacks

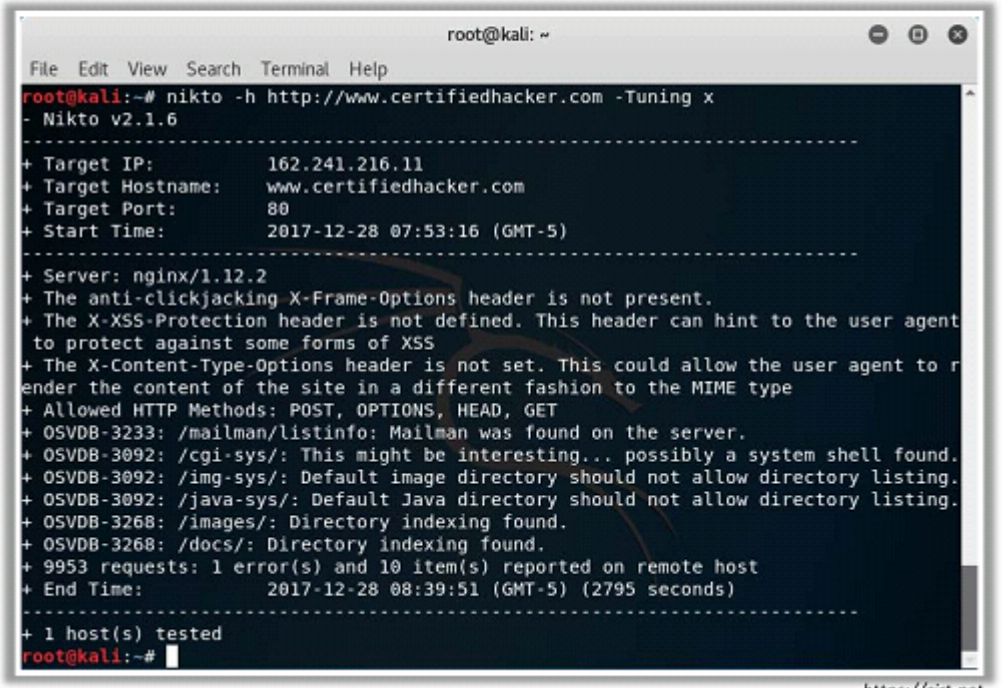


https://cirt.net/passwords

Finding Default Content of Web Server

- Most of the web application servers contain **default content and functionalities** allowing attackers to leverage attacks

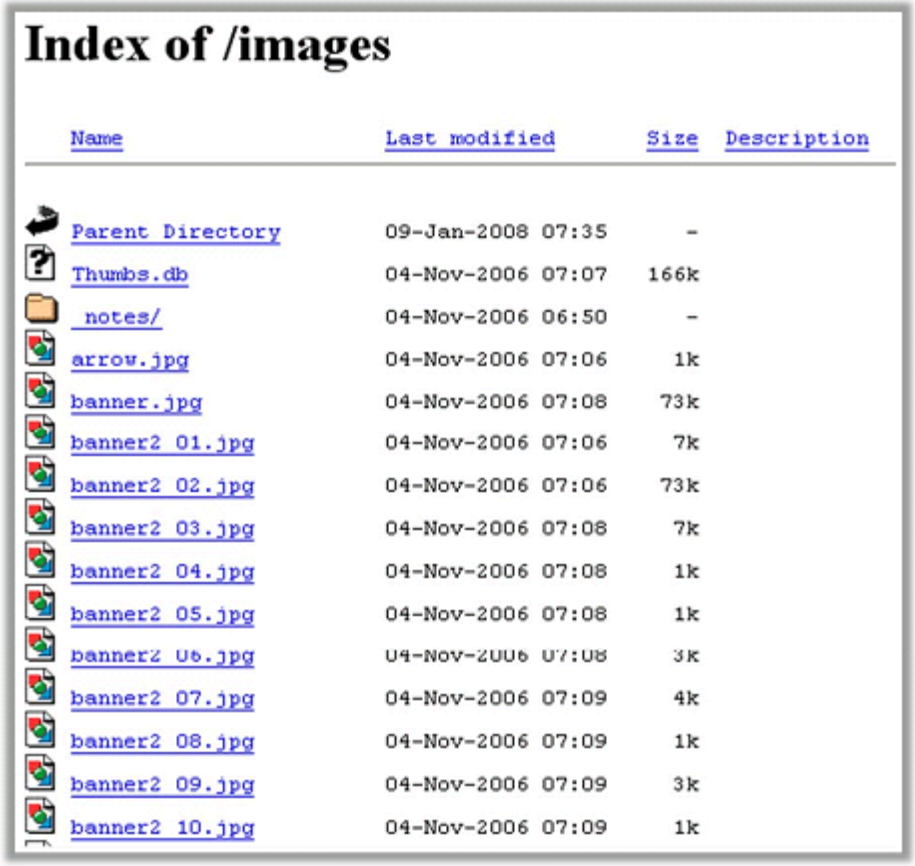
- Check for the following default contents and functionalities in the web servers
  - Administrators **debug and test functionality**
  - **Sample functionality** to demonstrate common tasks
  - Publically accessible **powerful functions**
  - Server **Installation manuals**
- Use tools like **Nikto2** (*https://cirt.net*) and exploit databases like **SecurityFocus** (http://www.securityfocus.com) to identify the default content



https://cirt.net

Finding Directory Listings of Web Server

- When a web server receives a request for the directory, it responds to the request in the following ways
  - Return default resource within directory
  - Return error
  - Return listing of directory content
- Directory listings sometimes possess the following vulnerabilities that allow the attackers to compromise web server
  - Improper access controls
  - Unintentional access to web root of servers
- After discovering the directory on the web server, make a request for the same directory and try to access the directory listings
- Try to exploit vulnerable web server software that gives access to the directory listings

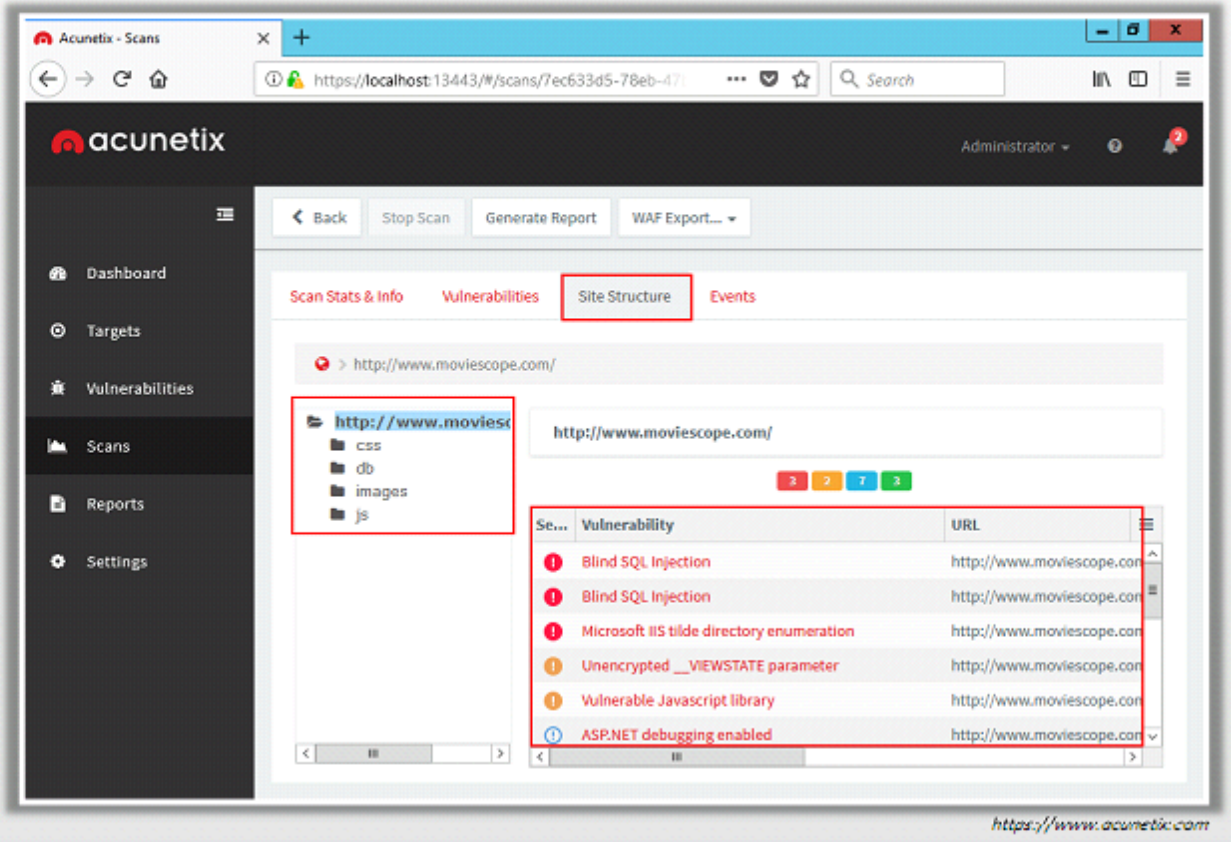


Index of /images

Name	Last modified	Size	Description
Parent Directory	09-Jan-2008 07:35	-	
Thumbs.db	04-Nov-2006 07:07	166k	
notes/	04-Nov-2006 06:50	-	
arrow.jpg	04-Nov-2006 07:06	1k	
banner.jpg	04-Nov-2006 07:08	73k	
banner2_01.jpg	04-Nov-2006 07:06	7k	
banner2_02.jpg	04-Nov-2006 07:06	73k	
banner2_03.jpg	04-Nov-2006 07:08	7k	
banner2_04.jpg	04-Nov-2006 07:08	1k	
banner2_05.jpg	04-Nov-2006 07:08	1k	
banner2_06.jpg	04-Nov-2006 07:08	3k	
banner2_07.jpg	04-Nov-2006 07:09	4k	
banner2_08.jpg	04-Nov-2006 07:09	1k	
banner2_09.jpg	04-Nov-2006 07:09	3k	
banner2_10.jpg	04-Nov-2006 07:09	1k	

