C|EH Certified Ethical Hacker

EC-Council

MODULE 13

HACKING WEB SERVERS

EC-COUNCIL OFFICIAL CURRICULA

This page is intentionally left blank.

LEARNING OBJECTIVES

- LO#01: Summarize Web Server Concepts
- LO#02: Demonstrate Different Web Server Attacks
- LO#03: Explain Web Server Attack Methodology
- LO#04: Explain Web Server Attack Countermeasures
- LO#05: Summarize Patch Management Concepts

Learning Objectives

Most organizations consider their web presence to be an extension of themselves. Organizations maintain websites associated with their business on the World Wide Web to establish their web presence. Web servers are a critical component of web infrastructure. A single vulnerability in web server configuration may lead to a security breach on websites. Therefore, web server security is critical to the normal functioning of an organization.

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

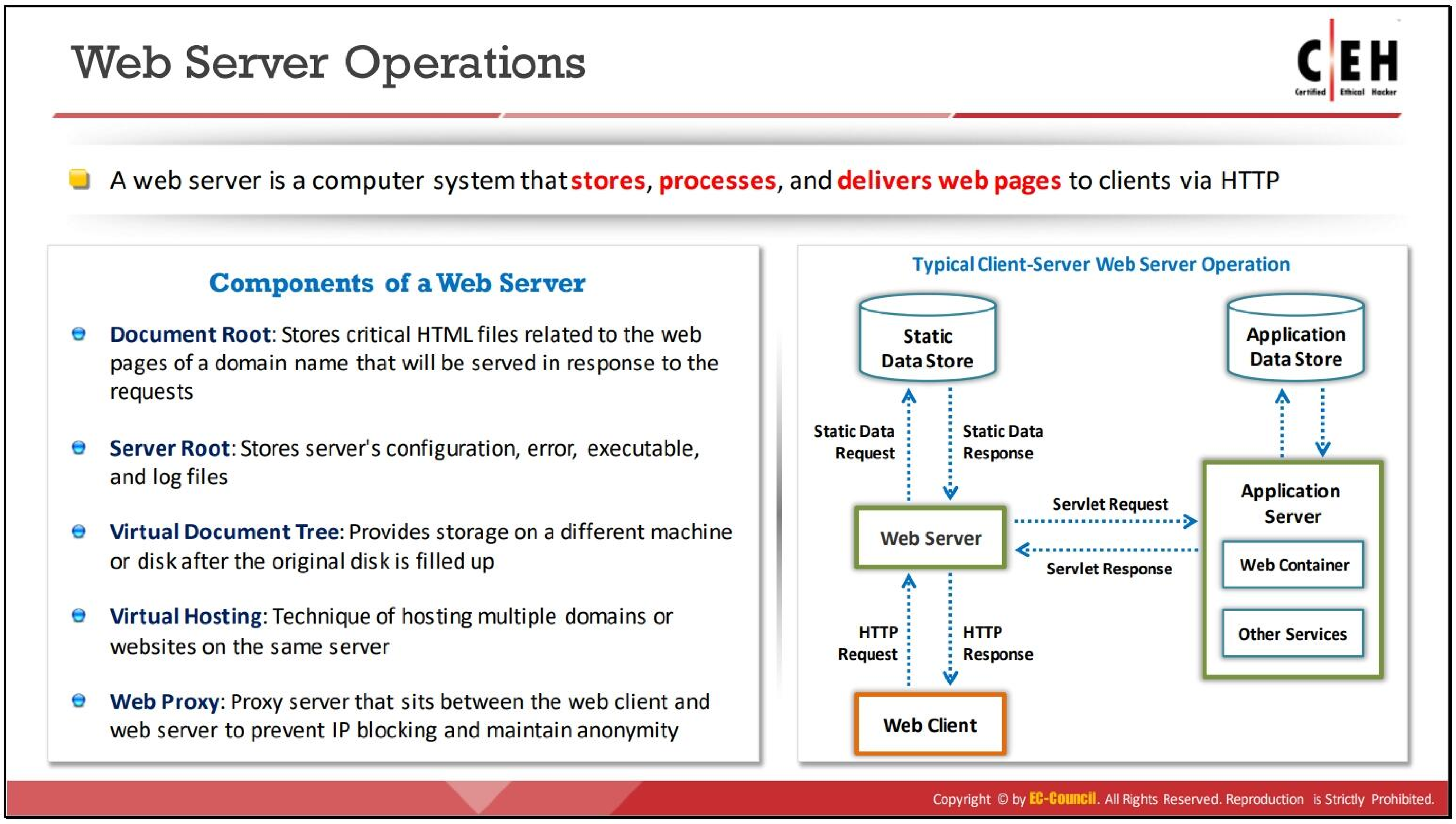
- Describe web server concepts
- Perform various web server attacks
- Describe web server attack methodology
- Use different web server attack tools
- Apply web server attack countermeasures
- Use different web server security tools
- Describe patch management concepts

LO#01: Summarize Web Server Concepts

Web Server Concepts

To understand web server hacking, it is essential to understand web server concepts, including what a web server is, how it functions, and other elements associated with it.

This section provides a brief overview of a web server and its architecture. It will also explain common factors or mistakes that allow attackers to hack a web server. This section also describes the impact of attacks on web servers.



Web Server Operations

A web server is a computer system that stores, processes, and delivers web pages to global clients via the Hypertext Transfer Protocol (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.

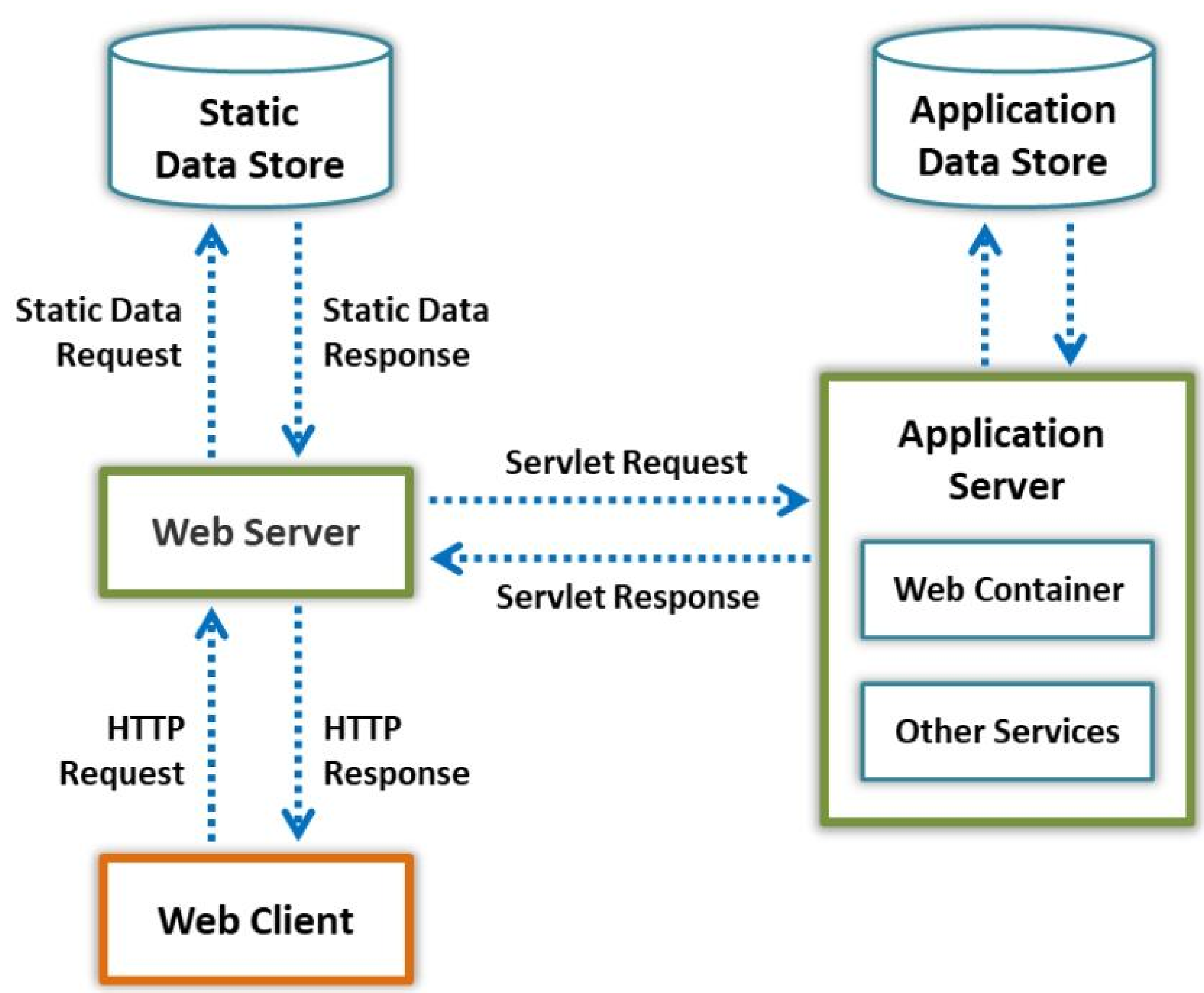


Figure 13.1: Typical client–server communication in web server operation

Components of a Web Server

A web server consists of the following components:

- **Document Root**

  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.

  For example, if the requested URL is *www.certifiedhacker.com* and the document root is named "certroot" and is stored in the directory */admin/web*, then */admin/web/certroot* is the document directory address.

  If the complete request is *www.certifiedhacker.com/P-folio/index.html*, the server will search for the file path */admin/web/certroot/P-folio/index.html*.

- **Server Root**

  It is the top-level root directory under the directory tree in which the server's configuration and error, executable, and log files are stored. It consists of the code that implements the server. The server root, in general, consists of four files. One file is dedicated to the code that implements the server, while the other three are subdirectories, namely, -conf, -logs, and -cgi-bin, which are used for configuration information, logs, and executables, respectively.

- **Virtual Document Tree**

  A virtual document tree provides storage on a different machine or disk after the original disk becomes full. It is case-sensitive and can be used to provide object-level security.

  In the above example under document root, for a request of *www.certifiedhacker.com/P-folio/index.html*, the server can also search for the file path */admin/web/certroot/P-folio/index.html* if the directory *admin/web/certroot* is stored in another disk.

- **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, in which company resources are intended to be accessed and managed globally.

  The following are the types of virtual hosting:

  - Name-based hosting
  - Internet Protocol (IP)-based hosting
  - Port-based hosting

- **Web Proxy**

  A proxy server is located between the web client and web server. Owing to the placement of web proxies, all requests from clients are passed on to the web server through the web proxies. They are used to prevent IP blocking and maintain anonymity.

Open-source Web Server Architecture

Open-source web server architecture typically uses Linux, Apache, MySQL, and PHP, often called the LAMP software bundle, as the principal components.

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

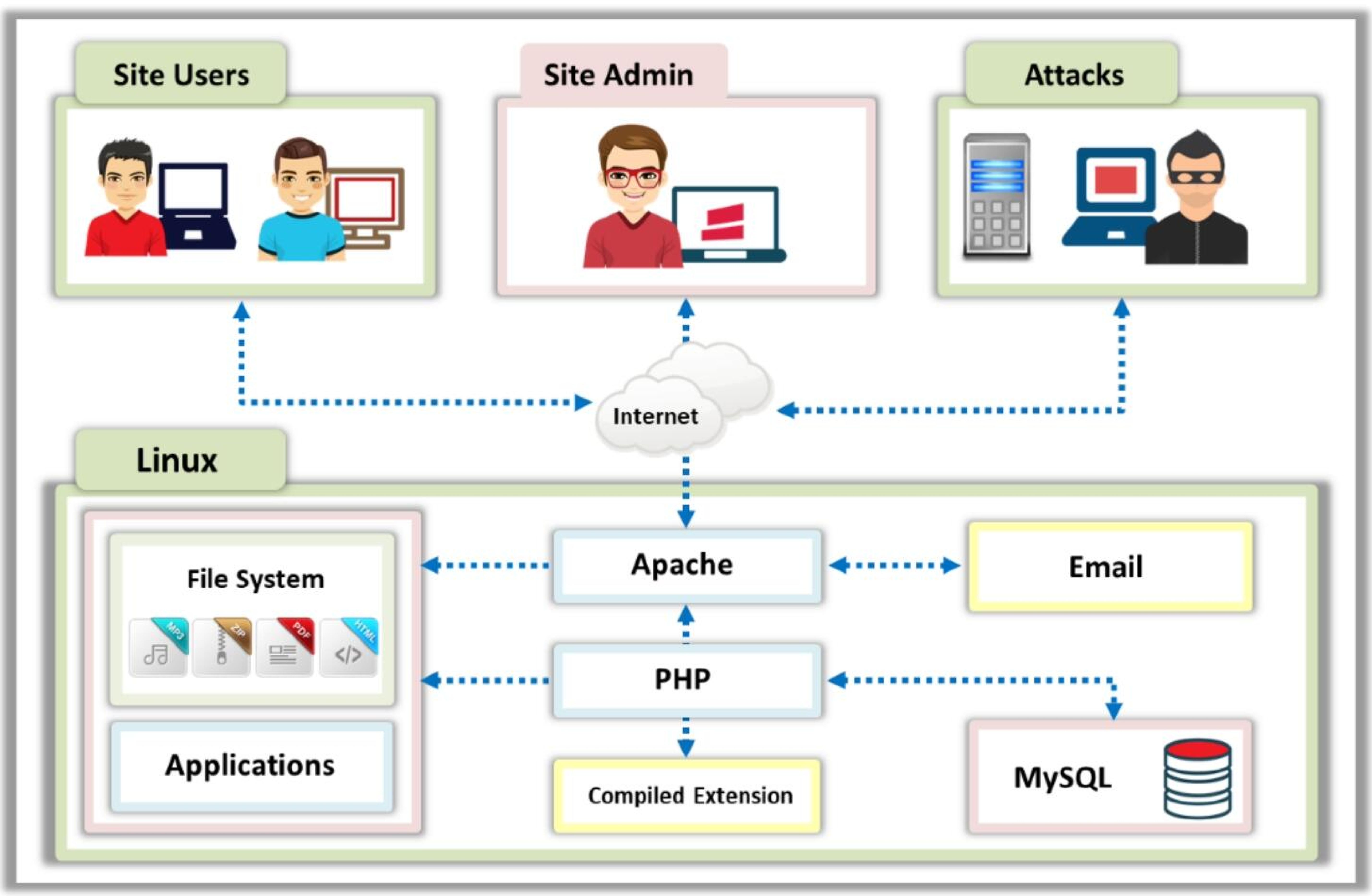


Figure 13.2: Functions of the principal components of the open-source web server architecture

IIS Web Server Architecture

The 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 Secure (HTTPS), 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 (WWW 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.

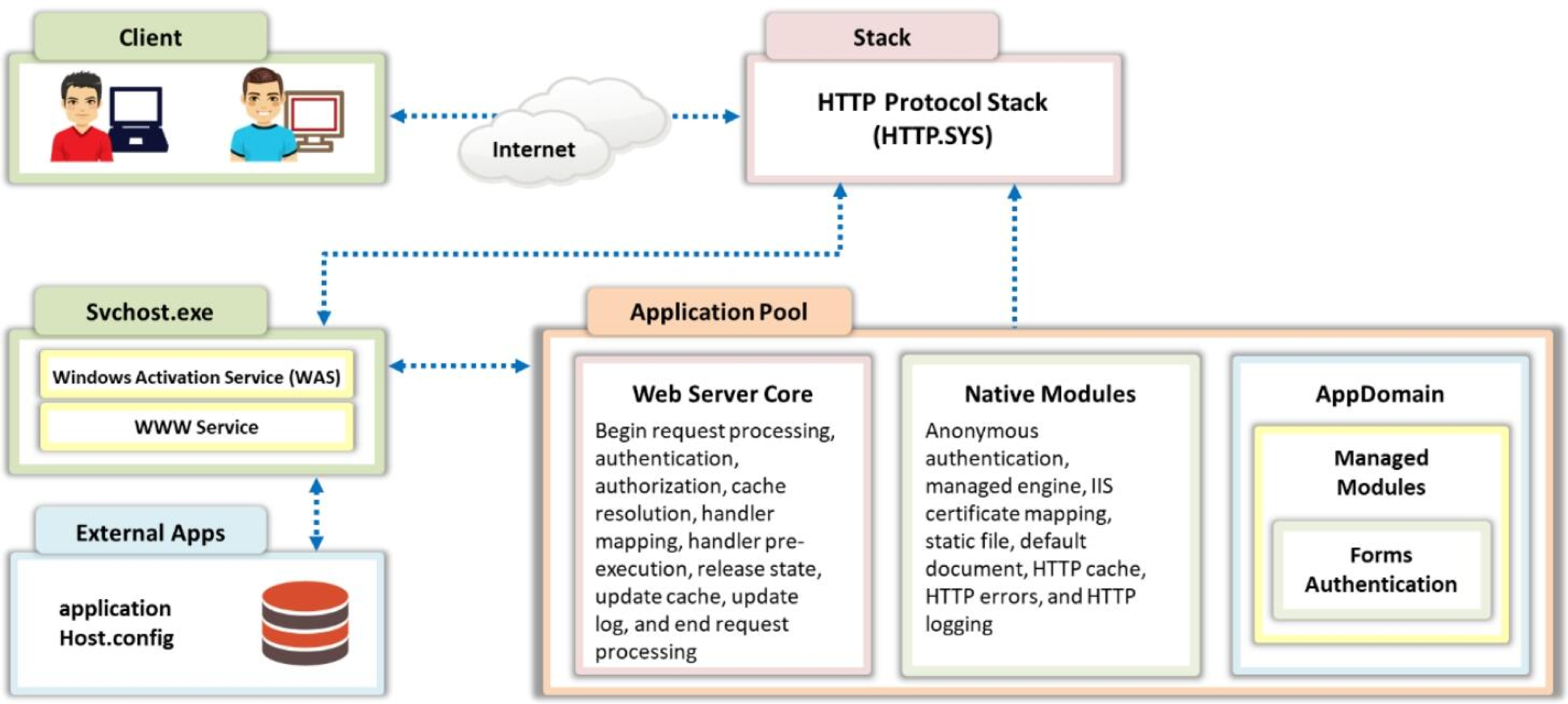
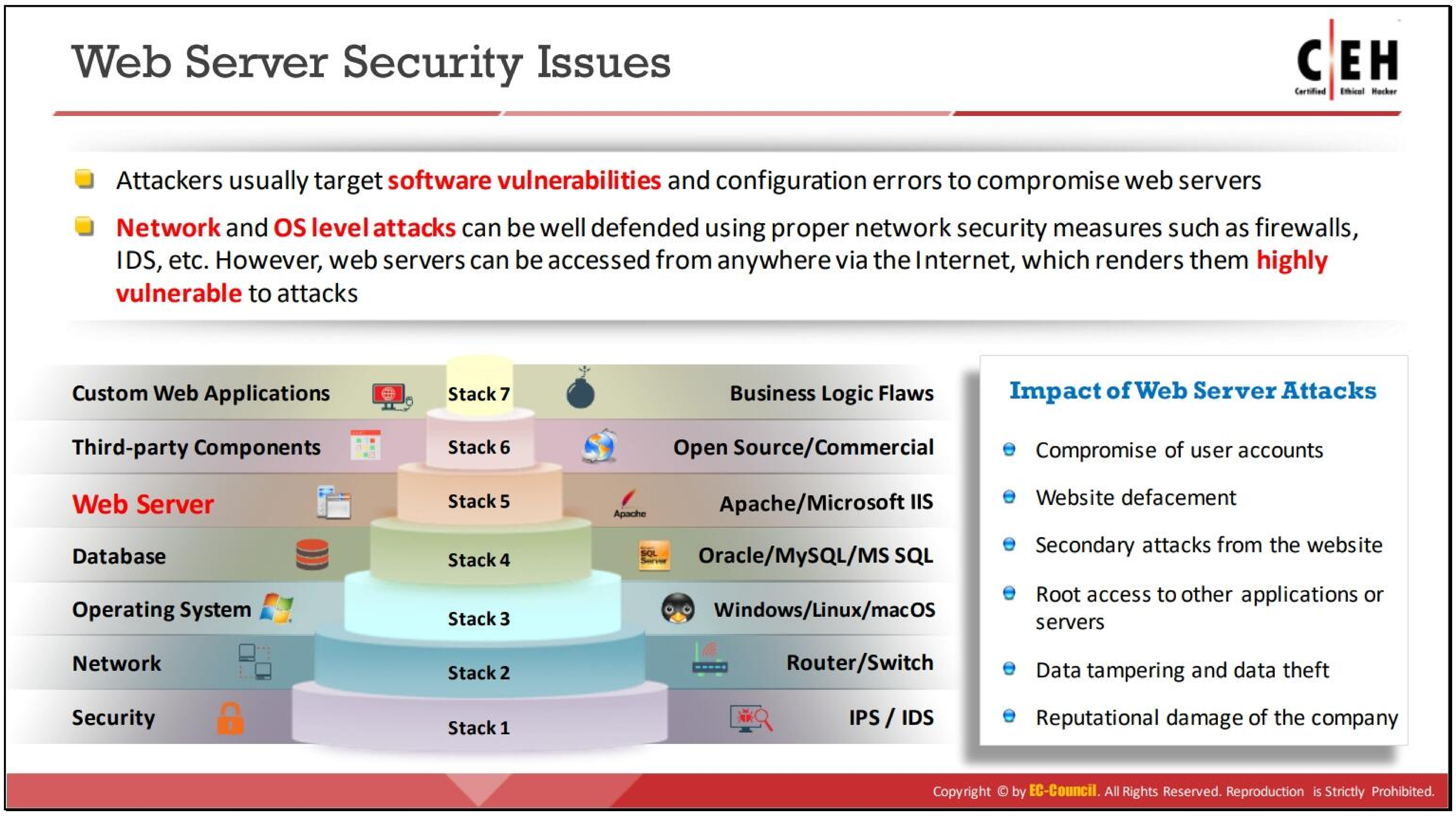


Figure 13.3: Components of the IIS web server architecture



Web Server Security Issues

A web server is a hardware/software application that hosts websites and makes them accessible over the Internet. A web server, along with a browser, successfully implements client–server model architecture. In this model, the web server plays the role of the server, and the browser acts as the client. To host websites, a web server stores the web pages of websites and delivers a particular web page upon request. Each web server has a domain name and an IP address associated with that domain name. A web server can host more than one website. Any computer can act as a web server if it has specific server software (a web server program) installed and is connected to the Internet.

Web servers are chosen based on their capability to handle server-side programming, security characteristics, publishing, search engines, and site-building tools. Apache, Microsoft IIS, Nginx, Google, and Tomcat are some of the most widely used web server software. An attacker usually targets vulnerabilities in the software component and configuration errors to compromise web servers.

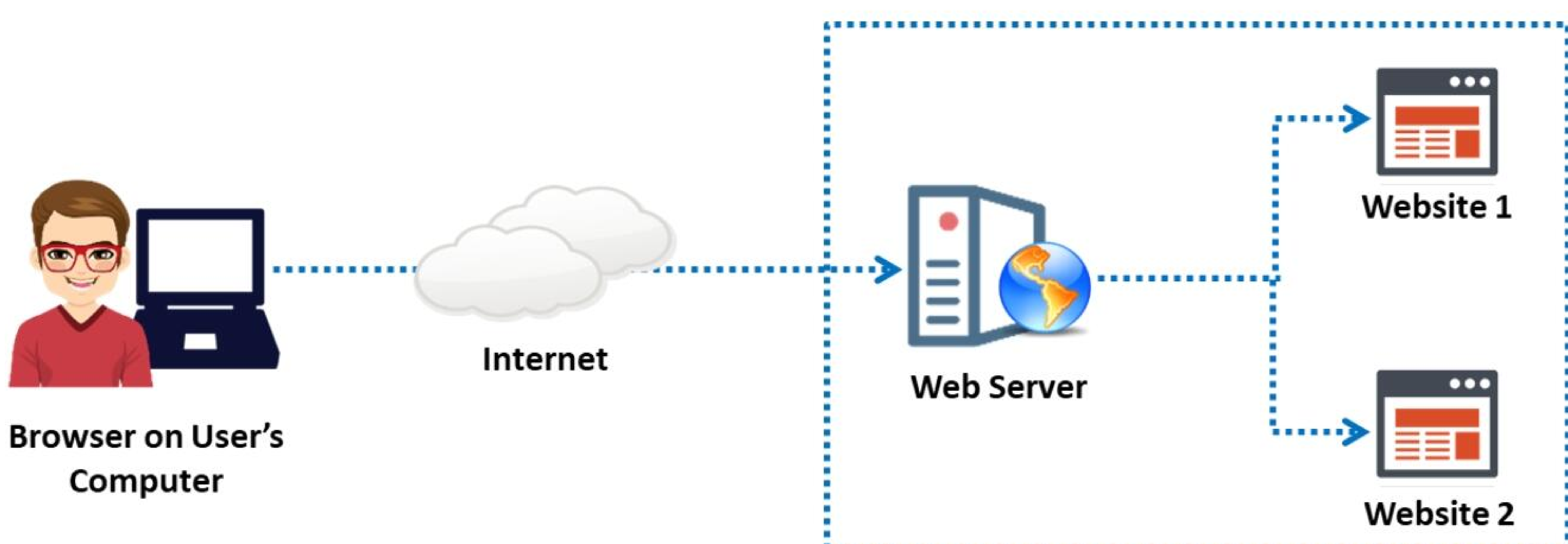


Figure 13.4: Conceptual diagram of a web server: the user visits websites hosted on a web server

Organizations can defend most network-level and OS-level attacks by adopting network security measures such as firewalls, intrusion detection systems (IDSs), and intrusion prevention systems (IPSs) and by following security standards and guidelines. This forces attackers to turn their attention to web-server- and web-application-level attacks because a web server that hosts web applications is accessible from anywhere over the Internet. This makes web servers an attractive target. Poorly configured web servers can create vulnerabilities in even the most carefully designed firewall systems. Attackers can exploit poorly configured web servers with known vulnerabilities to compromise the security of web applications. Furthermore, web servers with known vulnerabilities can harm the security of an organization. As shown in below figure, organizational security includes seven levels from stack 1 to stack 7.

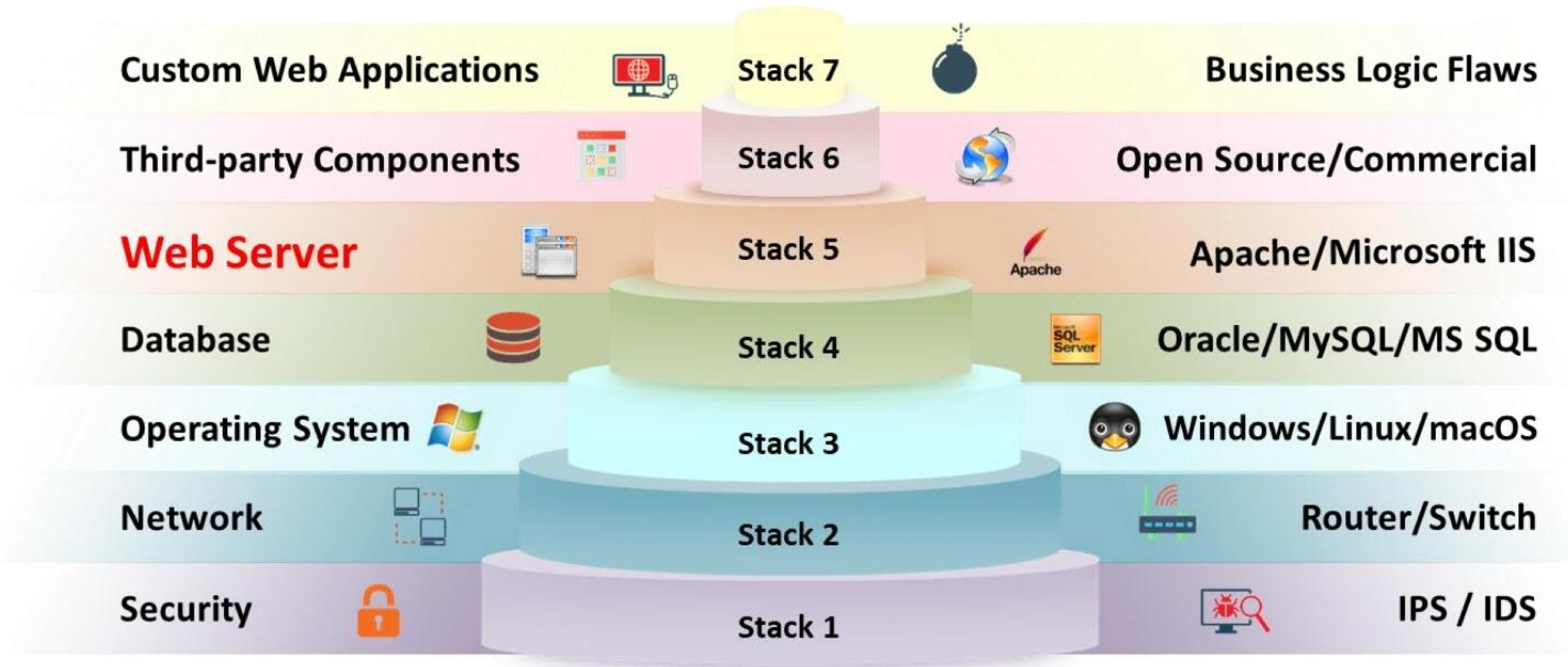


Figure 13.5: Levels of organizational security

Common Goals behind Web Server Hacking

Attackers perform web server attacks with certain goals in mind. These goals may be either technical or non-technical. For example, attackers may breach the security of a web server and steal sensitive information for financial gains or merely for the sake of curiosity.

The following are some common goals of web server attacks:

- Stealing credit-card details or other sensitive credentials using phishing techniques
- Integrating the server into a botnet to perform denial of service (DoS) or distributed DoS (DDoS) attacks
- Compromising a database
- Obtaining closed-source applications
- Hiding and redirecting traffic
- Escalating privileges

Some attacks are performed for personal reasons, rather than financial gains:

- For pure curiosity

- For completing a self-set intellectual challenge
- For damaging the target organization's reputation

Dangerous Security Flaws Affecting Web Server Security

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 following are some common oversights that make a web server vulnerable to attacks:

- 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

Attackers can cause various kinds of damage to an organization by attacking a web server. The following are some of the types of damage that attackers can cause to a web server.

- **Compromise of user accounts**: Web server attacks mostly focus on compromising user accounts. If the attacker compromises a user account, they can gain a large amount of useful information. The attacker can use the compromised user account to launch further attacks on the web server.
- **Website defacement**: Attackers can completely change the appearance of a website by replacing its original data. They deface the target website by changing the visuals and displaying different pages with messages of their own.
- **Secondary attacks from the website**: An attacker who compromises a web server can use the server to launch further attacks on various websites or client systems.
- **Root access to other applications or server**: Root access is the highest privilege level to log in to a server, irrespective of whether the server is a dedicated, semi-dedicated, or virtual private server. Attackers can perform any action once they attain root access to the server.
- **Data tampering**: An attacker can alter or delete the data of a web server and even replace the data with malware to compromise users who connect to the web server.
- **Data theft**: Data are among the primary assets of an organization. Attackers can attain access to sensitive data such as financial records, future plans, or the source code of a program.
- **Damage reputation of the company**: Web server attacks may expose the personal information of a company's customers to the public, damaging the reputation of the company. Consequently, customers lose faith in the company and become afraid of sharing their personal details with the company.

Why are Web Servers Compromised?

- **Improper** file and directory **permissions**
- Server installation with **default settings**
- Enabling of **unnecessary services**, including content management and remote administration
- **Security conflicts** with business ease-of-use case
- **Lack of proper security policies**, procedures, and maintenance
- **Improper authentication** with external systems
- **Default accounts** having 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
- Not using **dedicated server** for web services

Why are Web Servers Compromised?

There are inherent security risks associated with web servers, the local area networks (LANs) that host websites, and the end users who access these websites using browsers.

- **Webmaster's perspective**: From a webmaster's perspective, the greatest security concern is that a web server can expose the LAN or corporate intranet to threats posed by the Internet. These threats may be in the form of viruses, Trojans, attackers, or the compromise of data. Bugs in software programs are often sources of security lapses. Web servers, which are large and complex devices, also have these inherent risks. In addition, the open architecture of web servers allows arbitrary scripts to run on the server side while responding to remote requests. Any Common Gateway Interface (CGI) script installed in the web server may contain bugs that are potential security holes.
- **Network administrator's perspective**: From a network administrator's perspective, a poorly configured web server causes potential holes in the LAN's security. While the objective of the web server is to provide controlled access to the network, excess control can make the web almost impossible to use. In an intranet environment, the network administrator must configure the web server carefully so that legitimate users are recognized and authenticated, and groups of users are assigned distinct access privileges.
- **End user's perspective**: Usually, the end user does not perceive any immediate threat, because surfing the web appears both safe and anonymous. However, active content, such as ActiveX controls and Java applets, make it possible for harmful applications, such as viruses, to invade the user's system. In addition, active content from a website that is displayed by the user's browser can be used as a conduit for malicious software to bypass the firewall system and permeate the LAN.

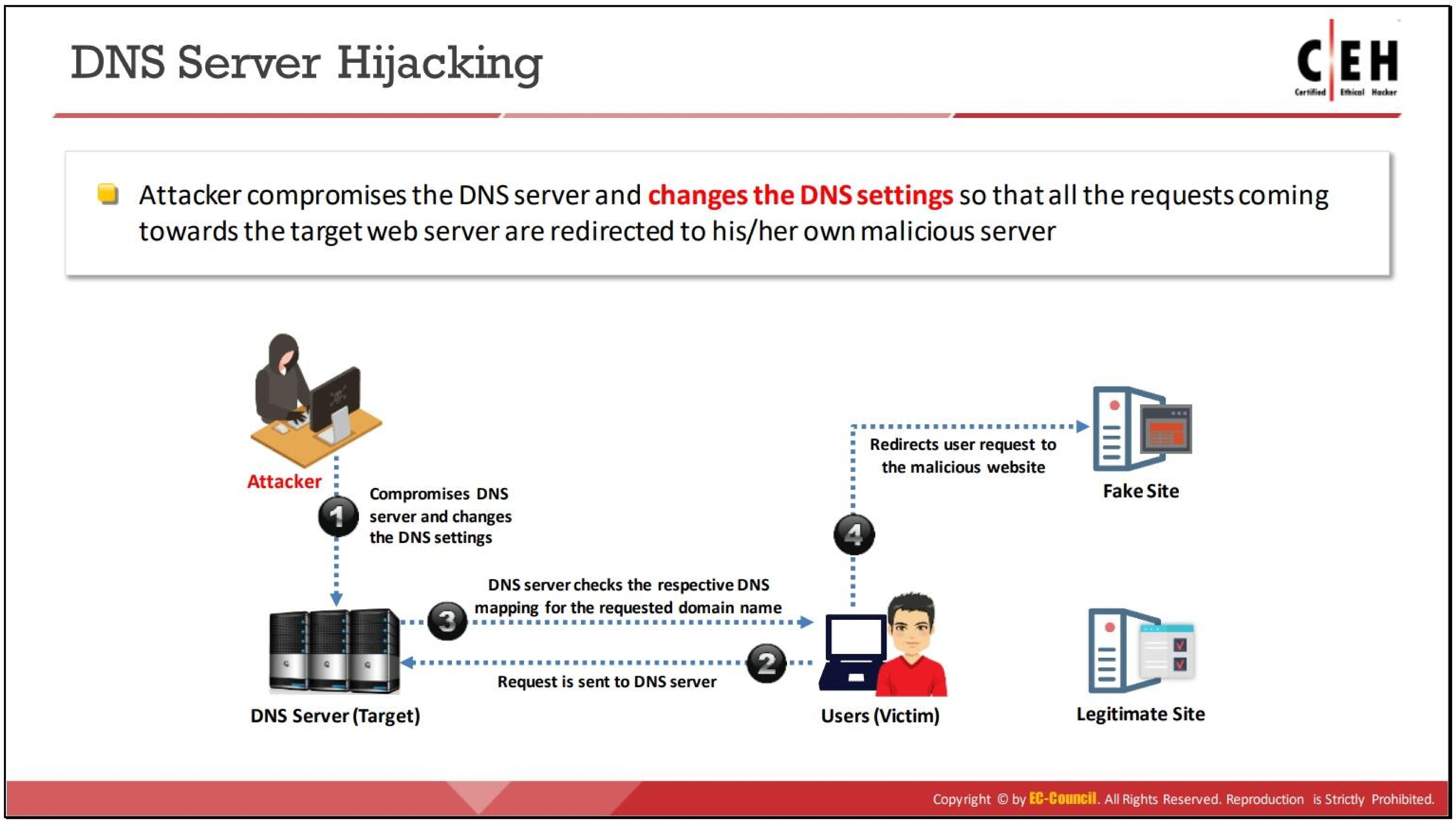
The following are some oversights that can compromise a web server:

- Improper file and directory permissions
- Installing the server with default settings
- Unnecessary services enabled, including content management and remote administration
- Security conflicts with the business' ease-of-use requirements
- Lack of proper security policy, procedures, and maintenance
- Improper authentication with external systems
- Default accounts with default or no passwords
- Unnecessary default, backup, or sample files
- Misconfigurations in the web server, OS, and networks
- Bugs in server software, OS, and web applications
- Misconfigured Secure Sockets Layer (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
- Not using dedicated server for web services

LO#02: Demonstrate Different 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, and web server password cracking. This section describes these attack techniques in detail.



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.

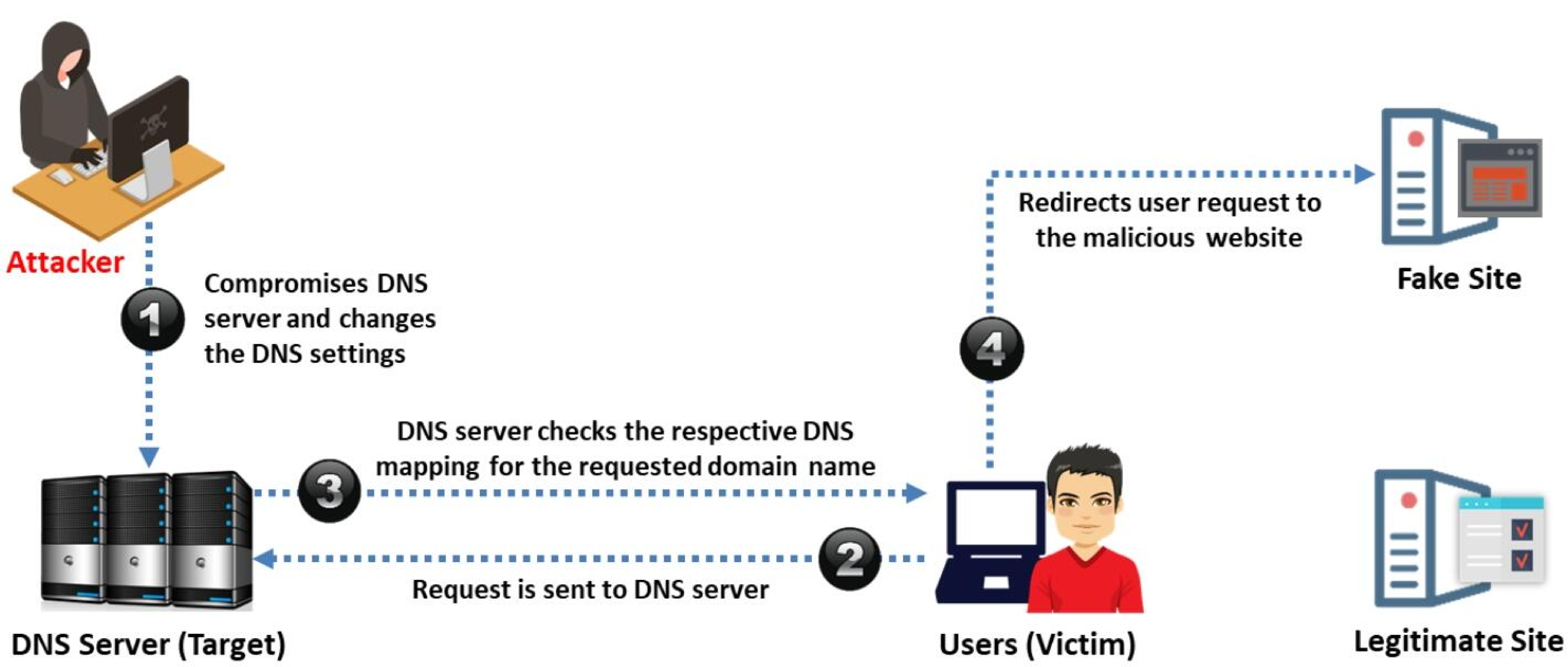
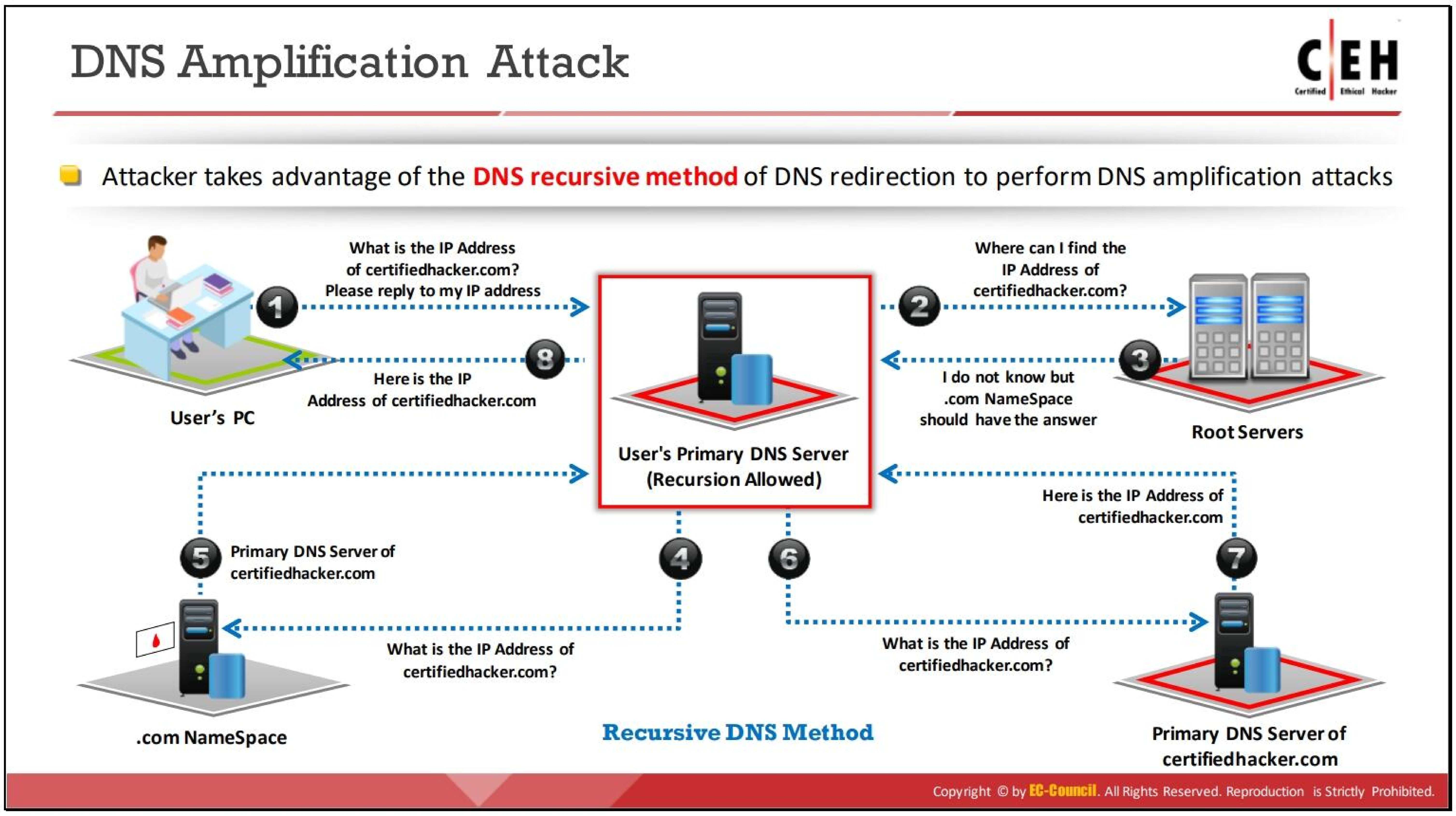
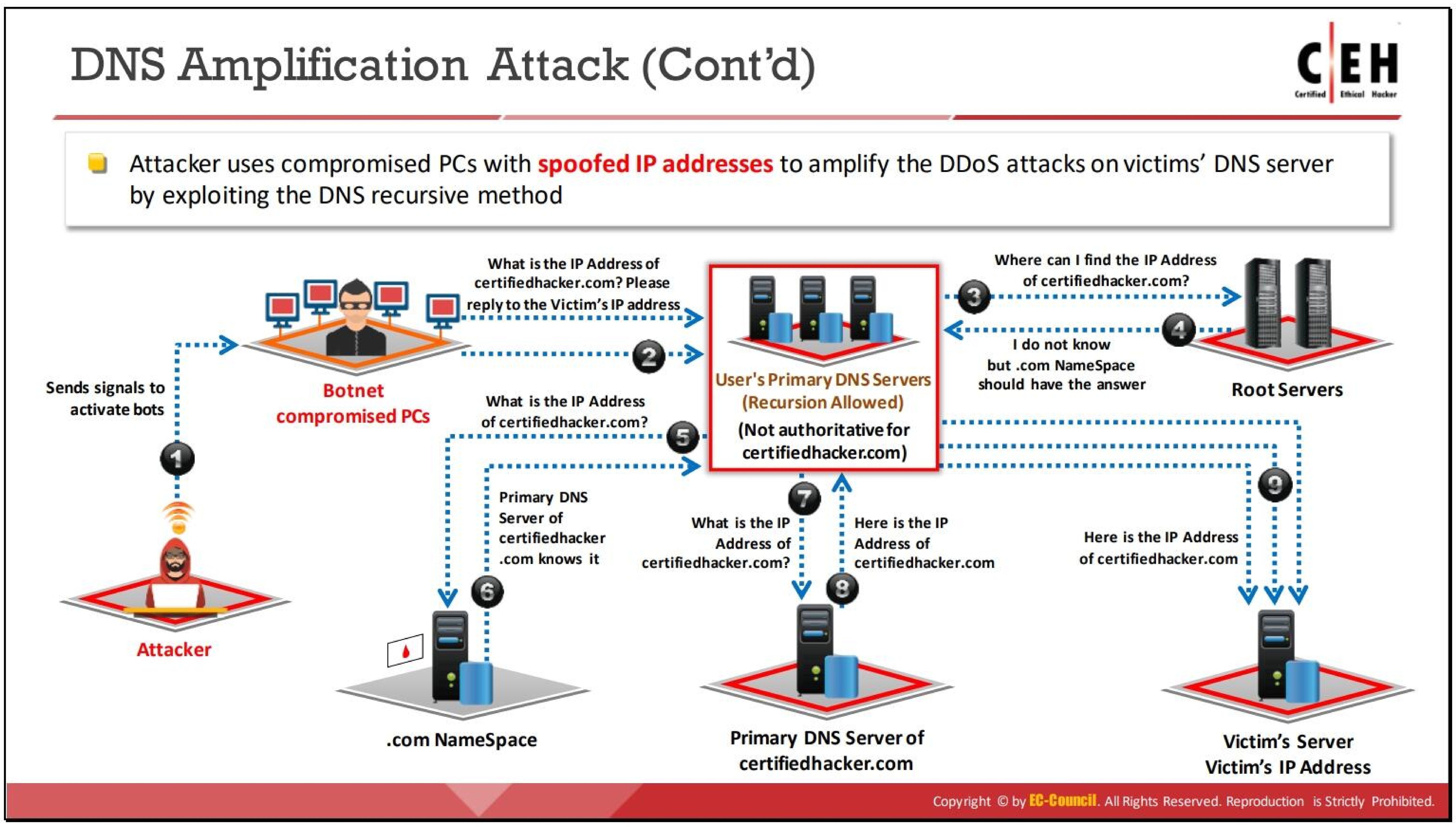


Figure 13.6: DNS server hijacking





DNS Amplification Attack

Recursive DNS query is a method of requesting DNS mapping. The query goes through DNS servers recursively until it fails to find the specified domain name to IP address mapping.

The following are the steps involved in processing recursive DNS requests; these steps are illustrated in the below figure.

- **Step 1:**

  Users who desire to resolve a domain name to its corresponding IP address send a DNS query to the primary DNS server specified in its Transmission Control Protocol (TCP)/IP properties.

- **Steps 2 to 7:**

  If the requested DNS mapping does not exist on the user's primary DNS server, the server forwards the request to the root server. The root server forwards the request to the .com namespace, where the user can find DNS mappings. This process repeats recursively until the DNS mapping is resolved.

- **Step 8:**

  Ultimately, when the system finds the primary DNS server for the requested DNS mapping, it generates a cache for the IP address in the user's primary DNS server.

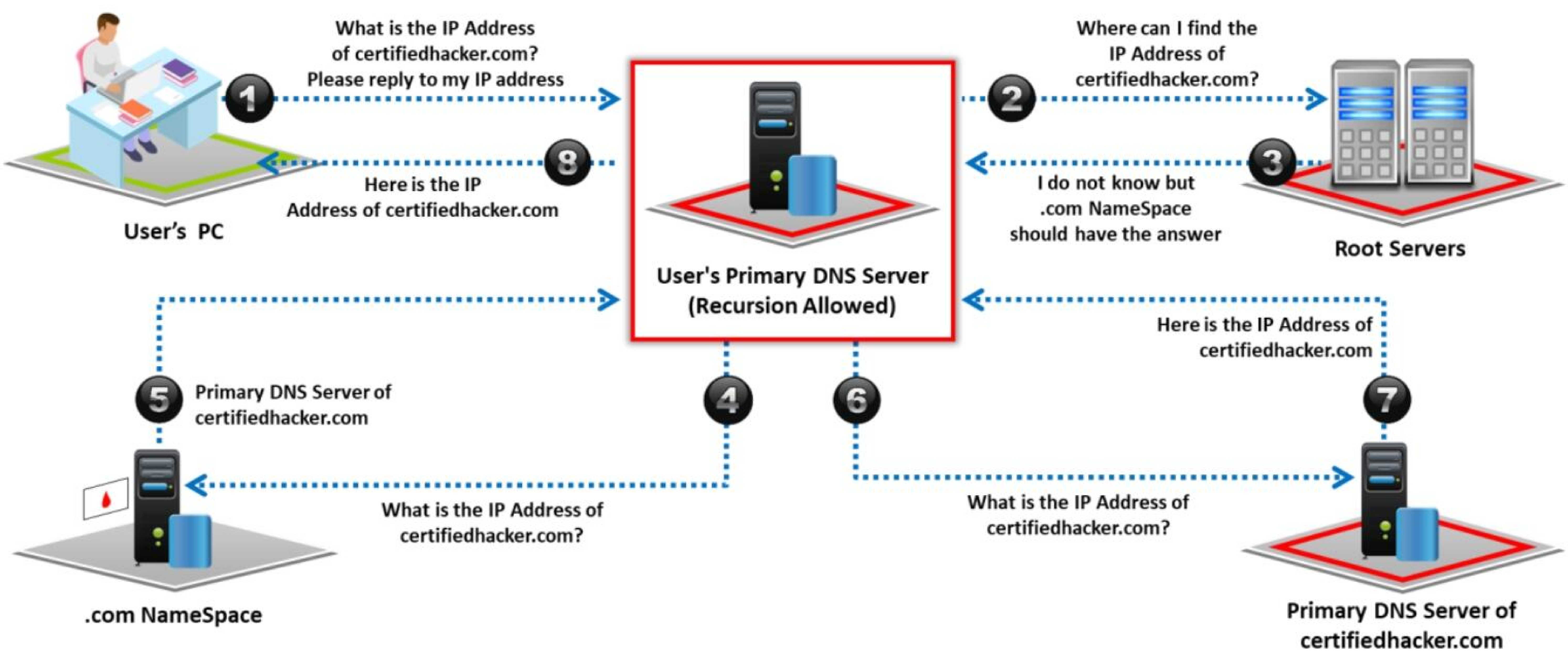


Figure 13.7: Recursive DNS query

Attackers exploit recursive DNS queries to perform a DNS amplification attack that results in DDoS attacks on the victim's DNS server.

The following are the steps involved in a DNS amplification attack; these steps are illustrated in the below figure.

- **Step 1:**

  The attacker instructs compromised hosts (bots) to make DNS queries in the network.

- **Step 2:**

  All the compromised hosts spoof the victim's IP address and send DNS query requests to the primary DNS server configured in the victim's TCP/IP settings.

- **Steps 3 to 8:**

  If the requested DNS mapping does not exist on the victim's primary DNS server, the server forwards the requests to the root server. The root server forwards the request to the .com or respective top-level domain (TLD) namespaces. This process repeats recursively until the victim's primary DNS server resolves the DNS mapping request.

- **Step 9:**

  After the primary DNS server finds the DNS mapping for the victim's request, it sends a DNS mapping response to the victim's IP address. This response goes to the victim because bots use the victim's IP address. The replies to copious DNS mapping requests from the bots result in DDoS on the victim's DNS server.

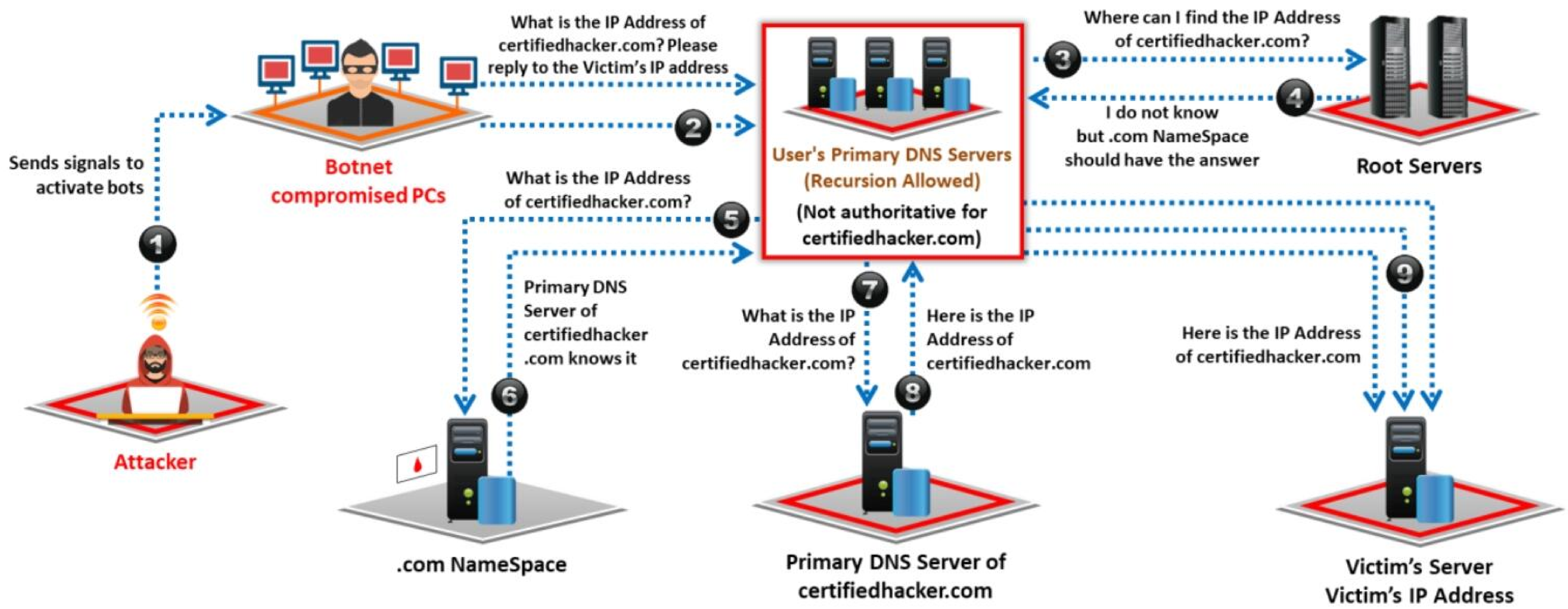
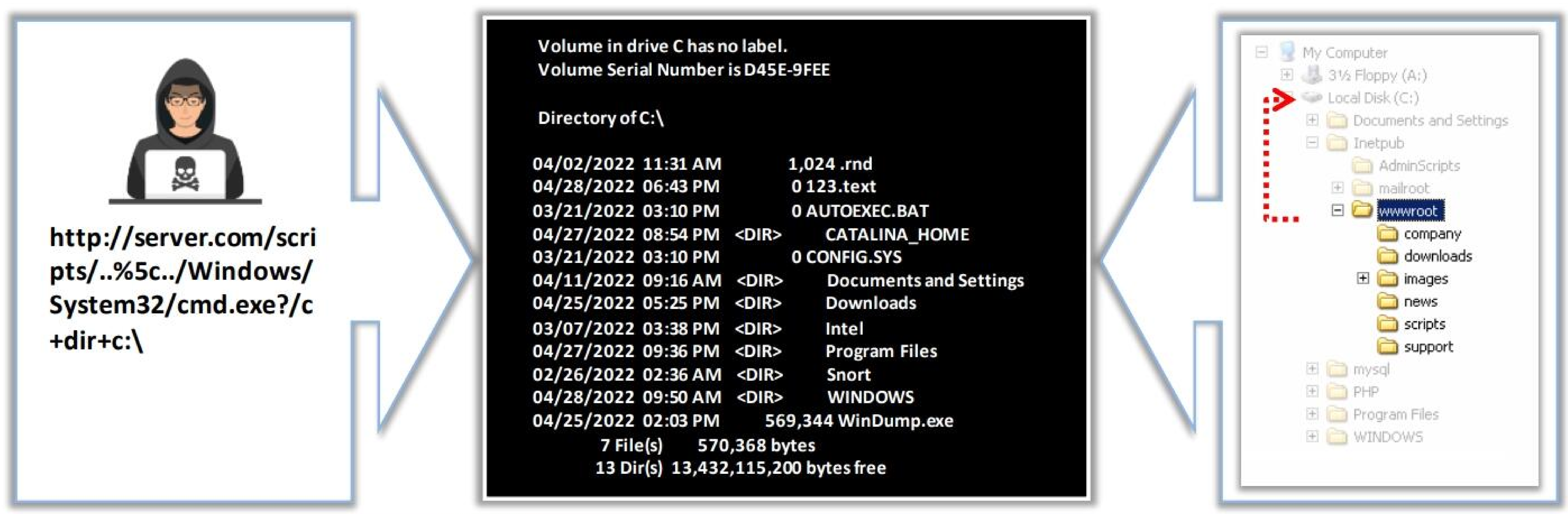


Figure 13.8: DNS amplification attack

Directory Traversal Attacks

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



Directory Traversal Attacks

An attacker may be able to perform a directory traversal attack owing to a vulnerability in the code of a web application. In addition, poorly patched or configured web server software can make the web server vulnerable to a directory traversal attack.

The design of web servers limits public access to some extent. Directory traversal is the exploitation of HTTP through which attackers can access restricted directories and execute commands outside the web server's root directory by manipulating a Uniform Resource Locator (URL). In directory traversal attacks, attackers use the dot-dot-slash (../) sequence to access restricted directories outside the web server's root directory. Attackers can use the trial-and-error method to navigate outside the root directory and access sensitive information in the system.

An attacker exploits the web server software (web server program) to perform directory traversal attacks. The attacker usually performs this attack with the help of a browser. A web server is vulnerable to this attack if it accepts input data from a browser without proper validation.

Figure 13.9: Directory traversal attack

Website Defacement

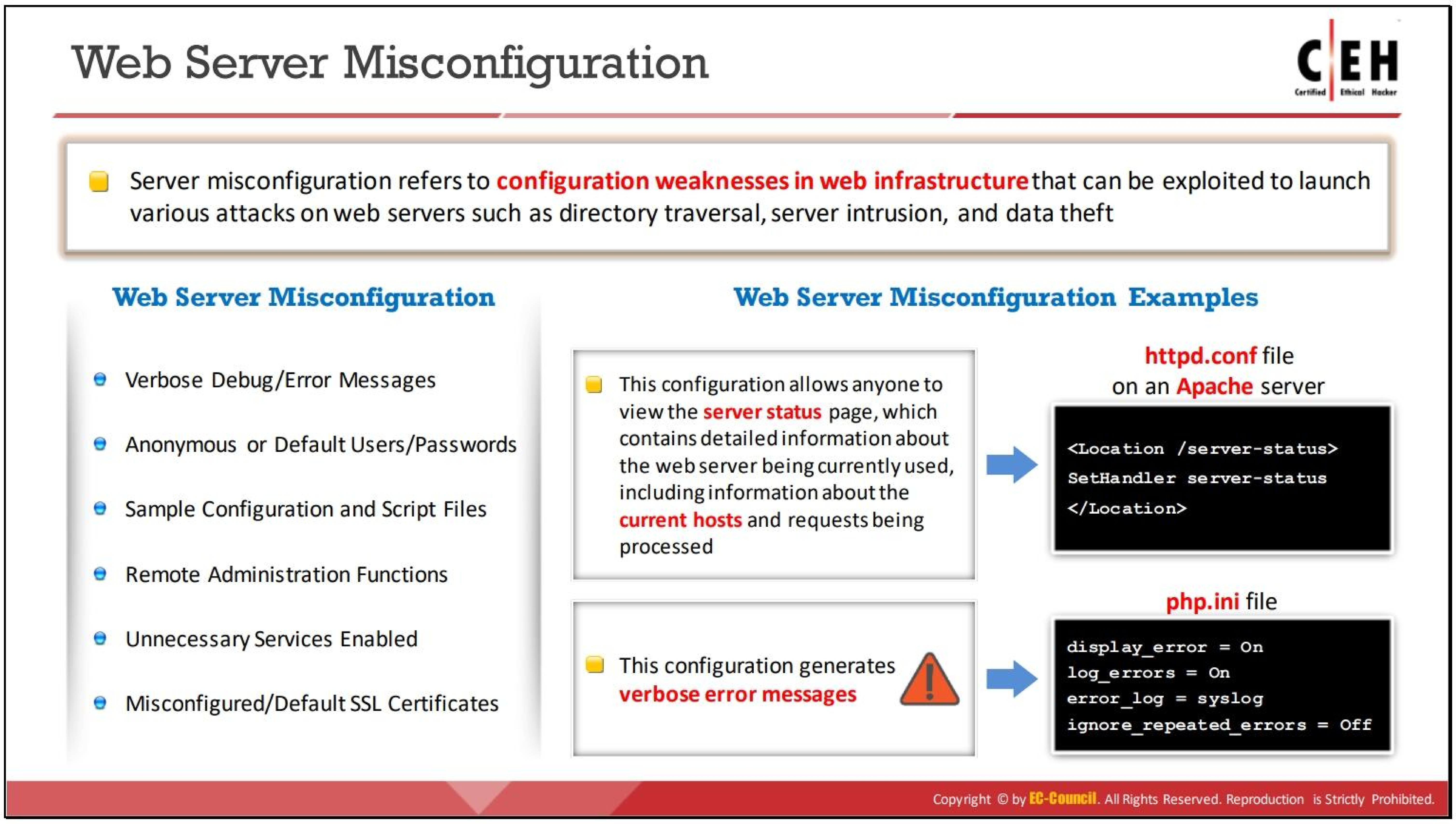
- Web defacement occurs when an intruder **maliciously alters the visual appearance of a web page** by inserting or substituting provocative, and frequently, offending data
- **Defaced pages expose visitors to some propaganda** or misleading information until the unauthorized changes are discovered and corrected
- Attackers use a variety of methods such as **MYSQL injection** to access a site in order to deface it

Website Defacement

Website defacement refers to unauthorized changes made to the content of a single web page or an entire website, resulting in changes to the visual appearance of the web page or website. Hackers break into web servers and alter the hosted website by injecting code to add images, popups, or text to a page in such a manner that the visual appearance of the page changes. In some cases, the attacker may replace the entire website instead of just changing a single page.

Figure 13.10: Screenshot displaying a website defacement attack

Defaced pages expose visitors to 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 website to deface it. In addition to changing the visual appearance of the target website, attackers deface websites for infecting the computers of visitors by making the website vulnerable to virus attacks. Thus, website defacement not only embarrasses the target organization by changing the appearance of its website but is also intended to harm its visitors.



Web Server Misconfiguration

Web server misconfiguration refers to the 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. The following are some 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

An Example of a Web Server Misconfiguration

"**Keeping the server configuration secure requires vigilance**"—Open Web Application Security Project (OWASP)

Administrators who configure web servers improperly may leave serious loopholes in the web server, thereby providing an attacker the chance to exploit the misconfigured web server to compromise its security and obtain sensitive information. The vulnerabilities of improperly configured web servers may be related to configuration, applications, files, scripts, or web pages. An attacker searches for such vulnerable web servers to launch attacks. The misconfiguration of a web server provides the attacker a path to enter the target network of an organization. These loopholes in the server can also help an attacker bypass user authentication. Once detected, these problems can be easily exploited and may result in the total compromise of a website hosted on the target web server.

As shown in the below figure, the configuration may allow 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.

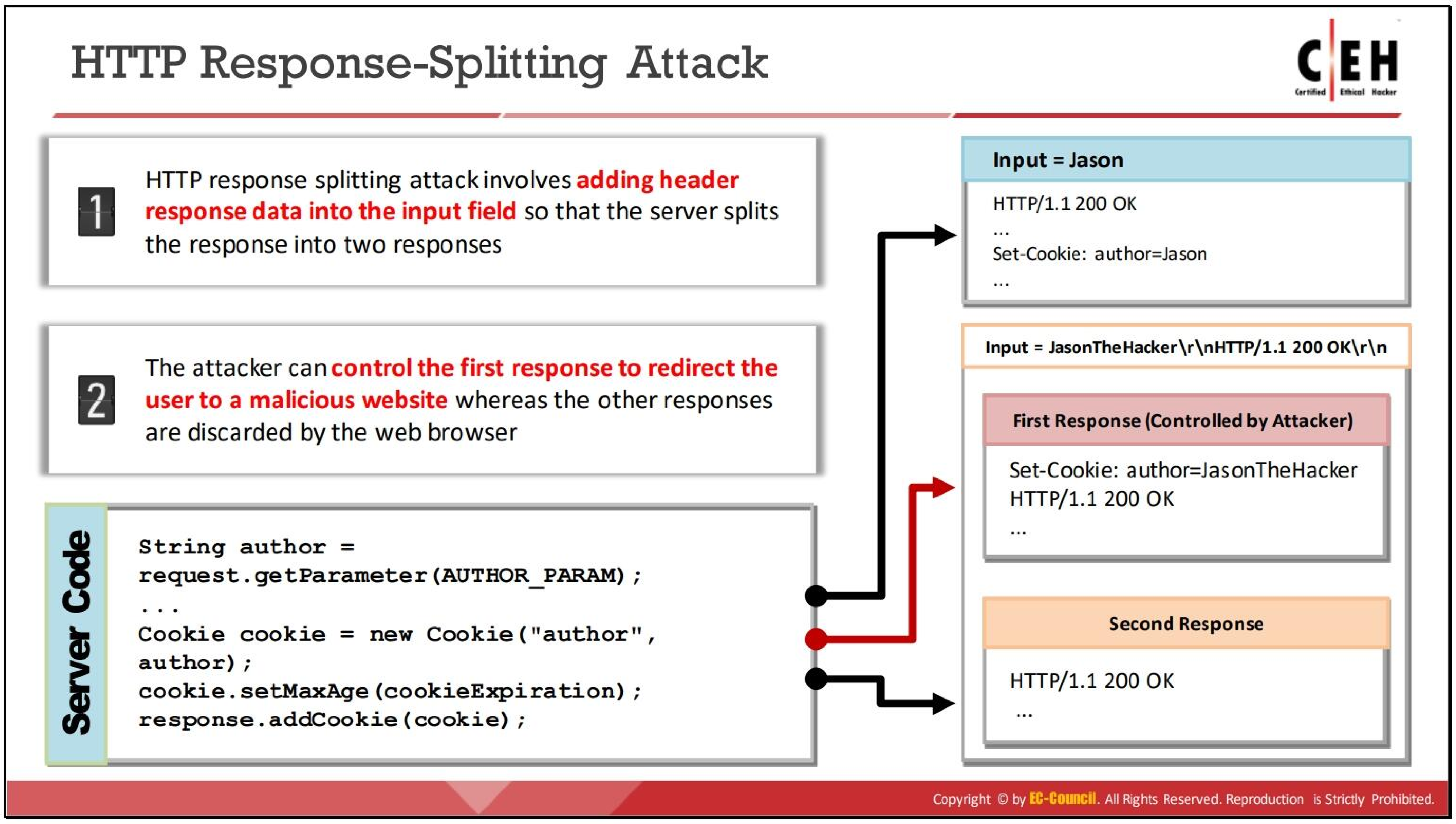
```
<Location /server-status>
SetHandler server-status
</Location>
```

Figure 13.11: Screenshot displaying the httpd.conf file on an Apache server

As shown in the below figure, the configuration may give verbose error messages.

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

Figure 13.12: Screenshot displaying the php.ini file



HTTP Response-Splitting Attack

An HTTP response-splitting attack is a web-based attack in which the attacker tricks the server by injecting new lines into response headers, along with arbitrary code. It involves adding header response data into the input field so that the server splits the response into two responses. This type of attack exploits vulnerabilities in input validation. Cross-site scripting (XSS), cross-site request forgery (CSRF), and Structured Query Language (SQL) injection are examples of this type of attack. In this attack, the attacker controls the input parameter and cleverly constructs a request header that elicits two responses from the server. The attacker alters a single request to appear as two requests by adding header response data into the input field. The web server, in turn, responds to each request. The attacker can pass malicious data to a vulnerable application, and the application includes the data in an HTTP response header. The attacker can control the first response to redirect the user to a malicious website, whereas the web browser will discard other responses.

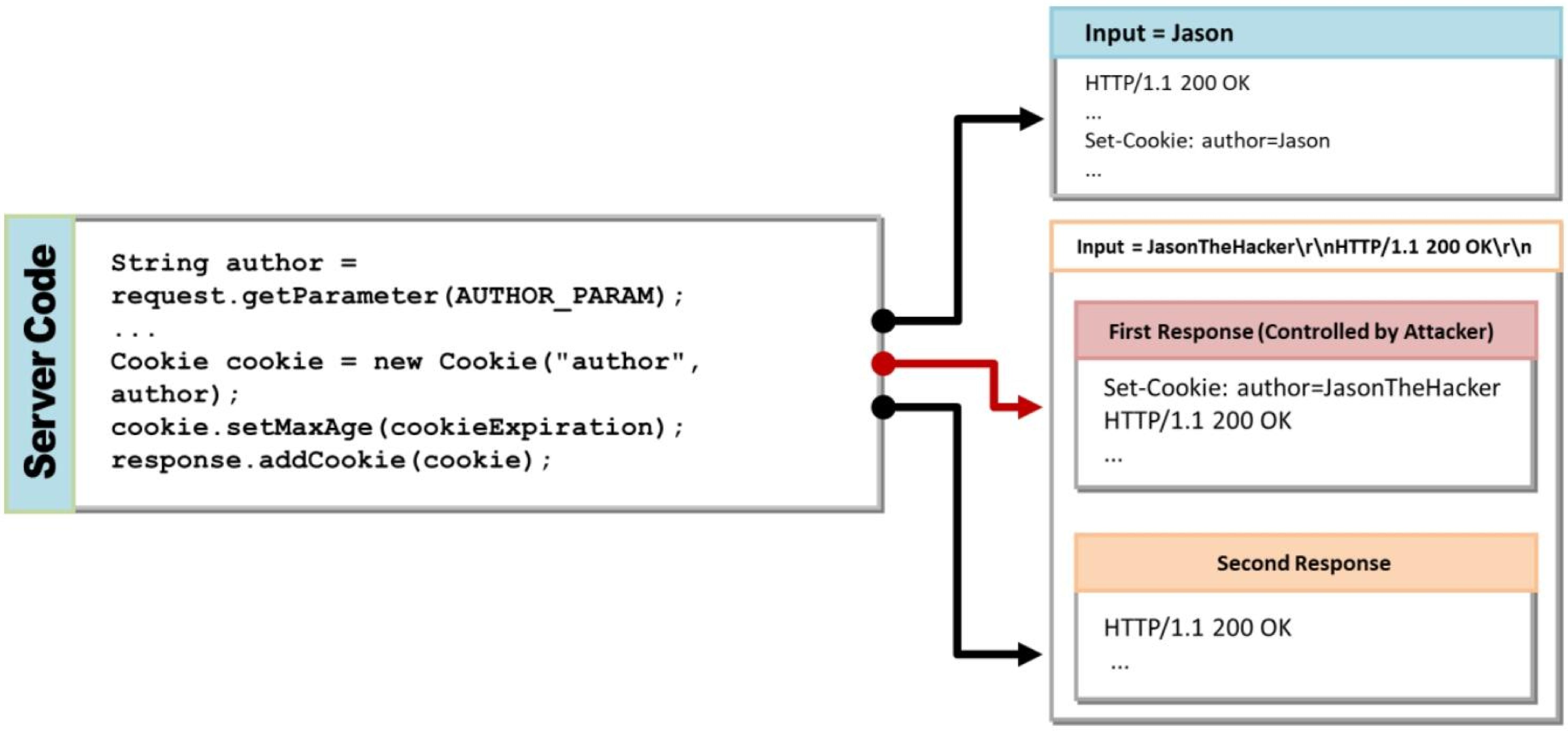


Figure 13.13: HTTP Response-Splitting attack

Example of an HTTP Response-Splitting Attack

In this example, 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.

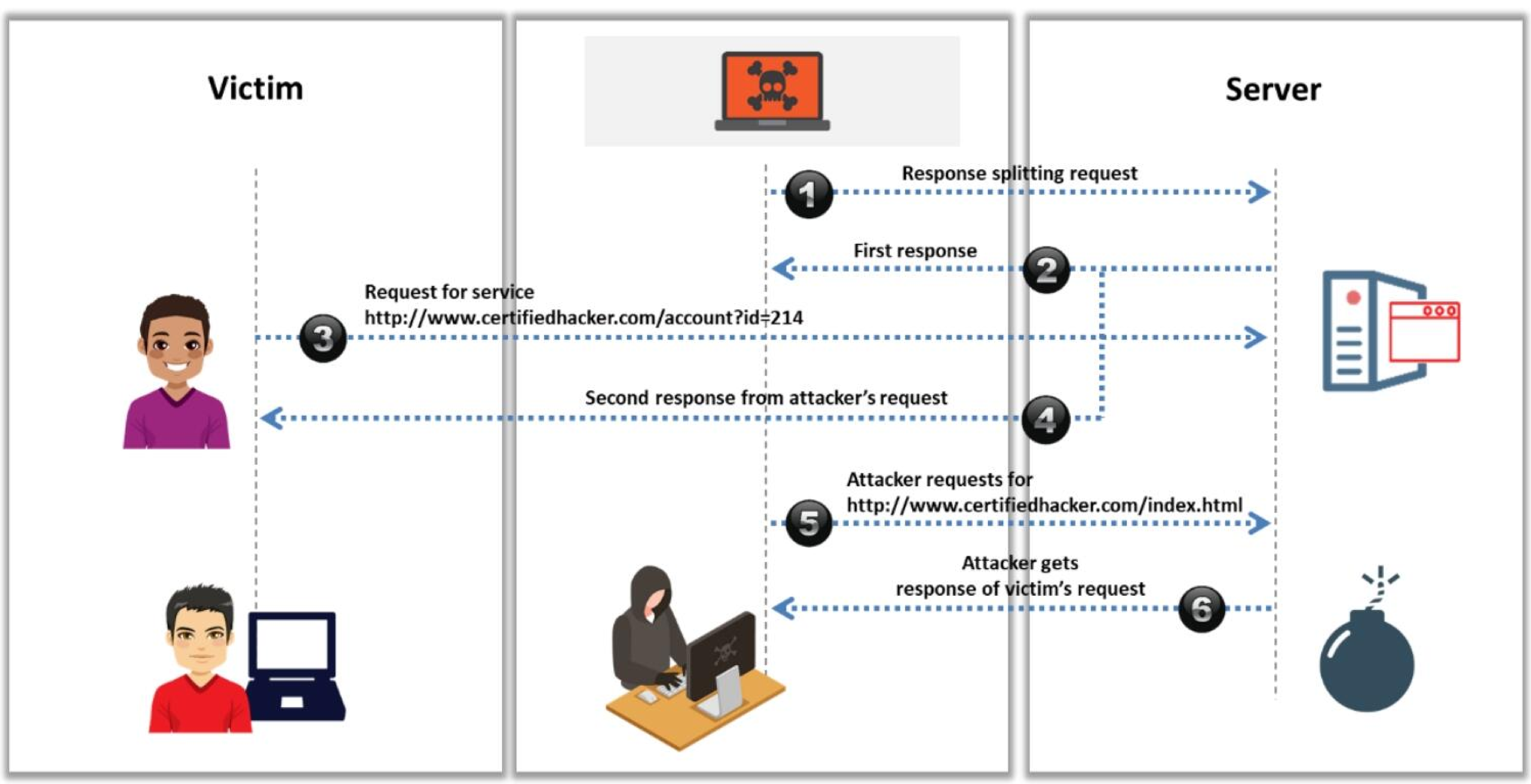
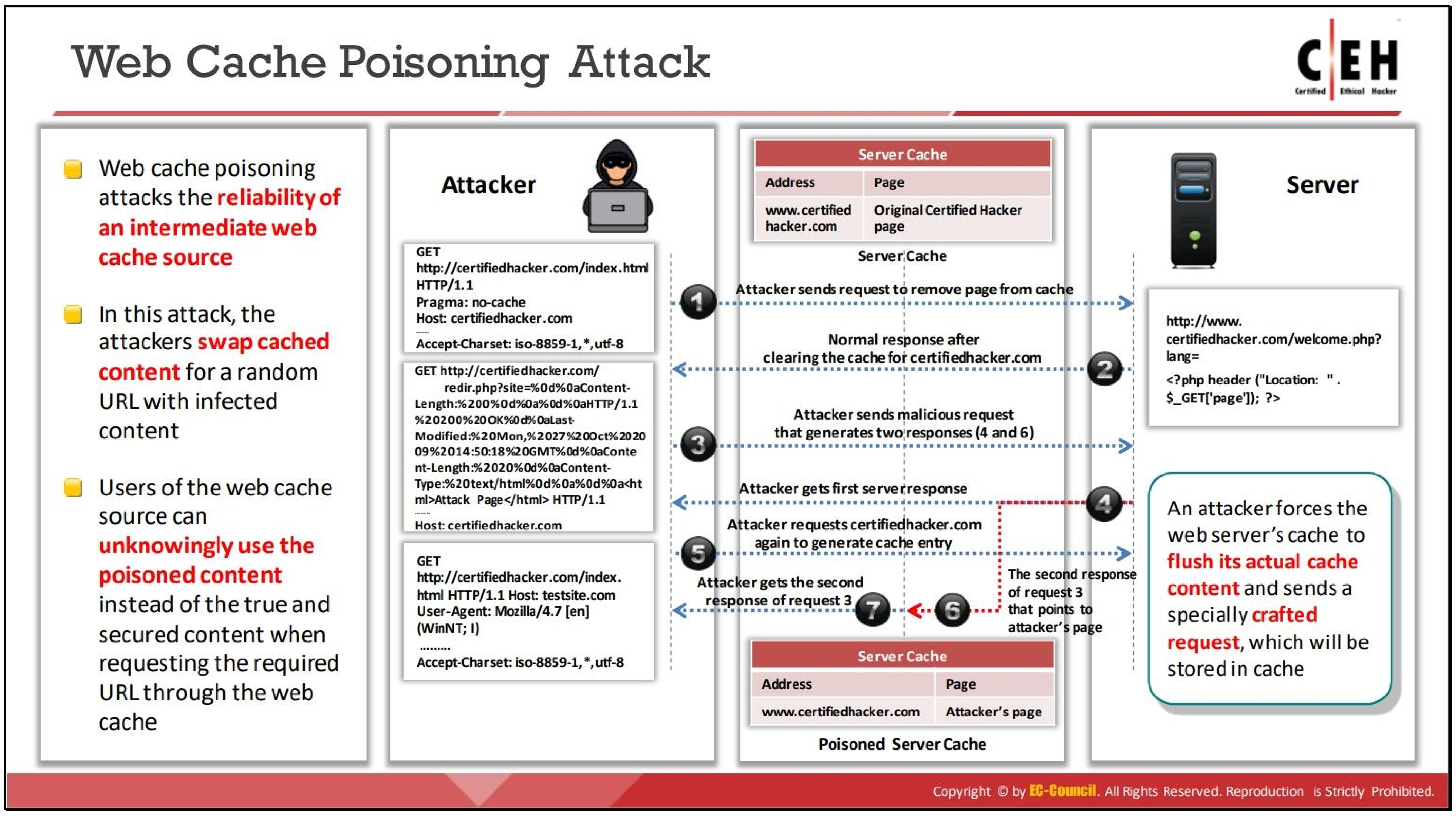


Figure 13.14: Example of an HTTP response-splitting attack



Web Cache Poisoning Attack

Web cache poisoning damages the reliability of an intermediate web cache source. In this attack, an attacker swaps cached content for a random URL with infected content. Users of the web cache source may unknowingly use the poisoned content instead of the true and secured content when requesting the required URL through the web cache.

An attacker forces the web server's cache to flush its actual cache content and sends a specially crafted request to store in the cache. In this case, all the users of that web server cache will receive malicious content until the servers flush the web cache. Web cache poisoning attacks are possible if the web server and application have HTTP response-splitting flaws.

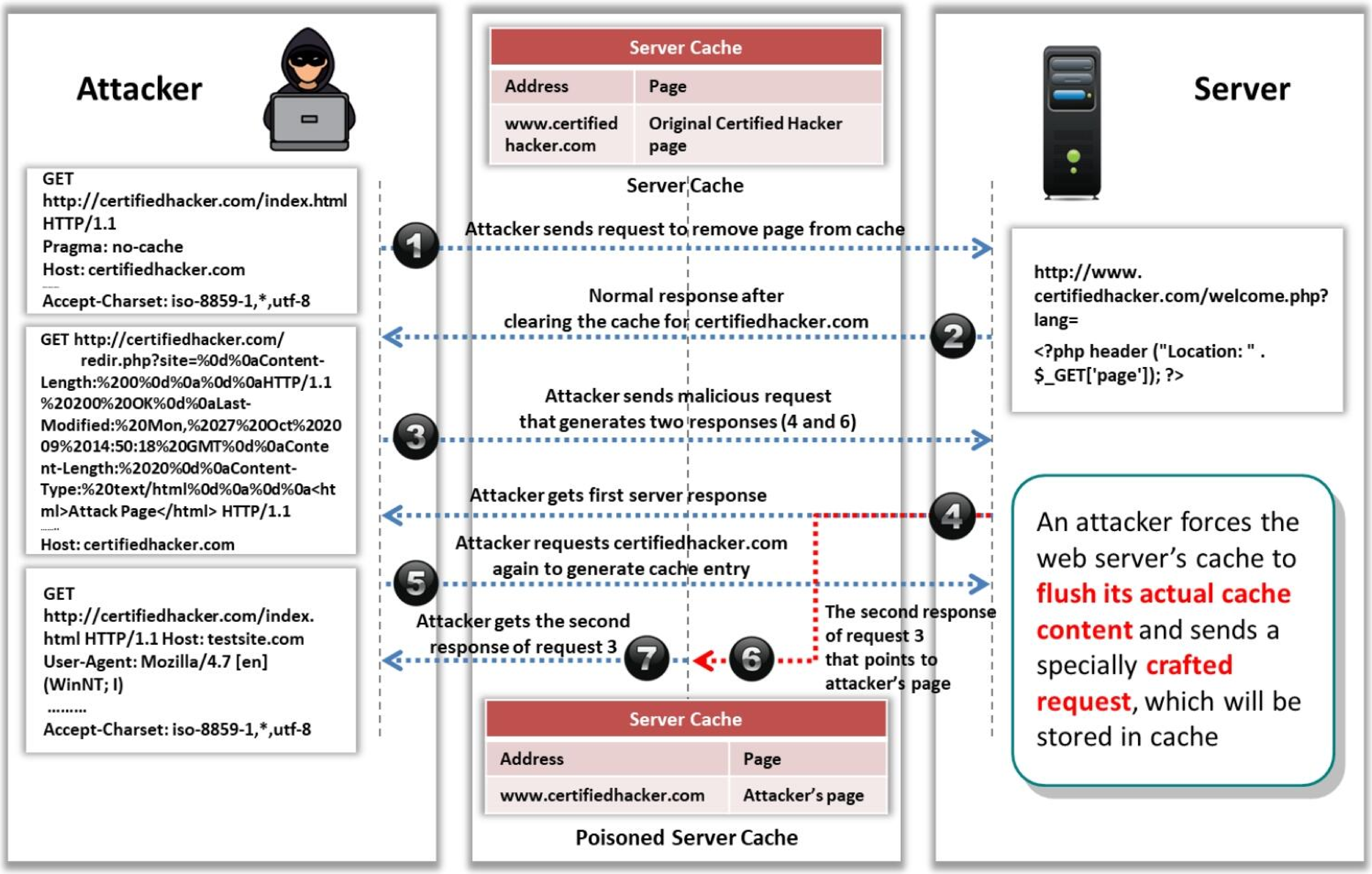
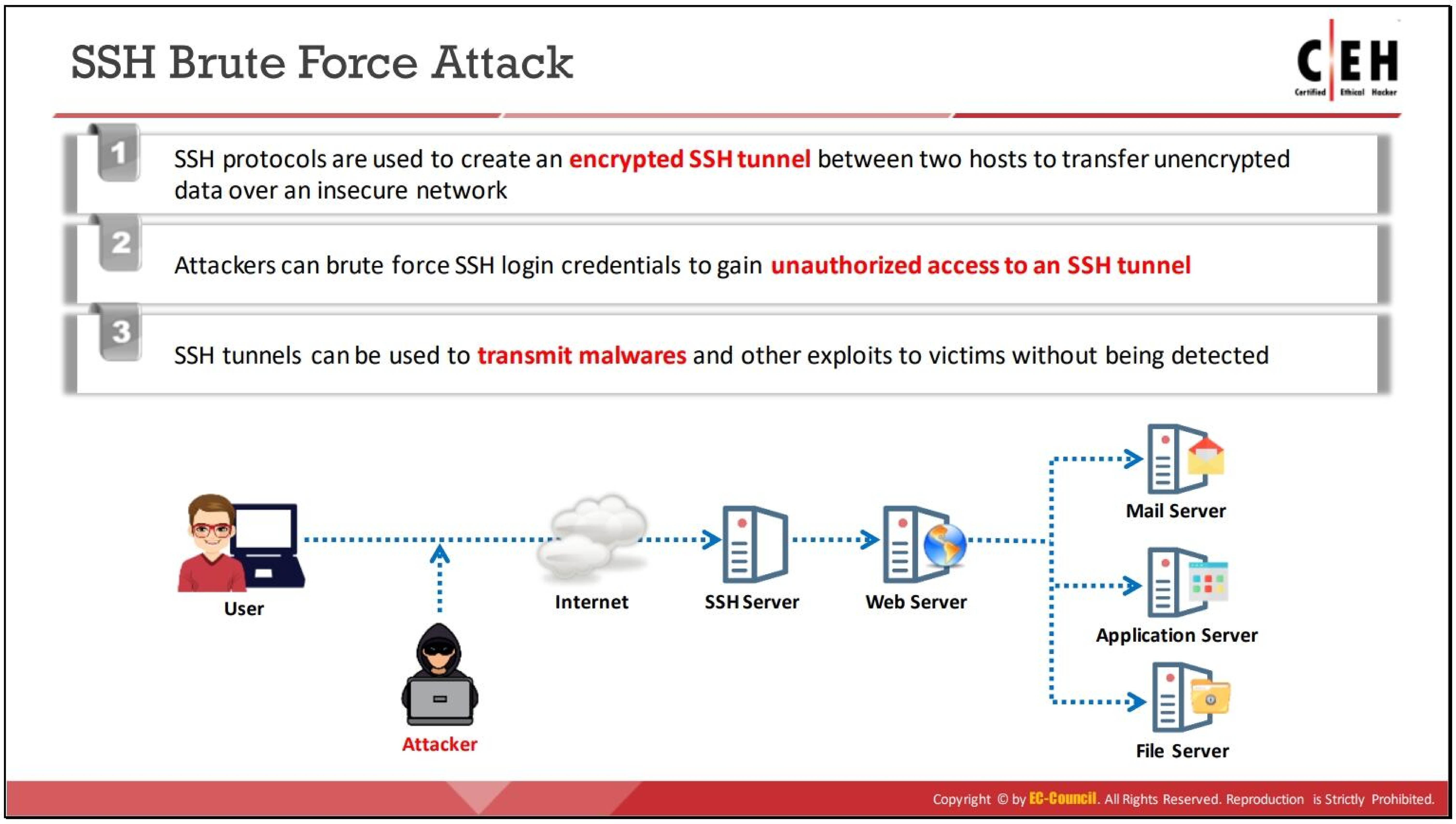


Figure 13.15: Web cache poisoning attack



SSH Brute Force Attack

Attackers use SSH protocols to create an encrypted SSH tunnel between two hosts to transfer unencrypted data over an insecure network. Usually, SSH runs on TCP port 22. To perform an attack on SSH, an attacker scans the entire SSH server using bots (performs a port scan on TCP port 22) to identify possible vulnerabilities. With the help of a brute-force attack, the attacker obtains login credentials to gain unauthorized access to an SSH tunnel. An attacker who obtains the login credentials of SSH can use the same SSH tunnels to transmit malware and other means of exploitation to victims without being detected. Attackers use tools such as Nmap and Ncrack on a Linux platform to perform an SSH brute-force attack.