Vulnerability Scanning

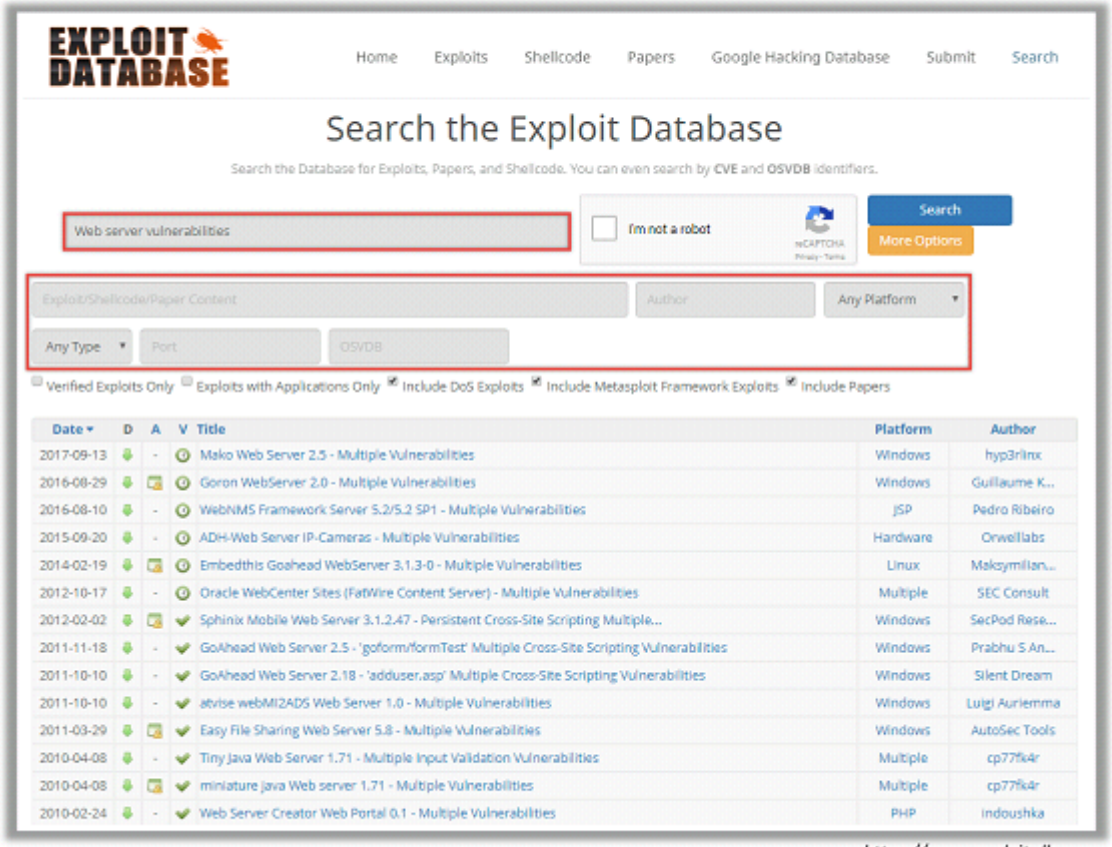
- Implement vulnerability scan to **identify weaknesses** in a network and determine if the system can be exploited
- Use vulnerability scanners such as WebInspect, Acunetix Web Vulnerability Scanner, etc. to find **hosts**, **services**, and **vulnerabilities**
- Sniff the network traffic to find out **active systems**, **network services**, **applications**, and vulnerabilities present
- Test the **web server infrastructure** for any misconfigurations, outdated content, and vulnerabilities using vulnerability scanners like **Acunetix Web Vulnerability Scanner**



https://www.acunetix.com

Finding Exploitable Vulnerabilities

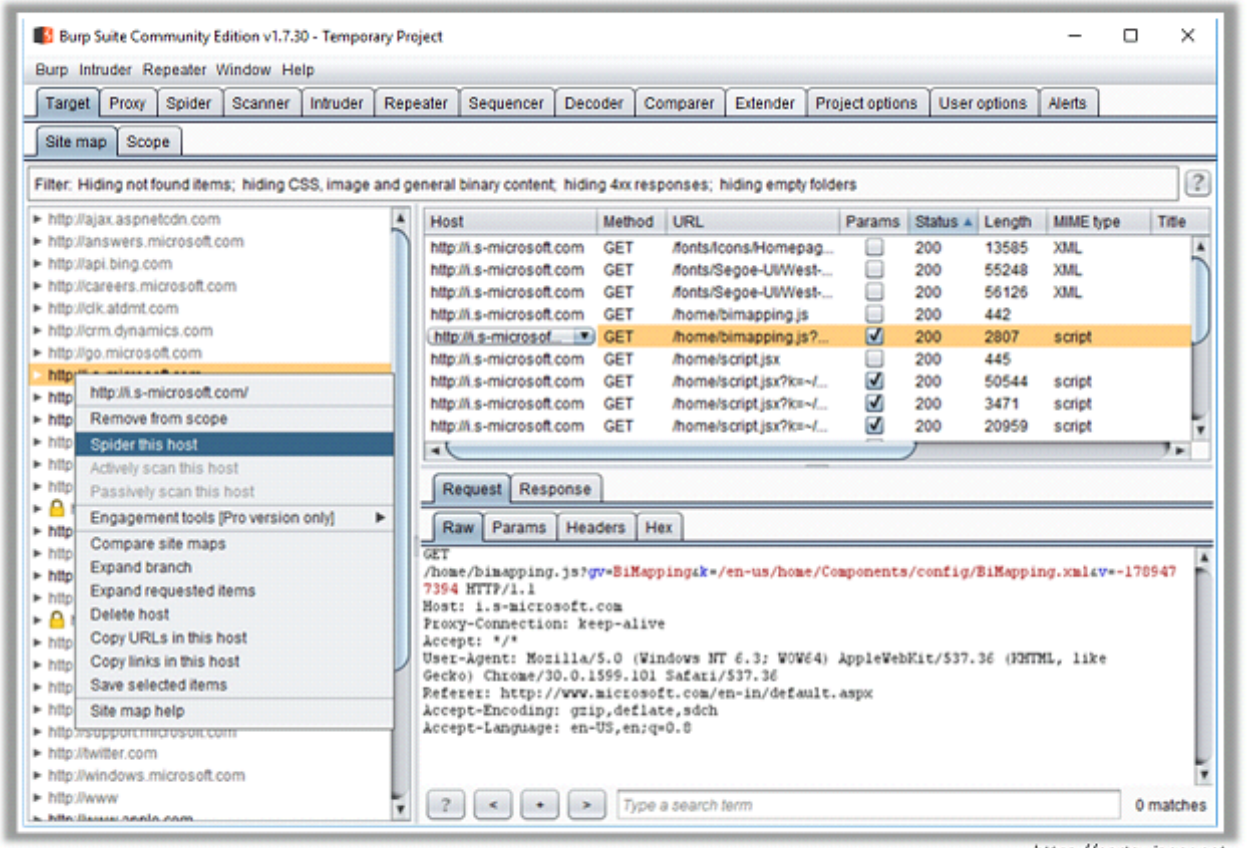
- Search for ~~a~~ web server exploitable vulnerabilities based on the web server OS and software application on exploit sites such as **SecurityFocus** (http://www.securityfocus.com), **Exploit Database** (https://www.exploit-db.com), etc.
- Search for vulnerability based on information gathered in the previous stages by using **More Options** button
- Exploiting these vulnerabilities allows one to execute a command or binary on a target machine to **gain higher privileges** than the existing ones or **bypass security mechanisms**



Session Hijacking

- Sniff valid session IDs to **gain unauthorized access** to the web server and snoop into data
- Use session hijacking techniques such as session fixation, session sidejacking, Cross-site scripting, etc., to **capture valid session cookies and IDs**
- Use tools such as **Burp Suite**, **Firesheep**, **JHijack**, etc. to automate session hijacking

Note: For complete coverage of Session Hijacking concepts and techniques refer to Module 11: Session Hijacking



https://portswigger.net

Web Server Passwords Hacking

- Use password cracking techniques such as **brute force attack**, **dictionary attack**, **password guessing** to crack web server passwords
- Use tools such as **Hashcat**, THC-Hydra, Ncrack, etc.


```
$ ./hashcat -m 15600 -a 3 hash.txt ?a?a?a?acat
hashcat (v3.6.0) starting...

OpenCL Platform #1: NVIDIA Corporation
======================================
* Device #1: GeForce GTX 1080, 2026/8107 MB allocatable, 20MCU
* Device #2: GeForce GTX 1080, 2028/8114 MB allocatable, 20MCU
* Device #3: GeForce GTX 1080, 2028/8114 MB allocatable, 20MCU
* Device #4: GeForce GTX 1080, 2028/8114 MB allocatable, 20MCU

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates

Applicable optimizers:
* Zero-Byte
* Single-Hash
* Single-Salt
* Brute-Force
* Slow-Hash-SIMD

Watchdog: Temperature abort trigger set to 90c
Watchdog: Temperature retain trigger set to 75c

$ethereum$p*262144*32383831373131303534383437373837...da3946:hashcat

Session..........: hashcat
Status...........: Cracked
Hash.Type........: Ethereum Wallet, PBKDF2-HMAC-SHA256
Hash.Target......: $ethereum$p*262144*32383831373131303534383437373837...da3946
Time.Started.....: Wed Jun  7 14:47:52 2017 (11 mins, 41 secs)
Time.Estimated...: Wed Jun  7 14:59:33 2017 (0 secs)
Guess.Mask.......: ?a?a?a?acat [7]
Guess.Queue......: 1/1 (100.00%)
Speed.Dev.#1.....:     5192 H/s (42.69ms)
Speed.Dev.#2.....:     5202 H/s (42.73ms)
Speed.Dev.#3.....:     5212 H/s (42.32ms)
Speed.Dev.#4.....:     5213 H/s (42.30ms)
Speed.Dev.#*.....:    20819 H/s
Recovered........: 1/1 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.........: 14146688/81450625 (17.37%)
Rejected.........: 0/14146688 (0.00%)
Restore.Point....: 0/857375 (0.00%)
Candidates.#1....: haricat -> hp[ycat
Candidates.#2....: h/rYcat -> h]~]cat
Candidates.#3....: myQBcat -> nNfEcat
Candidates.#4....: nBjLcat -> n uucat
HWMon.Dev.#1.....: Temp: 74c Fan: 69% Util:100% Core:1974MHz Mem:4513MHz Bus:1
HWMon.Dev.#2.....: Temp: 74c Fan: 80% Util:100% Core:1974MHz Mem:4513MHz Bus:1
HWMon.Dev.#3.....: Temp: 75c Fan: 64% Util:100% Core:1974MHz Mem:4513MHz Bus:1
HWMon.Dev.#4.....: Temp: 75c Fan: 79% Util:100% Core:1974MHz Mem:4513MHz Bus:1

Started: Wed Jun  7 14:47:43 2017
Stopped: Wed Jun  7 14:59:34 2017
```