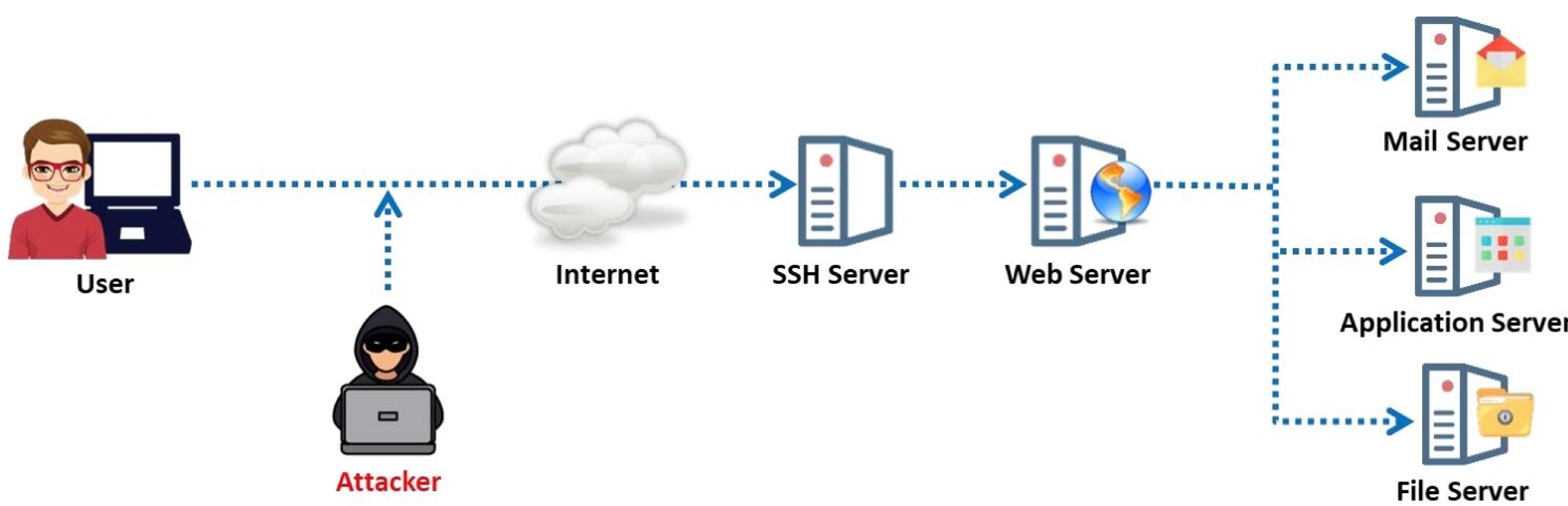


Figure 13.16: SSH Brute Force attack

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 mainly targets:

- **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, and keystroke logging
- Attackers usually begin hacking attempts with **password cracking** to prove to the web server that they are valid users
- Passwords can be cracked **manually** by guessing or by performing dictionary, brute force, and hybrid attacks using **automated tools** such as THC Hydra, and Ncrack

Web Server Password Cracking

An attacker attempts to exploit weaknesses to hack well-chosen passwords. The most common passwords found are password, root, administrator, admin, demo, test, guest, qwerty, pet names, and so on. The attacker mainly targets the following through web server password cracking:

- SMTP and FTP servers
- Web shares
- SSH tunnels
- Web form authentication

Attackers use different methods such as social engineering, spoofing, phishing, a Trojan horse or virus, wiretapping, and keystroke logging to perform web server password cracking. In many hacking attempts, the attacker starts with password cracking to prove to the web server that they are a valid user.

Web Server Password Cracking Techniques

Password cracking is the most common method of gaining unauthorized access to a web server by exploiting flawed and weak authentication mechanisms. Once the password is cracked, an attacker can use the password to launch further attacks.

We present some details of various tools and techniques used by attackers to crack passwords. Attackers can use password cracking techniques to extract passwords from web servers, FTP servers, SMTP servers, and so on. They can crack passwords either manually or with automated tools such as THC Hydra, Ncrack, and RainbowCrack.

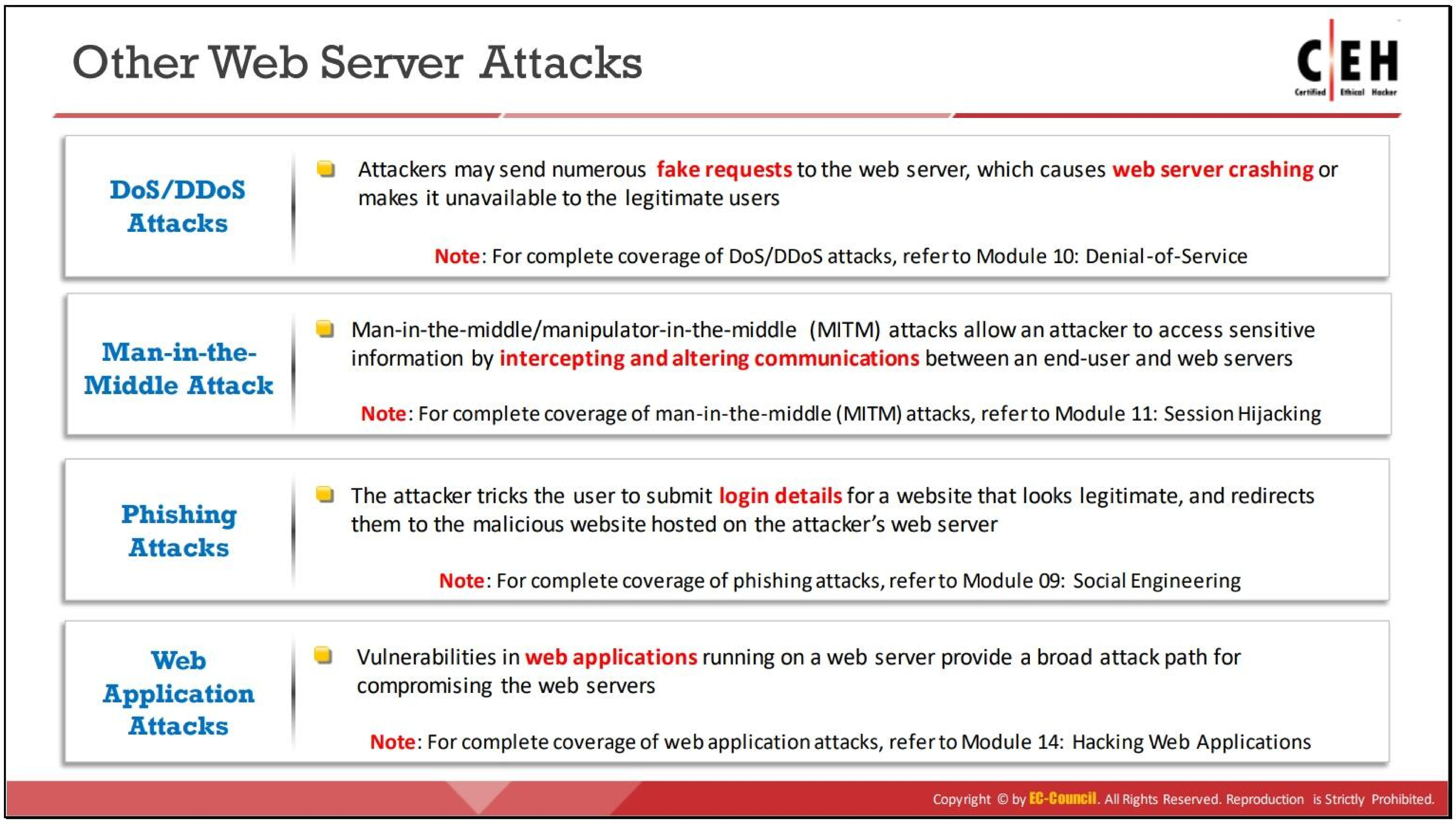
The following are some techniques attackers use to crack passwords:

- **Guessing**: This is the most common method of cracking passwords. In this method, the attacker guesses possible passwords either manually or by using automated tools provided with dictionaries. Most people tend to use their pets' names, loved ones' names, license plate numbers, dates of birth, or other weak passwords such as "QWERTY," "password," "admin," etc. so that they can remember them easily. The attacker exploits this human behavior to crack passwords.

- **Dictionary attack**: A dictionary attack uses a predefined file containing various combinations of words, and an automated program enters these words one at a time to check if any of them are the password. This might not be effective if the password includes special characters and symbols. If the password is a simple word, then it can be found quickly. Compared to a brute-force attack, a dictionary attack is less time-consuming.

- **Brute-force attack**: In the brute-force method, all possible character combinations are tested; for example, the test may include combinations of uppercase characters from A to Z, numbers from 0 to 9, and lowercase characters from a to z. This method is useful for identifying one-word or two-word passwords. If a password consists of uppercase and lowercase letters as well as special characters, it might take months or years to crack the password using a brute-force attack.

- **Hybrid attack**: A hybrid attack is more powerful than the above techniques because it uses both a dictionary attack and brute-force attack. It also uses symbols and numbers. Password cracking is easier with this method than with the above methods.



Other Web Server Attacks

DoS/DDoS Attacks

A 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. A web server DoS/DDoS attack often targets high-profile web servers such as bank servers, credit-card payment gateways, and even root name servers.

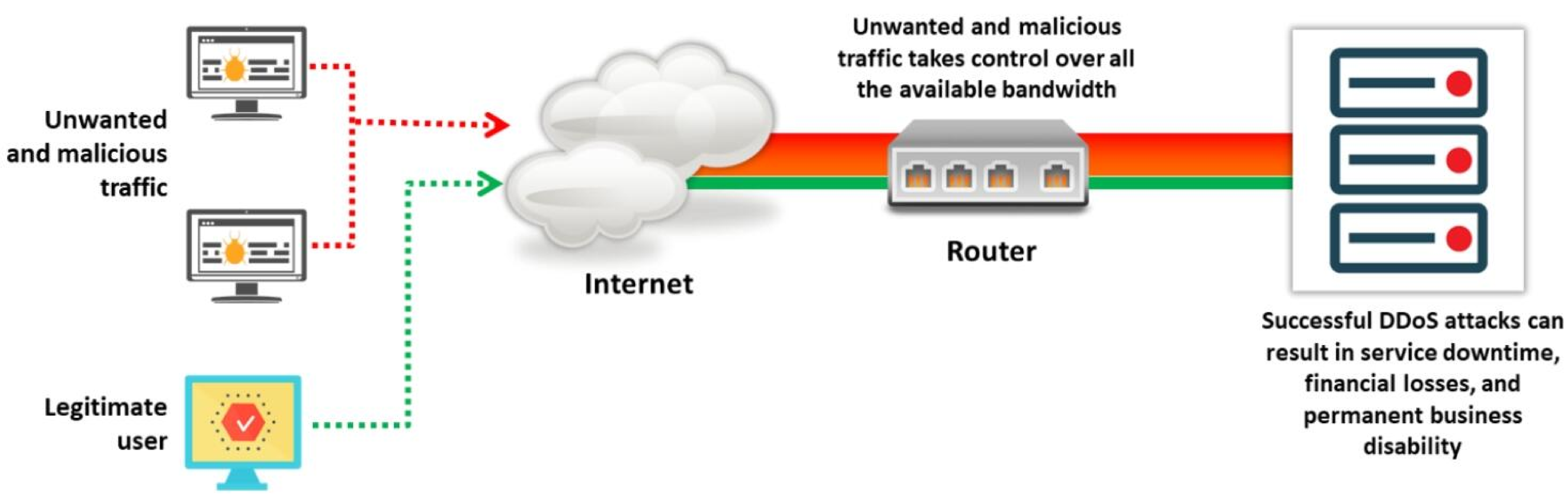


Figure 13.17: Web server DDoS attack

To crash a web server running an application, the attacker targets the following services to consume the web server's resources with fake requests:

- Network bandwidth
- Server memory
- Application exception handling mechanism

- CPU usage
- Hard-disk space
- Database space

Note: For complete coverage of DoS/DDoS attacks, refer to Module 10: Denial-of-Service.

Man-in-the-Middle Attack

Man-in-the-middle/manipulator-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, transferred over the Internet to the web server. The attacker lures the victim to connect to the web server by pretending to be a proxy. If the victim believes and accepts the attacker's request, then all the communication between the user and web server passes through the attacker. In this manner, the attacker can steal sensitive user information.

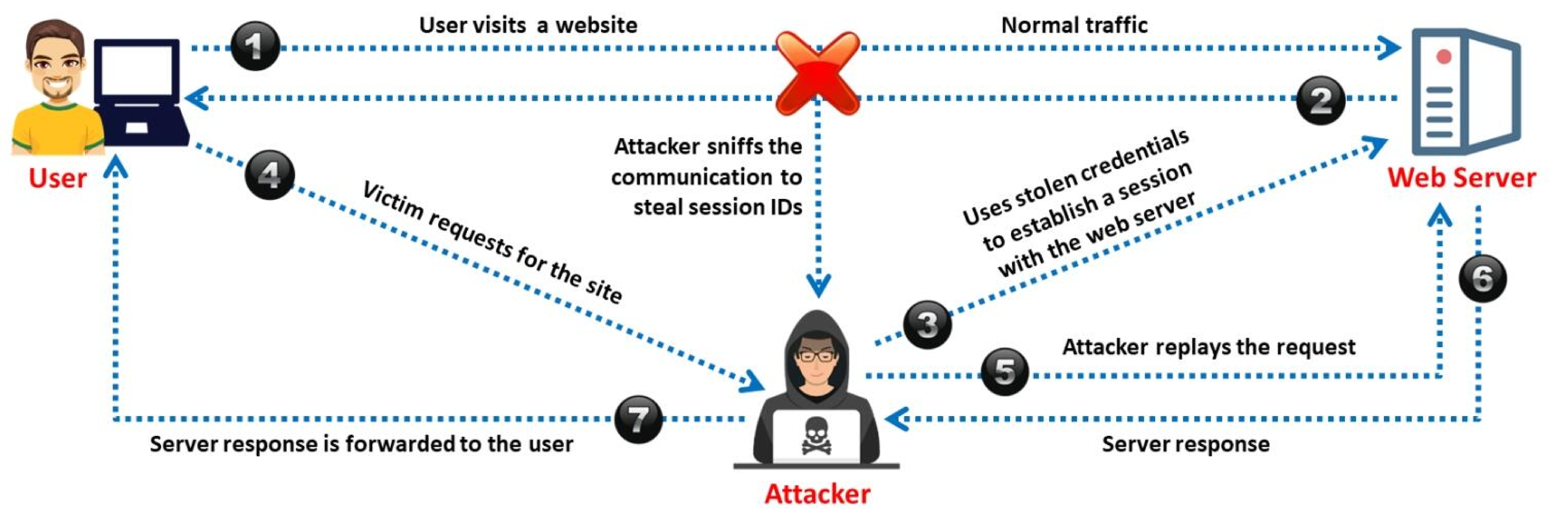


Figure 13.18: Man-in-the-middle/sniffing attack

Note: For complete coverage of man-in-the-middle (MITM) attacks, refer to Module 11: Session Hijacking.

Phishing Attacks

Attackers perform a phishing attack by sending an email containing a malicious link and tricking the user into clicking it. Clicking the link will redirect the user to a fake website that appears similar to the legitimate website. Attackers create such websites by hosting their address on web servers. When a victim clicks on the malicious link while believing the link to be a legitimate website address, the victim is redirected to the malicious website hosted on the attacker's server. The website prompts the user to enter sensitive information, such as usernames, passwords, bank account details, and social security numbers, and divulges the data to the attacker. Later, the attacker may be able to establish a session with the legitimate website by using the victim's stolen credentials to perform malicious operations on the target legitimate website.

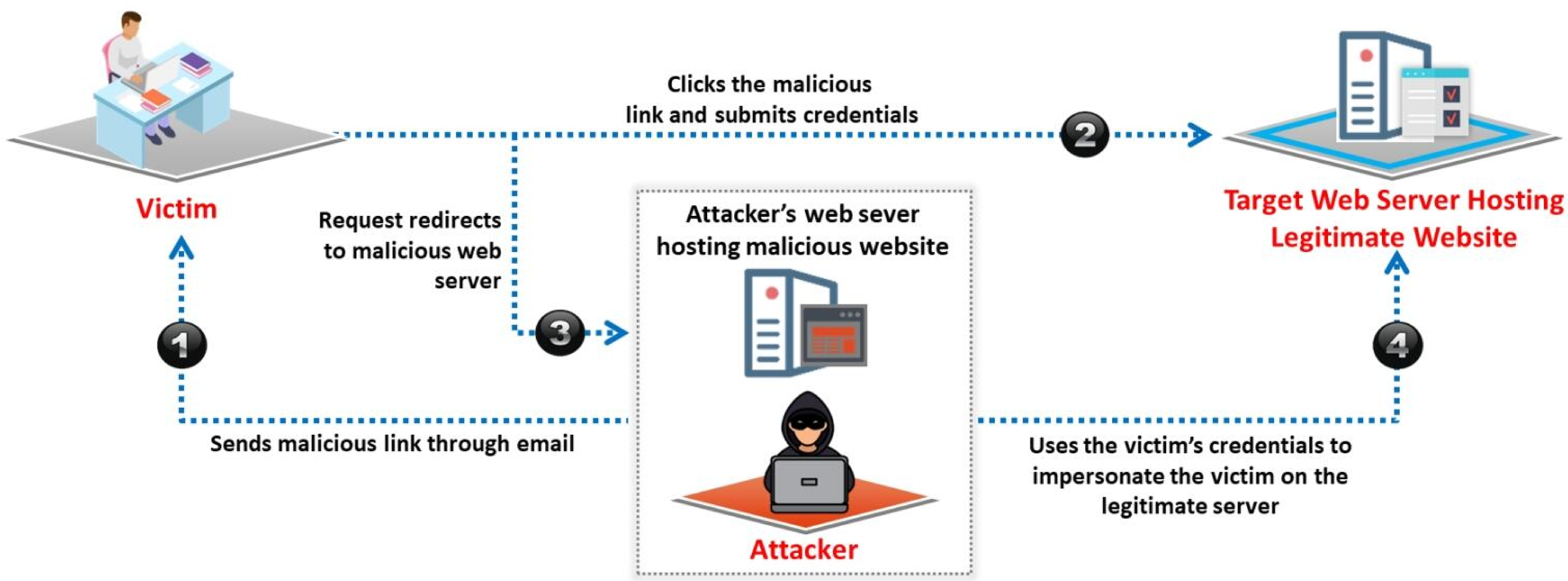


Figure 13.19: Phishing attacks

Note: For complete coverage of phishing attacks, refer to Module 09: Social Engineering.

Web Application Attacks

Even if web servers are configured securely or are secured using network security measures such as firewalls, a poorly coded web application deployed on the web server may provide a path for an attacker to compromise the web server's security. If web developers do not adopt secure coding practices while developing web applications, attackers may be able to exploit vulnerabilities and compromise web applications and web server security. An attacker can perform different types of attacks on vulnerable web applications to breach web server security.

- **Server-Side Request Forgery (SSRF) Attack:** Attackers exploit server-side request forgery (SSRF) vulnerabilities, which evolve from the unsafe use of functions in an application, in public web servers to send crafted requests to the internal or backend servers. The backend server believes that the request is made by the web server because they are on the same network and responds with the data stored in it.

- **Parameter/Form Tampering**: In this type of tampering attack, the attacker manipulates the parameters exchanged between the client and server to modify application data, such as user credentials and permissions as well as price and quantity of products.

- **Cookie Tampering**: Cookie-tampering attacks occur when a cookie is sent from the client side to the server. Different types of tools help in modifying persistent and non-persistent cookies.

- **Unvalidated Input and File Injection Attacks**: Unvalidated input and file-injection attacks are performed by supplying an unvalidated input or by injecting files into a web application.

- **Session Hijacking**: Session hijacking is an attack in which the attacker exploits, steals, predicts, and negotiates the real valid web session's control mechanism to access the authenticated parts of a web application.

- **SQL Injection Attacks**: SQL injection exploits the security vulnerability of a database for attacks. The attacker injects malicious code into the strings, which are later passed on to the SQL server for execution.
- **Directory Traversal**: Directory traversal is the exploitation of HTTP through which attackers can access restricted directories and execute commands outside of the web server's root directory by manipulating a URL.
- **Denial-of-Service (DoS) Attack**: A DoS attack is intended to terminate the operations of a website or server to make it unavailable for access by its intended users.
- **Cross-Site Scripting (XSS) Attacks**: In this method, an attacker injects HTML tags or scripts into a target website.
- **Buffer Overflow Attacks**: The design of most web applications helps them in sustaining some amount of data. If that amount exceeds the storage space available, the application may crash or exhibit some other vulnerable behavior. An attacker uses this advantage and floods the application with an excess amount of data, causing a buffer overflow attack.
- **Cross-Site Request Forgery (CSRF) Attack**: An attacker exploits the trust of an authenticated user to pass malicious code or commands to the web server.
- **Command Injection Attacks**: In this type of attack, a hacker alters the content of the web page by using HTML code and by identifying the form fields that lack valid constraints.
- **Source Code Disclosure**: Source-code disclosure is a result of typographical errors in scripts or misconfiguration, such as failure to grant executable permissions to a script or directory. Source-code disclosure can occasionally allow attackers to access sensitive information about database credentials and secret keys to compromise the web server.

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

LO#03: Explain Web Server Attack Methodology

Web Server Attack Methodology

1. **Information Gathering**
2. **Web Server Footprinting**
3. **Website Mirroring**
4. **Vulnerability Scanning**
5. **Session Hijacking**
6. **Web Server Passwords Hacking**

Web Server Attack Methodology

The previous section described attacks that can be performed to compromise a web server's security. This section explains how the attacker proceeds toward performing a successful attack on a web server. It also introduces web server hacking tools that attackers may use. These tools extract critical information during the hacking process.

A web server attack typically involves preplanned activities called an attack methodology that an attacker follows to reach the goal of breaching the target web server's security. Attackers hack a web server in multiple stages. At each stage, the attacker attempts to gather information about loopholes and to gain unauthorized access to the web server.

The following are the various stages of the attack methodology for web servers.

- **Information Gathering**

  Every attacker tries to collect as much information as possible about the target web server. The attacker gathers the information and then analyzes it to find lapses in the current security mechanisms of the web server.

- **Web Server Footprinting**

  The purpose of footprinting is to gather information about the security aspects of a web server with the help of tools or footprinting techniques. Through footprinting, attackers can determine the web server's remote access capabilities, its ports and services, and other aspects of its security.

- **Website Mirroring**

  Website mirroring is a method of copying a website and its content onto another server for offline browsing. With a mirrored website, an attacker can view the detailed structure of the website.

- **Vulnerability Scanning**

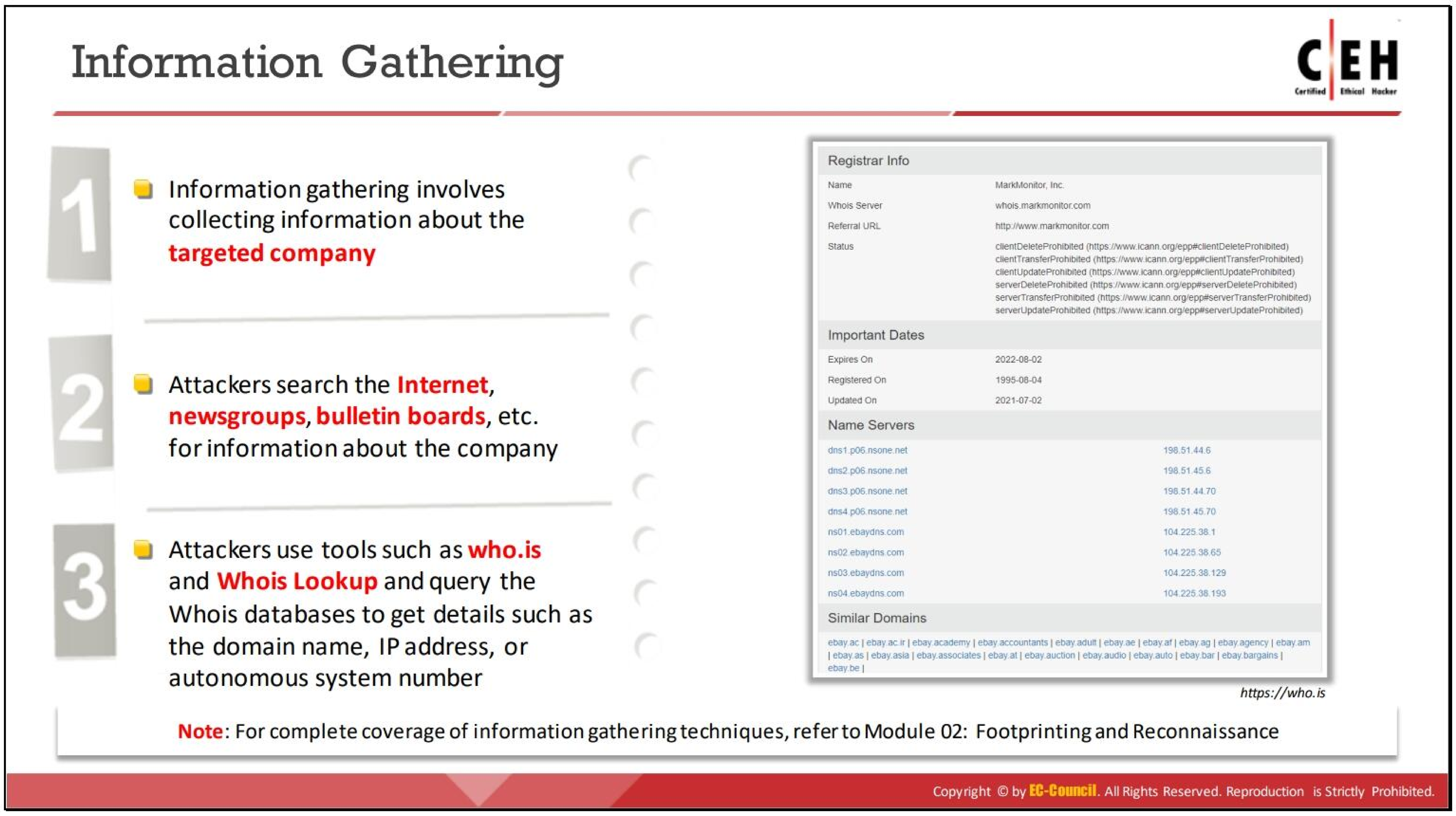
  Vulnerability scanning is a method of finding the vulnerabilities and misconfigurations of a web server. Attackers scan for vulnerabilities with the help of automated tools known as vulnerability scanners.

- **Session Hijacking**

  Attackers can perform session hijacking after identifying the current session of the client. The attacker takes complete control over the user session through session hijacking.

- **Web Server Passwords Hacking**

  Attackers use password-cracking methods such as brute-force attacks, hybrid attacks, and dictionary attacks to crack the web server's password.



Information Gathering

Information gathering is the first and one of the most important steps toward hacking a target web server. In this step, an attacker collects as much information as possible about the target server by using various tools and techniques. The information obtained from this step helps the attacker in assessing the security posture of the web server. Attackers may search the Internet, newsgroups, bulletin boards, and so on for gathering information about the target organization. Attackers can use tools such as who.is and Whois Lookup to extract information such as the target's domain name, IP address, and autonomous system number.

- **who.is**

  Source: *https://who.is*

  who.is is designed perform a variety of whois lookup functions. It lets the user perform a domain whois search, whois IP lookup, and whois database search for relevant information on domain registration and availability.

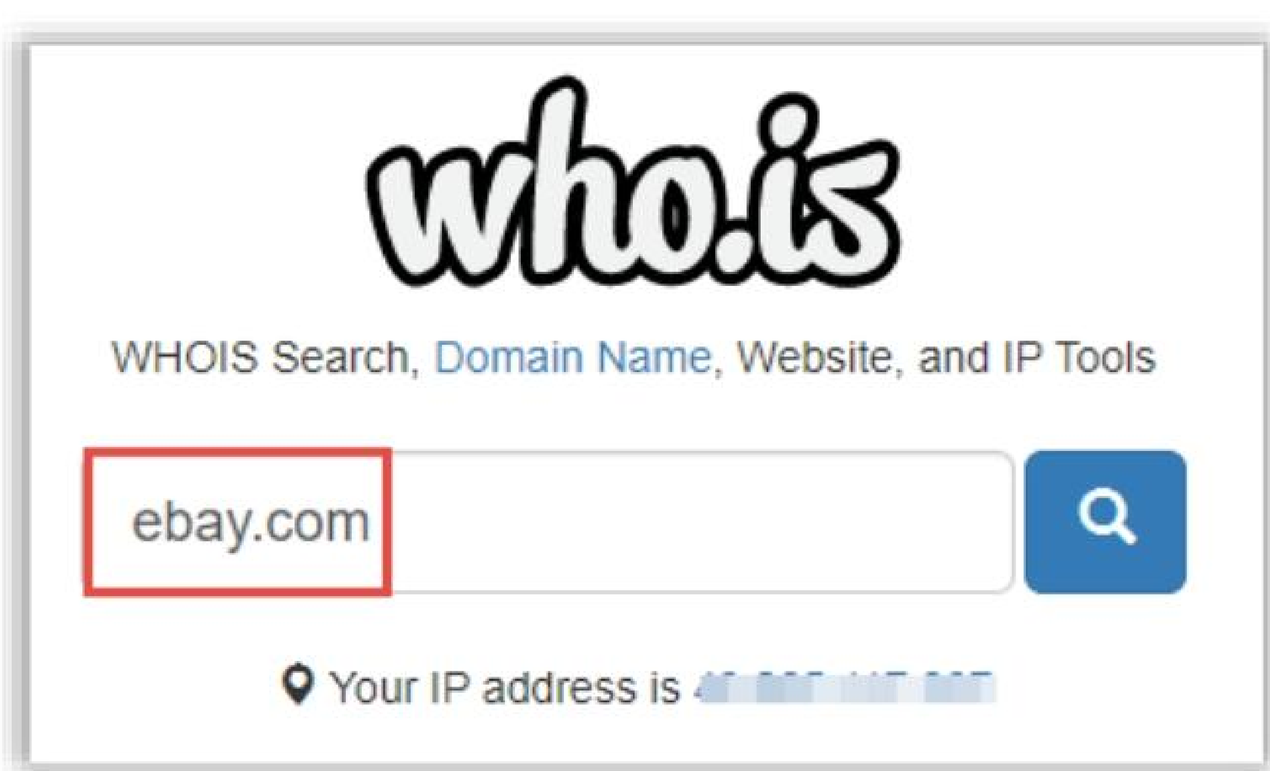


Figure 13.20: Screenshot of who.is

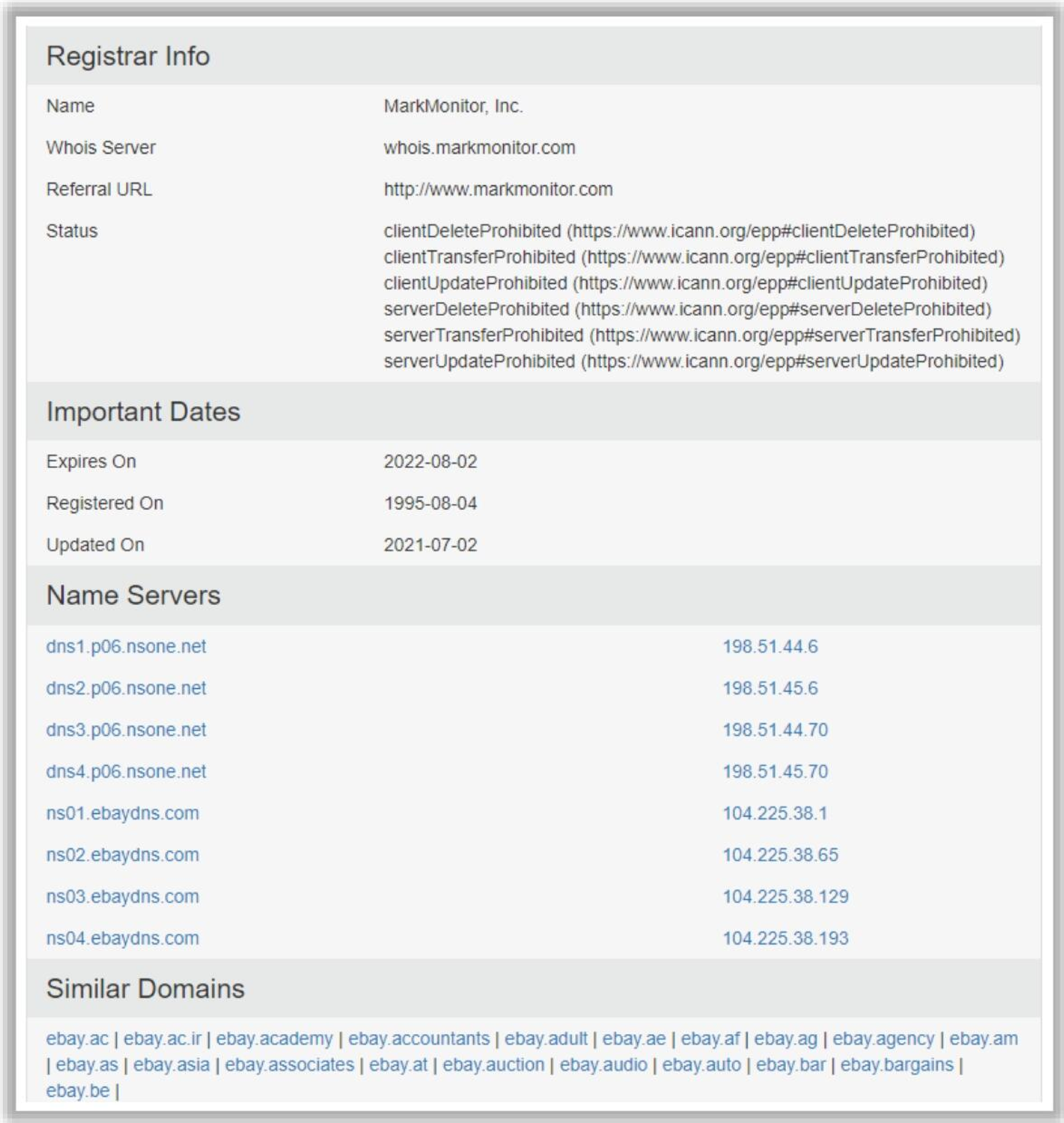


Figure 13.21: Screenshot displaying a who.is online search result

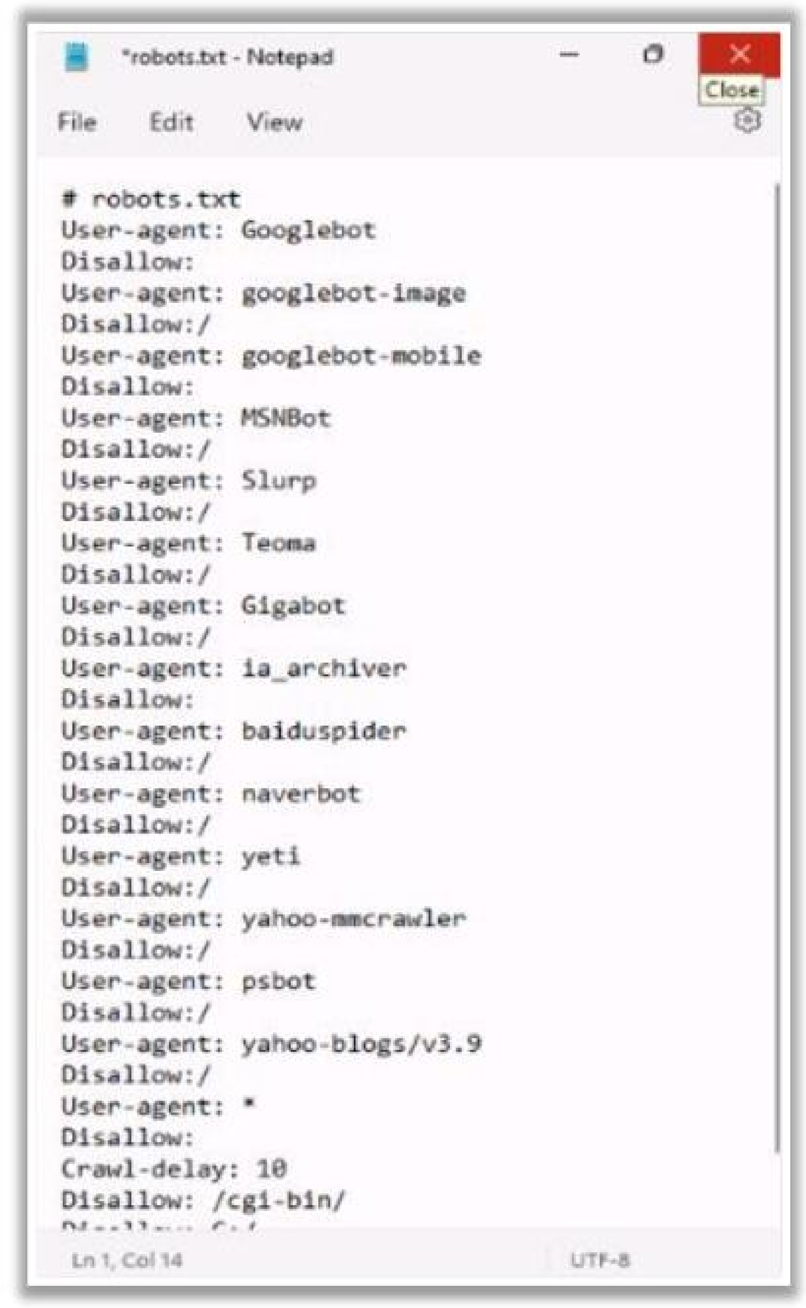
The following are some additional information-gathering tools:

- Whois Lookup (*https://whois.domaintools.com*)
- Whois (*https://www.whois.com*)
- Domain Dossier (*https://centralops.net*)
- Find Subdomains (*https://pentest-tools.com*)
- SmartWhois (*https://www.tamos.com*)

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 the Robots.txt file from the URL and retrieve sensitive information such as the **root directory structure** and **content management system information** about the target website
- An attacker can also download the Robots.txt file of a target website using the Wget tool

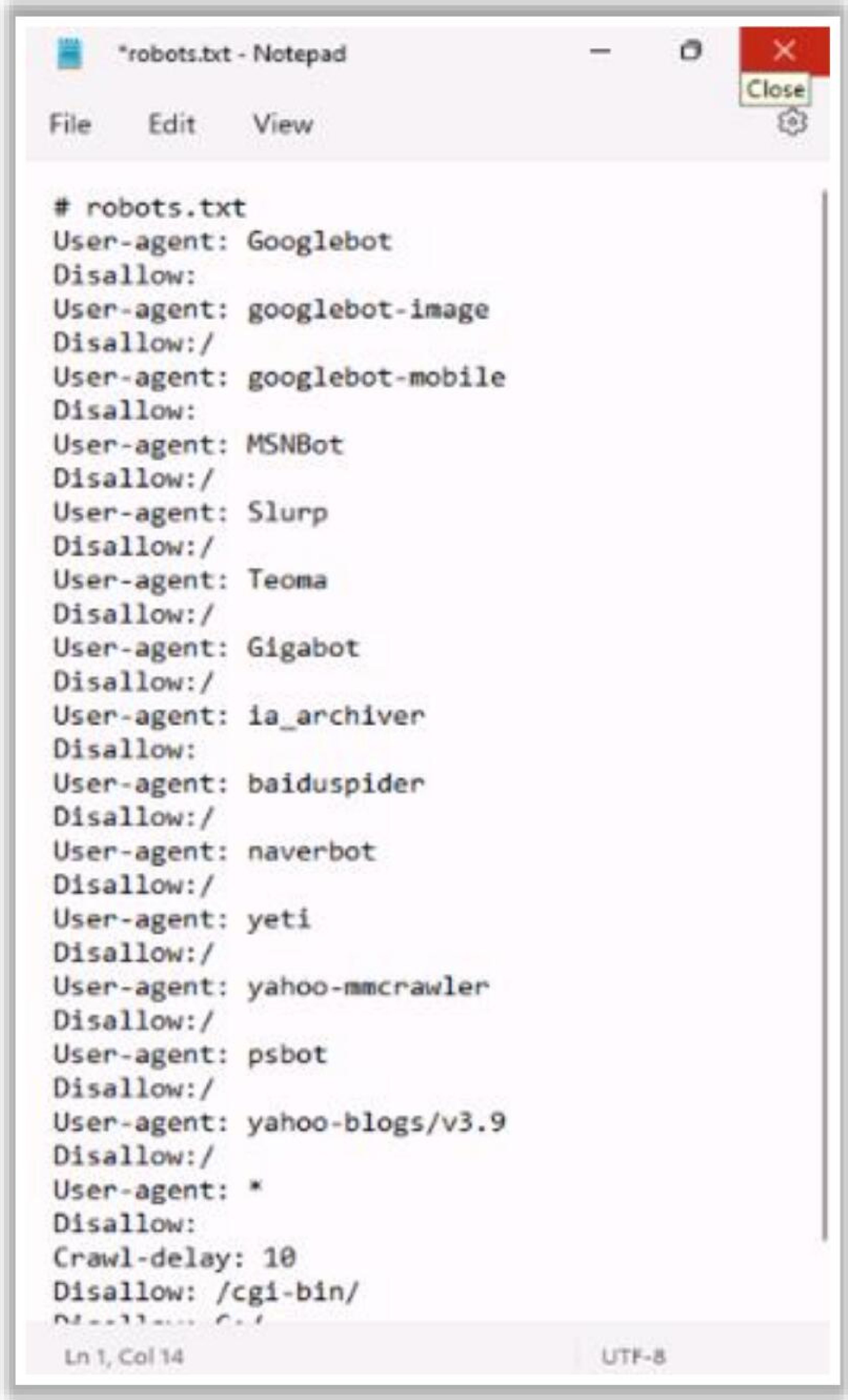


Information Gathering from Robots.txt File

A website owner creates a robots.txt file to list the files or directories a web crawler should index for providing search results. Poorly written robots.txt files can cause the complete indexing of website files and directories. If confidential files and directories are indexed, an attacker may easily obtain information such as passwords, email addresses, hidden links, and membership areas.

If the owner of the target website writes the robots.txt file without allowing the indexing of restricted pages for providing search results, an attacker can still view the robots.txt file of the site to discover restricted files and then view them to gather information.

An attacker types URL/robots.txt in the address bar of a browser to view the target website's robots.txt file. An attacker can also download the robots.txt file of a target website using the Wget tool.



*robots.txt - Notepad

File Edit View

```
# robots.txt
User-agent: Googlebot
Disallow:
User-agent: googlebot-image
Disallow:/
User-agent: googlebot-mobile
Disallow:
User-agent: MSNBot
Disallow:/
User-agent: Slurp
Disallow:/
User-agent: Teoma
Disallow:/
User-agent: Gigabot
Disallow:/
User-agent: ia_archiver
Disallow:
User-agent: baiduspider
Disallow:/
User-agent: naverbot
Disallow:/
User-agent: yeti
Disallow:/
User-agent: yahoo-mmcrawler
Disallow:/
User-agent: psbot
Disallow:/
User-agent: yahoo-blogs/v3.9
Disallow:/
User-agent: *
Disallow:
Crawl-delay: 10
Disallow: /cgi-bin/
```

Ln 1, Col 14 UTF-8

Figure 13.22: Screenshot displaying a robots.txt file

Web Server Footprinting/Banner Grabbing

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

Netcat

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

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

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

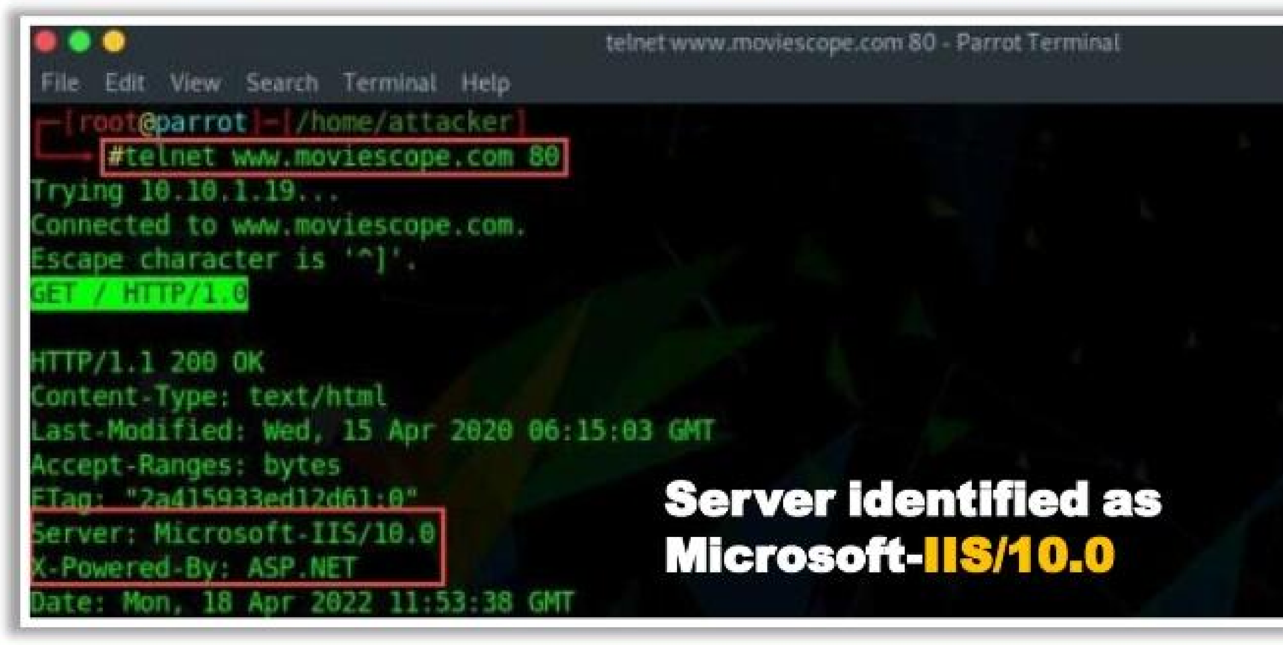
Telnet

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

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

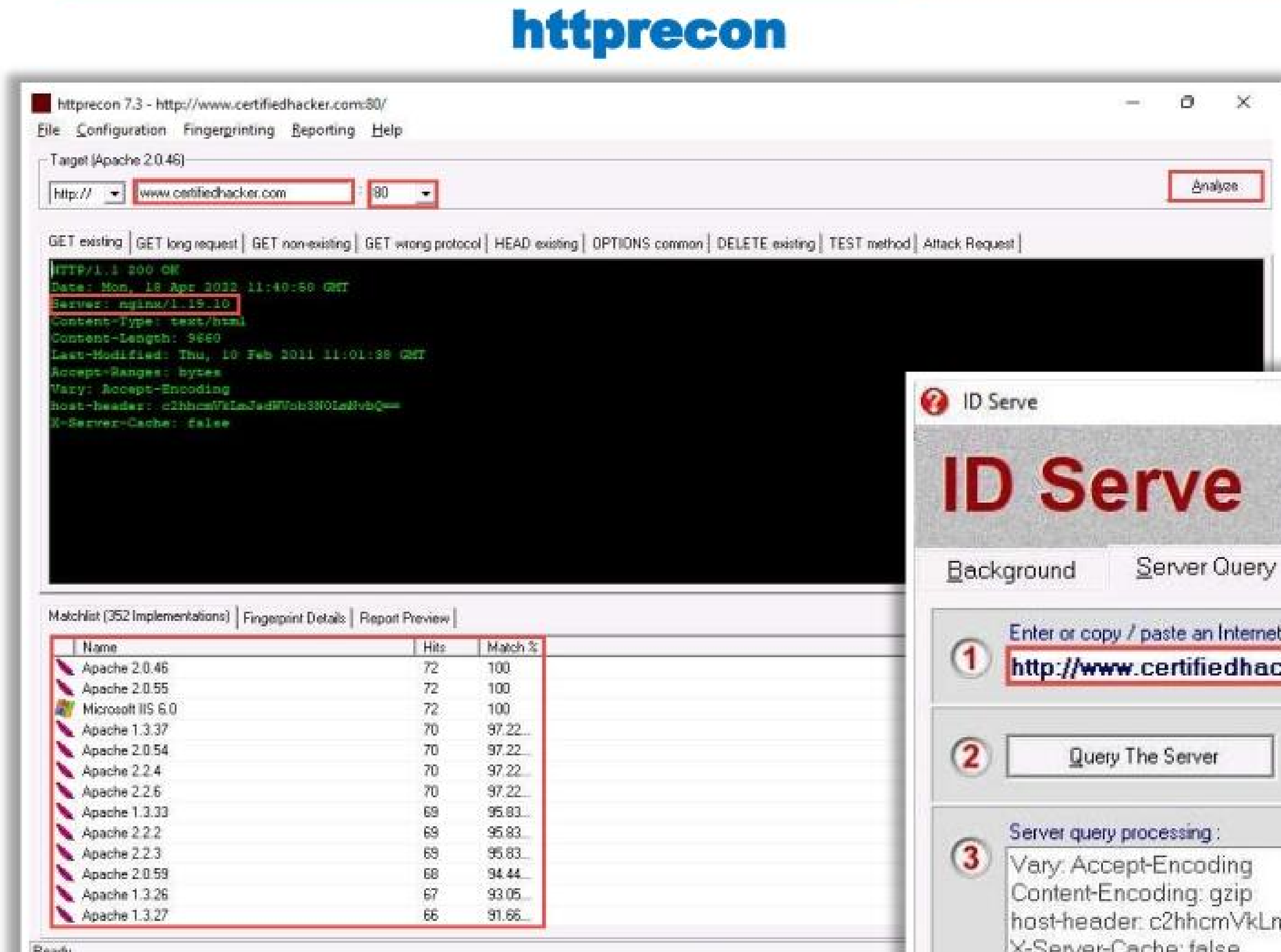
`GET / HTTP/1.0` - Press **[Enter] twice**

http://netcat.sourceforge.net



Web Server Footprinting Tools

httprecon



https://www.computec.ch

ID Serve

https://www.grc.com

- **Netcraft** https://www.netcraft.com
- **Uniscan** https://sourceforge.net
- **Nmap** https://nmap.org
- **Ghost Eye** https://github.com
- **Skipfish** https://code.google.com

Web Server Footprinting/Banner Grabbing

By performing web server footprinting, an attacker can gather valuable system-level data such as account details, OSs, software versions, server names, and database schema details. The Telnet utility can be used to footprint a web server and gather information such as server name, server type, OSs, and running applications running. Furthermore, footprinting tools such as ID Serve, httprecon, and Netcraft can be used to perform web server footprinting. These footprinting tools can extract information from the target server. Here, we examine the features and types of information these tools can collect from the target server.

Web Server Footprinting Tools

- **Netcat**

  Source: *http://netcat.sourceforge.net*

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

  The following are the commands used to perform banner grabbing for www.moviescope.com as an example to gather information such as server type and version.

  - `# nc –vv www.moviescope.com 80` – press [**Enter**]
  - `GET / HTTP/1.0` - press [**Enter**] twice

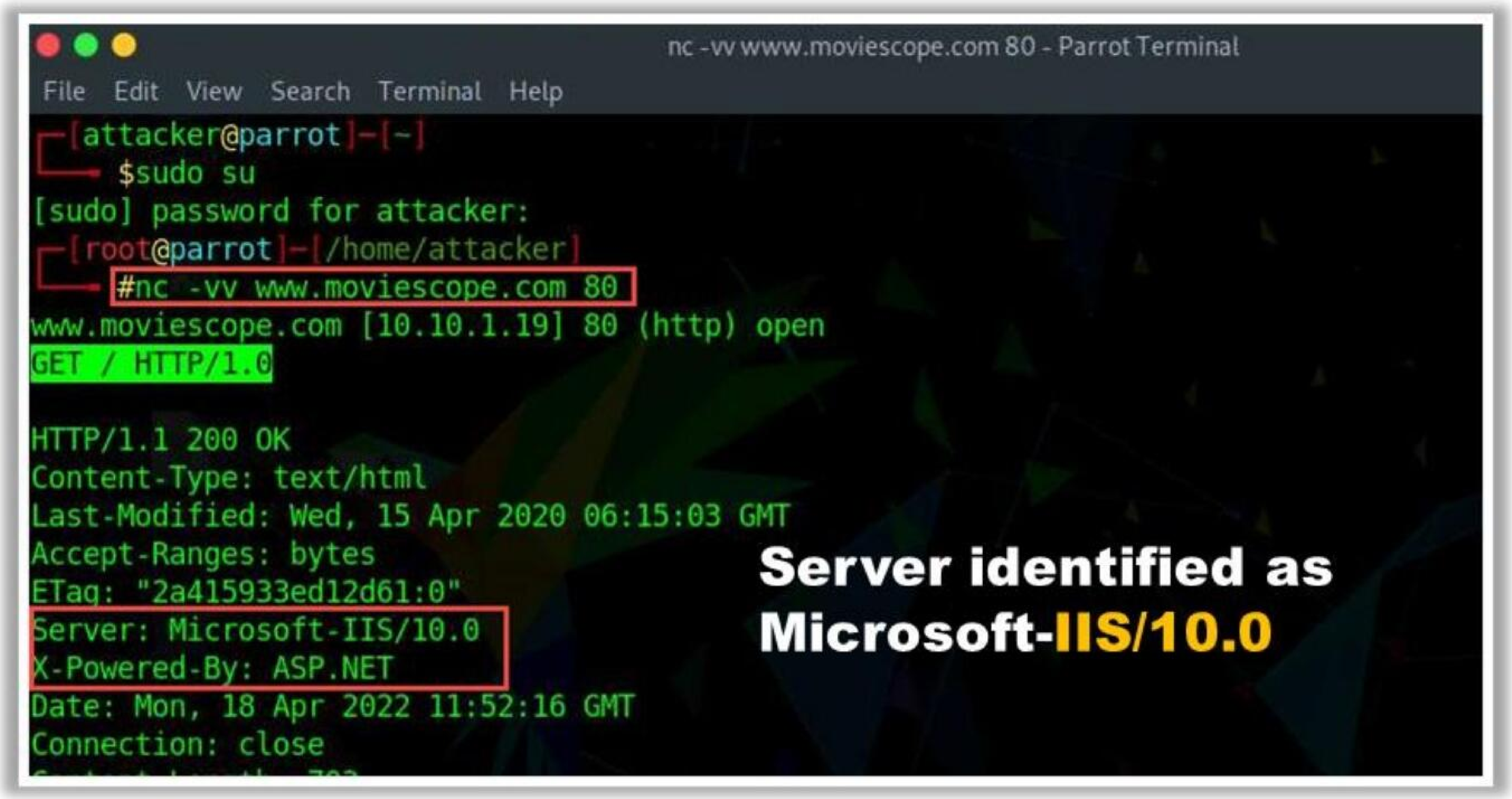


Figure 13.23: Netcat output

- **Telnet**

  Source: *https://docs.microsoft.com*

  Telnet is a client–server network protocol that is widely used on the Internet or LANs. It provides login sessions for a user on the Internet. A single terminal attached to another computer emulates the session by using Telnet. The primary security issues with Telnet are the following.

  - It does not encrypt data sent through the connection.
  - It lacks an authentication scheme.

Telnet enables an attacker to perform a banner-grabbing attack. It probes HTTP servers to determine the server field in the HTTP response header.

For instance, the following procedure is utilized to enumerate a host running on HTTP (TCP 80).

- Request Telnet to connect to a host on a specific port with the command **`# telnet www.moviescope.com 80`** and press **Enter**. A blank screen appears.
- Type **`GET / HTTP/1.0`** and press **Enter** twice.

The HTTP server responds with the information shown in the screenshot.

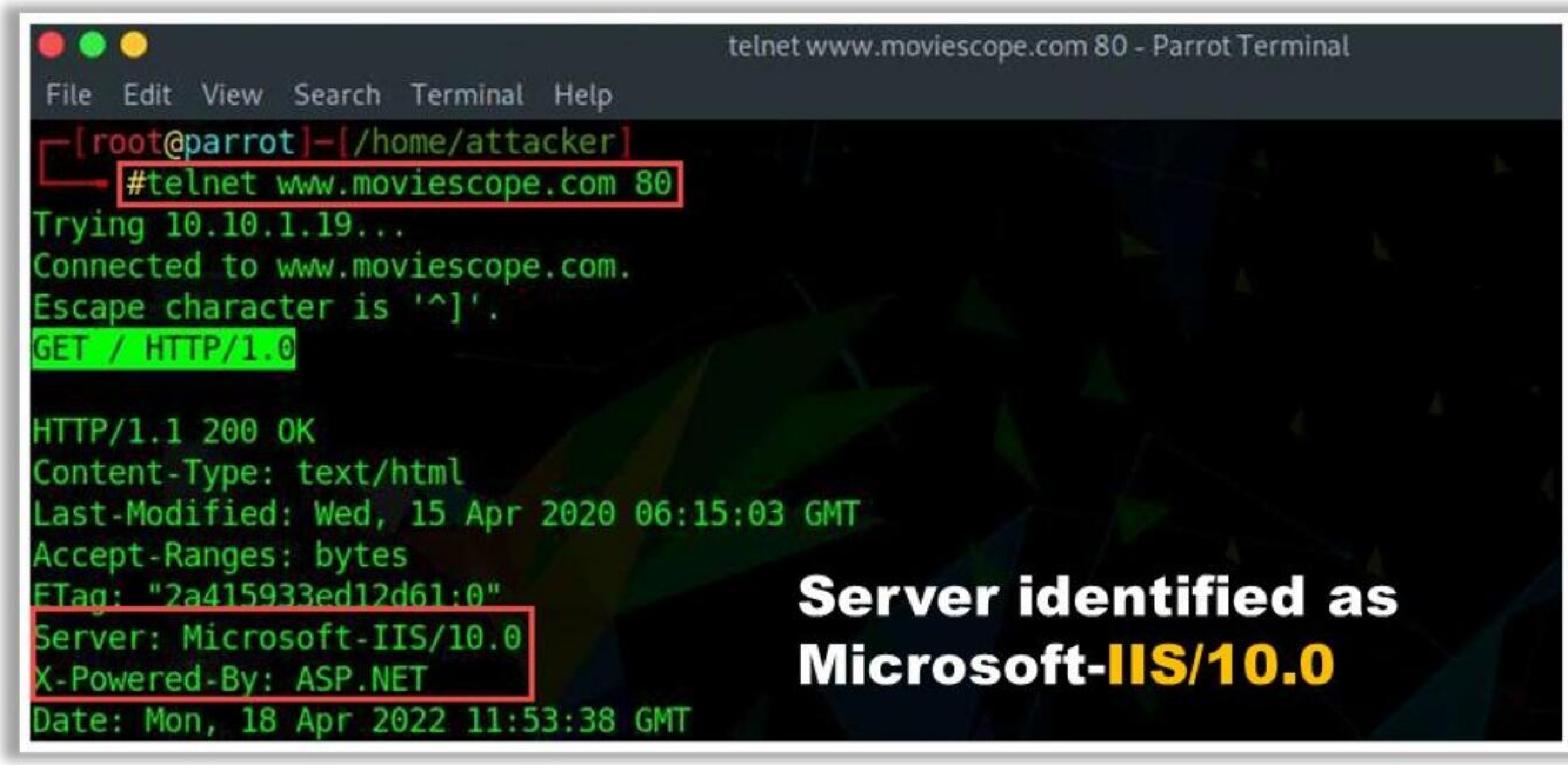


Figure 13.24: Telnet output

- **httprecon**

  Source: *https://www.computec.ch*

  httprecon is a tool for advanced web server fingerprinting. This tool performs banner-grabbing attacks, status code enumeration, and header ordering analysis on the target web server and provides accurate web server fingerprinting information.

  httprecon performs the following header analysis test cases on the target web server:

  - A legitimate GET request for an existing resource
  - An exceedingly long GET request (a Uniform Resource Identifier (URI) of >1024 bytes)
  - A common GET request for a non-existing resource
  - A common HEAD request for an existing resource
  - Enumeration with OPTIONS, which is allowed
  - The HTTP method DELETE, which is usually not permitted
  - The HTTP method TEST, which is not defined

- The protocol version HTTP/9.8, which does not exist
- A GET request including attack patterns (e.g., : ../ and %%)

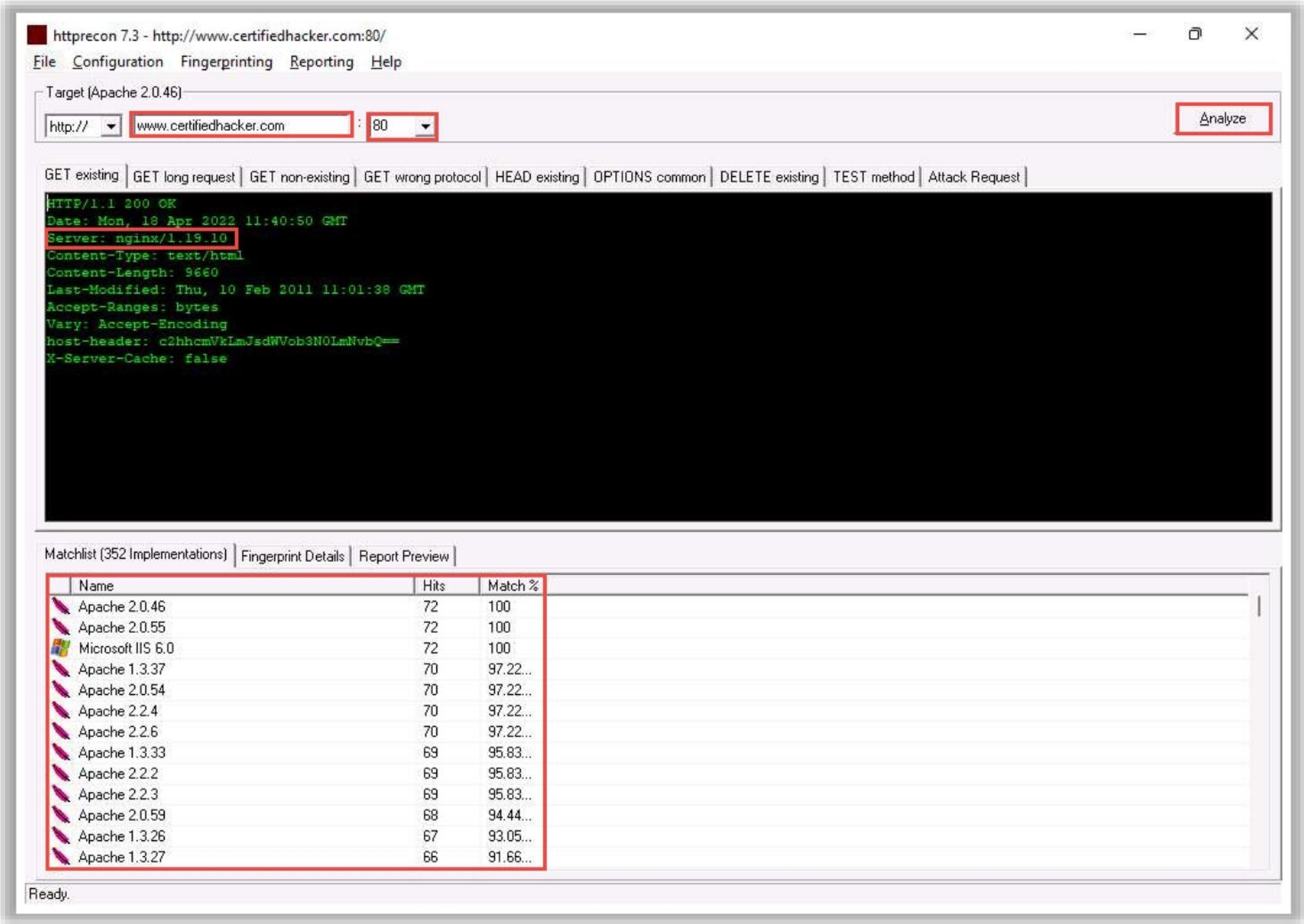


Figure 13.25: Screenshot of httprecon

- **ID Serve**

  Source: *https://www.grc.com*

  ID Serve is a simple Internet server identification utility. The following is a list of its capabilities.

  - **HTTP Server Identification**: ID Serve can identify the make, model, and version of a website's server software. ID Serve sends this information in the preamble of replies to web queries, but the information is not visible to the user.
  - **Non-HTTP Server Identification**: Most non-HTTP (non-web) Internet servers (e.g., FTP, SMTP, Post Office Protocol (POP), and NEWS) are required to transmit a line containing a numeric status code and a human-readable greeting to any connecting client. Therefore, ID Serve can also connect with non-web servers to receive and report the server's greeting message. This generally reveals the server's make, model, version, and other potentially useful information.
  - **Reverse DNS Lookup**: When ID Serve users enter a site's or server's domain name or URL, the application will use a DNS to determine the IP address of that domain.

However, it is occasionally useful to proceed in the other direction to determine the domain name associated with a known IP address. This process, known as reverse DNS lookup, is also built into ID Serve. ID Serve attempts to determine the associated domain name for any entered IP address.

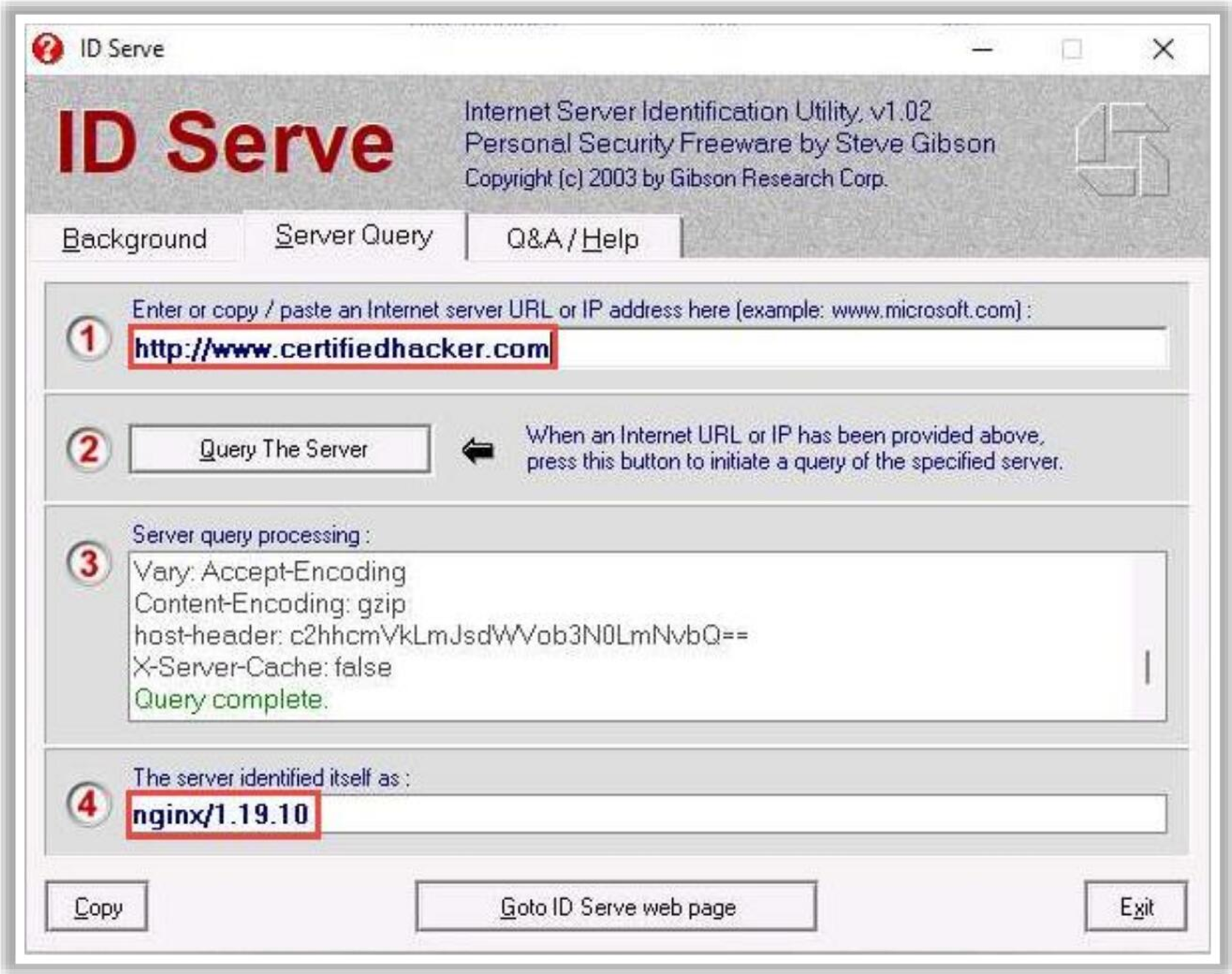
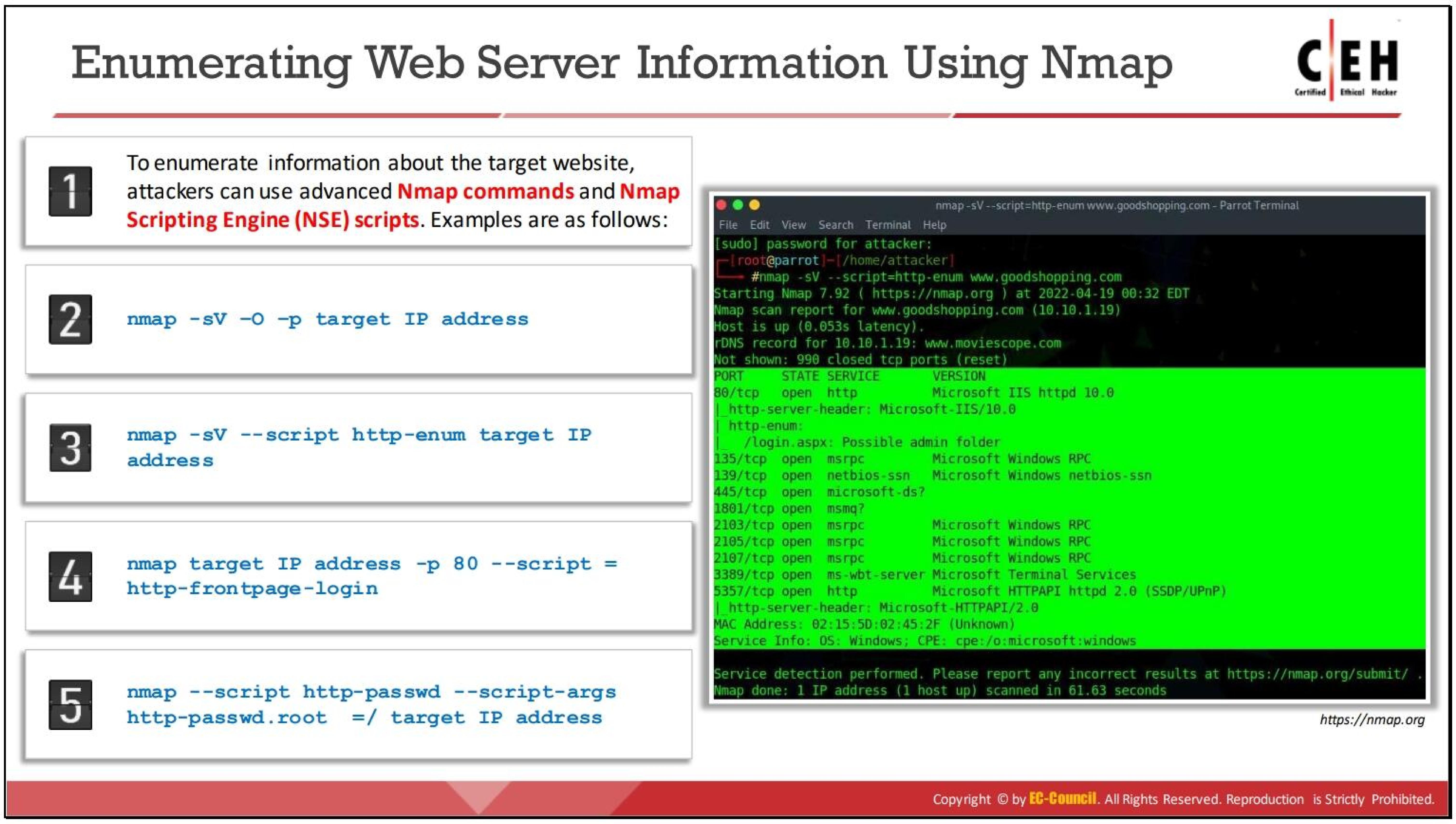


Figure 13.26: Screenshot of ID Serve

The following are some additional footprinting tools:

- Netcraft (*https://www.netcraft.com*)
- Uniscan (*https://sourceforge.net*)
- Nmap (*https://nmap.org*)
- Ghost Eye (*https://github.com*)
- Skipfish (*https://code.google.com*)



Enumerating Web Server Information Using Nmap

Source: *https://nmap.org*

Nmap, along with the Nmap Scripting Engine (NSE), can extract a large amount of valuable information from the target web server. In addition to Nmap commands, NSE provides scripts that reveal various types of useful information about the target server to an attacker.

An attacker uses the following Nmap commands and NSE scripts to extract information.

- Discover virtual domains with hostmap:

  $nmap --script hostmap <host>

- Detect a vulnerable server that uses the TRACE method:

  nmap --script http-trace -p80 localhost

- Harvest email accounts with http-google-email:

  $nmap --script http-google-email <host>

- Enumerate users with http-userdir-enum:

  nmap -p80 --script http-userdir -enum localhost

- Detect HTTP TRACE:

  $nmap -p80 --script http-trace <host>

- Check if the web server is protected by a web application firewall (WAF) or IPS:

  $nmap -p80 --script http-waf-detect --script-args="http-waf-detect.uri=/testphp.vulnweb.com/artists.php,http-waf-detect.detectBodyChanges" www.modsecurity.org

- Enumerate common web applications

  $nmap --script http-enum -p80 <host>

- Obtain robots.txt

  $nmap -p80 --script http-robots.txt <host>

The following are some additional Nmap commands used to extract web server information:

- **nmap -sV –O –p target IP address**
- **nmap -sV --script http-enum target IP address**
- **nmap target IP address -p 80 --script = http-frontpage-login**
- **nmap --script http-passwd --script-args http-passwd.root =/ target IP address**

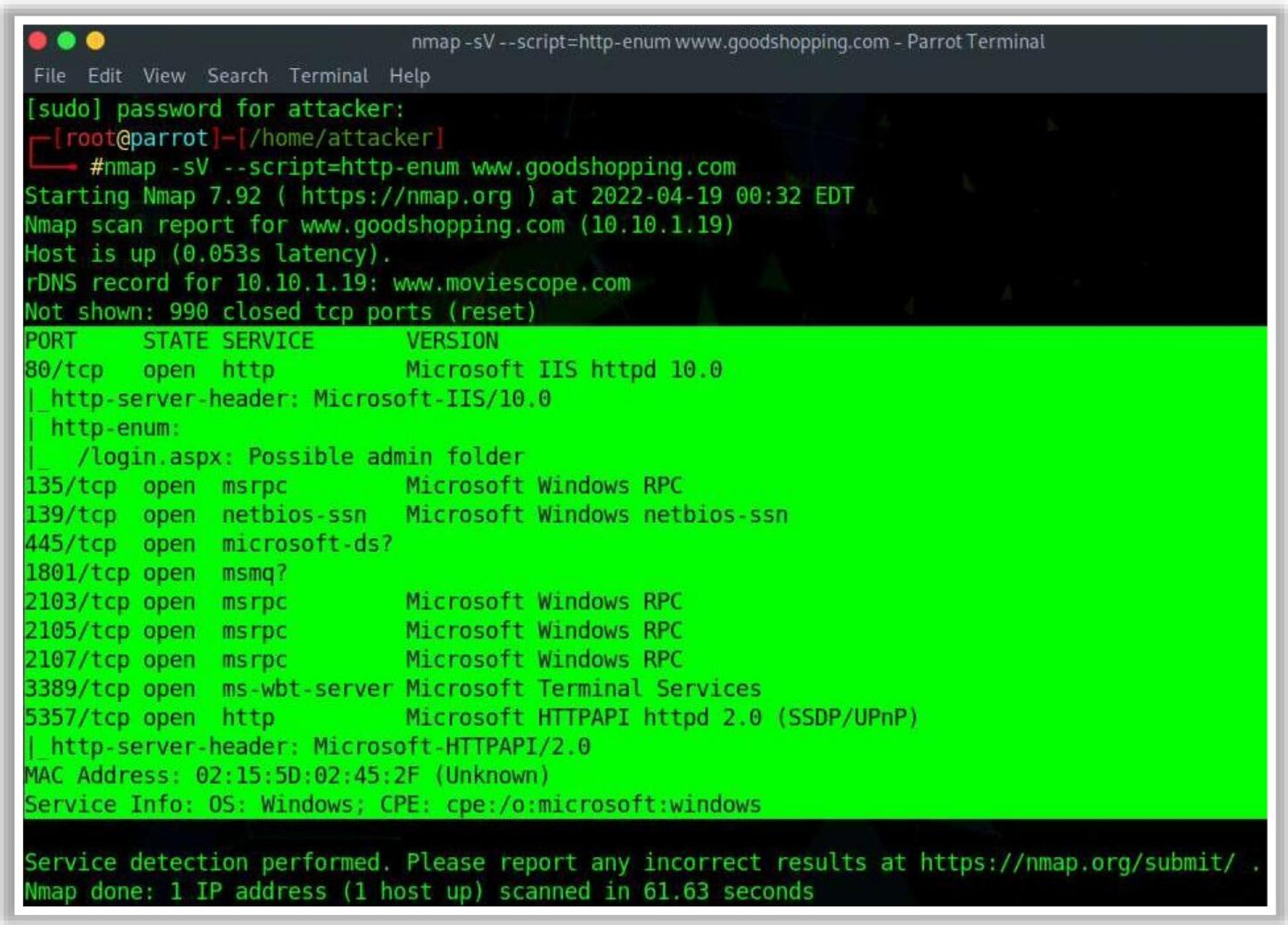
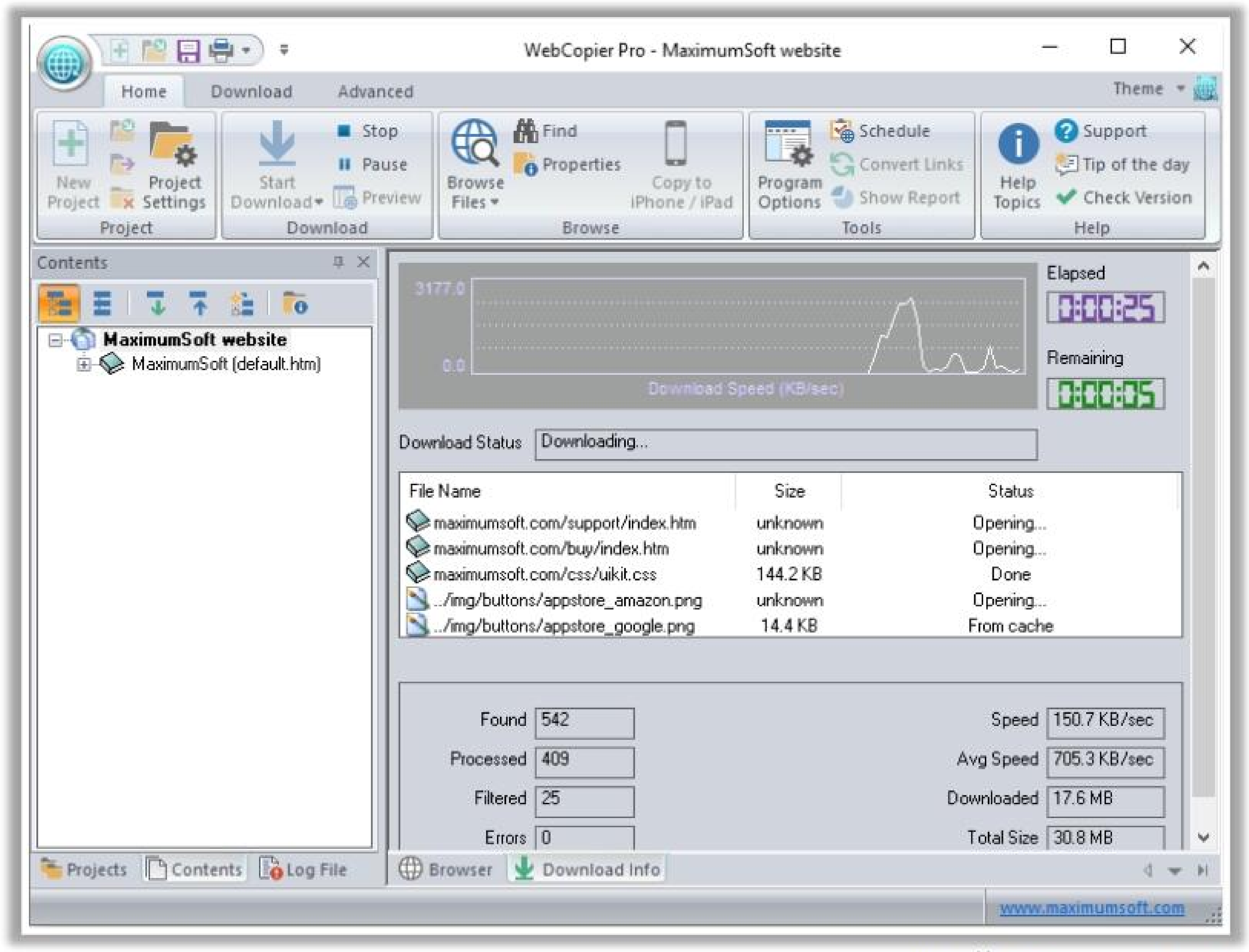


Figure 13.27: Screenshot of Nmap

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 **WebCopier Pro**, **HTTrack Web Site Copier**, **Website Ripper Copier**, etc. to mirror a website



https://www.maximumsoft.com

Website Mirroring

Website mirroring copies an entire website and its content onto a local drive. The mirrored website reveals the complete profile of the site's directory structure, file structure, external links, images, web pages, and so on. With a mirrored target website, an attacker can easily map the website's directories and gain valuable information. An attacker who copies the website does not need to be online to go through the target website. Furthermore, the attacker can gain valuable information by searching the comments and other items in the HTML source code of downloaded web pages. Many website mirroring tools can be used to copy a target website onto a local drive; examples include WebCopier Pro, HTTrack Web Site Copier, Website Ripper Copier, and Cyotek WebCopy.

- **WebCopier Pro**

  Source: *https://www.maximumsoft.com*

  WebCopier Pro is an offline browser to download websites and store them locally, so that they can be viewed/analyzed later. It allows attackers to analyze the website structure and find dead links.