https://hashcat.net

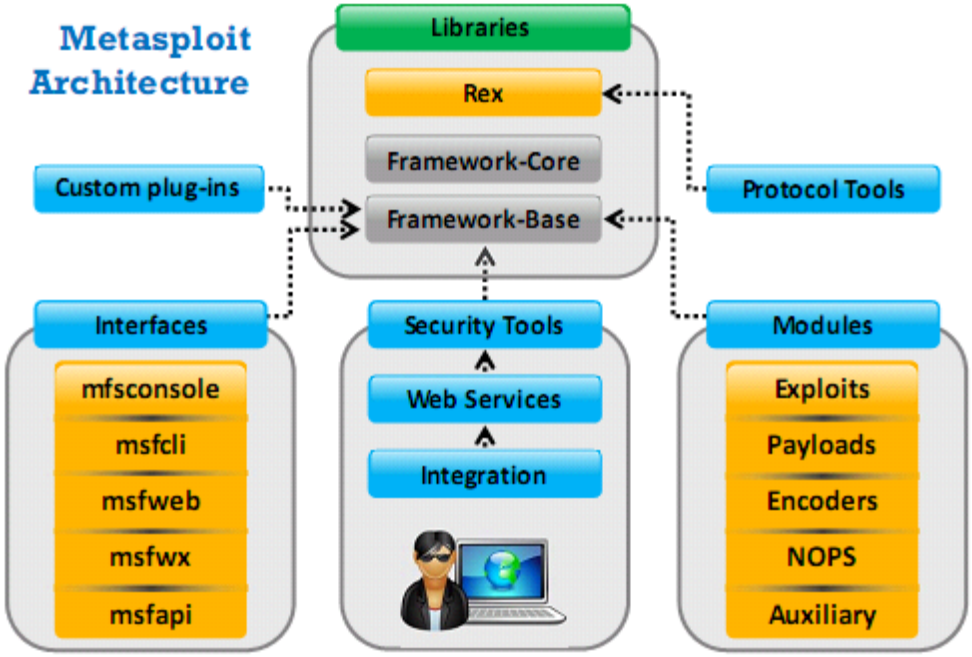
Using Application Server as a Proxy

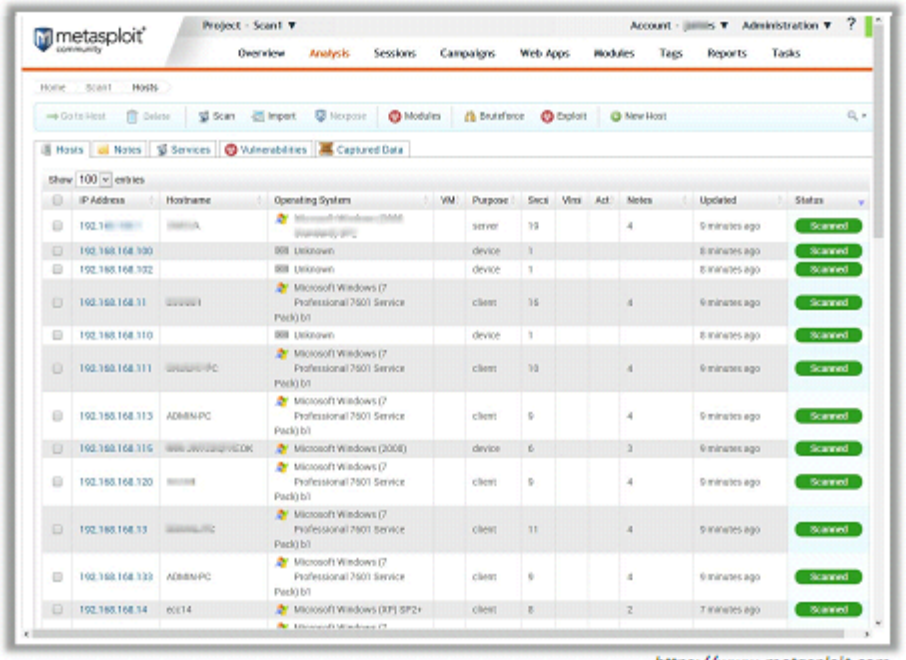
- Web servers with **forwarding** and **reverse HTTP proxy functions** enabled, are employed by the attackers to perform the following attacks:
  - Attacking third party systems on internet
  - Connecting to arbitrary hosts on the organization's internal network
  - Connecting back to other services running on the proxy host itself
- Attackers use **GET** and **CONNECT** requests **to use vulnerable web servers as proxies** to connect and obtain information from target systems through these proxy web servers

Module Flow

Metasploit

- The Metasploit Framework is a exploit development platform which supports fully automated **exploitation of web servers**, by abusing known vulnerabilities and leveraging weak passwords via Telnet, SSH, HTTP, and SNM





https://www.metasploit.com

Metasploit Exploit Module

- It is the basic module in Metasploit used to **encapsulate an exploit** with the help of which users target many platforms with a single exploit
- This module comes with **simplified meta-information fields**
- Using a Mixins feature, users can also **modify exploit behavior dynamically**, brute force attacks, and attempt passive exploits

Steps to exploit a system follow the Metasploit Framework

1. Configuring Active Exploit
2. Verifying the Exploit Options
3. Selecting a Target
4. Selecting the Payload
5. Launching the Exploit

Metasploit Payload and Auxiliary Module

Payload Module

- Payload module establishes a **communication channel** between the Metasploit framework and the victim host
- It combines the **arbitrary code** that is executed as a result of an exploit succeeding
- To generate **payloads**, first select a payload using the command as shown in the screenshot

Command Prompt

```
msf > use windows/shell_reverse_tcp
msf payload(shell_reverse_tcp) > generate -h
Usage: generate [options]
Generates a payload.
OPTIONS:
-b <opt> The list of characters to avoid: '\x00\xff'
-e <opt> The name of the encoder module to use.
-h Help banner.
-o <opt> A comma separated list of options in
         VAR=VAL format.
-s <opt> NOP sled length.
-t <opt> The output type: ruby, perl, c, or raw.
msf payload(shell_reverse_tcp) >
```

Auxiliary Module

- Auxiliary modules can be **used to perform arbitrary**, one-off actions such as port scanning, denial of service, and even fuzzing
- To run the auxiliary module, either use the `run` command, or use the `exploit` command

Command Prompt

```
msf > use dos/windows/smb/ms06_035_mailslot
msf auxiliary(ms06_035_mailslot) > set RHOST 1.2.3.4
RHOST => 1.2.3.4
msf auxiliary(ms06_035_mailslot) > run
[*] Mangling the kernel, two bytes at a time...
```

Metasploit NOPS Module

- NOP modules generate a no-operation instruction used for blocking out buffers
- Use `generate` command to generate a NOP sled of an arbitrary size and display it in a given format

OPTIONS:

`-b <opt>`: The list of characters to avoid: '\x00\xff'
`-h`: Help banner
`-s <opt>`: The comma separated list of registers to save
`-t <opt>`: The output type: ruby, perl, c, or raw

```
msf nop(opty2)>
```

Generates a NOP sled of a given length

```
msf > use x86/opty2
msf nop(opty2) > generate -h
Usage: generate [options] length
```

Command to generate a 50 byte NOP sled

```
msf nop(opty2) > generate -t c 50
unsigned char buf[] =
"\xf5\x3d\x05\x15\xf8\x67\xba\x7d\x08\xd6\x66\x9f\xb8\x2d
\xb6"
"\x24\xbe\xb1\x3f\x43\x1d\x93\xb2\x37\x35\x84\xd5\x14\x40
\xb4"
"\xb3\x41\xb9\x48\x04\x99\x46\xa9\xb0\xb7\x2f\xfd\x96\x4a
\x98"
"\x92\xb5\xd4\x4f\x91";
msf nop(opty2) >
```

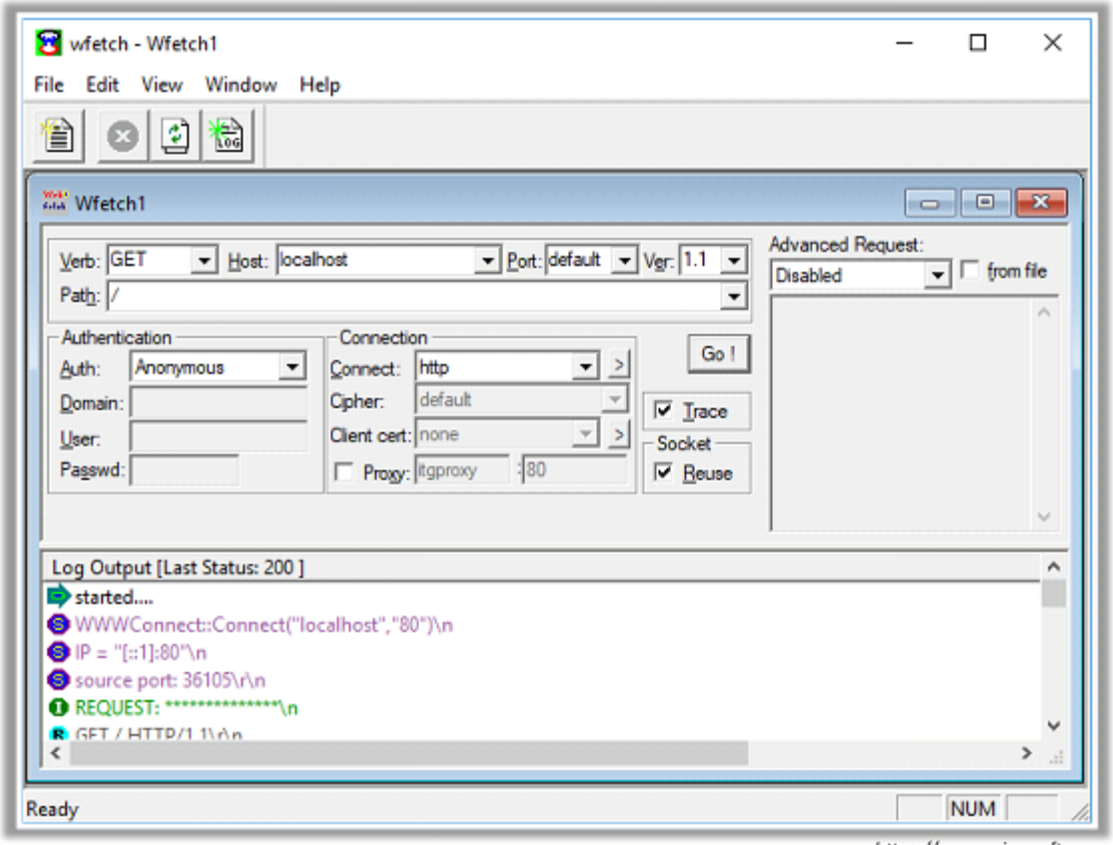
Web Server Attack Tools

Wfetch

- Wfetch allows the attacker to fully customize an HTTP request and send it to a Web server to see the raw HTTP request and response data
- It allows the attacker to test the performance of Web sites that contain new elements such as Active Server Pages (ASP) or wireless protocols