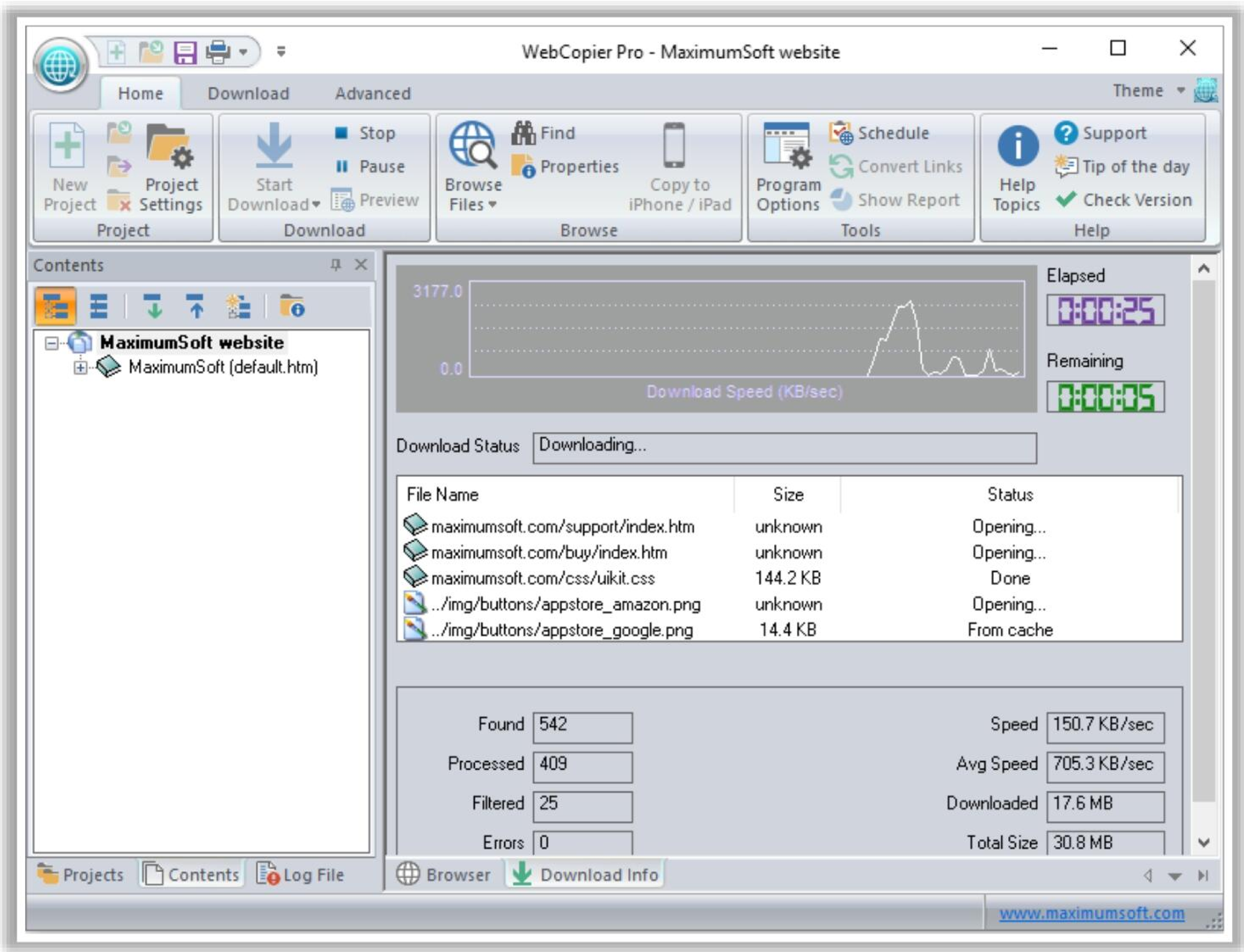


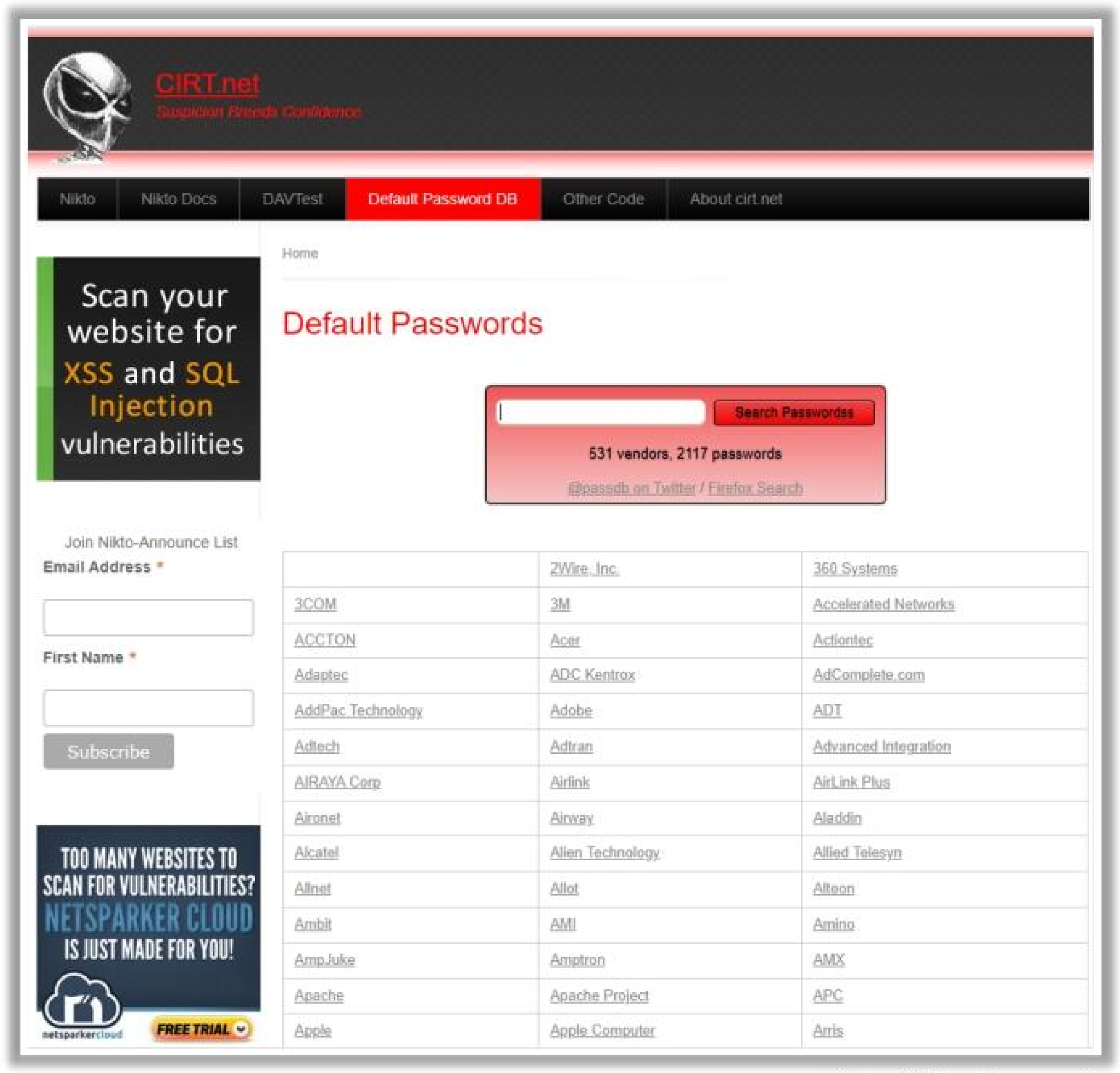
Figure 13.28: Screenshot of WebCopier Pro

The following are some additional website mirroring tools:

- HTTrack Web Site Copier (*https://www.httrack.com*)
- Website Ripper Copier (*https://www.tensons.com*)
- Cyotek WebCopy (*https://www.cyotek.com*)
- Portable Offline Browser (*http://www.metaproducts.com*)
- Offline Explorer Enterprise (*https://metaproducts.com*)

Finding Default Credentials of Web Server

- Many web server administrative interfaces are **publicly accessible** and are 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 use the 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** (*https://open-sez.me*), **cirt.net** (https://cirt.net/passwords), etc.
  - Attempt **password guessing** and **brute-forcing** attacks



https://cirt.net/passwords

Finding Default Credentials of Web Server

Administrators or security personnel use administrative interfaces to securely configure, manage, and monitor web application servers. Many web server administrative interfaces are publicly accessible and located in the root directory. Often, these administrative interface credentials are not properly configured and remain set to default. Attackers attempt to identify the running application interface of the target web server by performing port scanning. Once the running administrative interface is identified, the attacker uses the 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 such as Open Sez Me (*https://open-sez.me*) and cirt.net (*https://cirt.net/passwords*) to identify the default passwords
- Attempt password-guessing and brute-forcing attacks

These default credentials can grant access to the administrative interface, compromising the web server and allowing the attacker to exploit the main web application.

- **cirt.net**

  Source: *https://cirt.net/passwords*

  cirt.net is a lookup database for default passwords, credentials, and ports.

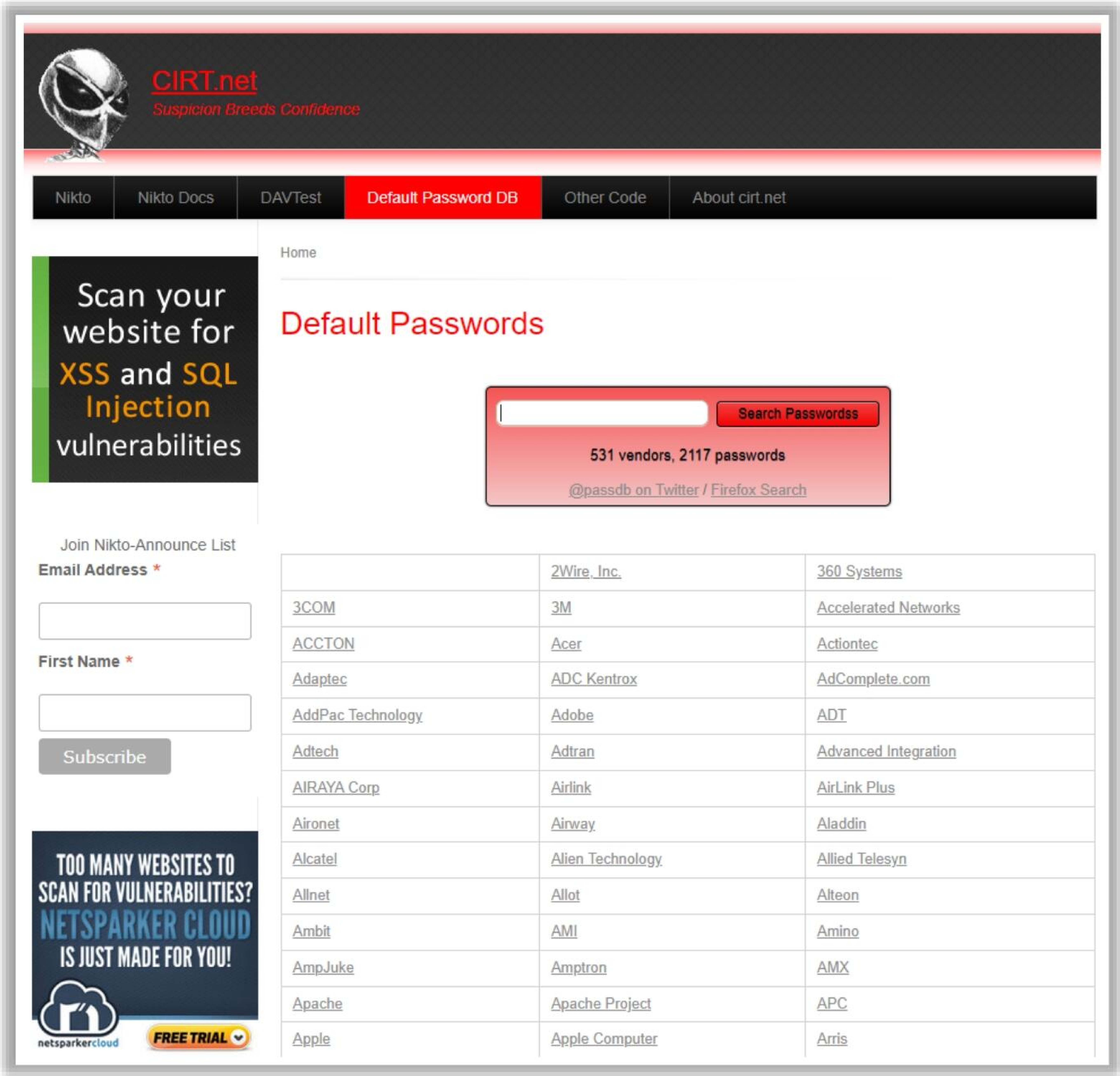


Figure 13.29: Screenshot displaying the default password DB page of cirt.net

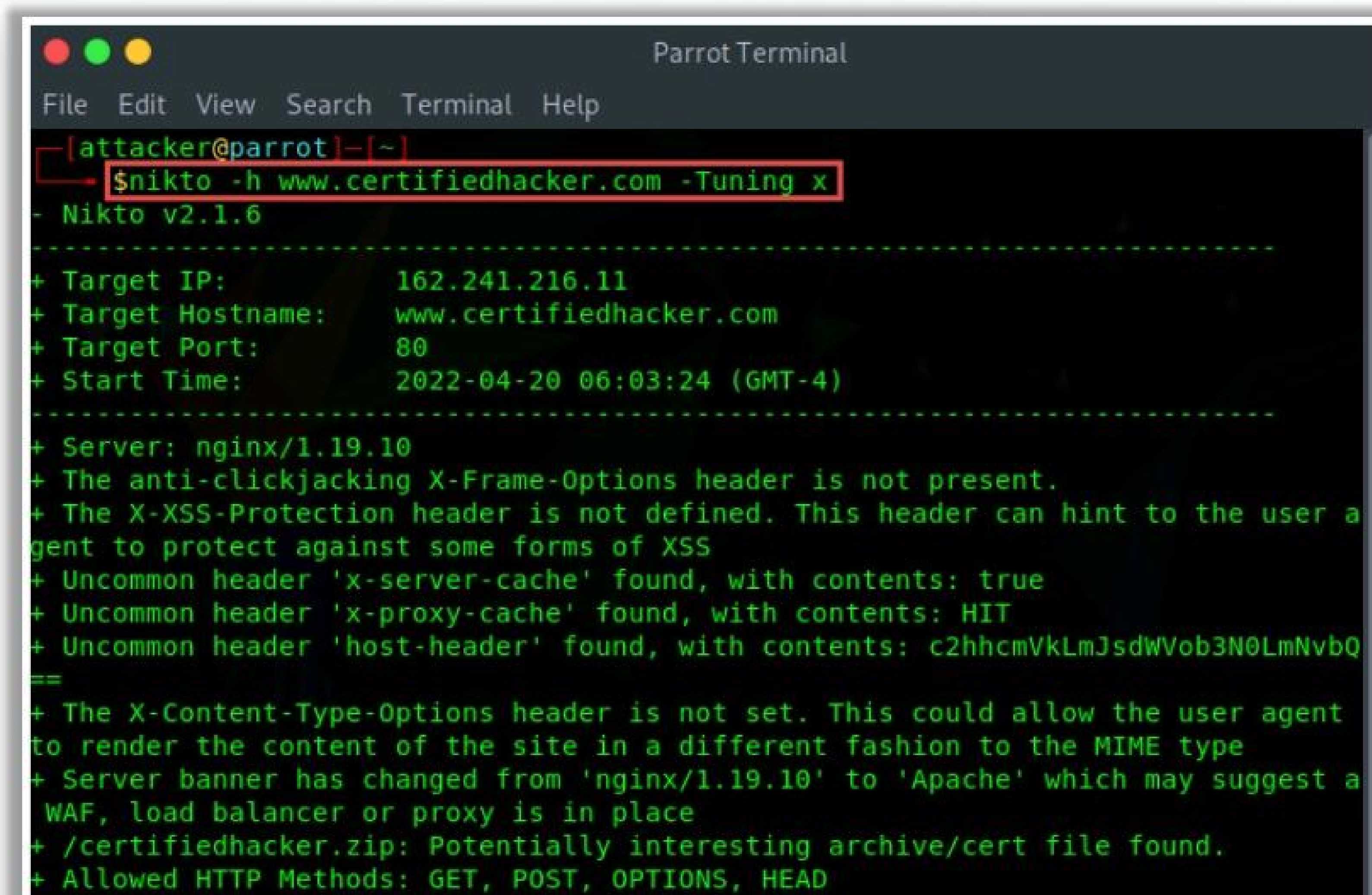
The following are some additional websites for finding the default passwords of web server administrative interfaces:

- *https://open-sez.me*
- *https://www.fortypoundhead.com*
- *http://www.defaultpassword.us*
- *https://default-password.info*
- *https://www.routerpasswords.com*

Finding Default Content of Web Server

- Most web application servers contain **default content and functionalities**, which allows attackers to leverage attacks

- Check for the following default contents and functionalities in the web servers
  - Administrator **debug and test functionality**
  - **Sample functionality** to demonstrate common tasks
  - Publicly accessible **powerful functions**
  - Server **installation manuals**
- Use tools like **Nikto2** (*https://cirt.net*) to identify the default content



https://cirt.net

Finding Default Content of Web Server

Most servers of web applications have default contents and functionalities that allow attackers to launch attacks. The following are some common default contents and functionalities that an attacker attempts to identify in web servers.

- **Administrators debug and test functionality**

  Functionalities designed for administrators to debug, diagnose, and test web applications and web servers contain useful configuration information and the runtime state of both the server and its running applications. Hence, these functionalities are the main targets for attackers.

- **Sample functionality to demonstrate common tasks**

  Many servers contain various sample scripts and pages designed to demonstrate certain application server functions and application programming interfaces (APIs). Often, web servers fail to secure these scripts from attackers, and these sample scripts either contain vulnerabilities that can be exploited by attackers or implement functionalities that allow attackers to exploit.

- **Publicly accessible powerful functions**

  Some web servers include powerful functionalities that are intended for administrative personnel and restricted from public use. However, attackers attempt to exploit such powerful functions to compromise the server and gain access. For example, some application servers allow web archives to be deployed over the same HTTP port as that used by the application. An attacker may use common exploitation frameworks such as Metasploit to perform scanning to identify default passwords, upload backdoors, and gain command-shell access to the target server.

- **Server installation manuals**

  An attacker attempts to identify server manuals, which may contain useful information about configuration and server installation. Accessing this information allows the attacker to prepare an appropriate framework to exploit the installed web server.

Tools such as Nikto2 can be used to identify default contents.

- **Nikto2**

  Source: *https://cirt.net*

  Nikto is a vulnerability scanner used extensively to identify potential vulnerabilities in web applications and web servers.

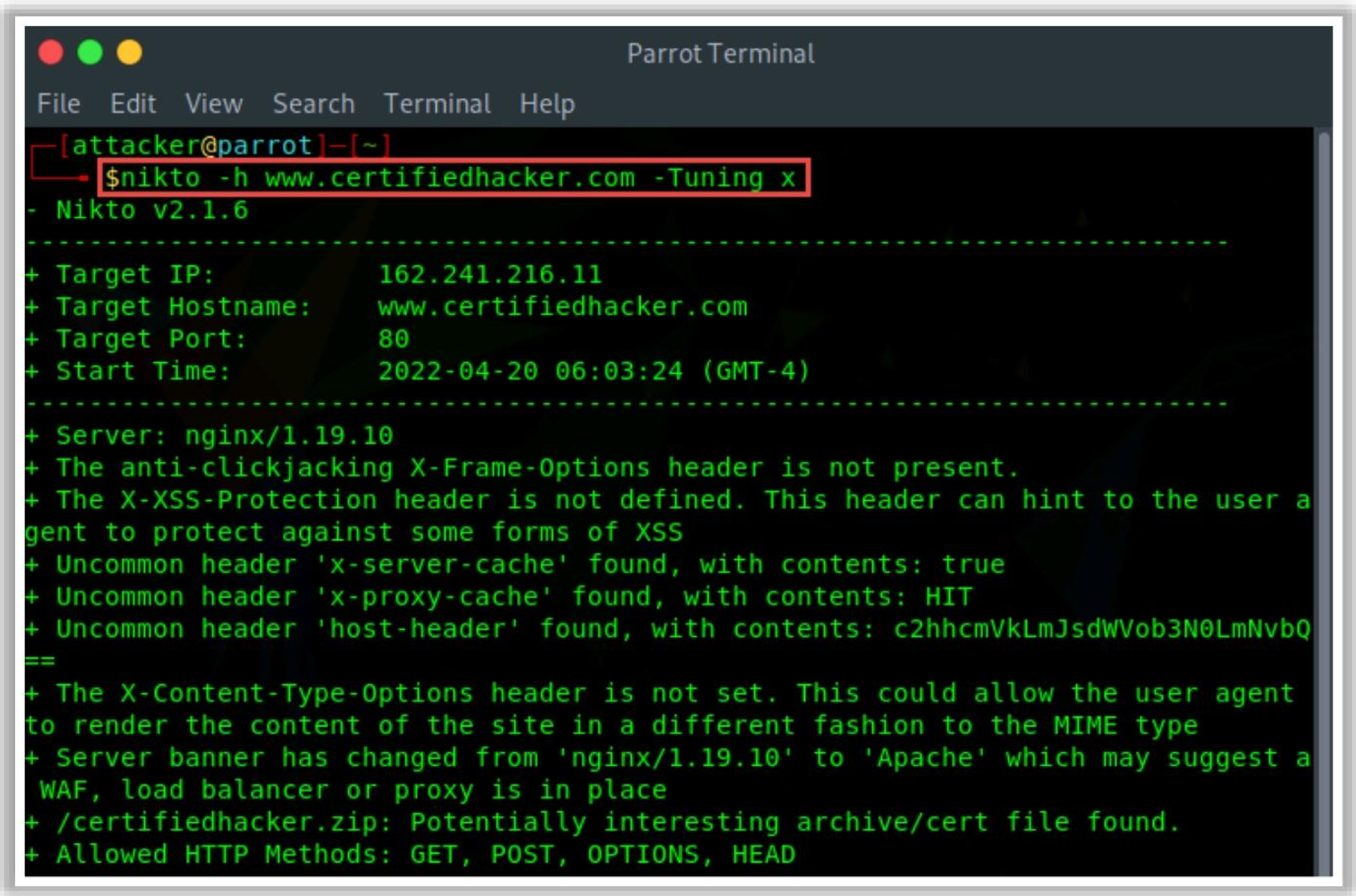
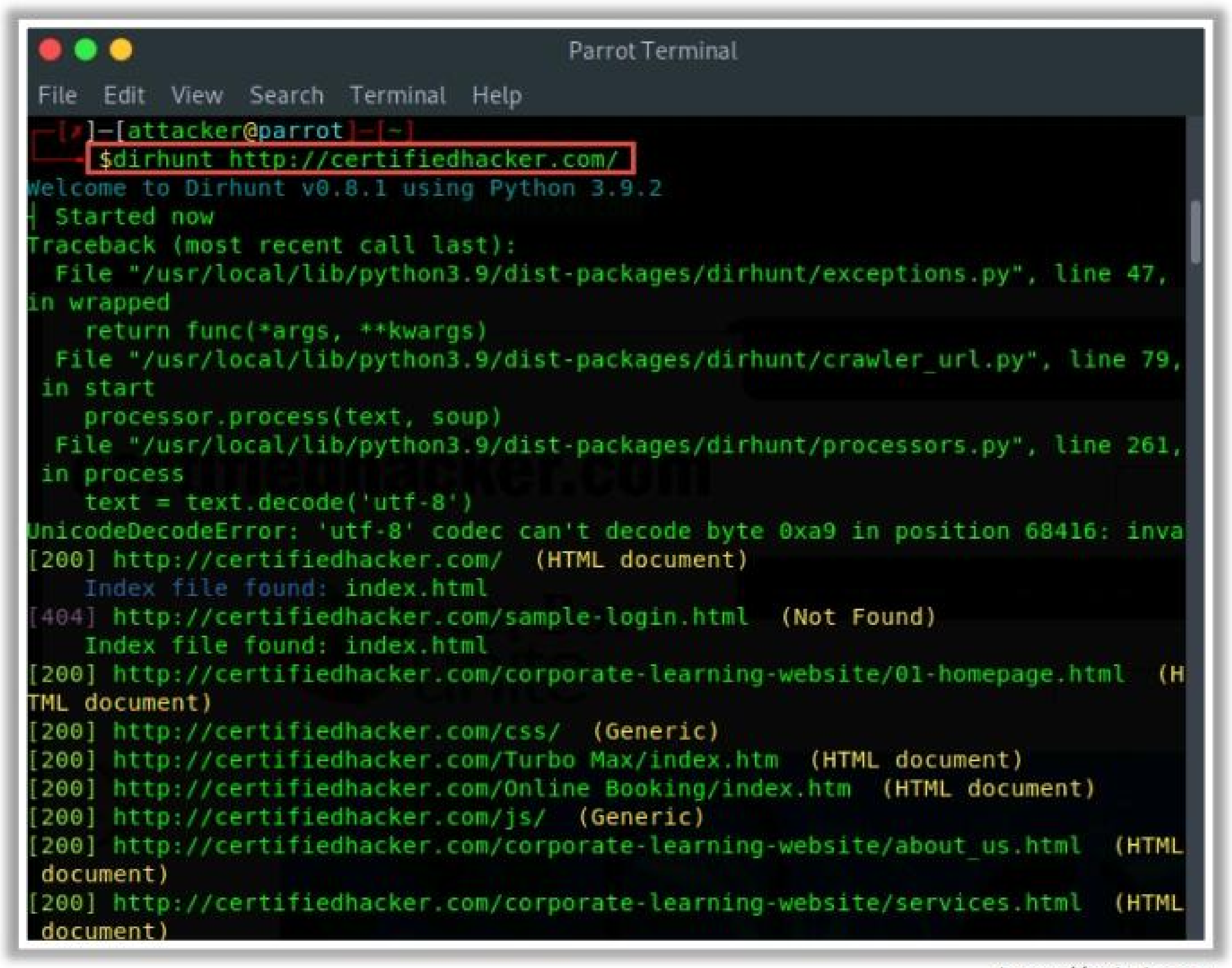


Figure 13.30: Screenshot of Nikto2

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:
  - Returns **default resource within the directory**
  - Returns **error**
  - Returns **listing of directory content**
- Directory listings sometimes possess the following **vulnerabilities** that allow the attackers to **compromise the web server**:
  - Improper **access controls**
  - Unintentional **access to the 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**
- Attackers use tools such as **Dirhunt** to search and analyze directories



https://github.com

Finding Directory Listings of Web Server

When a web server receives a request for a directory, rather than a file, the web server responds to the request in the following ways.

- **Return Default Resource within the directory**: The server may return a default resource within the directory, such as index.html.
- **Return Error**: The server may return an error, such as the HTTP status code 403, indicating that the request is not permitted.
- **Return listing of directory content**: The server may return a listing showing the contents of the directory. A sample directory listing is shown in the screenshot.

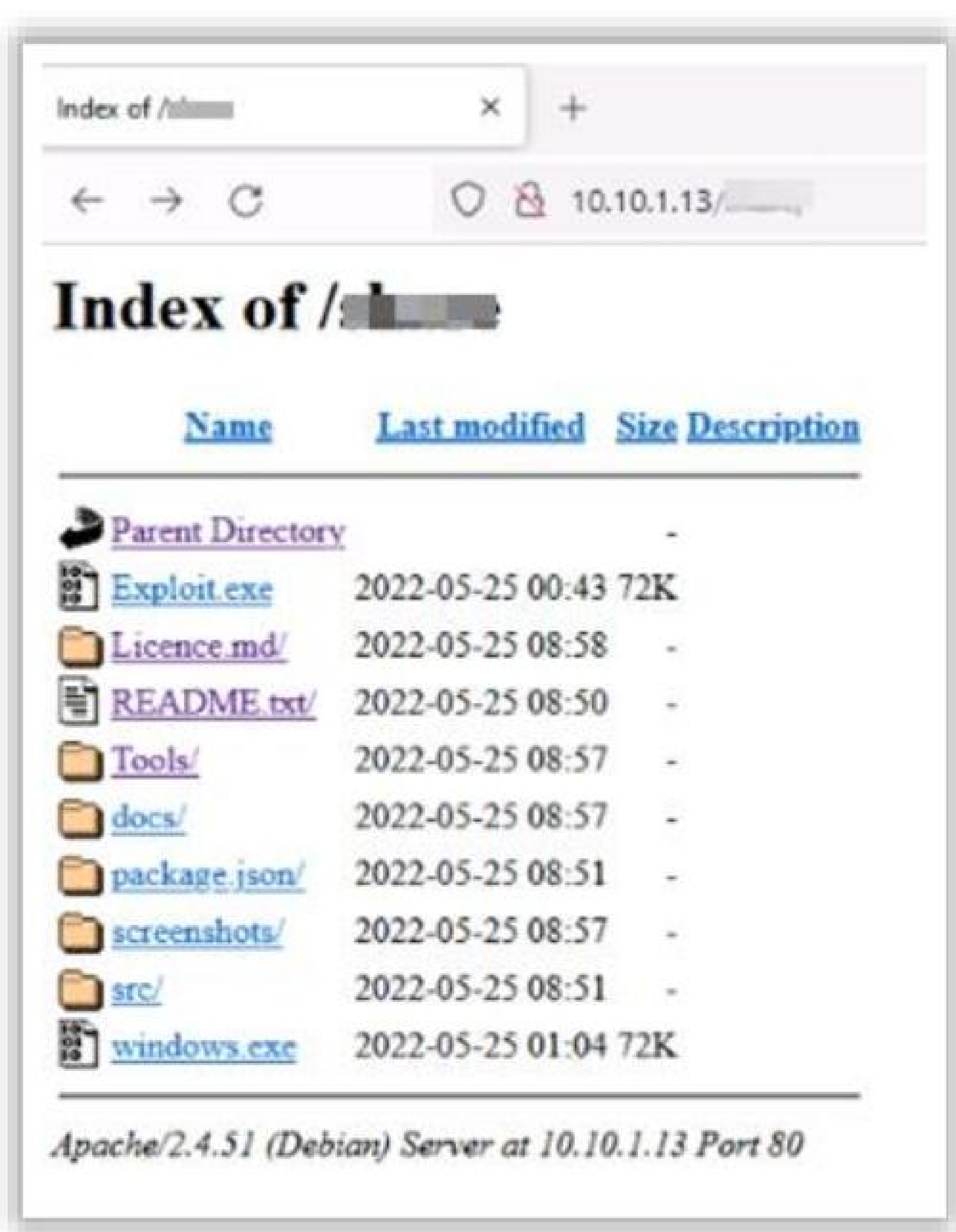


Figure 13.31: Screenshot displaying a sample directory listing

Though directory listings do not have significant relevance from a security perspective, they occasionally possess the following vulnerabilities that allow attackers to compromise web applications:

- Improper access controls
- Unintentional access to the web root of servers

In general, after discovering a directory on a web server, an attacker makes a request for that directory and attempts to access the directory listing. Attackers also attempt to exploit vulnerable web server software that grants access to directory listings.

Attackers use tools such as Dirhunt and Sitechecker to find directory listings of the target web server.

- **Dirhunt**

  Source: *https://github.com*

  Dirhunt is a web crawler optimized for searching and analyzing directories. This tool can find interesting results if the server has the "index of" mode enabled. Dirhunt is also useful if the directory listing is not enabled. It detects directories with false 404 errors, directories where an empty index file has been created to hide things, and so on.

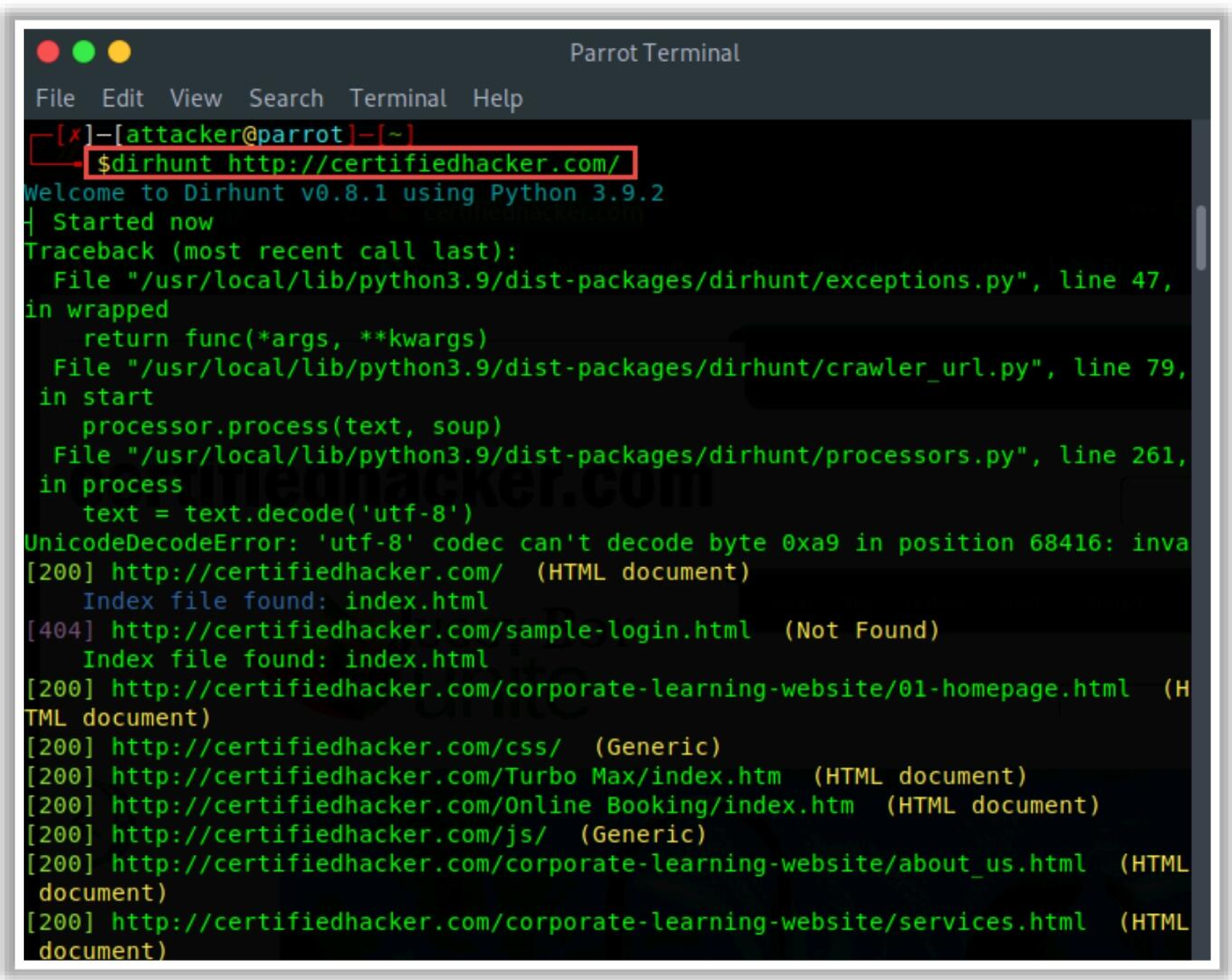
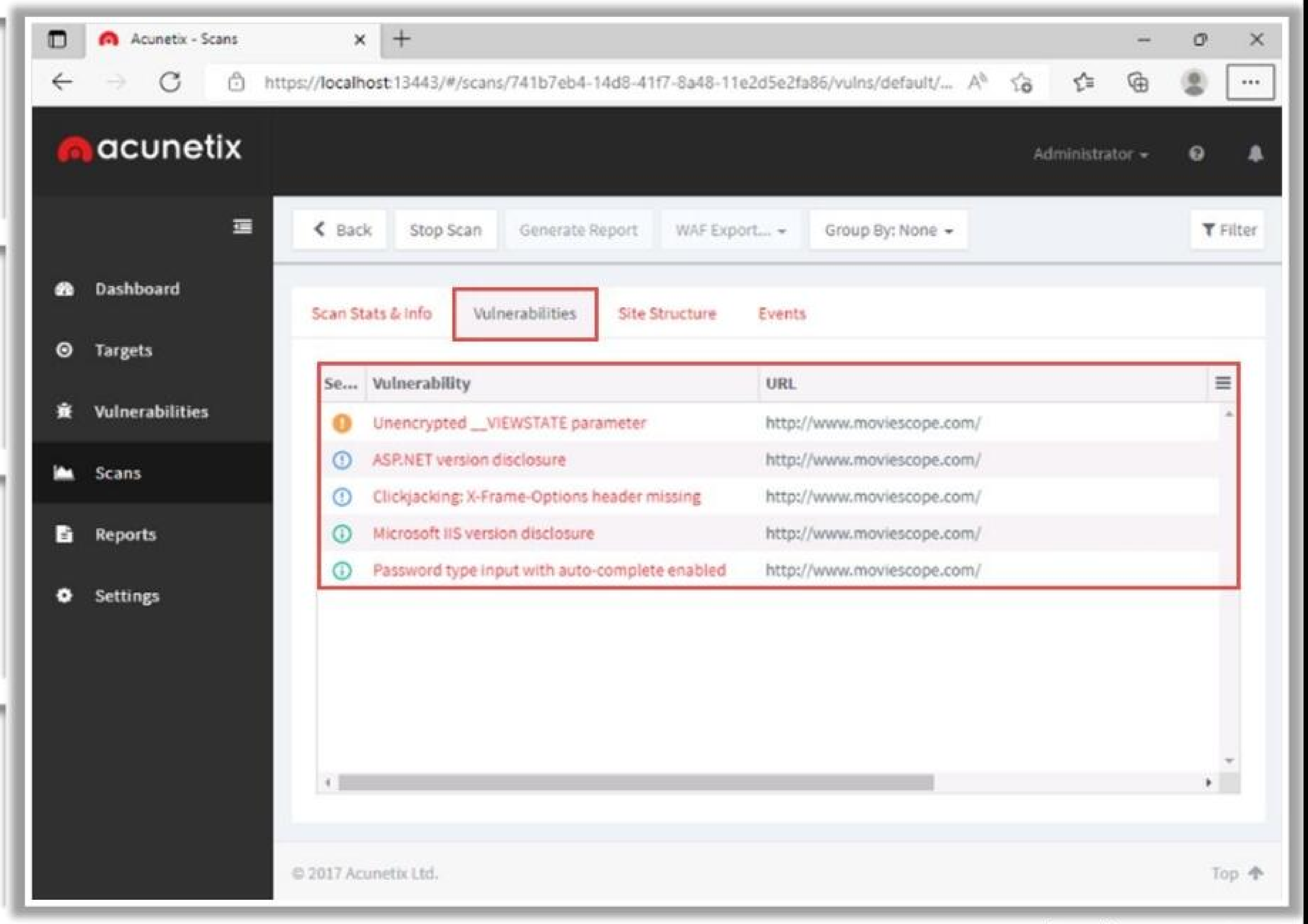


Figure 13.32: Screenshot of Dirhunt displaying directories and files

Vulnerability Scanning

- Implement vulnerability scan to **identify weaknesses** in a network and determine if the system can be exploited
- Use vulnerability scanners such as Acunetix Web Vulnerability Scanner, and Fortify WebInspect to find **hosts**, **services**, and **vulnerabilities**
- Sniff the network traffic to find any **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

Vulnerability Scanning

Vulnerability scanning is performed to identify vulnerabilities and misconfigurations in a target web server or network. Vulnerability scanning reveals possible weaknesses in a target server to exploit in a web server attack. In the vulnerability-scanning phase, attackers use sniffing techniques to obtain data on the network traffic to determine active systems, network services, and applications. Automated tools such as Acunetix Web Vulnerability Scanner are used to perform vulnerability scanning on a target server and find hosts, services, and vulnerabilities.

- **Acunetix Web Vulnerability Scanner**

  Source: *https://www.acunetix.com*

  Acunetix Web Vulnerability Scanner (WVS) scans websites and detects vulnerabilities. Acunetix WVS checks web applications for SQL injections, XSS, and so on. It includes advanced pen testing tools to ease manual security audit processes and creates professional security audit and regulatory compliance reports based on AcuSensor Technology. It supports the testing of web forms and password-protected areas, pages with CAPTCHA, single sign-on, and two-factor authentication mechanisms. It detects application languages, web server types, and smartphone-optimized sites. Acunetix crawls and analyzes different types of websites, including HTML5, Simple Object Access Protocol (SOAP), and Asynchronous JavaScript and Extensible Markup Language (AJAX). It supports the scanning of network services running on the server and the port scanning of the web server.

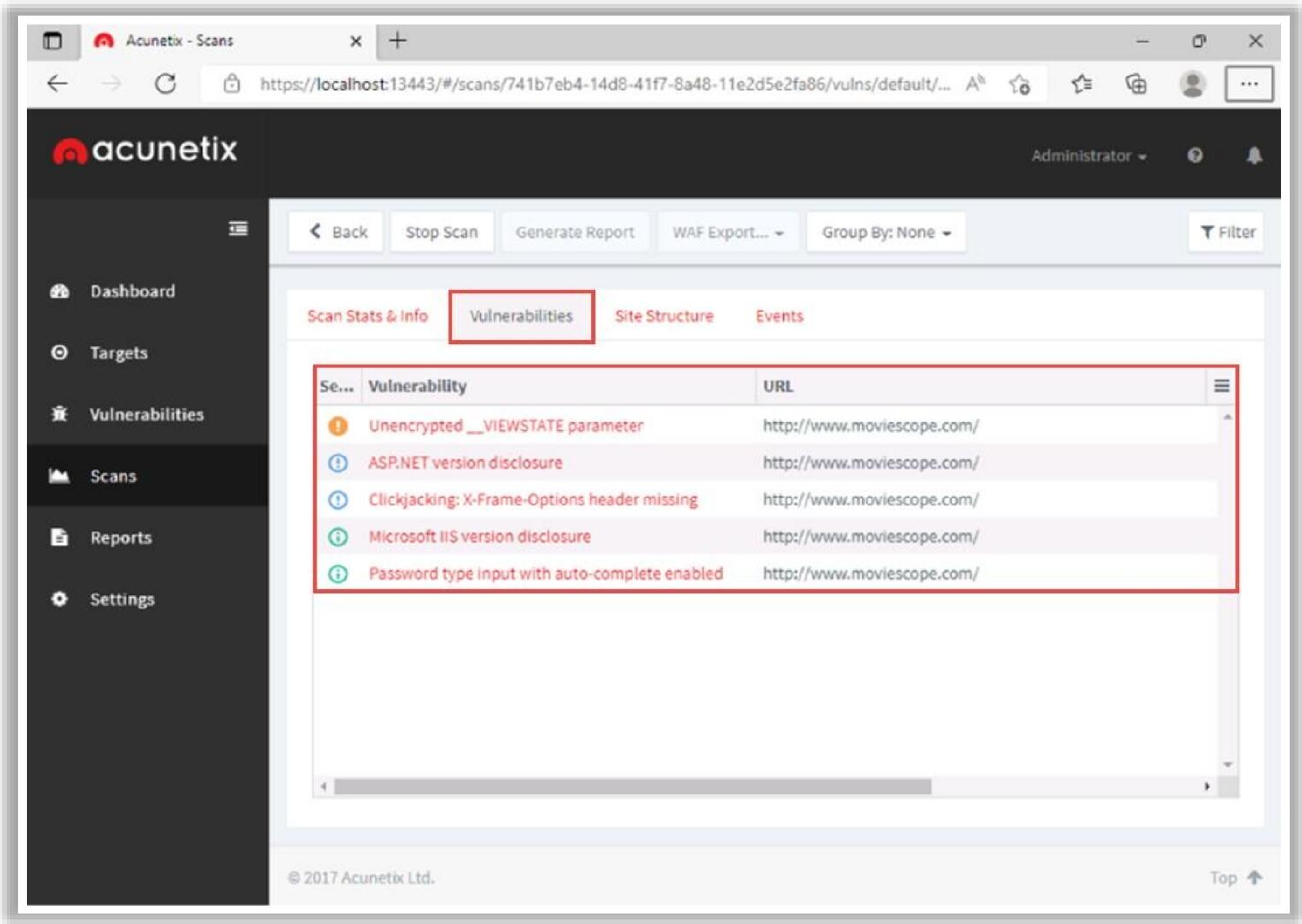


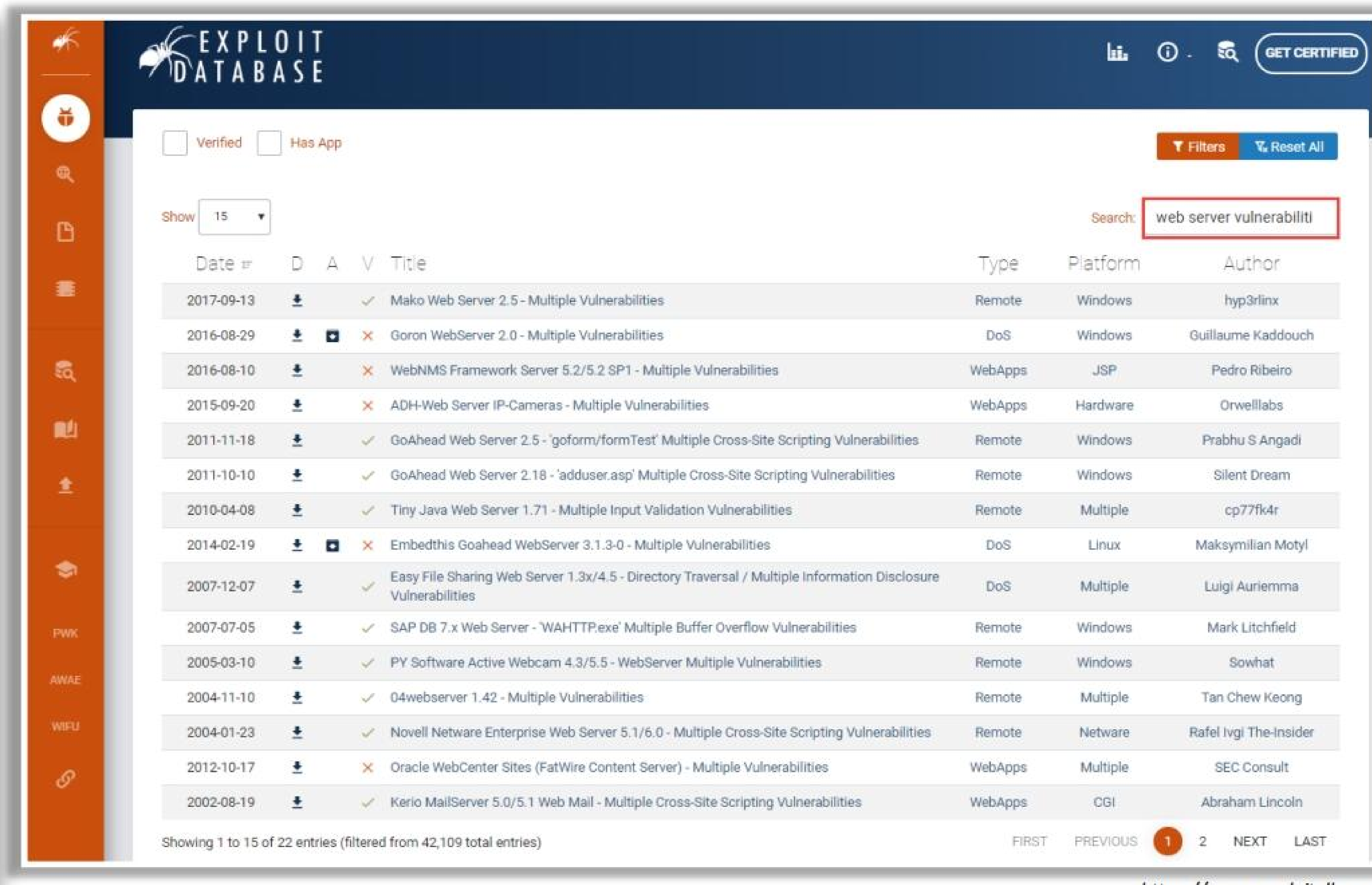
Figure 13.33: Screenshot of Acunetix Web Vulnerability Scanner

The following are some additional vulnerability scanning tools:

- Fortify WebInspect (*https://www.microfocus.com*)
- Tenable.io (*https://www.tenable.com*)
- ImmuniWeb (*https://www.immuniweb.com*)
- Invicti (*https://www.invicti.com*)

Finding Exploitable Vulnerabilities

- Search for web server exploitable vulnerabilities based on the web server OS and software application on exploit sites such as **Packet Storm** (*https://packetstormsecurity.com*), **Exploit Database** (*https://www.exploit-db.com*), etc.
- Search for vulnerability based on information gathered in the previous stages using the **Exploit Database**
- Exploiting these vulnerabilities allows for the execution of a command or binary on a target machine to **gain higher privileges** than the existing ones or **bypass security mechanisms**



https://www.exploit-db.com

Finding Exploitable Vulnerabilities

Flaws and programming errors in software design lead to security vulnerabilities. Attackers take advantage of these vulnerabilities to perform various attacks on the confidentiality, availability, or integrity of a system. Software vulnerabilities such as programming flaws in a program, service, or within the OS software or kernel can be exploited to execute malicious code.