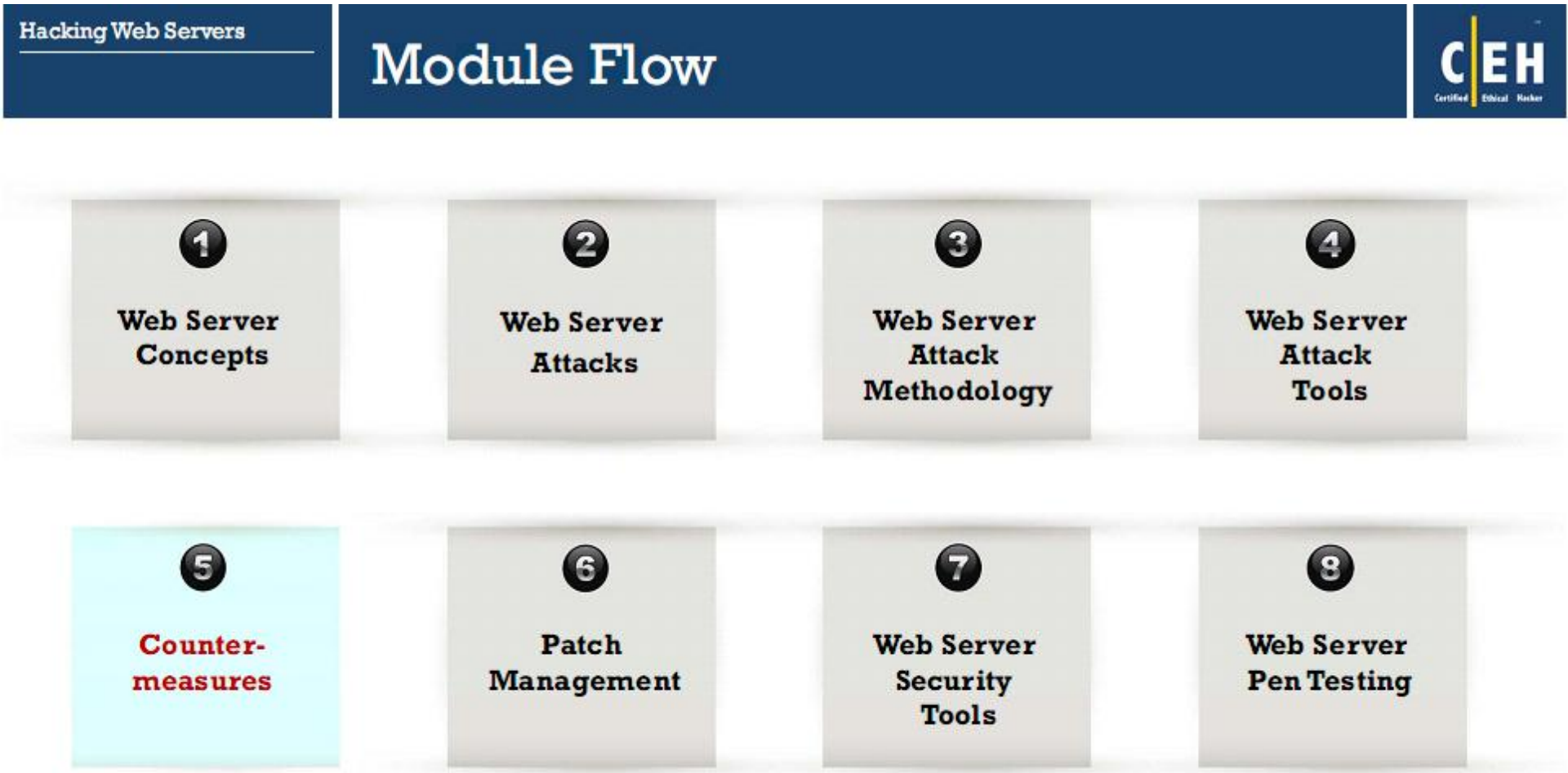
Web Server Attack Tools

- THC Hydra (*https://www.thc.org*)
- HULK DoS (*https://github.com*)
- MPack (*https://sourceforge.net*)
- w3af (*http://w3af.org*)



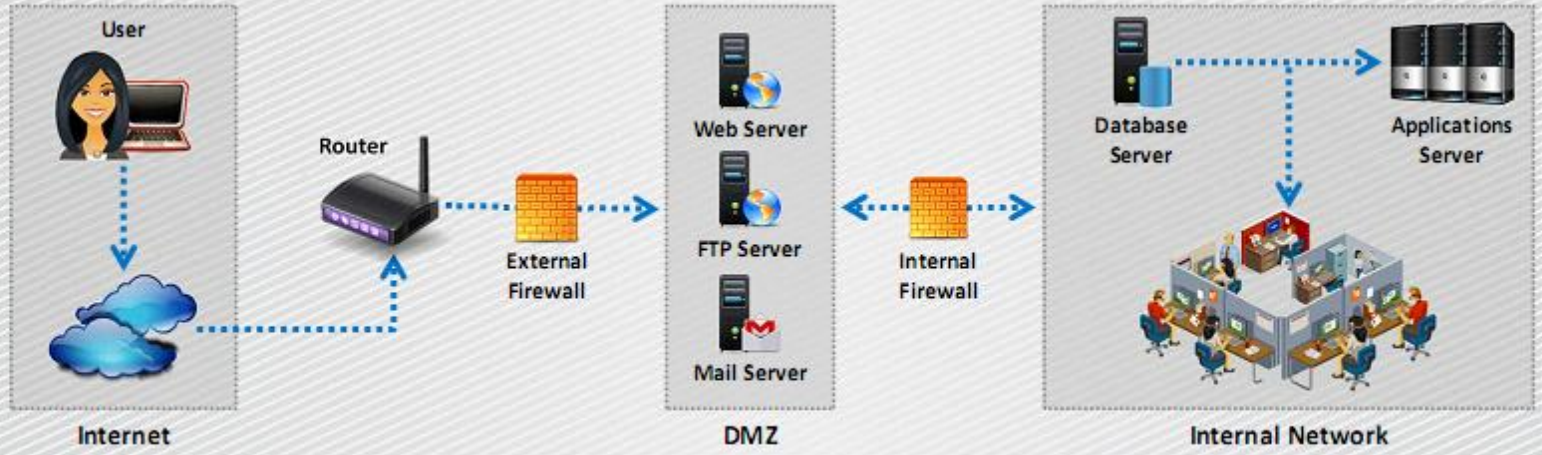
https://www.microsoft.com

Module Flow



Place Web Servers in Separate Secure Server Security Segment on Network

- An ideal **web hosting network** should be designed with at least **three segments** namely Internet segment, secure server security segment often called demilitarized zone (DMZ), and internal network
- Place the web server in **Server Security Segment** (DMZ) of the network, isolated from public network as well as internal network
- Firewalls should be in place for **internal network** as well as **Internet traffic** going towards DMZ



Countermeasures: Patches and Updates

01 Scan for existing vulnerabilities, patch, and update the server software regularly

02 Before applying any service pack, hotfix, or security patch, read and peer review all relevant documentation

03 Apply all updates, regardless of their type on an "as-needed" basis

04 Test the service packs and hotfixes on a representative non-production environment prior to being deployed to production

05 Ensure that service packs, hotfixes, and security patch levels are consistent on all Domain Controllers (DCs)

06 Ensure that server outages are scheduled and a complete set of backup tapes and emergency repair disks are available

07 Have a back-out plan that allows the system and enterprise to return to their original state, prior to the failed implementation

08 Schedule periodic service pack upgrades as part of operations maintenance and never try to have more than two service packs behind

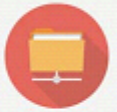
Countermeasures: Protocols

01 Block all unnecessary ports, Internet Control Message Protocol (ICMP) traffic, and unnecessary protocols such as NetBIOS and SMB

02 Harden the TCP/IP stack and consistently apply the latest software patches and updates to system software

03 If using insecure protocols such as Telnet, POP3, SMTP, FTP, take appropriate measures to provide secure authentication and communication, for example, by using IPSec policies

04 If remote access is needed, make sure that the remote connection is secured properly, by using tunneling and encryption protocols



05 Disable WebDAV if not used by the application or keep secure if it is required

Countermeasures: Accounts

1. Remove all unused modules and application extensions
2. Disable unused default user accounts created during installation of an operating system
3. When creating a new web root directory, grant the appropriate (least possible) NTFS permissions to the anonymous user being used from the IIS web server to access the web content
4. Eliminate unnecessary database users and stored procedures and follow the principle of least privilege for the database application to defend against SQL query poisoning
5. Use secure web permissions, NTFS permissions, and .NET Framework access control mechanisms including URL authorization
6. Slow down brute force and dictionary attacks with strong password policies, and then audit and be alert for logon failures
7. Run processes using least privileged accounts as well as least privileged service and user accounts

Countermeasures: Files and Directories

Eliminate unnecessary files within the **.jar files**

Disable serving of **directory listings**

Eliminate **sensitive configuration** information within the **byte code**

Eliminate the **presence of non-web files** such as archive files, backup files, text files, and header/include files

Avoid mapping **virtual directories** between two different servers, or over a network

Disable serving certain **file types** by creating a resource mapping

Monitor and check all **network services logs**, **website access logs**, **database server logs** (e.g., Microsoft SQL Server, MySQL, Oracle) and OS logs frequently

Ensure the presence of **web application** or **website files** and **scripts** on a separate partition or drive other than that of the operating system, logs, and any other system files

Detecting Web Server Hacking Attempts

Use **Website Change Detection System** to detect hacking attempts on the web server

Website Change Detection System involves:

Running specific script on the server that detects any changes made in the existing executable file or new file included on the server

Periodically comparing the **hash values** of the files on the server with their respective master hash value to detect the changes made in codebase

Alerting the user upon any change detection on the server

For example: **WebsiteCDS** is a script that goes through your entire web folder and detects any changes made to your code base and alerts you using email

How to Defend Against Web Server Attacks

01 Ports

- Audit the ports on the server regularly to ensure that an **insecure** or unnecessary service is not active on your web server
- Limit inbound traffic to **port 80 for HTTP** and **port 443 for HTTPS** (SSL)
- Encrypt or restrict **intranet traffic**

02 Server Certificates

- Ensure that **certificate data ranges** are valid and that certificates are used for their intended purpose
- Ensure that any certificate has not been revoked and **certificate's public key** is valid all the way to a trusted root authority

03 Machine.config

- Ensure that protected resources are mapped to **HttpForbiddenHandler** and unused **HttpModules** are removed
- Ensure that **tracing is disabled** <trace enable="false"/> and **debug compiles** are turned off

04 Code Access Security

- Implement **secure coding** practices
- Restrict **code access security policy** settings
- **Configure IIS** to reject URLs with "../" and install new patches and updates

How to Defend Against Web Server Attacks (Cont'd)

1. UrlScan is a security tool that **restricts** the types of HTTP requests that IIS will process
2. By blocking specific HTTP requests, the UrlScan security tool helps to **prevent potentially harmful requests** from reaching applications on the server
3. UrlScan screens all incoming requests to the server by filtering the requests based on **rules** that are set by the administrator
4. UrlScan can be configured to filter HTTP query string values and other HTTP headers to **mitigate SQL injection** attacks while the root cause is being fixed in the application
5. It provides **W3C formatted logs** for easier log file analysis through log parsing solutions like Microsoft Log Parser 2.2

https://www.iis.net

How to Defend Against Web Server Attacks (Cont'd)

01
- Apply **restricted ACLs** and block remote registry administration
- Secure the **SAM** (Stand-alone Servers Only)

02 Ensure that security related settings are **configured appropriately** and access to the metabase file is restricted with hardened **NTFS permissions**

03 **Remove** unnecessary ISAPI filters from the web server

04
- Remove all unnecessary file shares including the **default administration shares** if not required
- Secure the shares with restricted **NTFS permissions**

05 Relocate sites and virtual directories to **non-system partitions** and use IIS Web permissions to restrict access

06 Remove all unnecessary **IIS script mappings** for optional file extensions to avoid exploiting any bugs in the ISAPI extensions that handle these types of files

07 Enable a **minimum level of auditing** on your web server and use NTFS permissions to protect the log files

How to Defend Against Web Server Attacks (Cont'd)

- Do use a **dedicated machine** as a web server
- Create **URL mappings** to internal servers cautiously
- Do not install the **IIS server** on a domain controller
- Use server side **session ID tracking** and match connections with time stamps, IP addresses, etc.
- If a database server, such as **Microsoft SQL Server**, is to be used as a backend database, install it on a **separate server**
- Use **security tools** provided with web server software and **scanners** that automate and make the process of securing a web server easy
- Do physically protect the **web server machine** in a secure machine room
- Do not connect an IIS Server to the **Internet** until it is fully hardened
- Do not allow anyone to **locally log on** to the machine except for the administrator
- Do configure a **separate anonymous user account** for each application, if you host multiple web applications
- Limit the **server functionality** in order to support the web technologies that are going to be used
- Screen and filter the **incoming traffic request**

How to Defend against HTTP Response Splitting and Web Cache Poisoning

Server Admin

- Use latest **web server software**
- Regularly **update/patch OS** and web server
- Run **web Vulnerability Scanner**

Application Developers

- Restrict web application access to **unique IPs**
- Disallow **carriage return** (%0d or \r) and line feed (%0a or \n) characters
- Comply to **RFC 2616** specifications for HTTP/1.1

Proxy Servers

- Avoid sharing **incoming TCP connections** among different clients
- Use different TCP connections with the proxy for different **virtual hosts**
- Implement **"maintain request host header"** correctly

How to Defend against DNS Hijacking

Choose an **ICANN** accredited **registrar** and encourage them to set **Registrar-Lock** on the domain name

Safeguard the **registrant account information**

Include DNS hijacking into **incident response and business continuity planning**

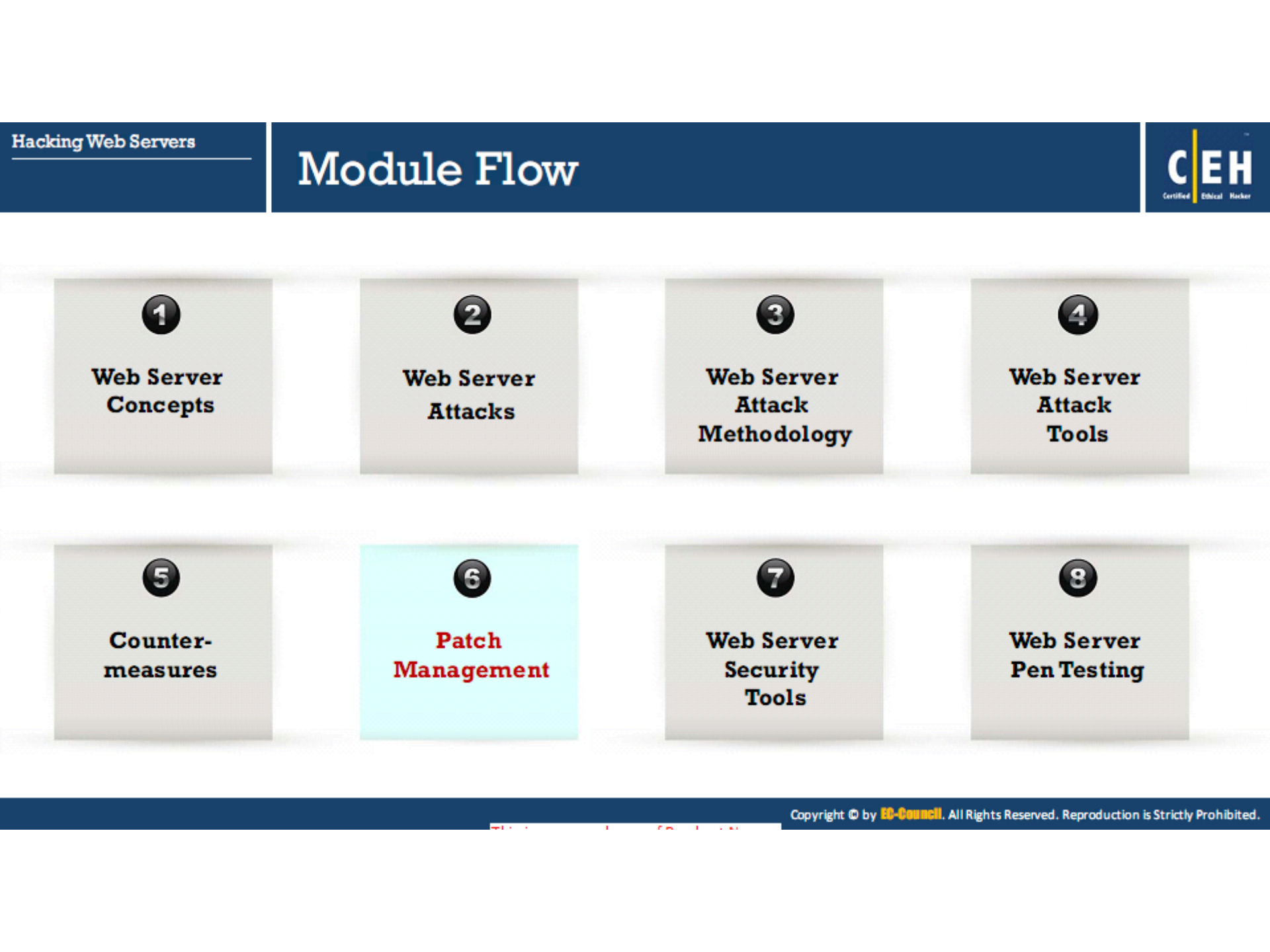
Use DNS monitoring tools/services to **monitor DNS server IP address and alert**

Avoid downloading **audio and video codecs** and other downloaders from untrusted websites

Install **antivirus** program and update it regularly

Change the **default router password** that comes with the factory settings

Module Flow



Patches and Hotfixes

Hotfixes are an **update to fix a specific customer issue** and not always distributed outside the customer organization

A patch is a **small piece of software designed to fix problems**, security vulnerabilities, and bugs and improve the performance of a computer program or its supporting data

Users may be notified through **emails** or through the **vendor's website**

A patch can be considered as a **repair job to a programming problem**

Hotfixes are sometimes packaged as a set of fixes called a **combined hotfix** or **service pack**

What is Patch Management?

"Patch management is a process used to ensure that the **appropriate patches** are installed on a system and help fix known vulnerabilities"

An automated patch management process

Detect	Use tools to detect missing security patches
Assess	Asses the issue(s) and its associated severity by mitigating the factors that may influence the decision
Acquire	Download the patch for testing
Test	Install the patch first on a testing machine to verify the consequences of the update
Deploy	Deploy the patch to the computers and make sure the applications are not affected
Maintain	Subscribe to get notifications about vulnerabilities as they are reported

Installation of a Patch

Identifying Appropriate Sources for Updates and Patches

- First make a **patch management plan** that fits the operational environment and business objectives
- Find appropriate **updates and patches** on the home sites of the applications or operating systems' vendors
- The recommended way of tracking issues relevant to **proactive patching** is to register with the home sites to **receive alerts**

Installation of a Patch

- Users can access and install security patches via the **World Wide Web**
- Patches can be installed in two ways

Manual Installation

- In this method, the user has to **download the patch** from the vendor and fix it

Automatic Installation

- In this method, the applications use the **Auto Update** feature to update themselves

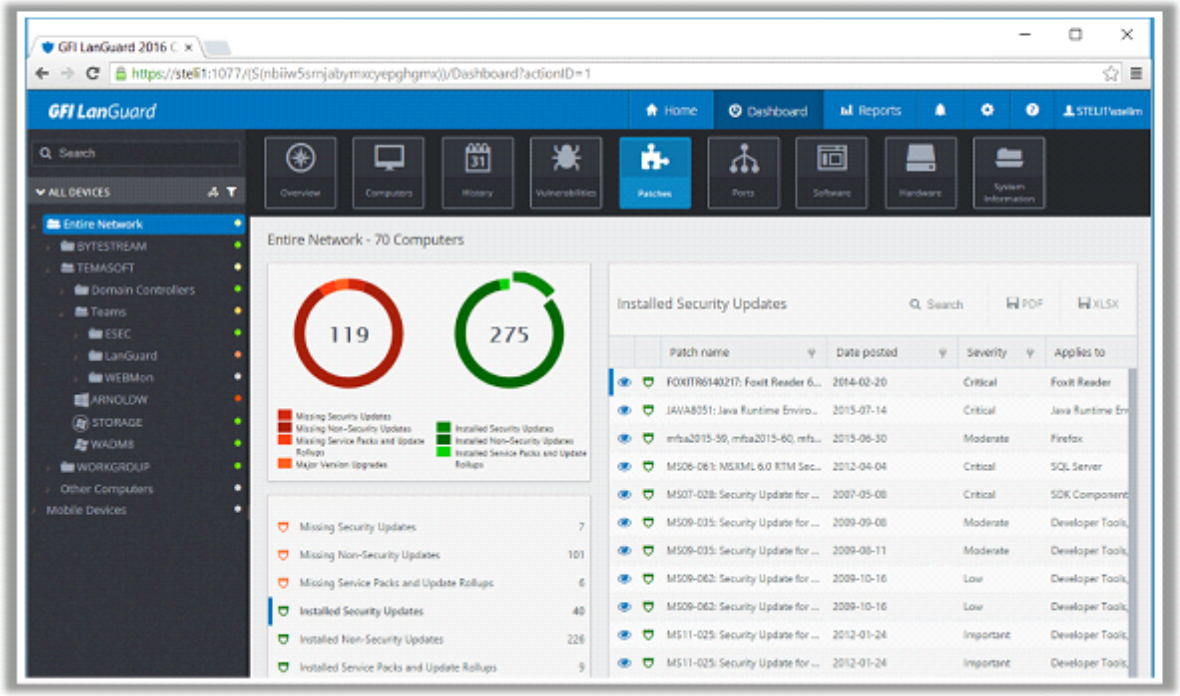
Implementation and Verification of a Security Patch or Upgrade

- Before installing any patch, **verify the source**
- Use proper **patch management program** to validate files versions and checksums before deploying security patches
- The patch management tool must be **able to monitor the patched systems**
- The **patch management team** should check for updates and patches regularly