Many public vulnerability repositories that are available online allow access to information about various software vulnerabilities. Attackers search on exploit sites such as Packet Storm (*https://packetstormsecurity.com*) and Exploit Database (*https://www.exploit-db.com*) for exploitable vulnerabilities of a web server based on its OS and software applications. Attackers use the information gathered in the previous stages to find the relevant vulnerabilities by using Exploit Database.

Exploiting these vulnerabilities allows attackers to execute a command or binary on a target machine to gain higher privileges than existing ones or to bypass security mechanisms. Attackers using these exploits can even access privileged user accounts and credentials.

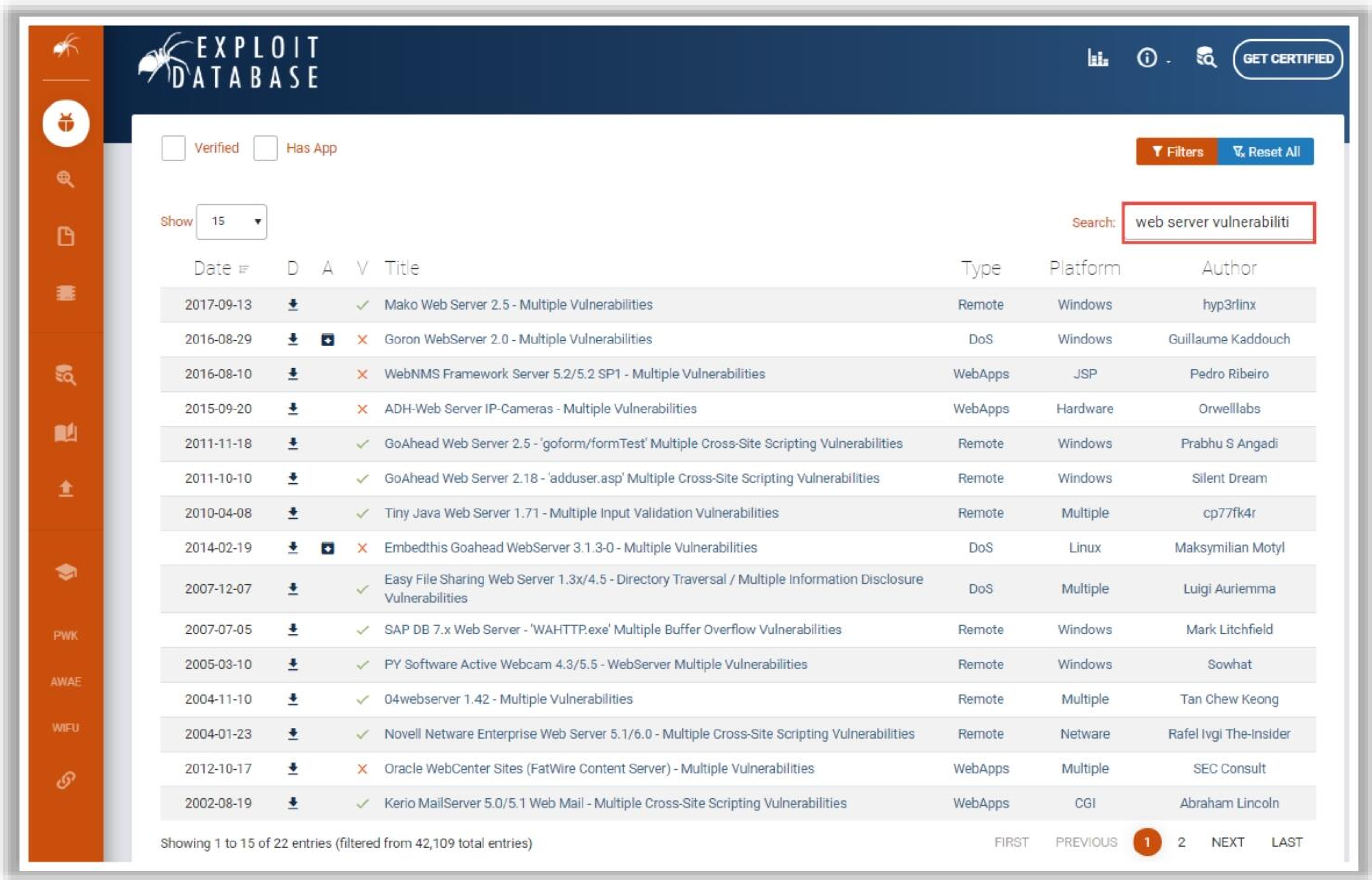
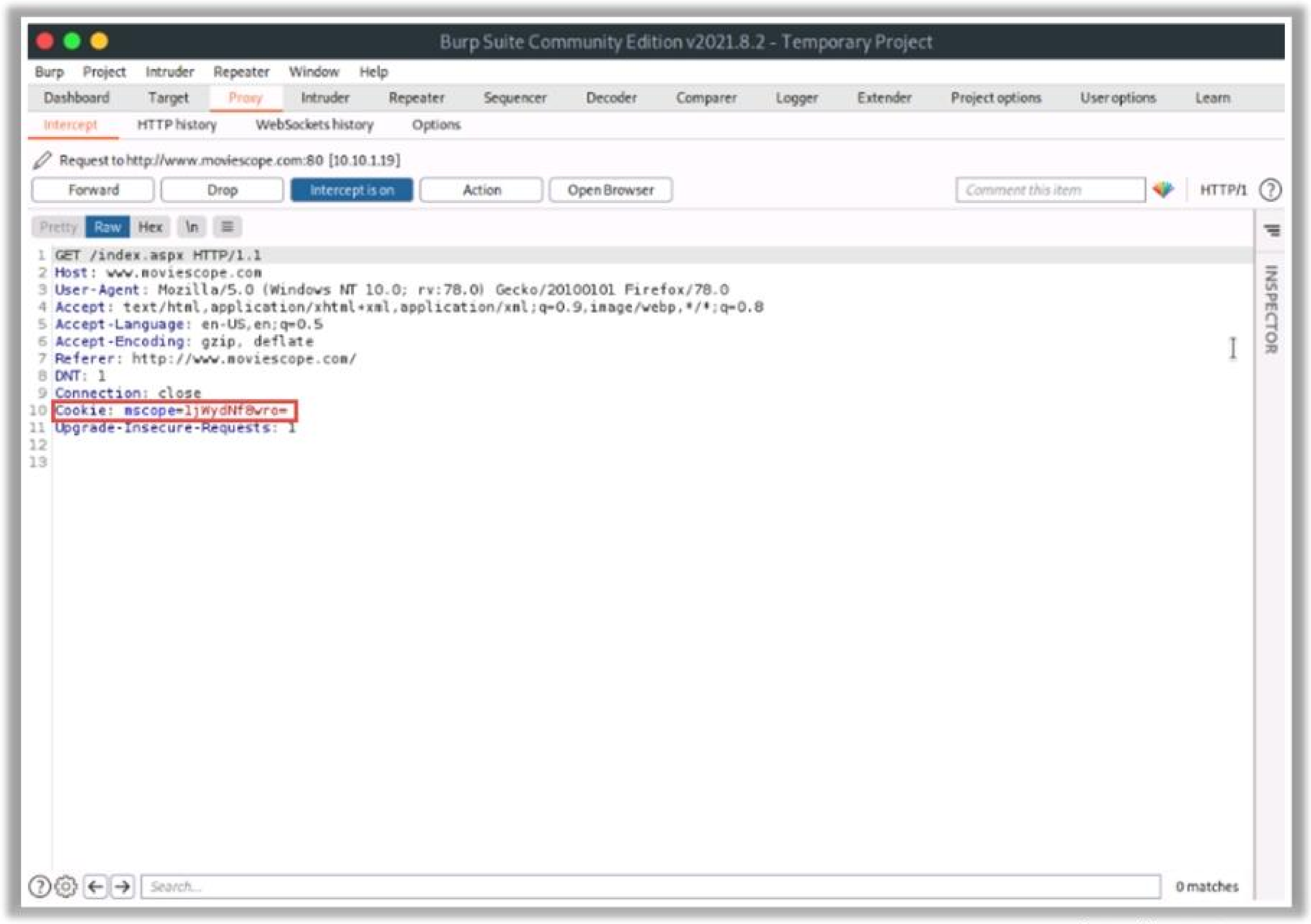


Figure 13.34: Screenshot of Google Hacking Database (GHDB)

Session Hijacking

- Sniff valid session IDs to **gain unauthorized access** to the web server to snoop 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**, **JHijack**, **Ettercap**, etc. to automate session hijacking

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



https://portswigger.net

Session Hijacking

Valid session IDs can be sniffed to gain unauthorized access to a web server and snoop its data. An attacker can hijack or steal valid session content using various techniques such as session token prediction, session replay, session fixation, sidejacking, and XSS. By using these techniques, the attacker attempts to capture valid session cookies and IDs in established sessions. The attacker uses tools such as Burp Suite, JHijack, and Ettercap to automate session hijacking.

- **Burp Suite**

  Source: *https://portswigger.net*

  Burp Suite is a web security testing tool that can hijack session IDs in established sessions. The Sequencer tool in Burp Suite tests the randomness of session tokens. With this tool, an attacker can predict the next possible session ID token and use that to take over a valid session.

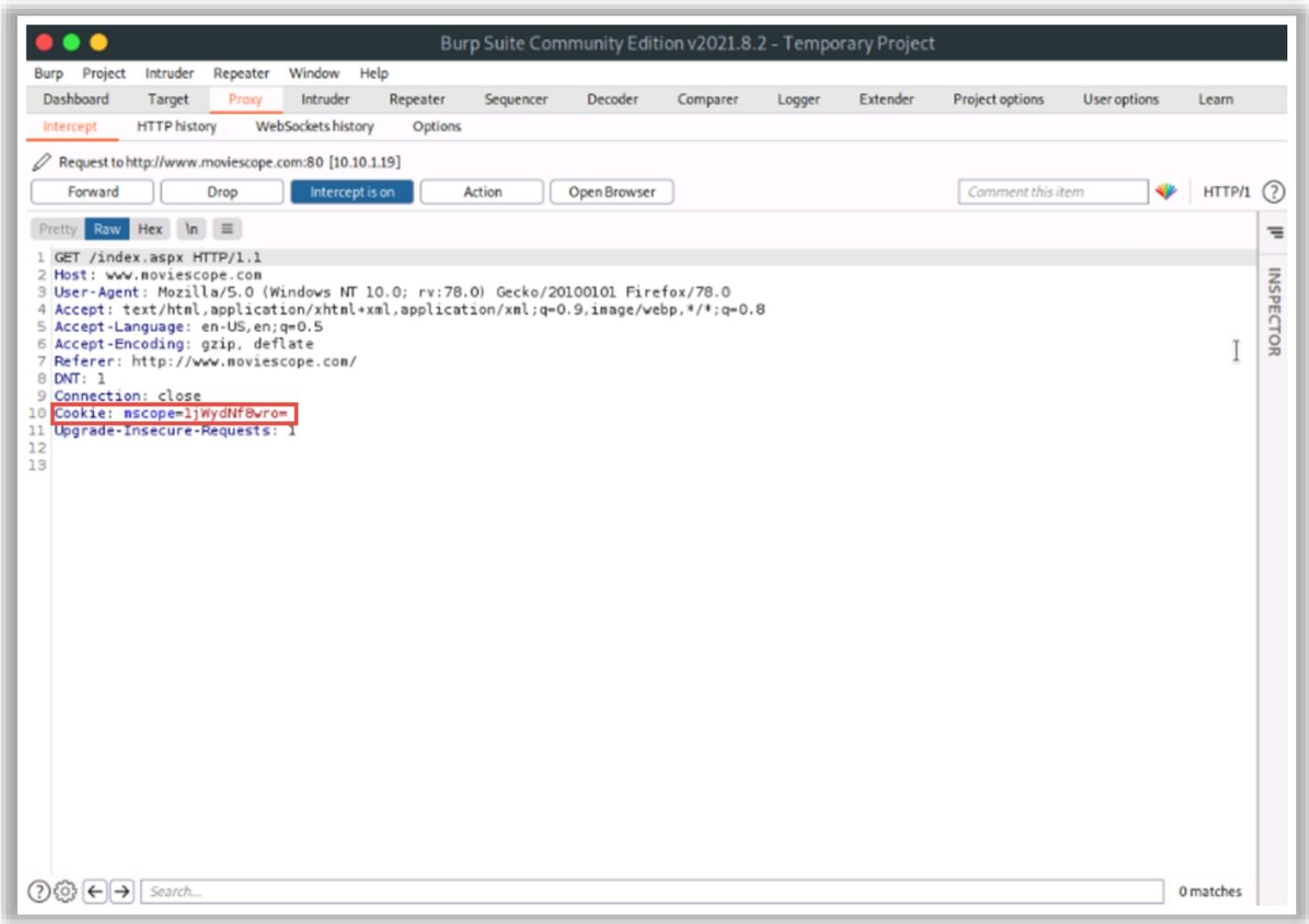


Figure 13.35: Screenshot of Burp Suite

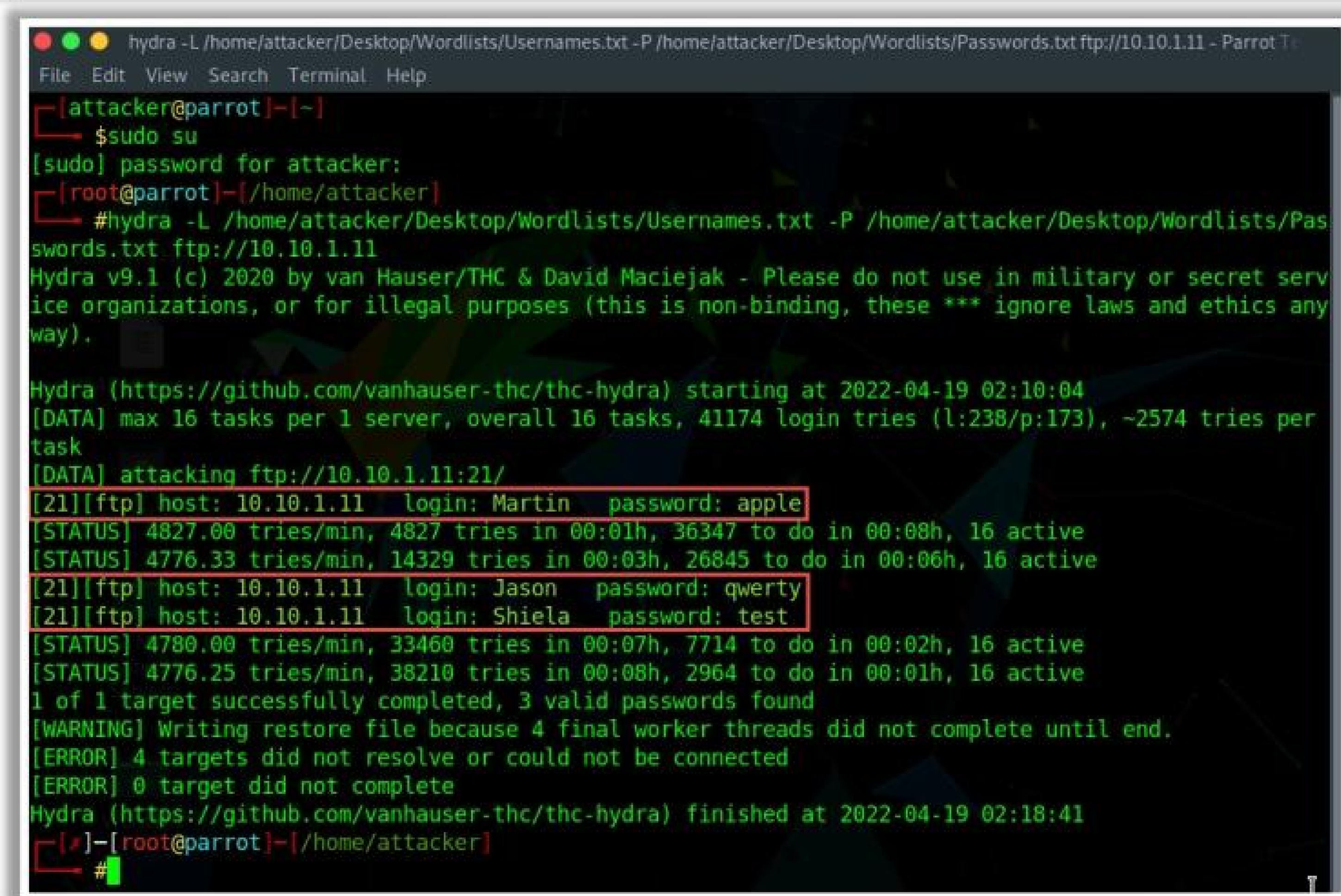
The following are some additional session hijacking tools:

- JHijack (*https://sourceforge.net*)
- Ettercap (*https://www.ettercap-project.org*)
- CookieCatcher (*https://github.com*)
- Cookie Cadger (*https://github.com*)

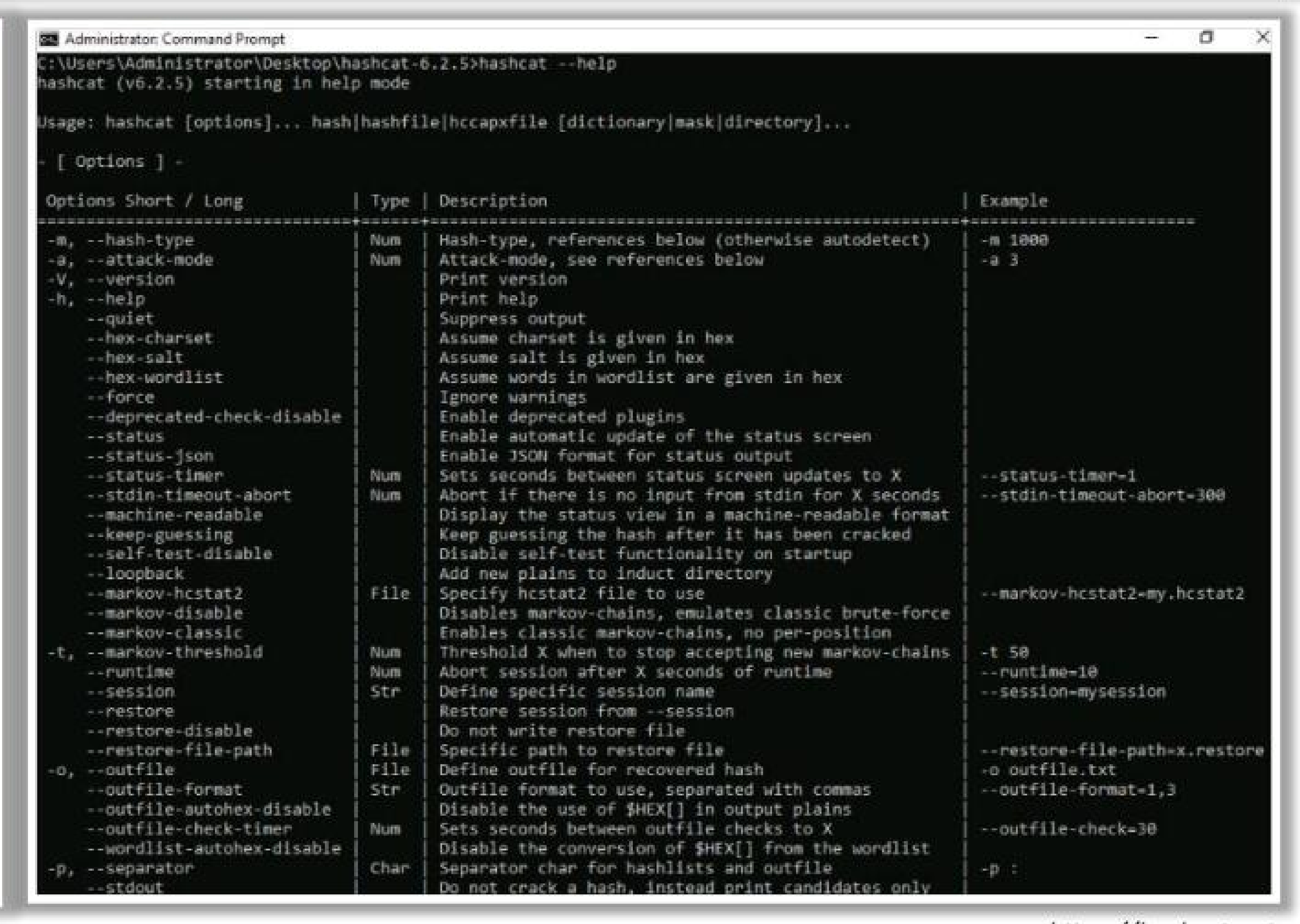
Note: For complete coverage of concepts and techniques related to session hijacking, refer to Module 11: Session Hijacking.

Web Server Password Hacking

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



https://github.com



https://hashcat.net

Web Server Password Hacking

In this phase of web server hacking, an attacker attempts to crack web server passwords. The attacker may employ all possible techniques of password cracking to extract passwords, including password guessing, dictionary attacks, brute-force attacks, hybrid attacks, precomputed hashes, rule-based attacks, distributed network attacks, and rainbow attacks. The attacker needs patience to crack passwords because some of these techniques are tedious and time-consuming. The attacker can also use automated tools such as Hashcat, THC Hydra, and Ncrack to crack web passwords and hashes.

- **Hashcat**

  Source: *https://hashcat.net*

  Hashcat is a cracker compatible with multiple OSs and platforms and can perform multi-hash (MD4, 5; SHA – 224, 256, 384, 512; RIPEMD-160; etc.), multi-device password cracking. The attack modes of this tool are straight, combination, brute force, hybrid dict + mask, and hybrid mask + dict.

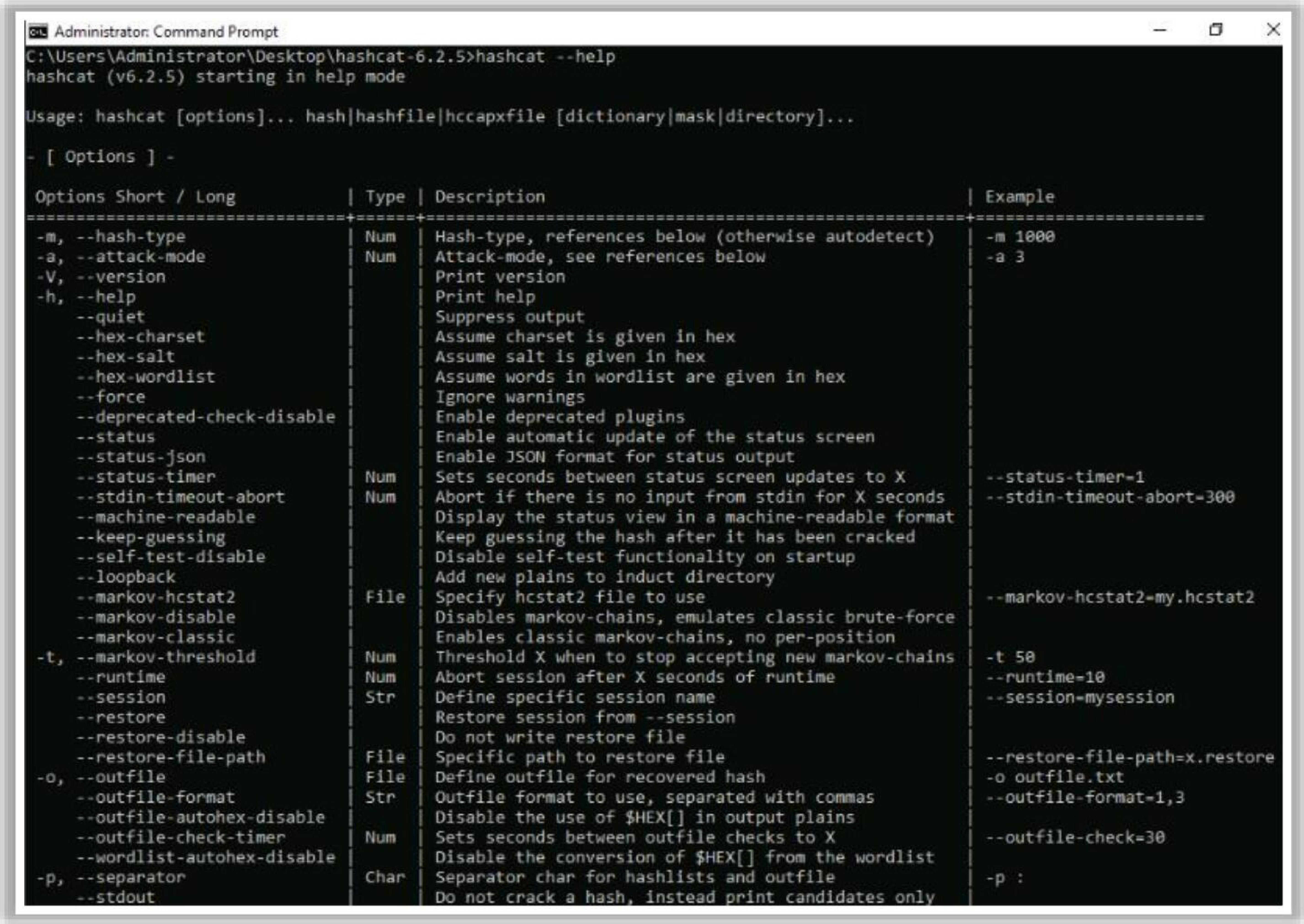


Figure 13.36: Screenshot of Hashcat password cracker

- **THC Hydra**

  Source: *https://github.com*

  THC Hydra is a parallelized login cracker that can attack numerous protocols. This tool is a proof-of-concept code that provides researchers and security consultants the possibility to demonstrate how easy it would be to gain unauthorized remote access to a system.

  Currently, this tool supports the following protocols: Asterisk; Apple Filing Protocol (AFP); Cisco Authentication, Authorization, and Accounting (AAA); Cisco auth; Cisco enable; Concurrent Versions System (CVS); Firebird; FTP; HTTP-FORM-GET; HTTP-FORM-POST; HTTP-GET; HTTP-HEAD; HTTP-POST; HTTP-PROXY; HTTPS-FORM-GET; HTTPS-FORM-POST; HTTPS-GET; HTTPS-HEAD; HTTPS-POST; HTTP-Proxy; ICQ; Internet Message Access Protocol (IMAP); Internet Relay Chat (IRC); Lightweight Directory Access Protocol (LDAP); Memcached; MongoDB; Microsoft SQL Server; MySQL; Network Control Protocol (NCP); Network News Transfer Protocol (NNTP); Oracle Listener; Oracle system identifier (SID); Oracle; PC-Anywhere; personal computer Network File System (PC-NFS); POP3; Postgres; Radmin; Remote Desktop Protocol (RDP); Rexec; Rlogin; Rsh; Real Time Streaming Protocol (RTSP); SAP R/3; Session Initiation Protocol (SIP); Server Message Block (SMB); Simple Mail Transfer Protocol (SMTP); SMTP Enum; Simple Network Management Protocol (SNMP) v1+v2+v3; SOCKS5; SSH (v1 and v2); SSH key; Subversion;

TeamSpeak (TS2); Telnet; VMware-Auth; Virtual Network Computing (VNC); and Extensible Messaging and Presence Protocol (XMPP).

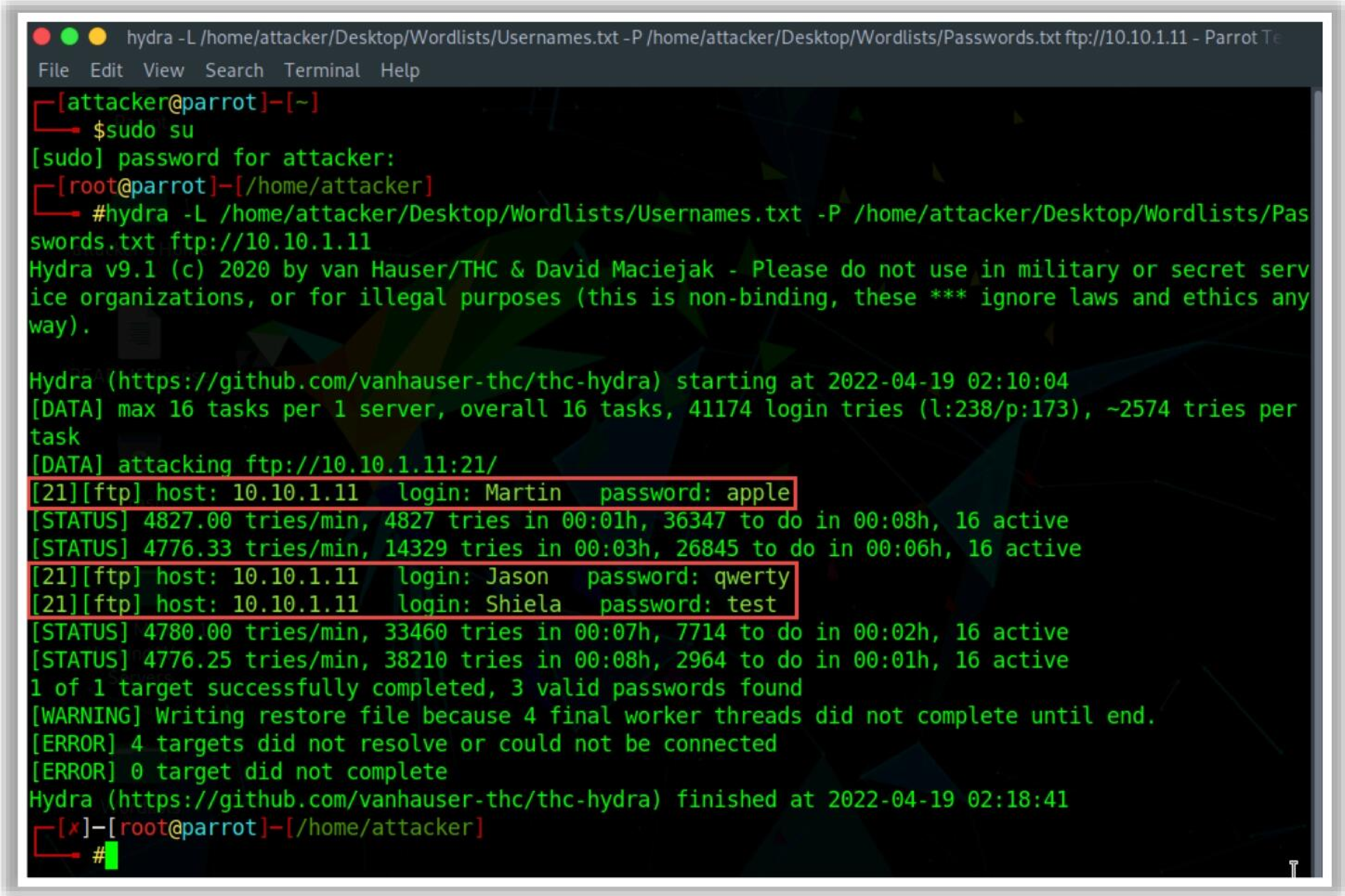
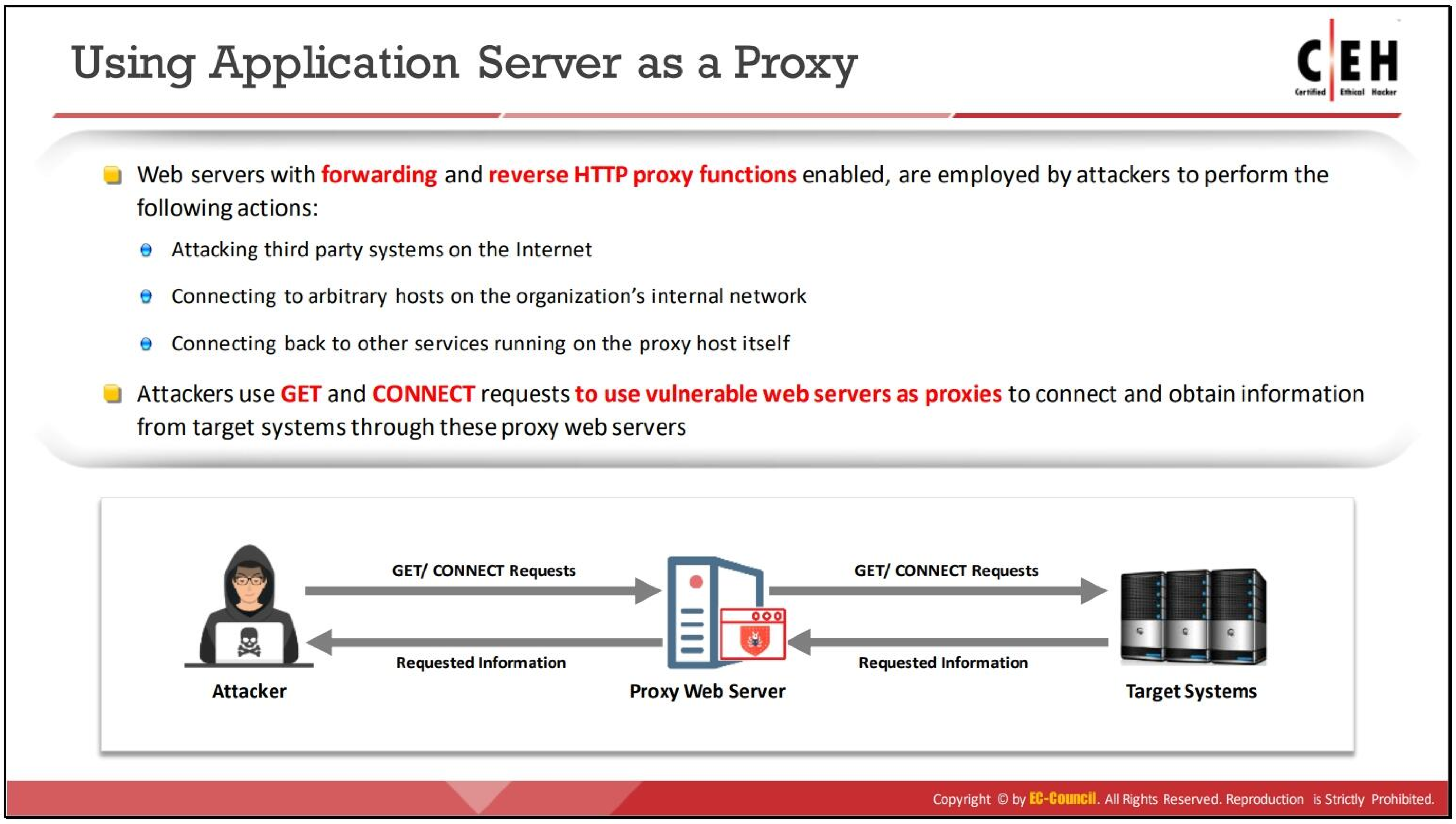


Figure 13.37: Screenshot of THC Hydra password cracker

The following are some additional password cracking tools:

- Ncrack (*https://nmap.org*)
- Rainbow crack (*https://project-rainbowcrack.com*)
- Wfuzz (*http://www.edge-security.com*)
- Wireshark (*https://www.wireshark.org*)



Using Application Server as a Proxy

Web servers are occasionally configured to perform functions such as forwarding or reverse HTTP proxy. Web servers with these functions enabled are employed by attackers to perform the following attacks:

- Attacking third-party systems on the 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 to and obtain information from target systems through these web servers.

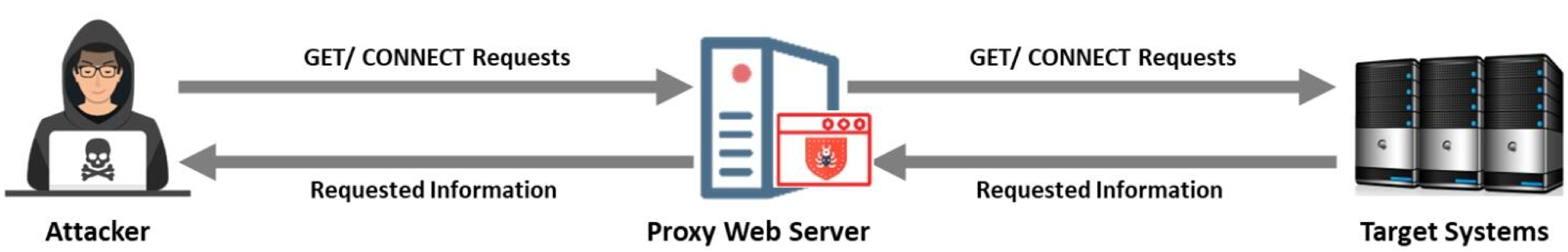
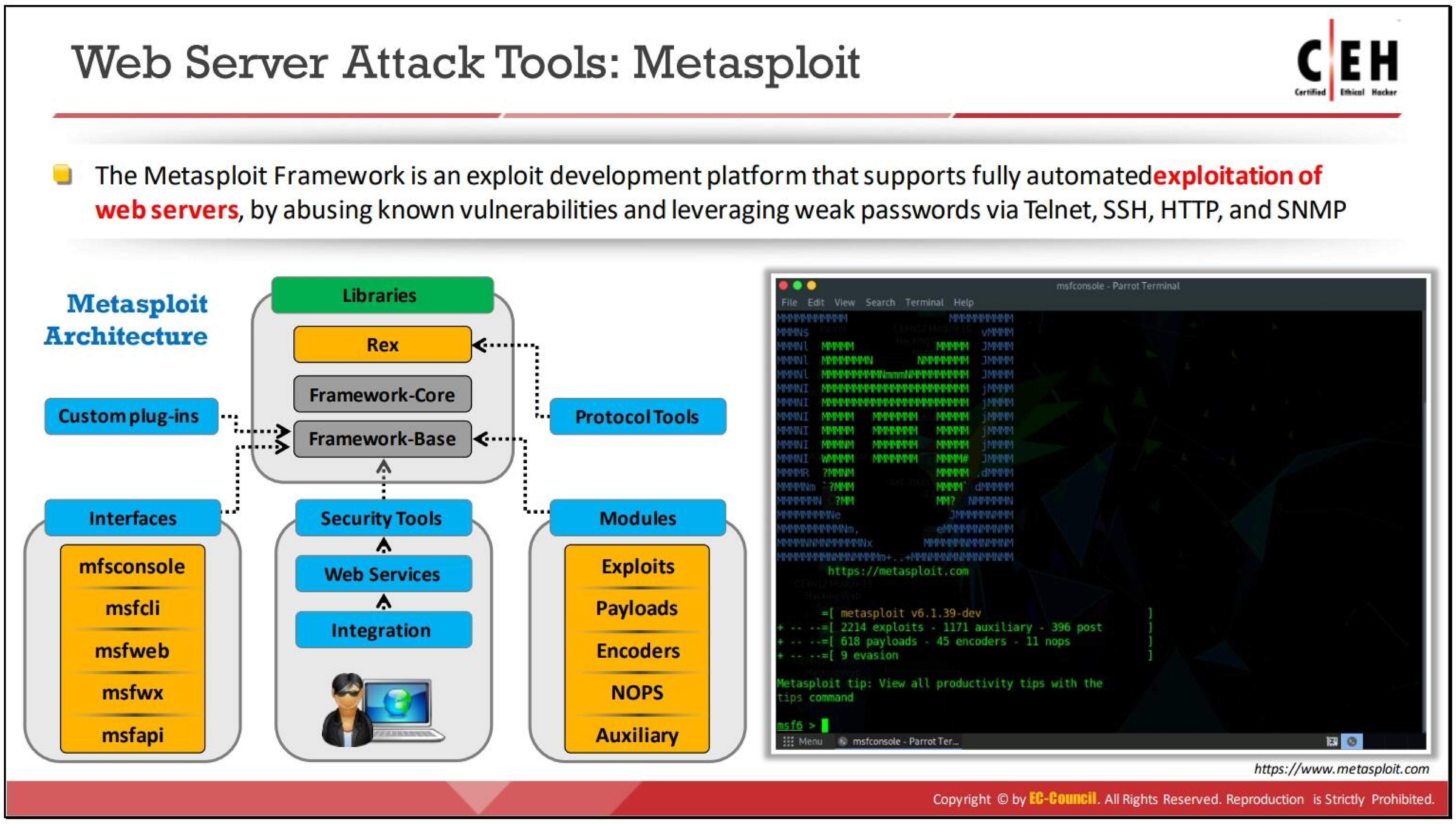


Figure 13.38: Illustration of the use of an application server as a proxy



Web Server Attack Tools: Metasploit

Source: *https://www.metasploit.com*

The Metasploit Framework is a penetration-testing toolkit, exploit development platform, and research tool that includes hundreds of working remote exploits for various platforms. It performs fully automated exploitation of web servers by abusing known vulnerabilities and leveraging weak passwords via Telnet, SSH, HTTP, and SNMP.

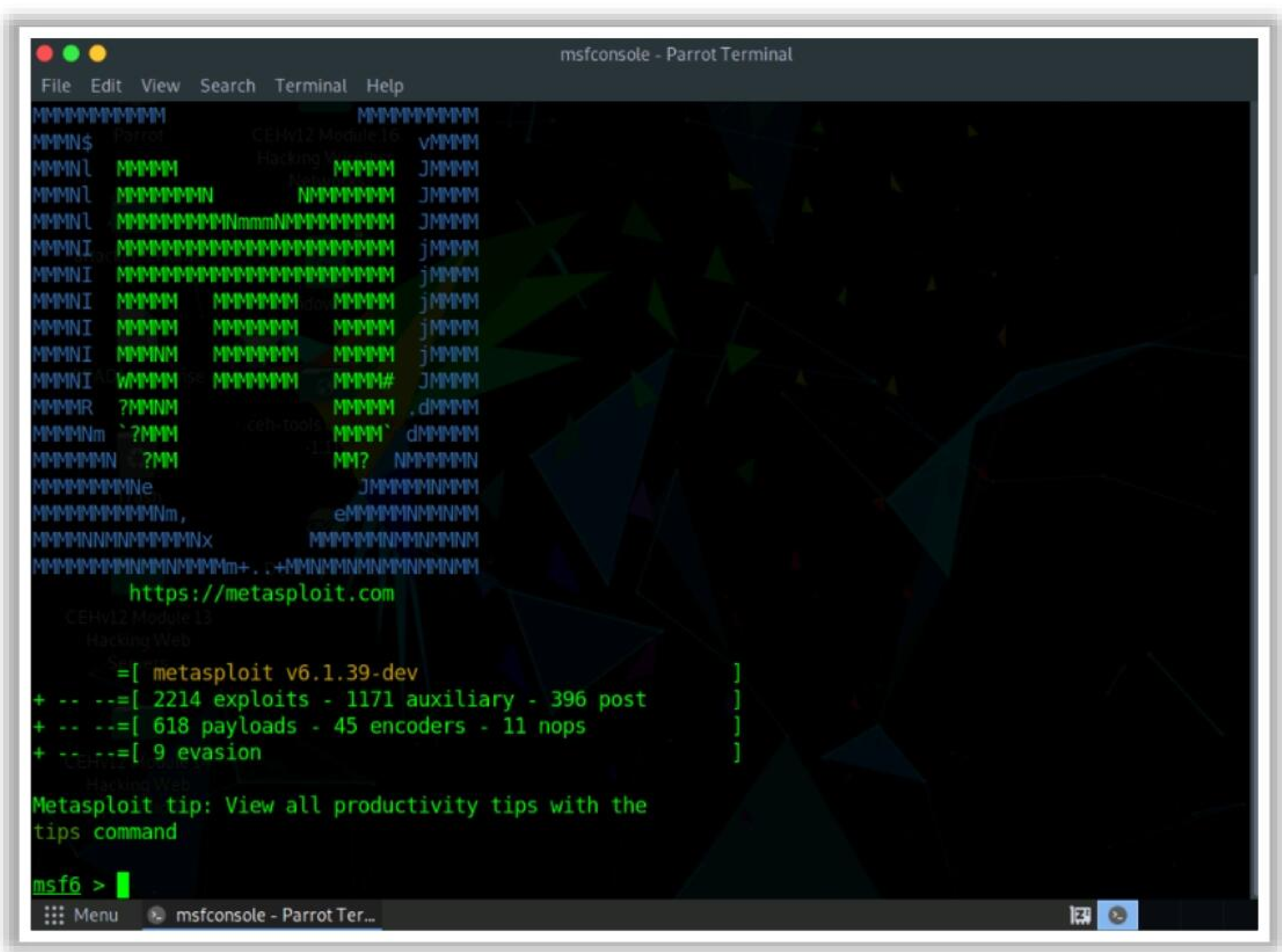


Figure 13.39: Screenshot of Metasploit

An attacker may use the following features of Metasploit to perform a web server attack:

- Closed-loop vulnerability validation
- Phishing simulations
- Social engineering
- Manual brute forcing
- Manual exploitation
- Evade-leading defensive solutions

Metasploit enables pen testers to perform the following:

- Quickly complete pen-test assignments by automating repetitive tasks and leveraging multi-level attacks
- Assess the security of web applications, network and endpoint systems, as well as email users
- Tunnel any traffic through compromised targets to pivot deep into a network
- Customize the content and template of executive, audit, and technical reports

Metasploit Architecture

The Metasploit Framework is an open-source exploitation framework that provides security researchers and pen testers with a uniform model for the rapid development of exploits, payloads, encoders, no operation (NOP) generators, and reconnaissance tools. The framework reuses large chunks of code that a user would otherwise have to copy or re-implement on a per-exploit basis. The framework is modular in architecture and encourages the reuse of code across various projects. The framework can be broken down into a few different pieces, the lowest level of which is the framework core. The framework core is responsible for implementing all the required interfaces that allow interaction with exploit modules, sessions, and plugins. It supports vulnerability research, exploit development, and the creation of custom security tools.