Patch Management Tools

GFI LanGuard

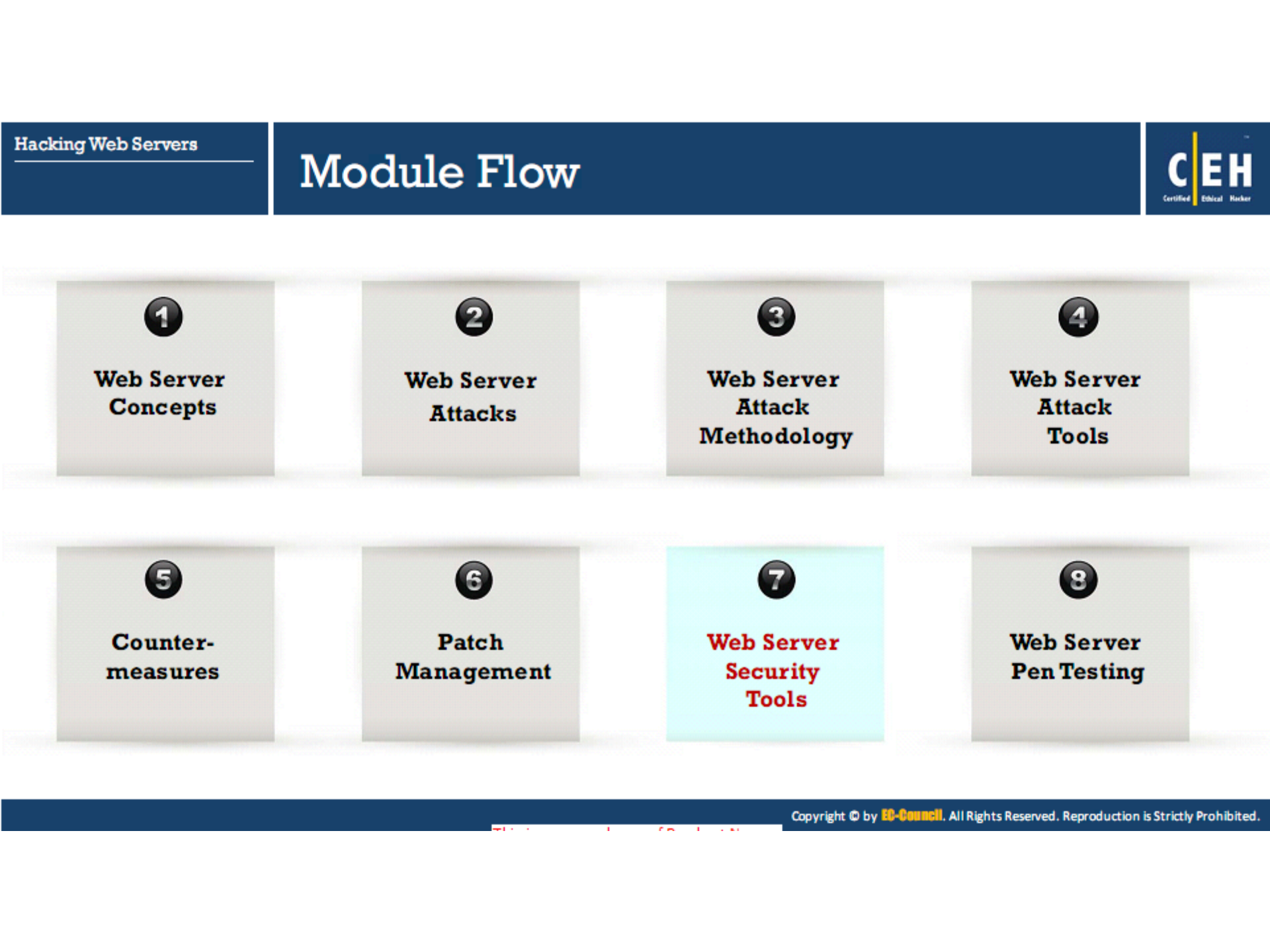
GFI LanGuard's patch management scans your network automatically and also **installs and manages security and non-security patches**



https://www.gfi.com

- **Symantec Client Management Suite**
  https://www.symantec.com
- **MaaS360 Patch Analyzer**
  https://www.ibm.com
- **Solarwinds Patch Manager**
  https://www.solarwinds.com
- **Kaseya Security Patch Management**
  https://www.kaseya.com
- **Software Vulnerability Manager**
  https://www.flexerasoftware.com

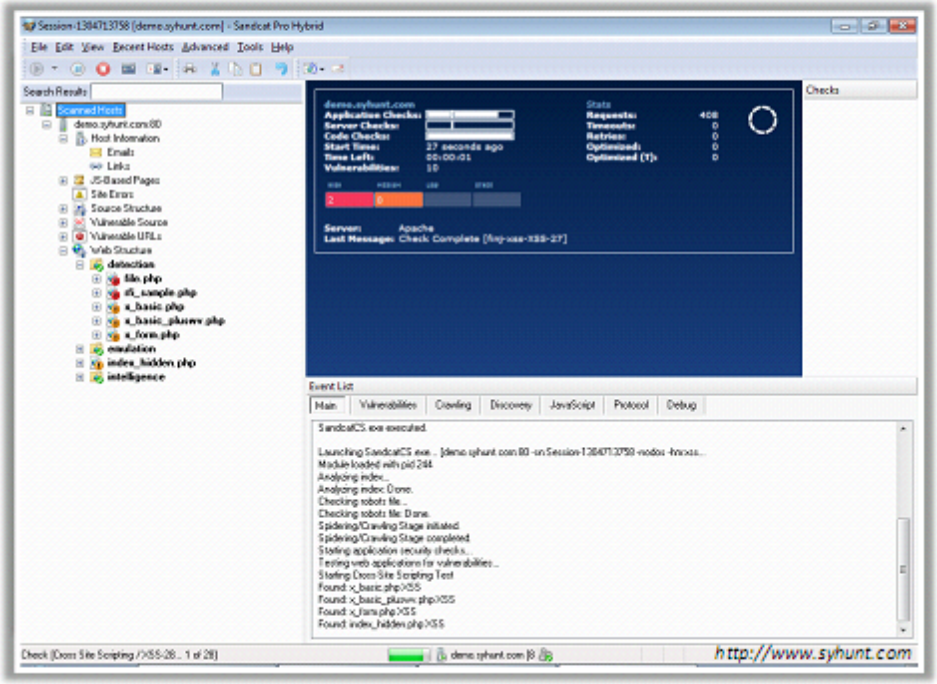
Module Flow



Web Application Security Scanners

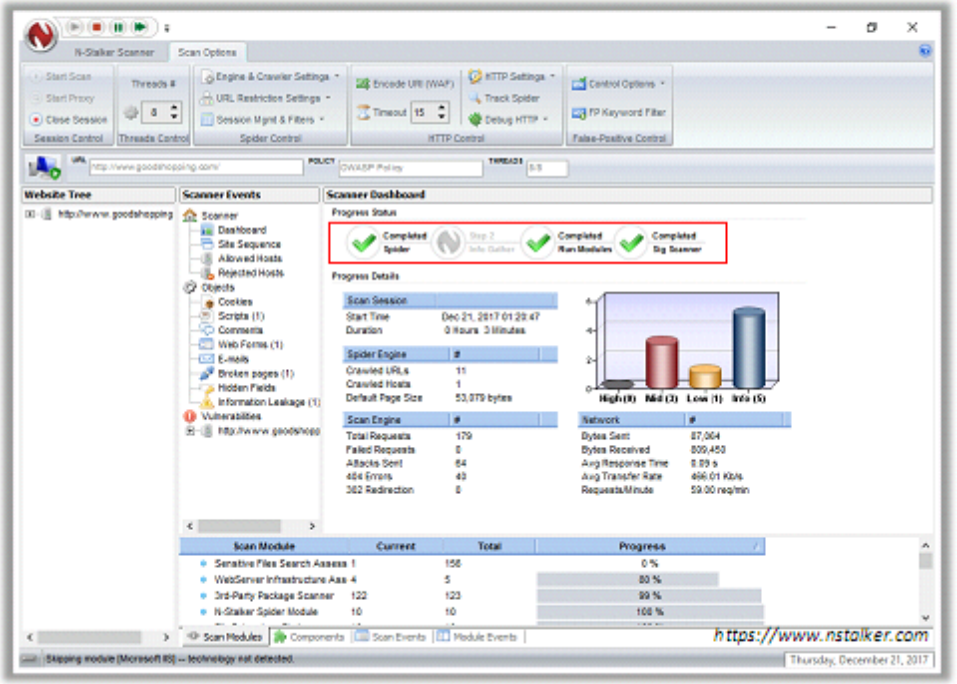
Syhunt Hybrid

Syhunt Hybrid helps to automate **web application security** testing and guard organization's **web infrastructure** against various web application security threats



N-Stalker

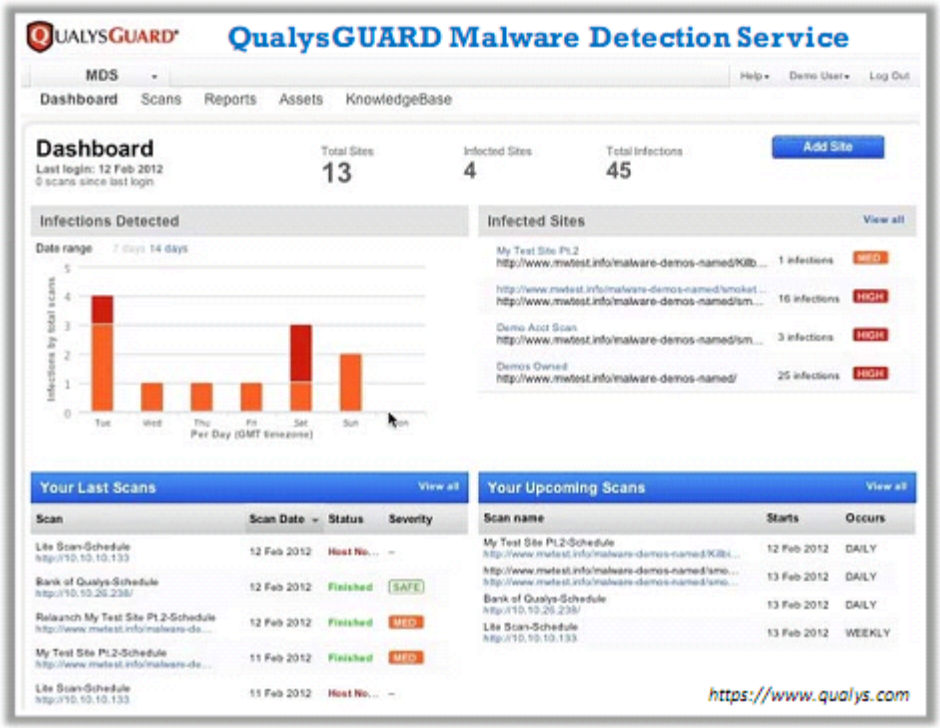
N-Stalker is a **WebApp Security Scanner** to search for vulnerabilities such as SQL injection, XSS, and known attacks