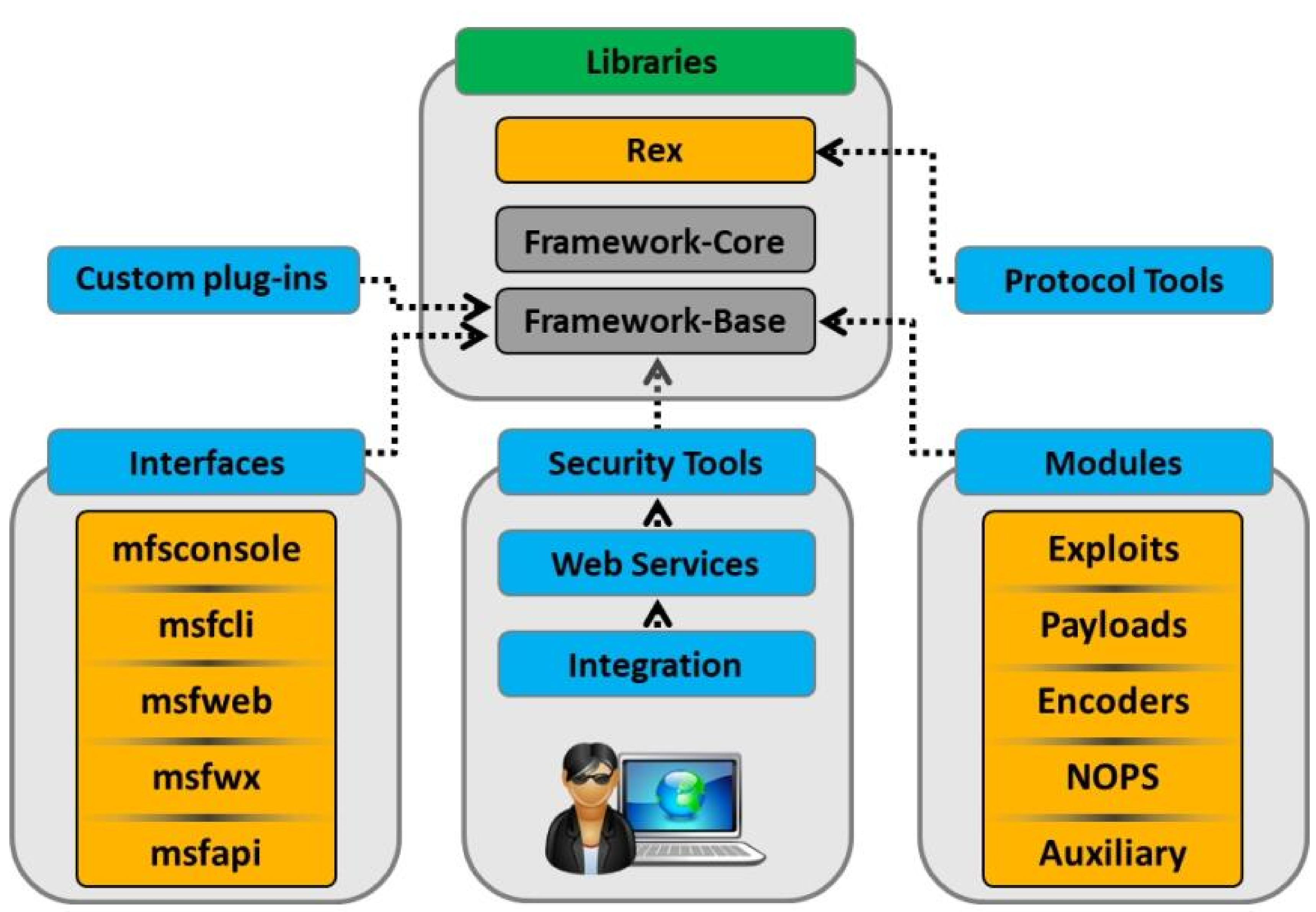


Figure 13.40: Metasploit architecture

Metasploit Exploit Module

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

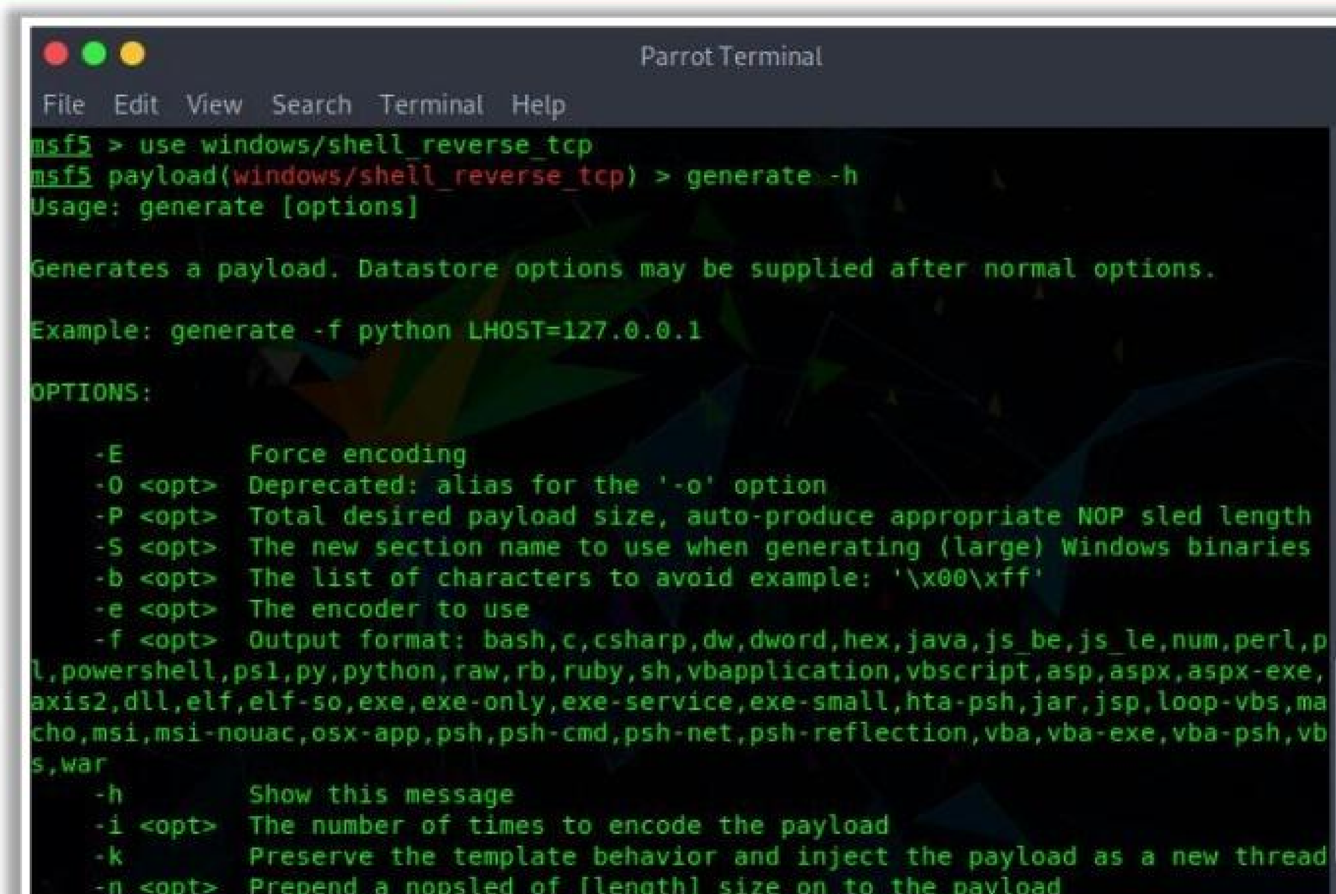
Steps to exploit a system using the Metasploit Framework

1. **Configure an Active Exploit**
2. **Verify the Exploit Options**
3. **Select a Target**
4. **Select a Payload**
5. **Launch the Exploit**

Metasploit Payload and Auxiliary Modules

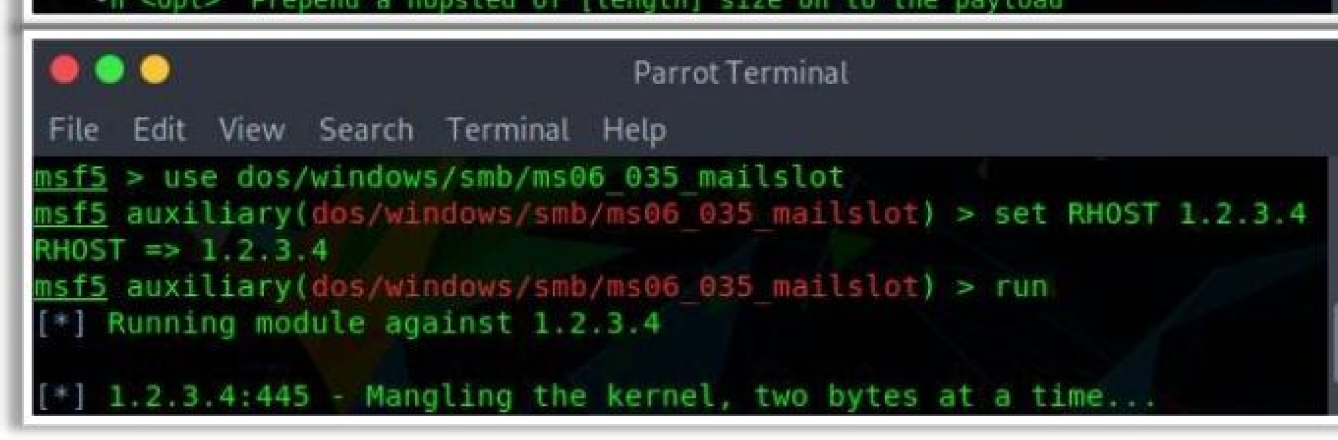
Payload Module

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



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 an auxiliary module, either use the **`run`** command, or **`exploit`** command



Metasploit NOPS Module

- NOPS modules generate a no-operation instruction used for blocking out buffers
- Use **generate** command to generate a NOP sled of arbitrary size and display it in a specific 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)>`

Command to generate 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) >
```


Metasploit Modules

- **Metasploit Exploit Module**

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

  A system can be exploited with the Metasploit Framework through the following steps:

  - Configure an active exploit
  - Verify the exploit options
  - Select a target
  - Select a payload
  - Launch the exploit

- **Metasploit Payload Module**

  An exploit carries a payload in its backpack when it breaks into a system and then leaves the backpack there. The following three types of payload modules are provided by the Metasploit Framework.

  - **Singles**: Self-contained and completely standalone
  - **Stagers**: Sets up a network connection between the attacker and victim
  - **Stages**: Downloaded by stager modules

A Metasploit payload module can upload and download files from the system, take screenshots, and collect password hashes. It can even take over the screen, mouse, and keyboard to control a computer remotely. The payload Module establishes a communication channel between the Metasploit framework and victim host. It combines arbitrary code that is executed as the result of an exploit succeeding. To generate payloads, a payload is first selected using the command shown in the screenshot.

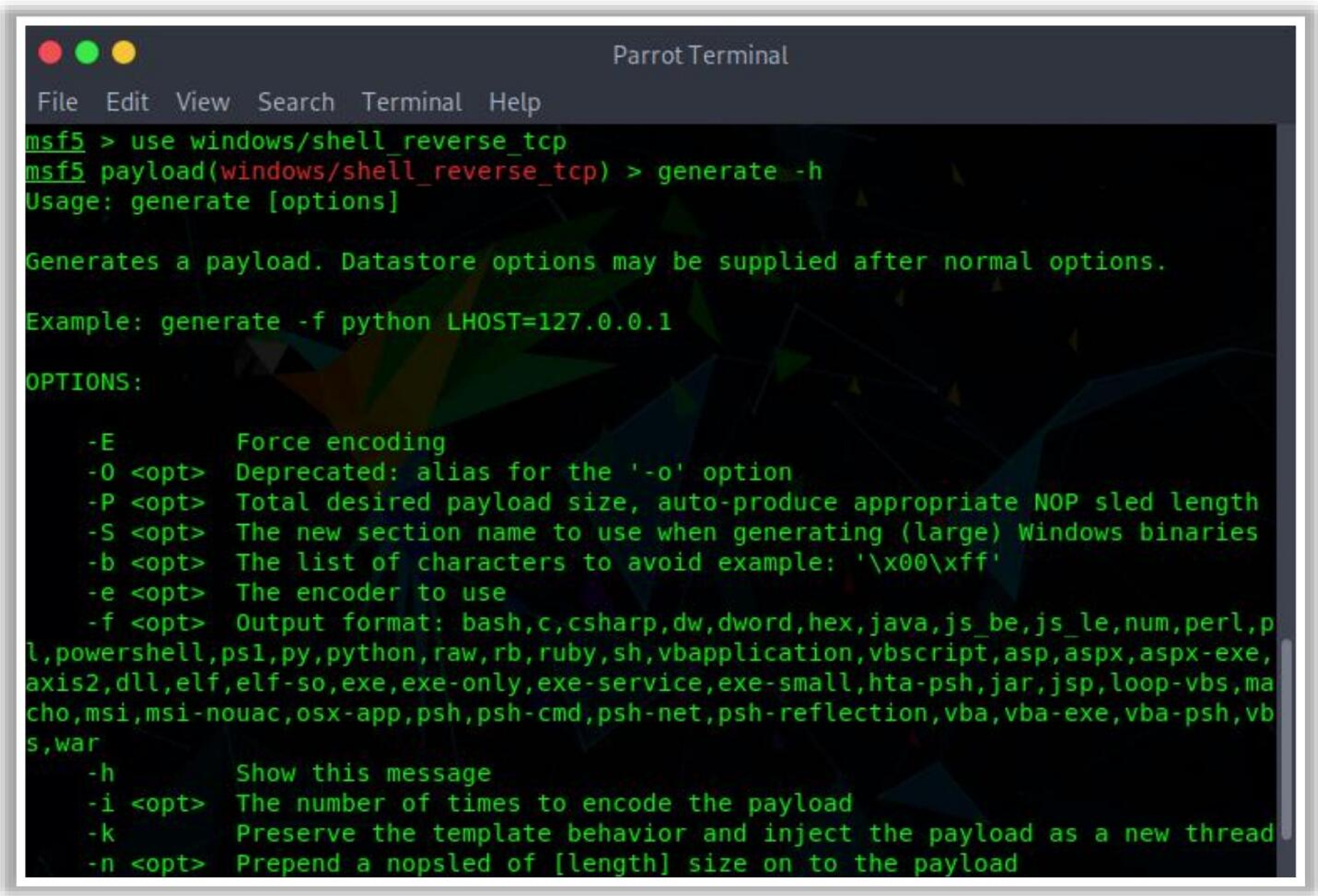


Figure 13.41: Screenshot displaying the Metasploit payload command

- **Metasploit Auxiliary Module**

  Auxiliary modules of Metasploit can be used to perform arbitrary, one-off actions such as port scanning, DoS, and even fuzzing. It includes tools and modules that assess the security of the target as well as auxiliary modules such as scanners, DoS modules, and fuzzers. The **`show auxiliary`** command in Metasploit can be used to list all the available auxiliary modules in Metasploit. All modules in Metasploit other than the ones used to exploit are auxiliary modules. Metasploit uses auxiliary modules as an extension for various purposes other than exploitation. Auxiliary modules are stored in the modules/auxiliary/ directory of the framework's main directory. The **`run`** command or the **`exploit`** command can be used to run an auxiliary module.

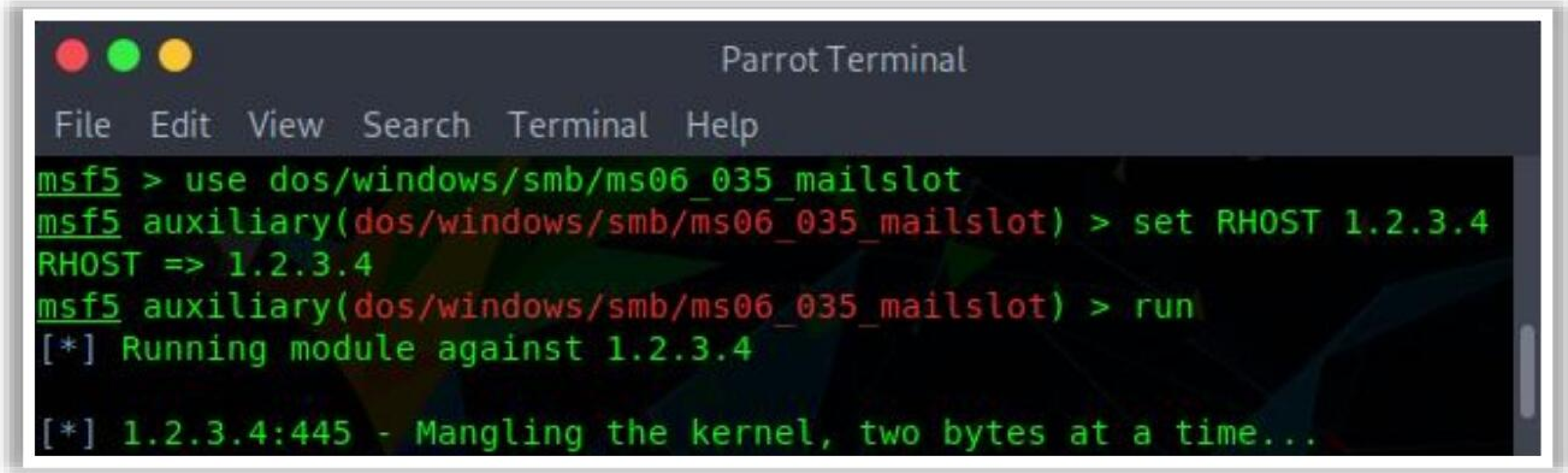


Figure 13.42: Screenshot displaying auxiliary module commands of Metasploit

The basic definition of an auxiliary module is as follows:

```
require 'msf/core'
p "My Auxiliary Module"
class Metasploit3 < Msf::Auxiliary
end         # for the class definition
```

- **Metasploit NOPS Module**

  NOP modules generate no-operation instructions used for blocking out buffers. The **`generate`** command can be used to generate a NOP sled of arbitrary size and display it in a given format.

  Options:

  -b <opt>: A list of characters to avoid ('\x00\xff')

  -h: Help banner

  -s <opt>: A comma separated list of registers to save

  -t <opt>: The output type (Ruby, Perl, C, or raw)

  msf nop(opty2)>

  The following command is used to generate a NOP sled of a given length:

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

  The following command is used 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) >
```

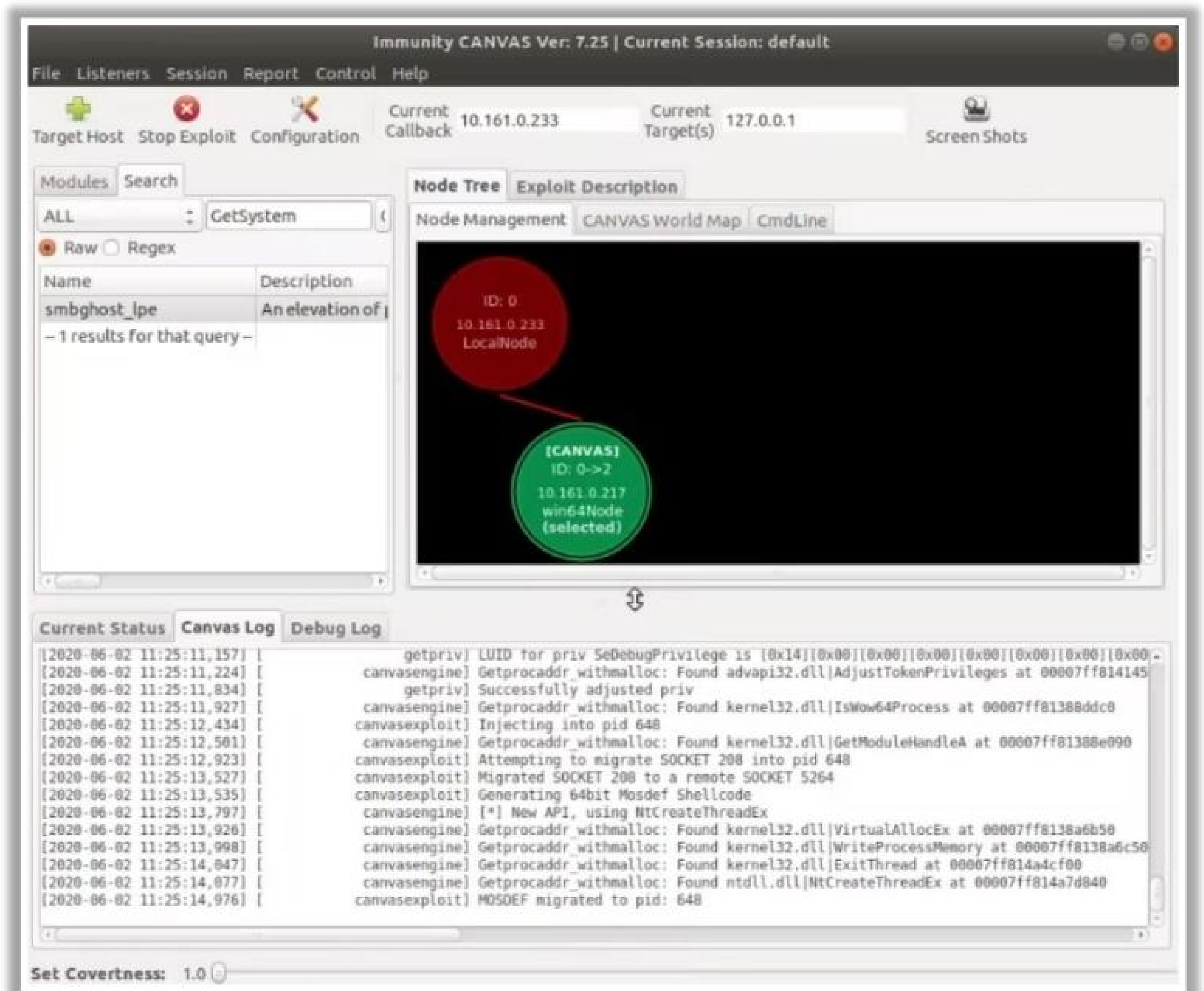
Web Server Attack Tools

Immunity's CANVAS

- Immunity's CANVAS provides hundreds of exploits, an **automated exploitation system**, and a comprehensive, reliable exploit development framework, to penetration testers and security professionals
- It provides features such as client-side exploitation, privilege escalation, **advanced web attack technology**, and remote kernel exploitation

Web Server Attack Tools

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



https://www.immunityinc.com

Web Server Attack Tools

Immunity's CANVAS

Source: *https://www.immunityinc.com*

Immunity's CANVAS provides penetration testers and security professionals with hundreds of exploits, an automated exploitation system, and a comprehensive, reliable exploit development framework. It provides features such as client-side exploitation, privilege escalation, HTTP tunneled privilege escalation, remote kernel exploitation, advanced backdoor technology, and advanced web attack technology.

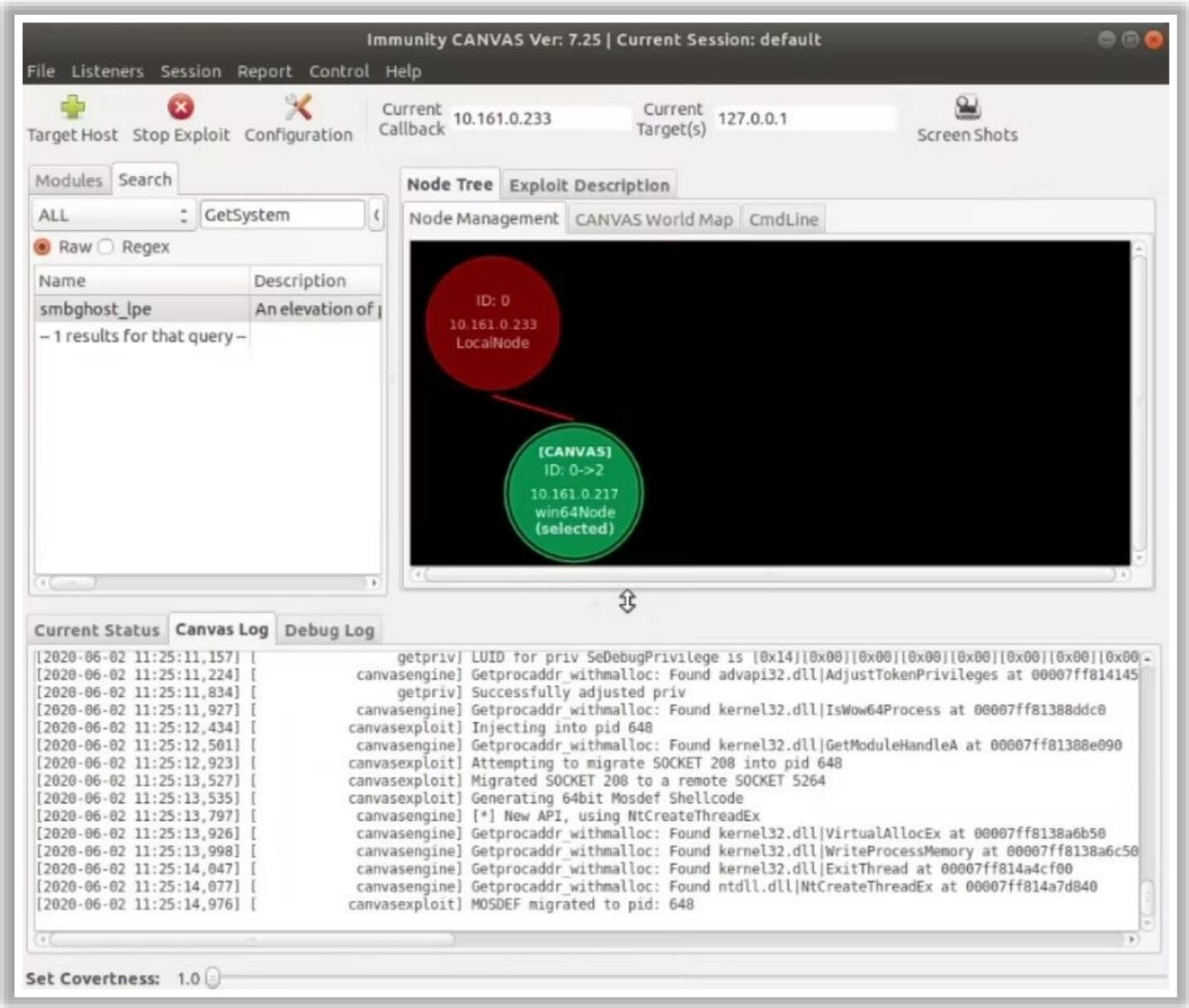


Figure 13.43: Screenshot of Immunity CANVAS

The following are some additional web server attack tools:

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

LO#04: Explain Web Server Attack Countermeasures

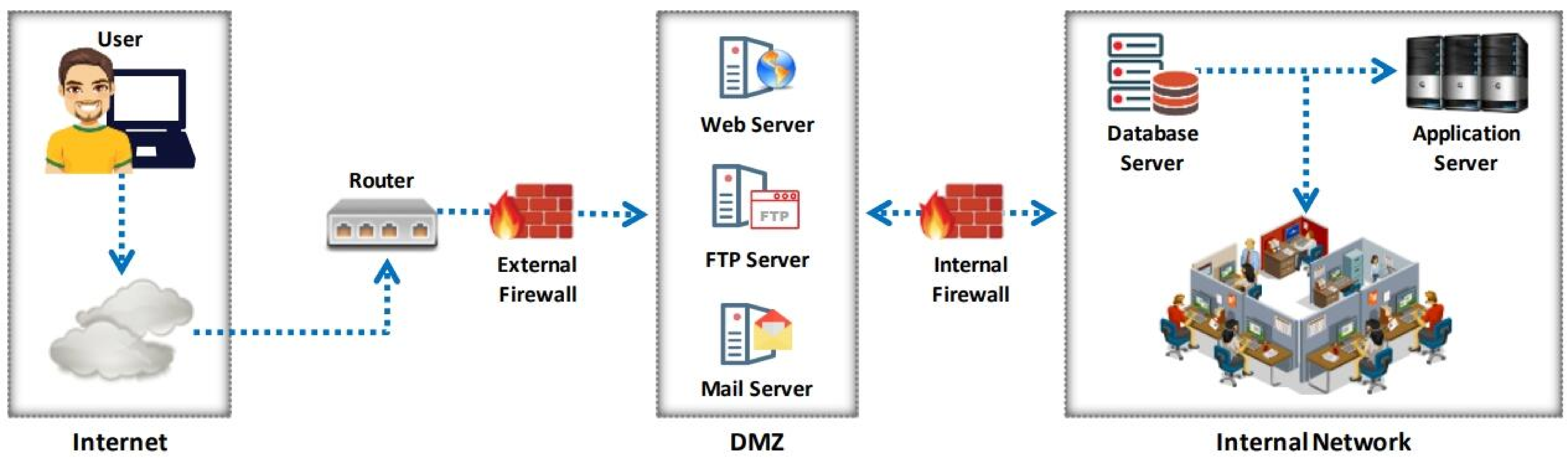
Web Server Attack Countermeasures

In previous sections, we discussed the benefits of a well-informed web server security posture, the danger posed by web server attacks, the methodology used in web server attacks, and the tools that assist an attacker in performing web server attacks. In this section, we discuss the tools and techniques used for securing web servers. This section also discusses various methods to detect web server attacks, countermeasures, and defense techniques.

Further, this section describes common security tools to secure a web server against possible attacks. These tools scan for vulnerabilities in a target server and web applications, send alerts in the case of hacking attempts, scan for malware in the web server, and perform other security assessment activities.

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, an Internet segment, a secure server security segment often called a demilitarized zone (DMZ), and an internal network
- The web server should be placed in the **Server Security Segment** (DMZ) of the network, isolated from the public and internal networks
- Firewalls should be placed for the **internal network** and **Internet traffic**, directed toward the DMZ



Place Web Servers in Separate Secure Server Security Segment on Network

An ideal web hosting network should be designed with three segments: an Internet segment; a secure server security segment, which is often called the demilitarized zone (DMZ); and an internal network. The first step in securing web servers is to place them separately in the DMZ, which is isolated from the public network and from the internal web-hosting network. Placing web servers in a separate segment adds security barriers between the web servers and the internal network as well as between the web servers and the outside public network. This separation allows the administrator to place firewalls and apply access control based on security rules for the internal network as well as for Internet traffic toward the DMZ. Such a web-hosting network can prevent attacks on the web server by outside attackers or malicious insiders.

Network segmentation divides a network into different segments, each having its own hub or switch. It allows network administrators to protect one segment from others by enforcing firewalls and security rules depending on the level of security desired. In a segmented network, an attacker who compromises one segment of the network will not be able to compromise the security of other segments of the network. Let us example a sample web-hosting network that is segmented by the administrator in such a manner that the web server is placed in a DMZ.

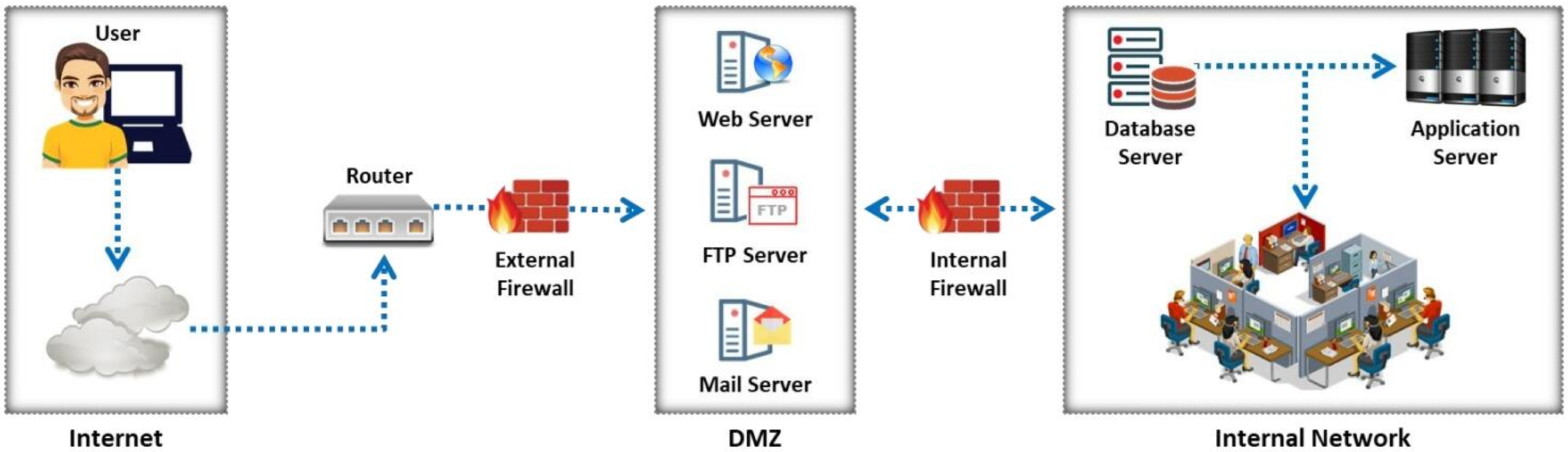
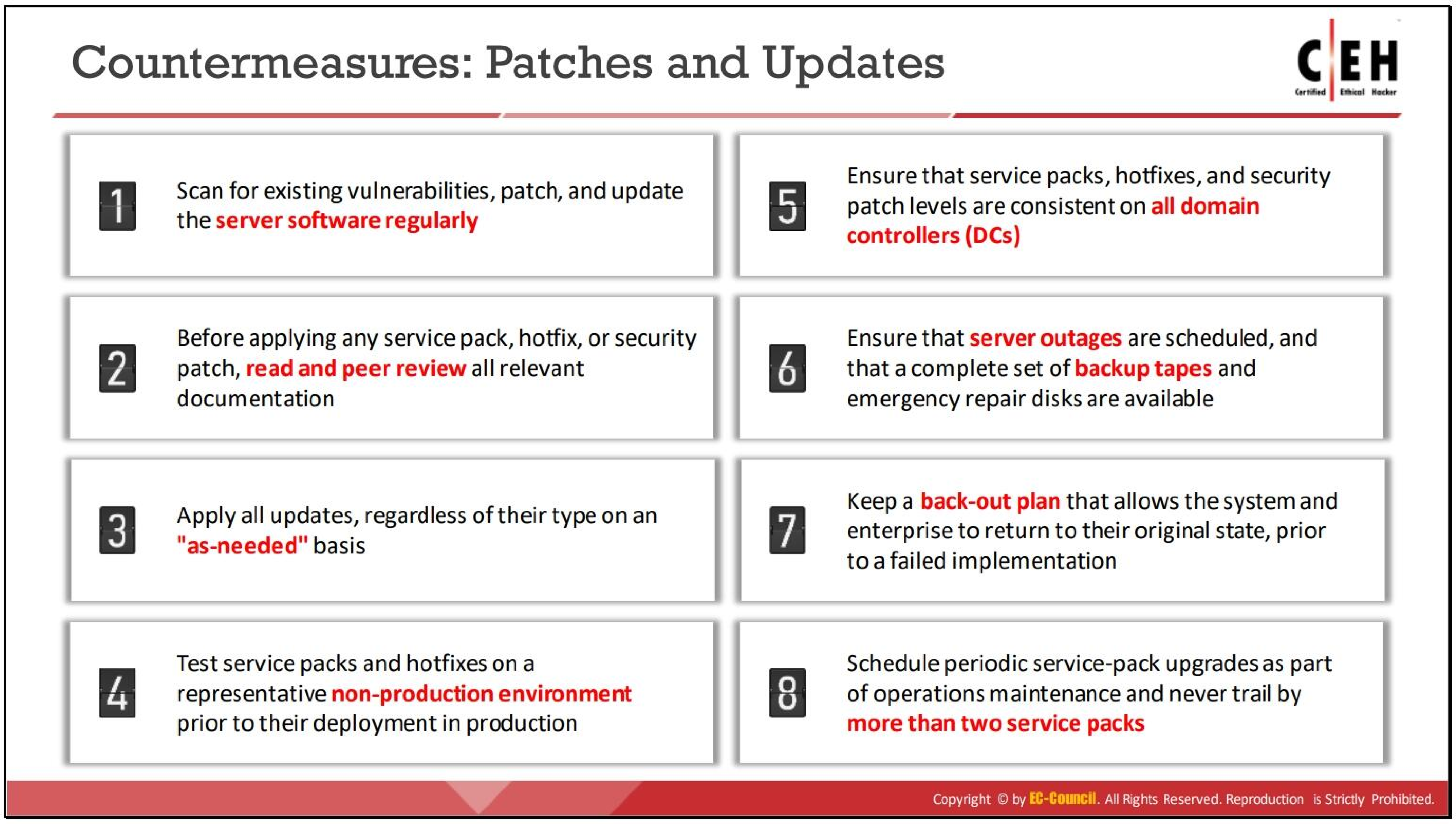


Figure 13.44: Illustration of three different segments in a web-hosting network



Countermeasures: Patches and Updates

The following are various countermeasures for secure update and patch management of web servers:

- Scan for existing vulnerabilities; patch and update the server software regularly.
- Before applying any service pack, hotfix, or security patch, read and peer review all relevant documentation.
- Apply all updates, regardless of their type, on an “as-needed” basis.
- Test service packs and hotfixes on a representative non-production environment prior to their deployment in production.
- Ensure that service packs, hotfixes, and security patch levels are consistent on all domain controllers (DCs).
- Ensure that server outages are scheduled and that a complete set of backup tapes and emergency repair disks are available.
- Keep a back-out plan that allows the system and enterprise to return to their original state, prior to a failed implementation.
- Schedule periodic service-pack upgrades as part of operations maintenance and never trail by more than two service packs.
- Disable all unused script extension mappings.
- Avoid using default configurations dispatched with web servers.

- Use virtual patches in the organization because they provide additional identification/logging capabilities.
- Establish a disaster recovery plan to handle patch management failures.
- Conduct an extensive risk assessment to determine which segments of the network are most vulnerable or at high risk that need to be patched first.
- Make a detailed inventory of all the endpoints, services, and dependencies.
- While deploying a patch to the full system, ensure that it is performed in testing environment first.
- Deploy an alerting system for patches.
- Use a patch management application or system such as SolarWinds Patch Manager to automate the procedure.
- Regularly conduct monitoring and reporting to ensure your patch management processes and updates are performing effectively.
- Reduce your exposure to third-party risks by limiting the number of software versions you employ.
- All patch and update operations should be validated and documented for accessibility, analysis, and confirmation.
- Make a standardized patch management and security update methodology as part of the SDLC.

Countermeasures: Protocols and Accounts

Protocols

- Block all unnecessary **ports, ICMP traffic**, and unnecessary protocols such as NetBIOS and SMB
- Harden the TCP/IP stack and consistently apply the **latest software patches** and updates to system software
- If insecure protocols such as **Telnet**, **POP3**, **SMTP**, and **FTP** are used, then take appropriate measures to provide secure authentication and communication, for example, by using IPsec policies
- If remote access is needed, ensure that remote connections are secured properly by using **tunneling and encryption protocols**
- Disable **WebDAV** if it is not used by the application or keep it secure if it is required

Accounts

- Remove all unused **modules and application extensions**
- **Disable unused default user accounts** created during the installation of an OS
- When creating a new web root directory, **grant appropriate (least possible) NTFS permissions** to anonymous users of the IIS web server to access the web content
- **Eliminate unnecessary database users** and stored procedures and follow the principle of least privilege for the database application to defend against SQL query poisoning
- Use secure web permissions, NTFS permissions, and **.NET Framework access control mechanisms** including URL authorization
- Slow down brute force and dictionary attacks with **strong password policies** and implement audits and alerts for login failures

Countermeasures: Protocols and Accounts

Countermeasures: Protocols

The following are various countermeasures for using secure protocols on web servers:

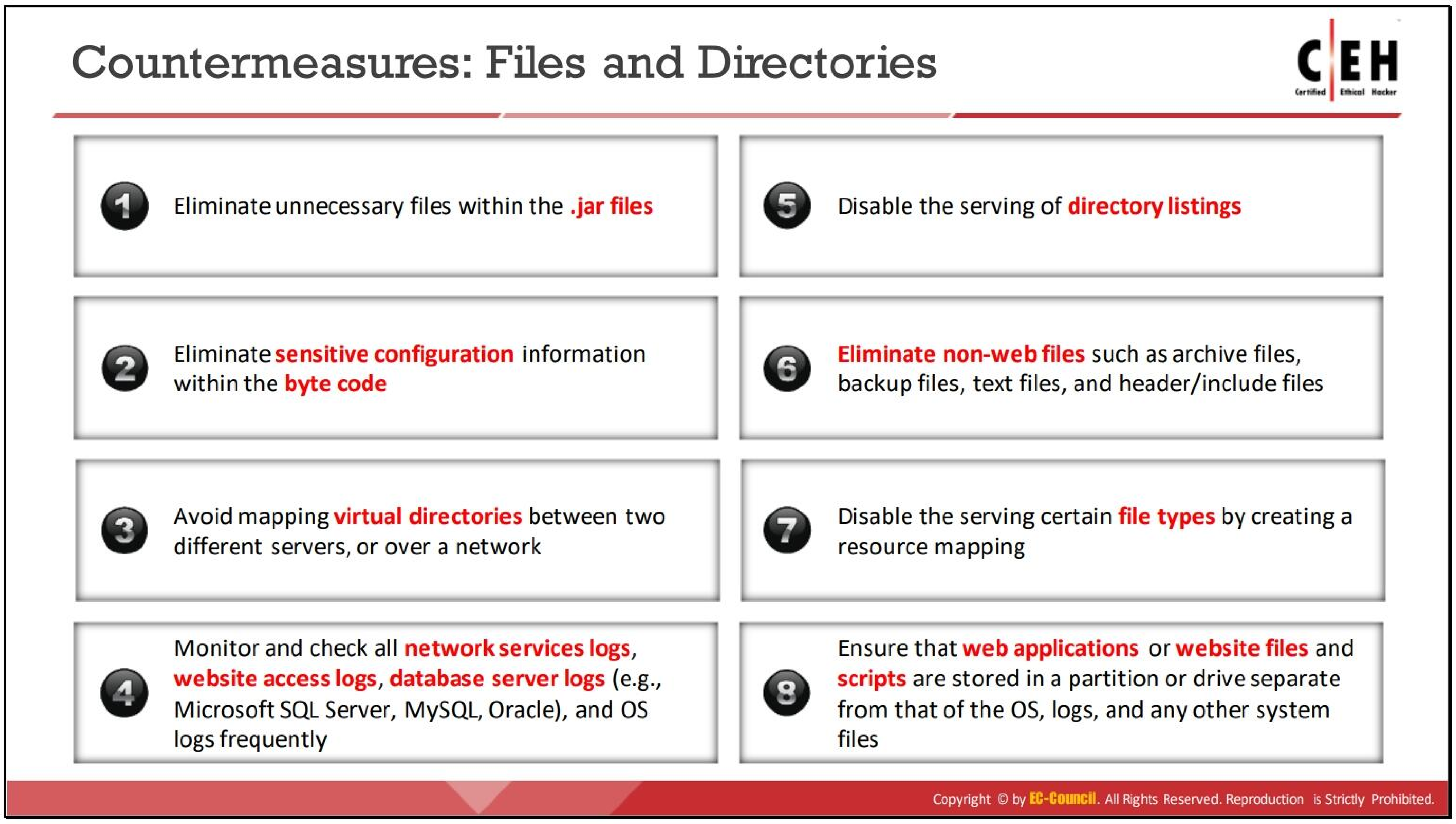
- Block all unnecessary ports, ICMP traffic, and unnecessary protocols such as NetBIOS and SMB.
- Harden the TCP/IP stack and consistently apply the latest software patches and updates to system software.
- If insecure protocols such as Telnet, POP3, SMTP, and FTP are used, then take appropriate measures to provide secure authentication and communication, for example, by using IPsec policies.
- If remote access is needed, ensure that remote connections are secured properly by using tunneling and encryption protocols.
- Disable Web Distributed Authoring and Versioning (WebDAV) if it is not used by the application, or keep it secure if it is required.
- Use secure protocols such as Transport Layer Security (TLS)/SSL for communicating with the web server.
- Ensure that unidentified FTP servers operate in an innocuous part of the directory tree that is different from the web server's tree.
- Ensure that the HTTP service banner is properly configured to cover the details of the host device like OS version and type.