Web Server Security Scanners

ScanMyServer

- ScanMyServer is used to **find security vulnerabilities** in a web site or a web server
- It can **generate comprehensive test reports** and also can assist in fixing security problems that might exist in company's website or web server

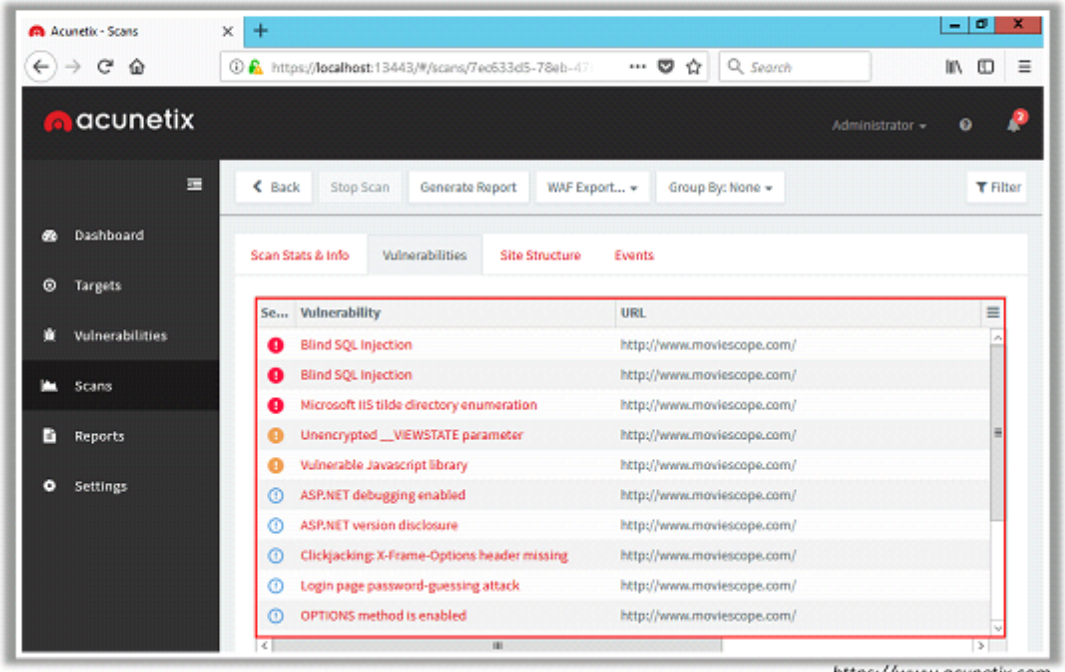


QualysGUARD Malware Detection Service proactively scan their web sites for **malware**, providing automated alerts and in-depth reporting to enable prompt identification and resolution

Web Server Security Tools

Acunetix WVS

Acunetix WVS **checks web applications** for SQL injections, cross-site scripting, etc. and creates professional security audit and regulatory compliance reports



https://www.acunetix.com

Fortify WebInspect
https://software.microfocus.com

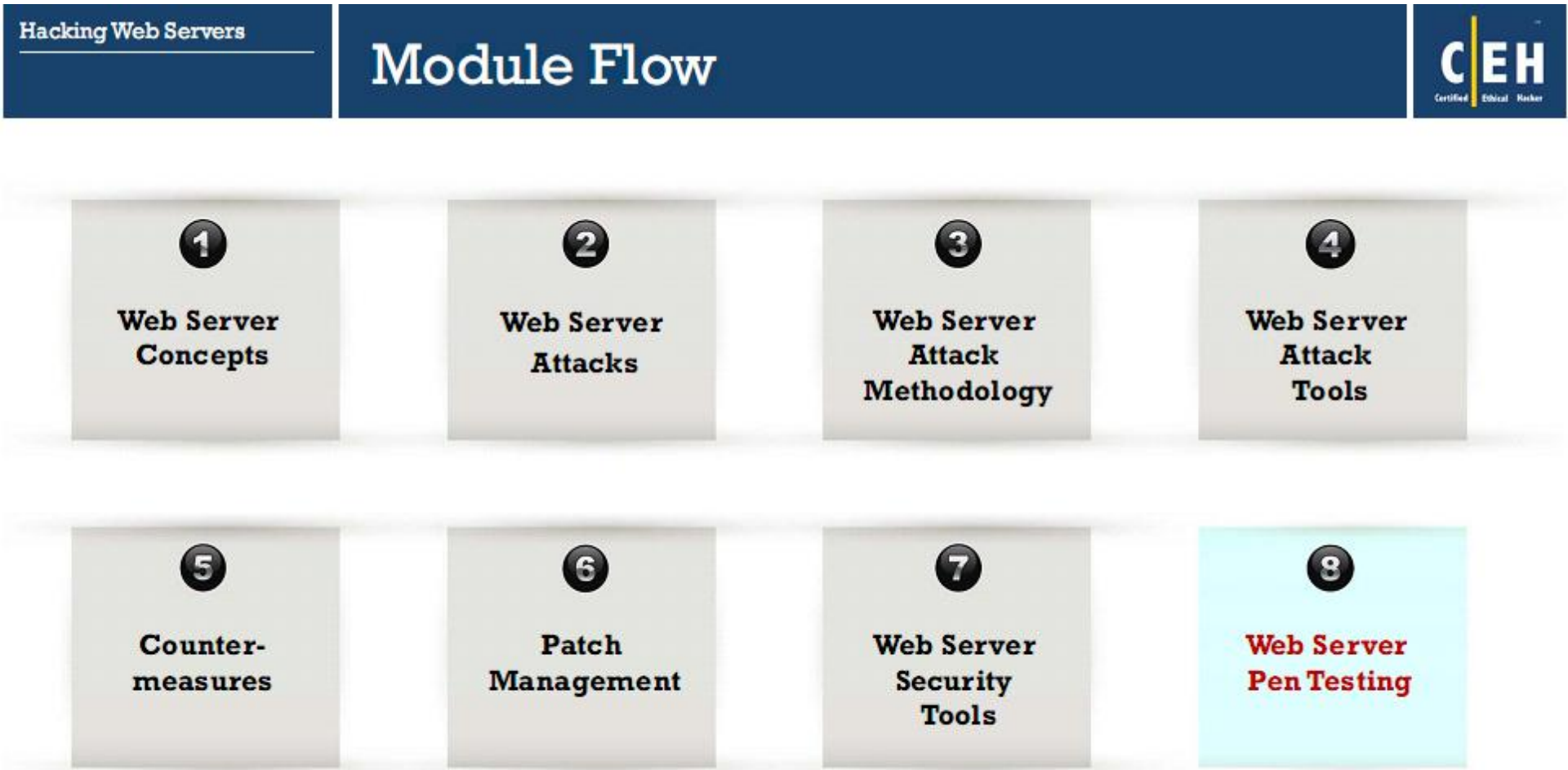
Retina CS
https://www.beyondtrust.com

Nscan
https://nscan.hypermart.net

NetIQ Secure Configuration Manager
https://www.netiq.com

SAINT Scanner
http://www.saintcorporation.com

Module Flow



Web Server Penetration Testing

- Web server pen testing is used to **identify**, **analyze**, and **report vulnerabilities** such as authentication weaknesses, configuration errors, protocol related vulnerabilities, etc. in a web server
- The best way to perform penetration testing is to **conduct a series of methodical and repeatable tests**, and to work through all of the different application vulnerabilities

Why Web Server Pen Testing?

Verification of Vulnerabilities

To exploit the vulnerability in order to test and fix the issue

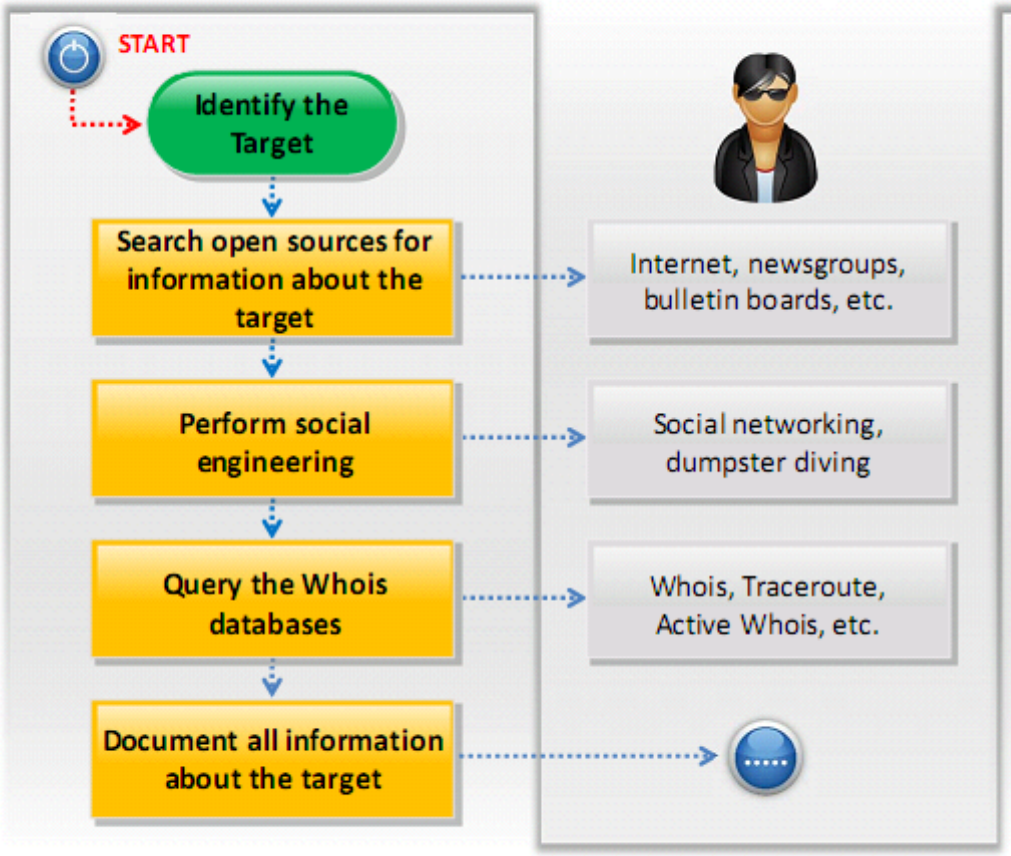
Remediation of Vulnerabilities

To retest the solution against vulnerability to ensure that it is completely secure

Identification of Web Infrastructure

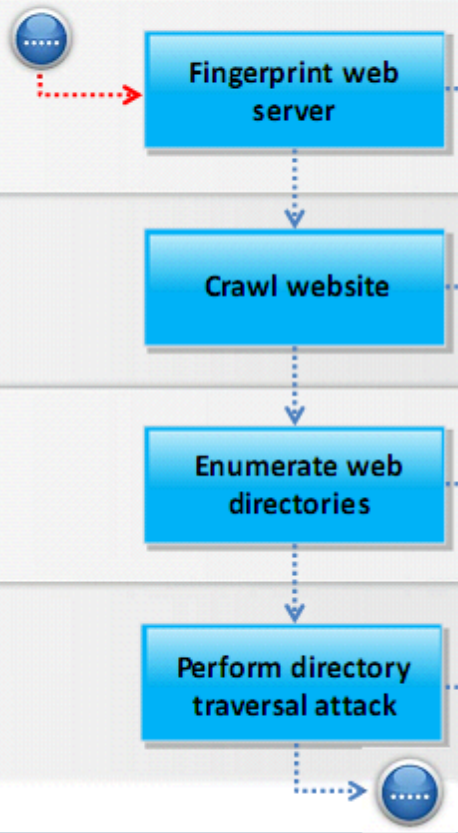
To identify make, version, and update levels of web servers; this helps in selecting exploits to test for associated published vulnerabilities

Web Server Penetration Testing (Cont'd)



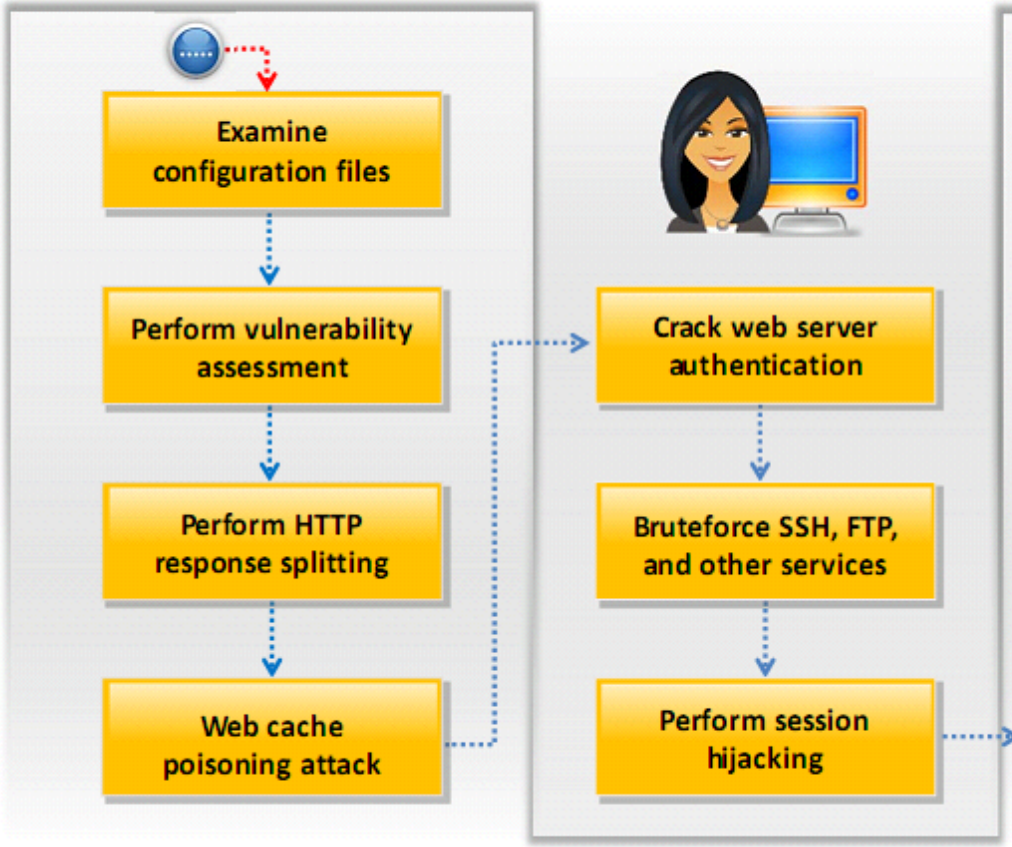
- Web server penetration testing starts with **collecting as much information** as possible about an organization ranging from its physical location to operating environment
- Use **social engineering techniques** to collect information such as human resources, contact details, etc. that may help in **web server authentication testing**
- Use **Whois database query tools** to get the details about the target such as domain name, IP address, administrative contacts, Autonomous System Number, DNS, etc.
- **Note**: Refer Module 02: Footprinting and Reconnaissance for more information gathering techniques

Web Server Penetration Testing (Cont'd)



- Fingerprint web server to gather information such as server name, server type, operating systems, applications running, etc. using tools such as Netcraft, httprecon, and ID Serve
- Crawl website to gather specific types of information from web pages, such as email addresses
- Enumerate web server directories to extract important information such as web functionalities, login forms, etc.
- Perform directory traversal attack to access restricted directories and execute commands outside of the web server's root directory

Web Server Penetration Testing (Cont'd)



- Perform vulnerability scanning to **identify weaknesses** in a network using tools such as **Acunetix**, **WebInspect**, **Nessus**, etc. and determine if the system can be exploited
- Perform HTTP response splitting attack to pass malicious data to a vulnerable application that includes the data in an HTTP response header
- Perform web cache poisoning attack to force the web server's cache to **flush its actual cache content** and send a specially **crafted request**, which will be stored in cache
- Bruteforce SSH, FTP, and other services login credentials to gain **unauthorized access**
- Perform session hijacking to **capture valid session cookies and IDs**. Use tools such as **Burp Suite**, **Firesheep**, **JHijack**, etc. to automate session hijacking

Web Server Penetration Testing (Cont'd)

Perform MITM attack

- Perform MITM attack to access sensitive information by **intercepting and altering communications** between an end-user and web servers

Perform web application pen testing

- **Note**: Refer Module 14: Hacking Web Applications for more information on how to conduct web application pen testing

Examine web server logs

- Use tools such as **Webalizer**, **AWStats**, **Ktmatu Relax**, etc. to examine web server logs

Exploit frameworks → **Document all the findings**

- Use tools such as **Metasploit**, **w3af**, etc. to exploit frameworks and document all the findings

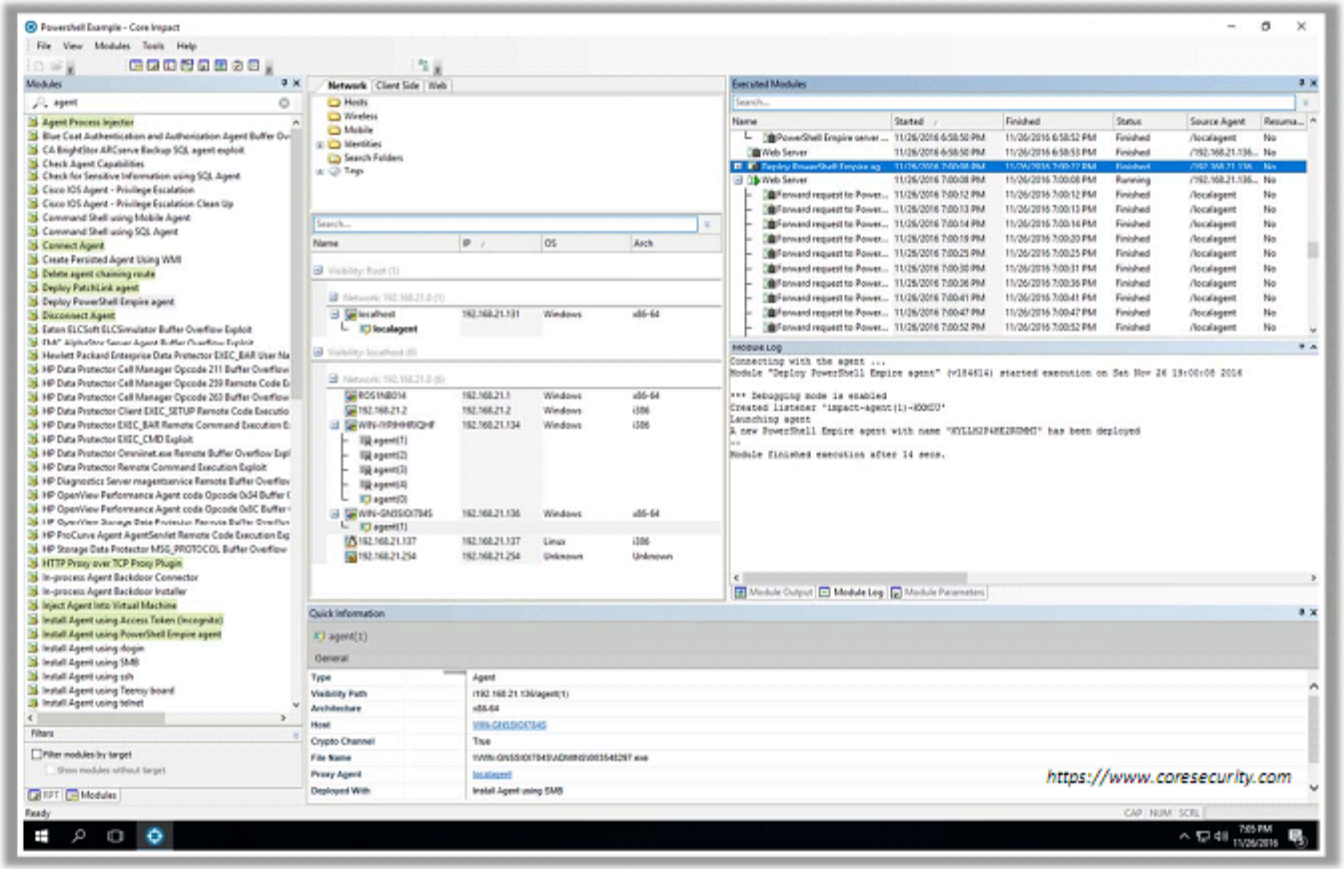
Web Server Pen Testing Tools

CORE Impact

- CORE Impact finds vulnerabilities on an organization's web server
- This tool allows a user to evaluate the security posture of a web server using the same techniques employed by today's cyber-criminals

Web Server Pen Testing Tools

- Immunity CANVAS (*http://www.immunitysec.com*)
- Arachni (*http://www.arachni-scanner.com*)
- WebSurgery (*http://sunrisetech.gr*)



Module Summary

- Web servers assume critical importance in the realm of Internet security
- Vulnerabilities exist in different releases of popular web servers and respective vendors patch these often
- The inherent security risks owing to the compromised web servers have impact on the local area networks that host these websites, even on the normal users of web browsers
- Looking through the long list of vulnerabilities that had been discovered and patched over the past few years, it provides an attacker ample scope to plan attacks to unpatched servers
- Different tools/exploit codes aid an attacker in perpetrating web server's hacking
- Countermeasures include scanning for the existing vulnerabilities and patching them immediately, anonymous access restriction, incoming traffic request screening, and filtering