- Isolate the supporting servers such as LDAP servers from the local subnet to filter out the traffic through a firewall before entering the local network.
- Ensure that all the file transfer applications through the web server are done through FTPS for better data encryption and protection.

Countermeasures: Accounts

The following countermeasures can be adopted to secure user accounts on a web server:

- Remove all unused modules and application extensions.
- Disable unused default user accounts created during the installation of an OS.
- When creating a new web root directory, grant the appropriate (least possible) NTFS permissions to anonymous users of the IIS web server to access the web content.
- Eliminate unnecessary database users and stored procedures and follow the principle of least privilege for the database application to defend against SQL query poisoning.
- Use secure web permissions, NTFS permissions, and .NET Framework access control mechanisms, including URL authorization.
- Slow down brute-force and dictionary attacks with strong password policies and implement audits and alerts for login failures.
- Run processes using least privileged accounts as well as least privileged services and user accounts.
- Limit the administrator or root-level access to the minimum number of users and maintain a record of the same.
- Maintain logs of all user activity in an encrypted form on the web server or in a separate machine on the intranet.
- Disable all noninteractive accounts that should exist but do not require an interactive login.
- Use secure VPN networks such as OpenVPN while accessing multi-server platforms or accessing data from cross-server network models, which helps use one account for multiple server access.
- Use password managers such as KeePass to maintain a proper password policy for multiple user accounts.
- Enable the Separation of Duties (SoD) feature on the server config settings.
- Force users to periodically change passwords for their accounts by creating a password expiry policy.
- Enable the user account locking feature by setting a limit on the number of failed login attempts.
- Implement 2FA or MFA as an additional layer of security for user accounts.



Countermeasures: Files and Directories

The following countermeasures can be adopted for securing files and directories on a web server:

- Eliminate unnecessary files within .jar files.
- Eliminate sensitive configuration information within the byte code.
- Avoid mapping virtual directories between two different servers or over a network.
- Monitor and check all network services logs, website access logs, database server logs (e.g., Microsoft SQL Server, MySQL, and Oracle), and OS logs frequently.
- Disable the serving of directory listings.
- Eliminate non-web files such as archive files, backup files, text files, and header/include files.
- Disable the serving of certain file types by creating a resource map.
- Ensure that web applications or website files and scripts are stored in a partition or drive separate from that of the OS, logs, and any other system files.
- Run the web server within a sandbox directory for preventing access to system files.
- Avoid all non-web file types from being referenced in a URL.
- Run the web server processes with the least required privileges and give access only to the necessary directories.
- Exclude meta characters while processing user inputs to ensure proper filtering of inputs.
- Employ file integrity checkers to verify web content and intrusion detection.

Detecting Web Server Hacking Attempts

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

Website Change Detection System involves:

1. **Running specific script** on the server that detects any changes made in the existing executable file or new file included on the server
2. Periodically comparing the **hash values** of the files on the server with their respective master hash value to detect the changes made in codebase
3. **Alerting the user** upon any change detected on the server

For example: **DirectoryMonitor** is an automated tool that goes through all your web folders and detects any changes made to your codebase and alerts you via an email, if changes are detected

Detecting Web Server Hacking Attempts

An attacker who gains access to a web server by compromising security through known vulnerabilities present in the web server may attempt to plant backdoors (scripts). These backdoors allow the attacker to gain access, launch phishing attacks, or send spam emails. The victim remains unaware of the web server attack until the server is blacklisted on spam mails or until the attacker redirects the visitors of a target site hosted on the web server to some other site. Thus, a web server attack is difficult to detect unless such malicious events occur. By the time these events occur, it may be too late to react because the attacker would have already succeeded. Therefore, a mechanism to detect a web server hacking attempt in its early stages is required to prevent harm to the web server.

When an attacker installs a backdoor on a web server, the size of files infected with the backdoor automatically increases. A website change detection system (WDS) is a script that runs on the server to detect changes made to any executable file or the presence of any new file on the web server, such as HTML, JavaScript (JS), PHP, Active Server Pages (ASP), Perl, and Python files. It works by periodically comparing the hash values of the files on the server with their respective master hash values to detect any changes to the codebase. If it detects any change on the server, it alerts the user to take necessary action. Thus, WDS helps in detecting web server hacking attempts in the early stages of an attack. For example, DirectoryMonitor is an automated tool that goes through entire web folders, detects any changes made to the codebase, and alerts the user through an email.

How to Defend Against Web Server Attacks

1 Ports

- Regularly audit the ports on the server 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**

2 Server Certificates

- Ensure that **certificate data ranges** are valid and that the certificates are used for their intended purpose
- Ensure that no certificate has been revoked and the **certificate's public key** is valid all the way to a trusted root authority

3 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

4 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.
   - Apply **restricted ACLs** and block remote registry administration
   - Secure the **SAM** (Stand-alone servers only)
2. Ensure that security-related settings are **configured appropriately** and that access to the metabase file is restricted with hardened **NTFS permissions**
3. **Remove** unnecessary ISAPI filters from the web server
4.
   - Remove all unnecessary file shares, including the **default administration shares**, if they are not required
   - Secure the shares with restricted **NTFS permissions**
5. Relocate sites and virtual directories to **non-system partitions** and use IIS web permissions to restrict access
6. Remove all unnecessary **IIS script mappings** for optional file extensions to avoid exploitation of any bugs in the ISAPI extensions that handle these types of files
7. Enable a **minimum level of auditing** on the web server and use NTFS permissions to protect log files

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

8. Use a **dedicated machine** as a web server
9. Create **URL mappings** to internal servers cautiously
10. Do not install the **IIS server** on a domain controller
11. Use server-side **session ID tracking** and match connections with timestamps, IP addresses, etc.
12. If a database server such as **Microsoft SQL Server** is to be used as a backend database, install it on a **separate server**
13. Use security tools provided with **web server software** and **scanners** that automate and simplify the process of securing a web server
14. Physically protect the **web server machine** in a secure machine room
15. Do not connect an IIS Server to the **Internet** until it is fully hardened
16. Do not allow anyone to **locally log in** to the machine except the administrator
17. Configure a **separate anonymous user account** for each application if multiple web applications are hosted
18. Limit the **server functionality** to support only the web technologies to be used
19. Screen and filter **incoming traffic requests**

How to Defend Against Web Server Attacks

Defenses against web server attacks include the following.

- **Ports**

  Monitor all ports on the web server regularly to prevent unnecessary traffic toward the target web server. If traffic is not monitored, the target web server will be vulnerable to malware attacks. Do not allow public access to port 80 for HTTP or to port 443 for HTTPS; traffic to these ports should be limited. If port 80 is kept open, the server will be vulnerable to DoS attacks, which consume server resources. Intranet traffic should either be encrypted or restricted to secure the web server.

  Attackers attempt to hide their identity by spoofing the IP address of a legitimate user. By processing the security log file, either using the "deny this IP address" rule in the firewall ruleset file or by creating a "routed blackhole" command, the target system can defend against web server attacks.

- **Server Certificates**

  Server certificates guarantee security and are signed by a trusted authority. However, an attacker may compromise certified servers using forged certificates to intercept secure communications by performing MITM attacks. There are various techniques to avoid such MITM attacks. The following are some of them.

  - Use the direct validation of certificates.
  - Use a novel protocol that does not depend on third parties for certificate validation.
  - Allow domains to directly and securely examine their certificates by using previously established user authentication credentials.

- Use a robust cryptographic construction that enhances server identity validation and resolves the limitations of third-party solutions.
- Ensure that the certificate data ranges are valid and that certificates are used for their intended purpose.
- Ensure that the certificate has not been revoked and that the certificate's public key is valid all the way to a trusted root authority.

- **Machine.config**

  The machine.config file provides a mechanism of securing information by changing machine-level settings. It affects all other applications. The machine.config file includes machine settings for the .Net framework, which affect the security. The following can be performed with the machine.config file:

  - Ensure that protected resources are mapped to HttpForbiddenHandler and that unused HttpModules are removed
  - Ensure that tracing is disabled <trace enable="false"/> and debug compiles are turned off
  - Verify that ASP.NET errors are not reverted to the client
  - Verify session state settings

- **Code Access Security**

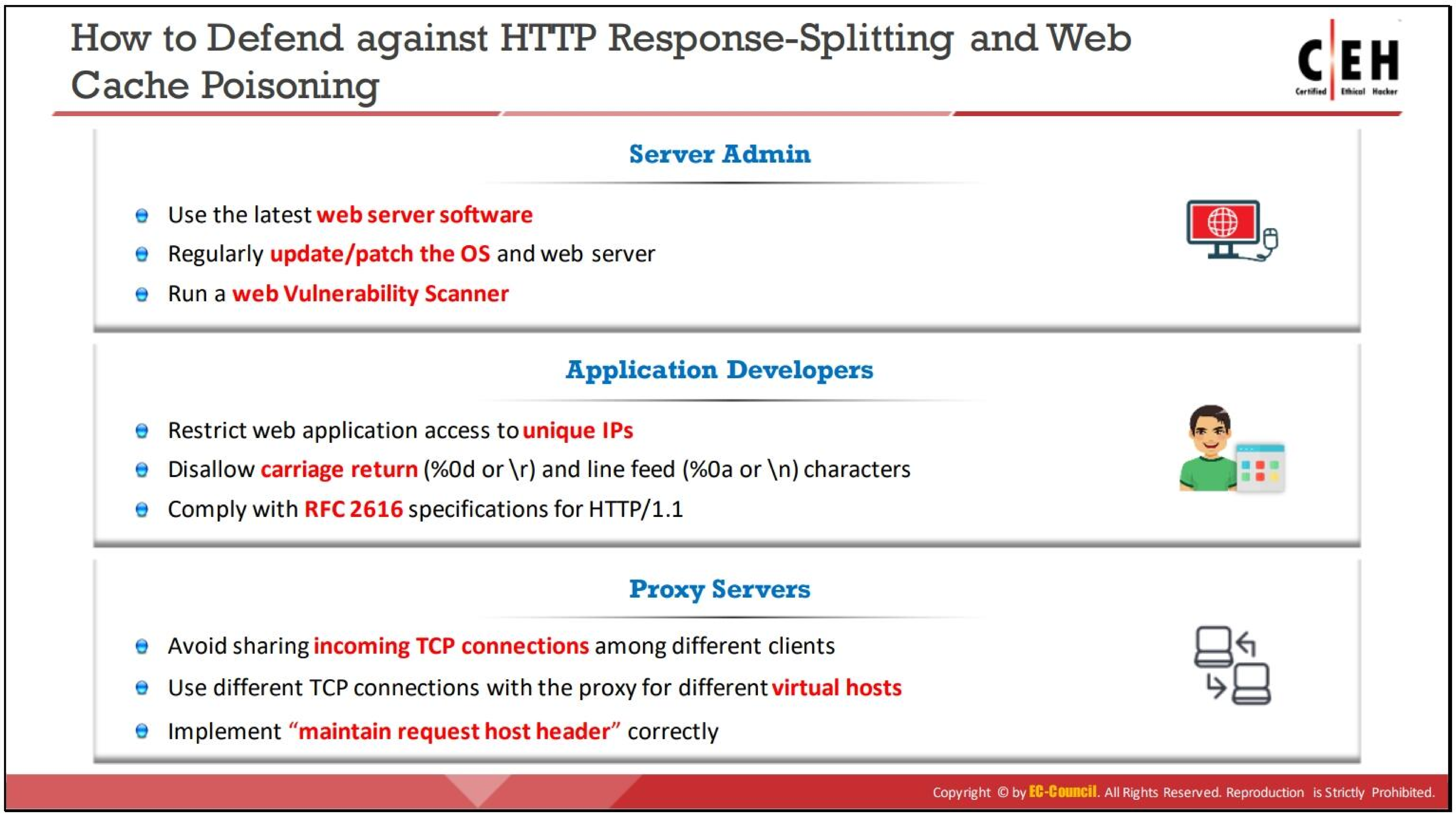
  The following measures can be adopted to ensure code access security.

  - Implement secure coding practices to avoid source-code disclosure and input validation attacks.
  - Restrict code access security policy settings to ensure that there are no permissions to execute code downloaded from the Internet or intranet.
  - Configure IIS to reject URLs with "../" to prevent path traversal, lockdown system commands and utilities with restrictive access control lists (ACLs), and install new patches and updates.
  - If targets do not implement code access security in their web servers, then there is a possibility of execution of malicious code.

The following are some other measures to defend against web server attacks:

- Apply restricted ACLs and block remote registry administration.
- Secure the SAM (stand-alone servers only).
- Ensure that security-related settings are configured appropriately and that access to the metabase file is restricted with hardened NTFS permissions.
- Remove unnecessary Internet Server Application Programming Interface (ISAPI) filters from the web server.

- Remove all unnecessary file shares, including the default administration shares, if they are not required.
- Secure the shares with restricted NTFS permissions.
- Relocate sites and virtual directories to non-system partitions and use IIS web permissions to restrict access.
- Remove all unnecessary IIS script mappings for optional file extensions to avoid exploitation of any bugs in the ISAPI extensions that handle these types of files.
- Enable a minimum level of auditing on the web server and use NTFS permissions to protect log files.
- 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 timestamps, 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 simplify the process of securing a web server.
- 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 in to the machine except the administrator.
- Configure a separate anonymous user account for each application if multiple web applications are hosted.
- Limit the server functionality to support only the web technologies to be used.
- Screen and filter incoming traffic requests.
- Store website files and scripts on a separate partition or drive.
- Use an effective anti-bot mitigation service such as DataDome to detect botnets in real time.



How to Defend against HTTP Response-Splitting and Web Cache Poisoning

While setting cookies, remove carriage returns (CRs) and linefeeds (LFs) before inserting data into an HTTP response header. The best practice is to use third-party products to test for the existence of security holes and defend against CRLF injection. Ensure that data application engines are up to date.

The User Datagram Protocol (UDP) source port randomization technique defends servers against blind response forgery. Limit the number of simultaneous recursive queries and increase the times-to-live (TTLs) of legitimate records.

The following are some methods to defend against HTTP response-splitting attacks and web cache poisoning:

- **Server Admin**
  - Use the latest web server software
  - Regularly update/patch the OS and web server
  - Run a web vulnerability scanner
- **Application Developers**
  - Restrict the web application's access to unique IPs
  - Disallow CR (%0d or \r) and LF (%0a or \n) characters
  - Comply with RFC 2616 specifications for HTTP/1.1
  - Parse all user inputs or other forms of encoding before using them in HTTP headers

- **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

1. Choose an **ICANN** accredited **registrar** and encourage them to set **Registrar-Lock** on the domain name
2. Safeguard the **registrant's account information**
3. Include DNS hijacking in **incident response and business continuity planning**
4. Use DNS monitoring tools/services to **monitor the IP address of the DNS server** and set alerts
5. Avoid downloading **audio and video codecs** and other downloaders from untrusted websites
6. Install an **antivirus** program and update it regularly
7. Change the **default router password** that comes with the factory settings

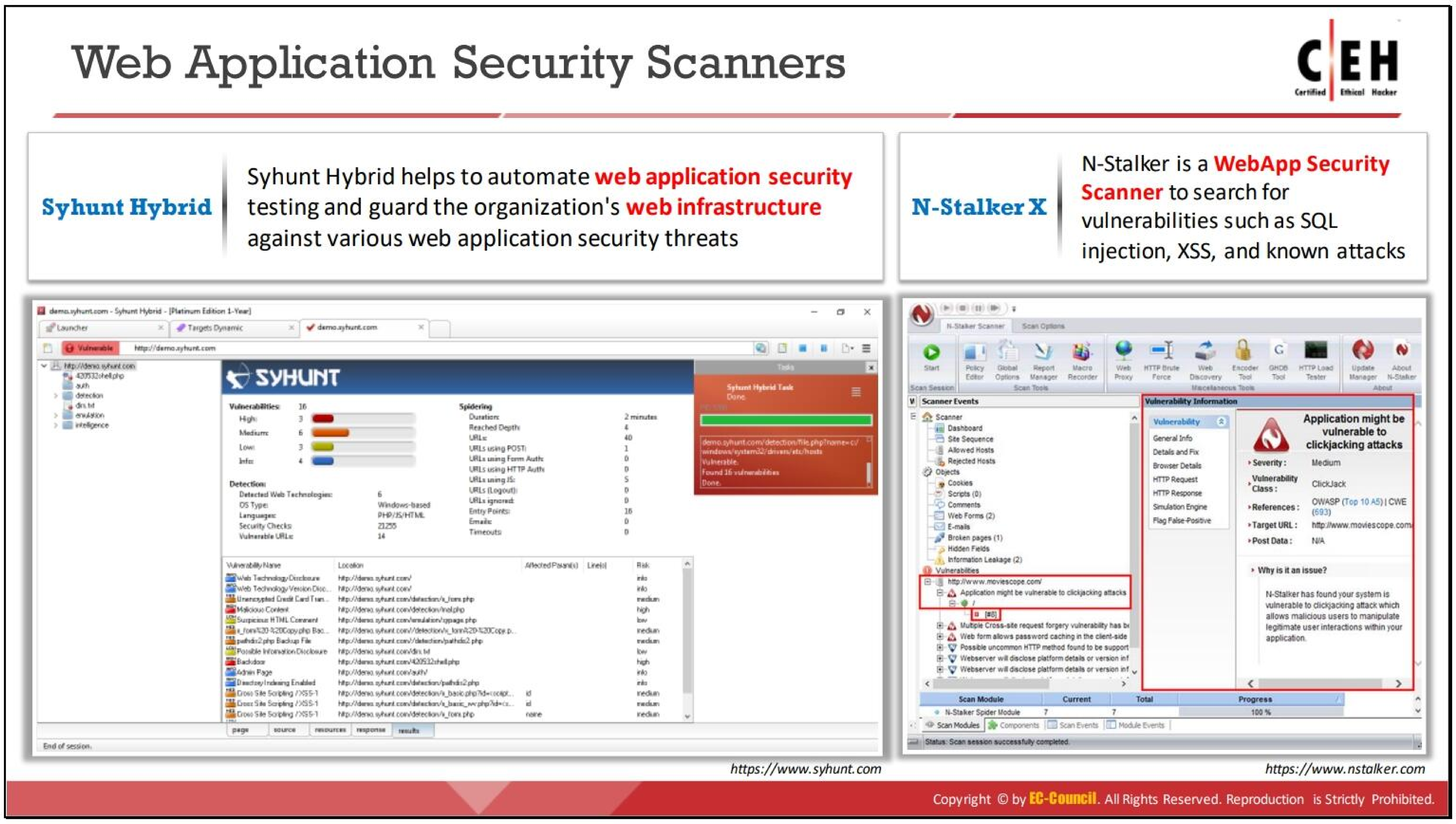
How to Defend against DNS Hijacking

The following techniques can be used to defend against DNS hijacking:

- Choose a registrar accredited by the Internet Corporation for Assigned Names and Numbers (ICANN) and encourage them to set REGISTRAR-LOCK on the domain name.
- Safeguard the registrant's account information.
- Include DNS hijacking in incident response and business continuity planning.
- Use DNS monitoring tools/services to monitor the IP address of the DNS server and set alerts.
- Avoid downloading audio and video codecs and other downloaders from untrusted websites.
- Install an antivirus program and update it regularly.
- Change the default router password.
- Restrict zone transfers and use script blockers in the browser.
- **Domain Name System Security Extensions (DNSSEC)**: It adds an extra layer to DNS that prevents it from being hacked.
- **Strong password policies and user management**: The use of strong passwords further enhances security.
- **Better service-level agreements (SLAs) from DNS service providers**: When signing up for DNS servers with DNS service providers, learn who to contact when an issue occurs,

how to receive good-quality reception and support, and whether the DNS server's infrastructure is hardened against attacks.

- **Configuring a master–slave DNS within your network**: Use a master–slave DNS and configure the master without Internet access. Maintain two slave servers so that even if an attacker hacks a slave, it will update only when it receives an update from the master.
- **Constant monitoring of DNS servers**: The constant monitoring of DNS servers ensures that a domain name returns the correct IP address.
- **Ensure router safety**: Change the default username and password of the router. Keep the firmware up-to-date for ensuring safety from new vulnerabilities.
- **Use VPN service**: Establish virtual private network (VPN)-encrypted tunnels for secure private communication over the Internet. This feature protects messages from eavesdropping and unauthorized access.
- Use firewall protection services to safeguard the original DNS resolvers and filter out the rogue DNS resolver traffic.
- Maintain proper protection systems such as MFA and hardware security to ensure controlled access to the DNS server.
- Install script blocker extensions in the browser.
- Use only secured and reputed VPN networks instead of free VPN services, which can track your activities and record them for future use.



Web Application Security Scanners

- **Syhunt Hybrid**

  Source: *https://www.syhunt.com*

  The Syhunt Hybrid scanner automates web application security testing and guards the organization's web infrastructure against web application security threats. Syhunt Dynamic crawls websites and detects XSS, directory transversal problems, fault injection, SQL injection, attempts to execute commands, and several other attacks. Syhunt Hybrid creates signatures to detect application vulnerabilities and prevents logout. It analyzes JavaScript (JS), logs suspicious responses, and tests errors for review.

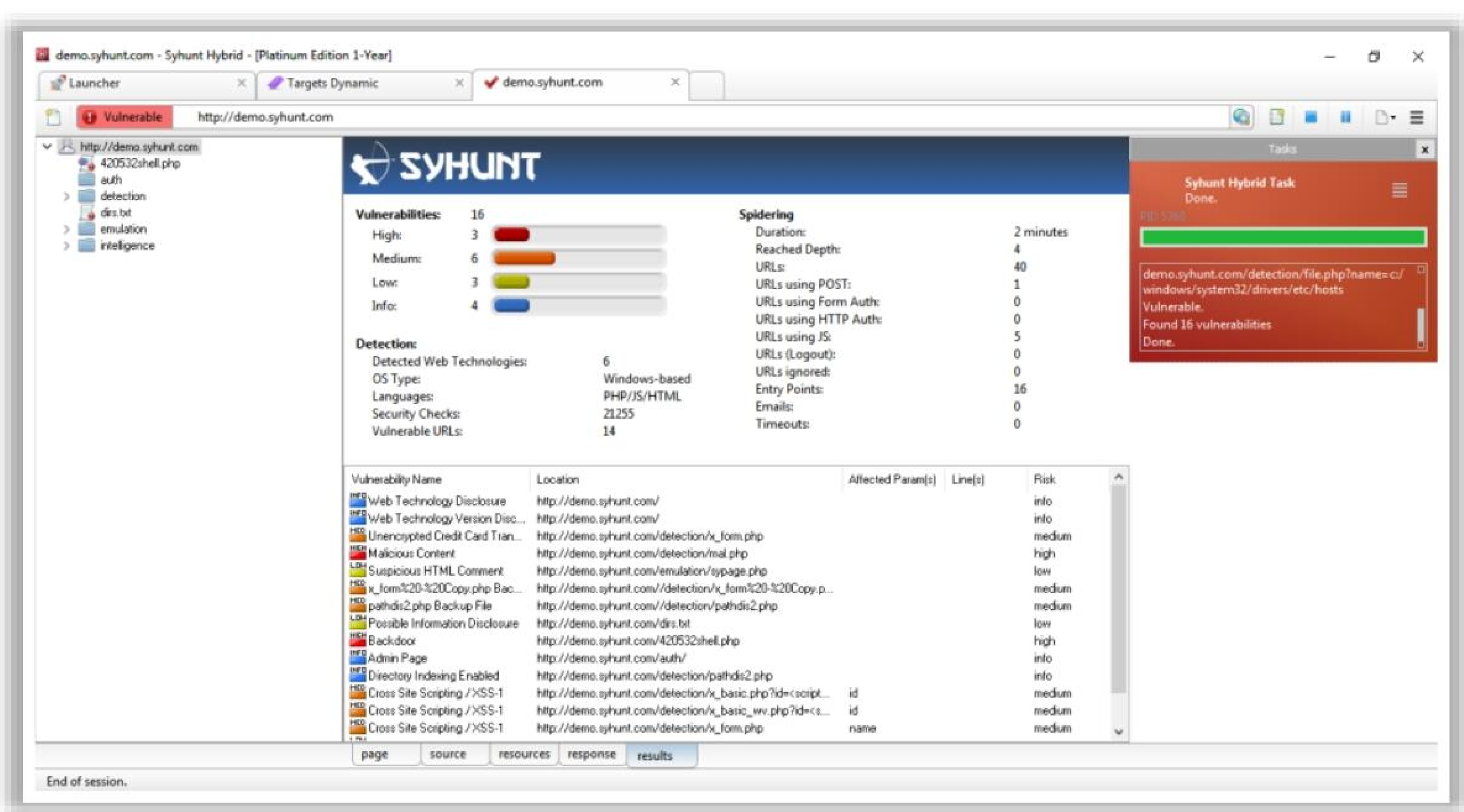


Figure 13.45: Screenshot of Syhunt Hybrid web application security scanner

- **N-Stalker X**

  Source: *https://www.nstalker.com*

  N-Stalker is a web application security scanner that searches for vulnerabilities to attacks such as clickjacking, SQL injection, and XSS. It allows spider crawling throughout the application and the creation of web macros for form authentication. It also provides proxy capabilities for “drive-thru” attacks and identifies components through reverse proxies that distribute different platforms in the same application URL.

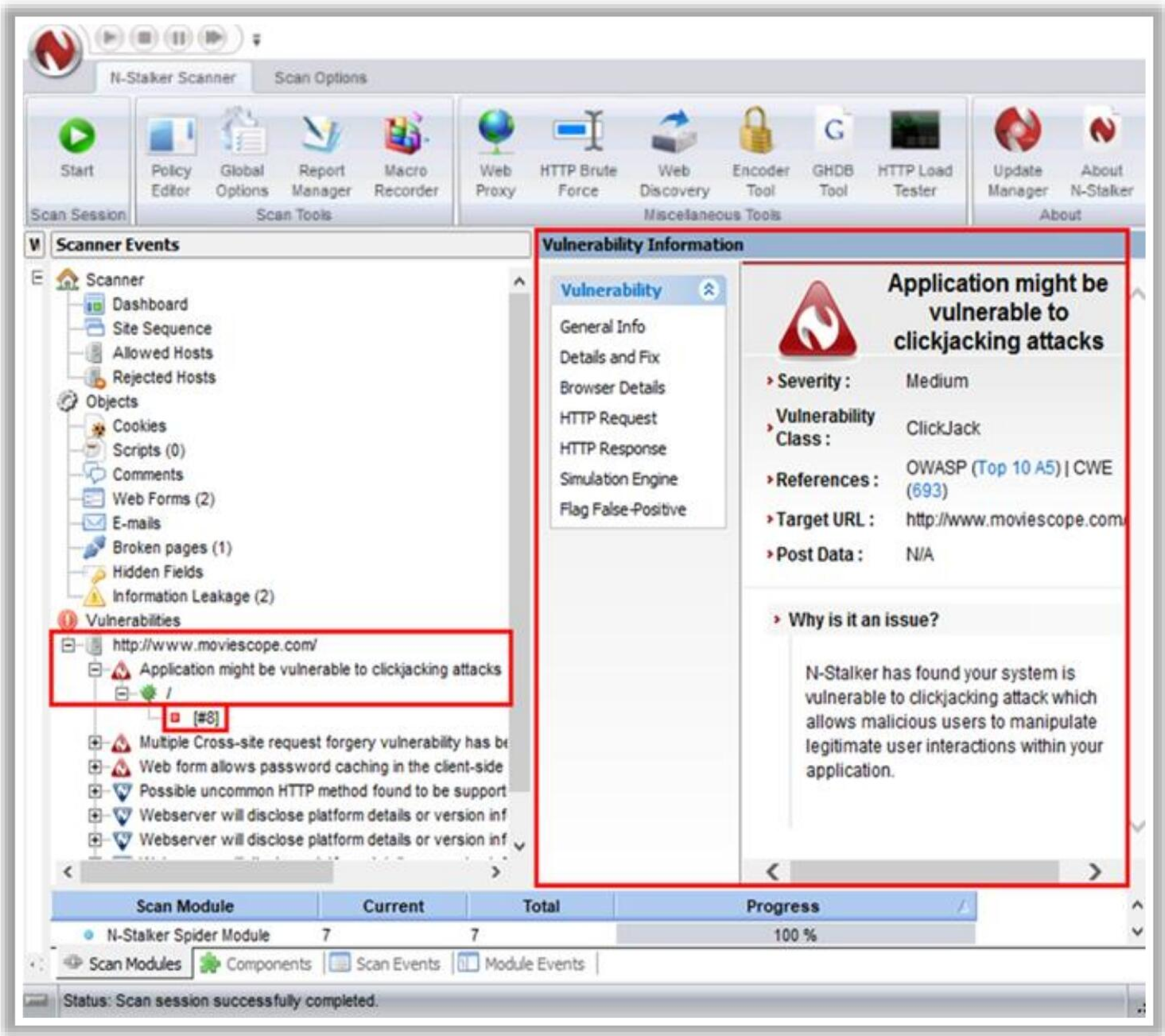
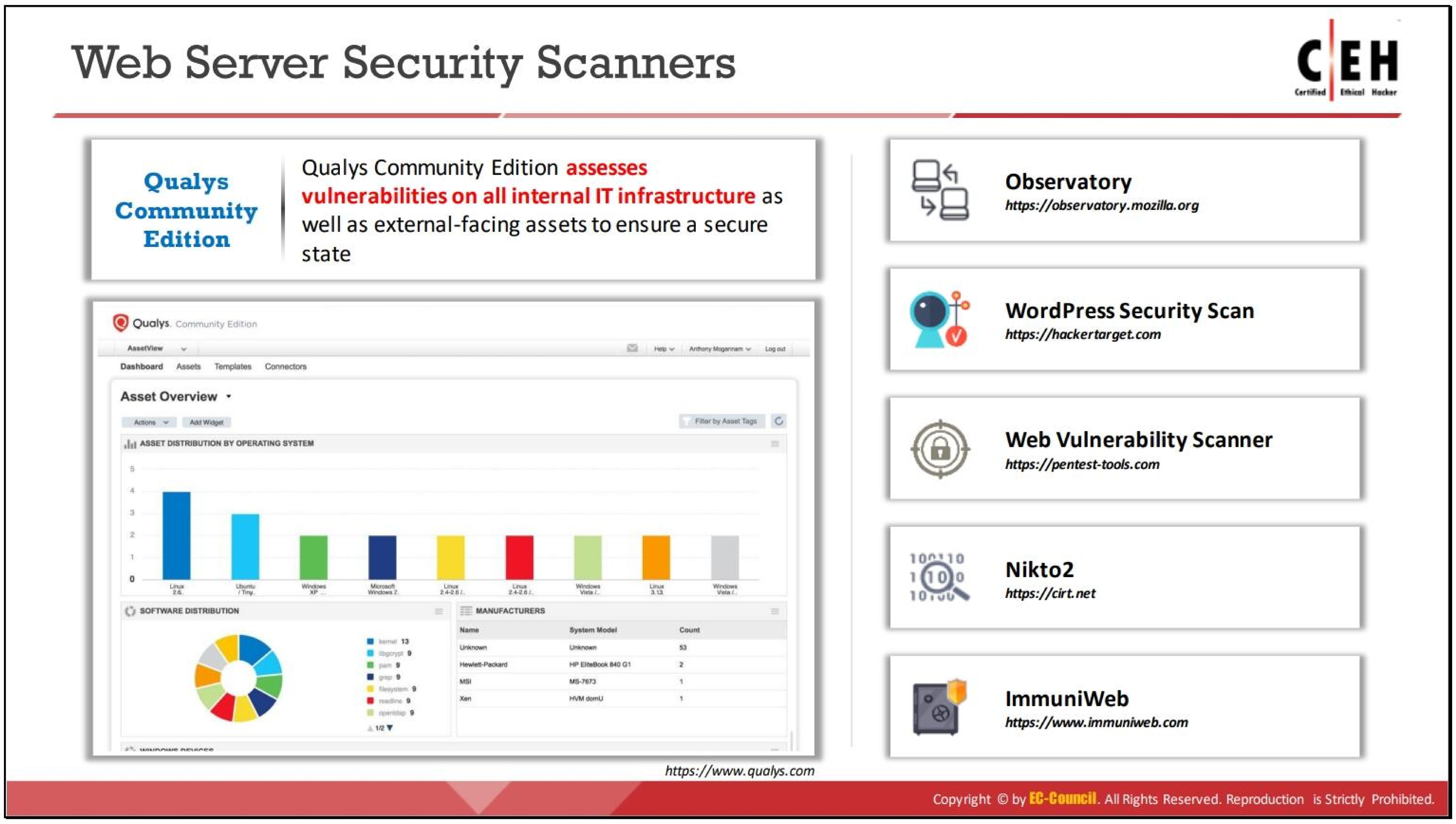


Figure 13.46: Screenshot of N-Stalker X

The following are some additional web application security scanners:

- Invicti (*https://www.invicti.com*)
- Burp Suite (*https://www.portswigger.net*)
- Wapiti (*https://wapiti-scanner.github.io*)
- WebScarab (*https://www.owasp.org*)

- WPSec (*https://wpsec.com*)
- Tinfoil Security (*https://www.tinfoilsecurity.com*)
- Skipfish (*https://code.google.com*)
- Detectify (*https://detectify.com*)
- Fortify on Demand (*https://www.microfocus.com*)
- OWASP Zed Attack Proxy (ZAP) (*https://www.zaproxy.org*)
- SonarQube (*https://www.sonarqube.org*)
- Arachni (*https://www.arachni-scanner.com*)
- w3af (*https://w3af.org*)
- Grabber (*http://rgaucher.info*)
- Vega (*https://subgraph.com*)



Web Server Security Scanners

- **Qualys Community Edition**

  Source: *https://www.qualys.com*

  Qualys Community Edition discovers IT assets, manages vulnerabilities, scans web applications, and maintains cloud assets inventory. It offers vulnerability management to identify dangerous bugs and remediate them immediately. Qualys can also assess vulnerabilities on all internal IT infrastructure as well as external-facing assets to ensure a secure state.

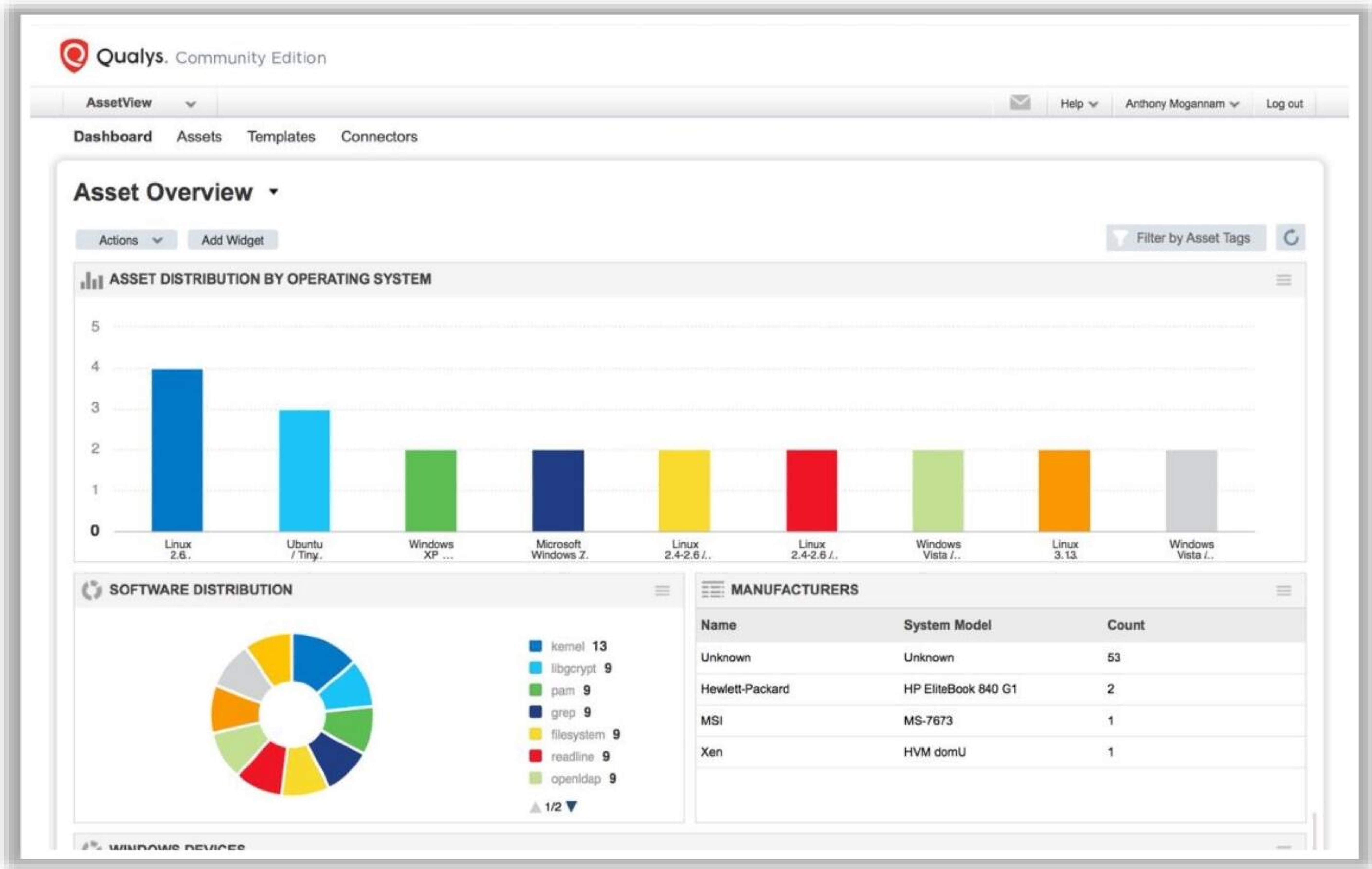
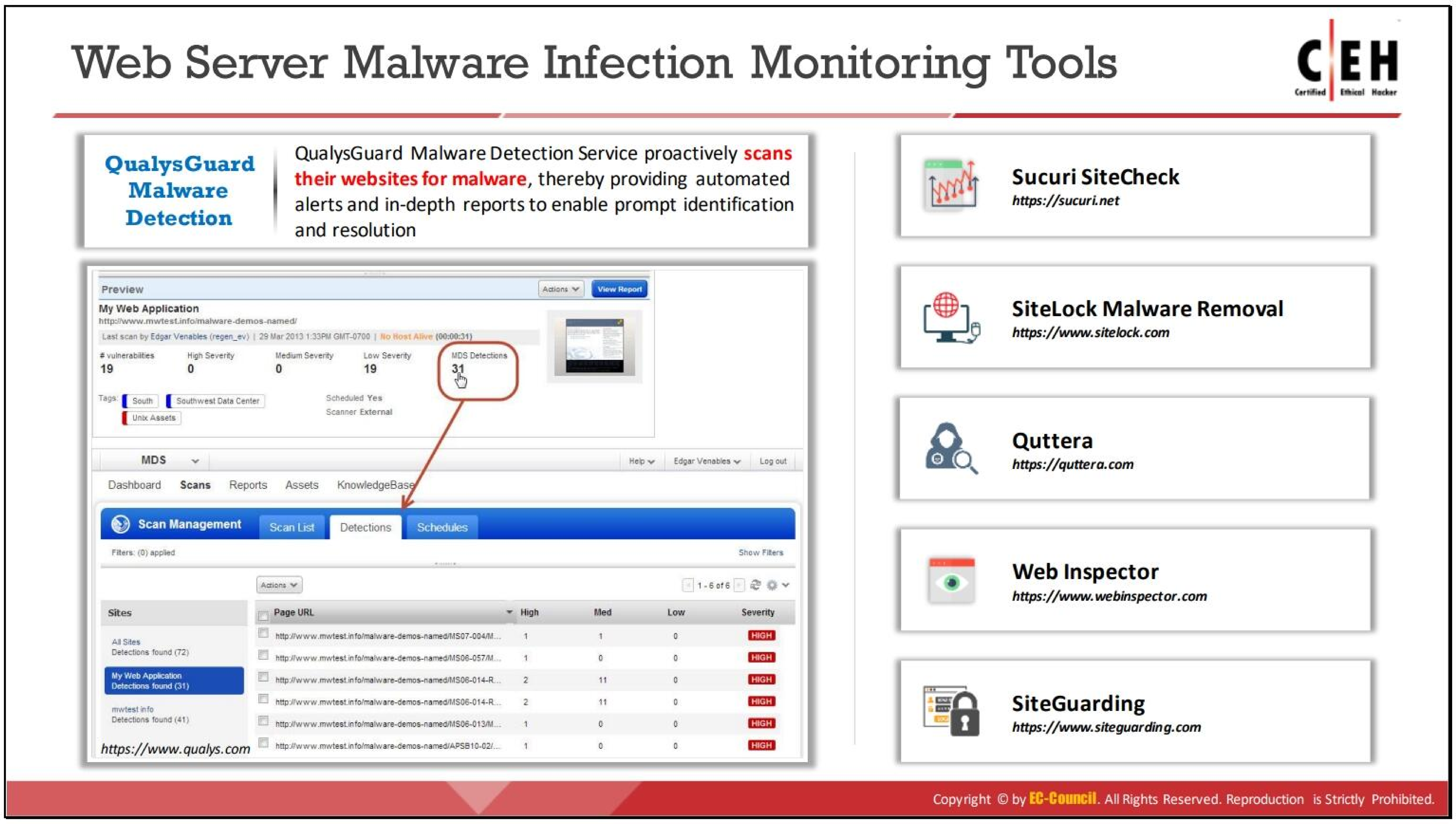


Figure 13.47: Screenshot of Qualys Community Edition

The following are some additional web server security scanners:

- Observatory (*https://observatory.mozilla.org*)
- WordPress Security Scan (*https://hackertarget.com*)
- Web Vulnerability Scanner (*https://pentest-tools.com*)
- Nikto2 (*https://cirt.net*)
- ImmuniWeb (*https://www.immuniweb.com*)



Web Server Malware Infection Monitoring Tools

- **QualysGuard Malware Detection**

  Source: *https://www.qualys.com*

  QualysGuard Malware Detection allows organizations to proactively scan their websites for malware and provides automated alerts and in-depth reporting to enable prompt identification and resolution. It enables organizations to protect their customers from malware infections and safeguard their brand reputation.

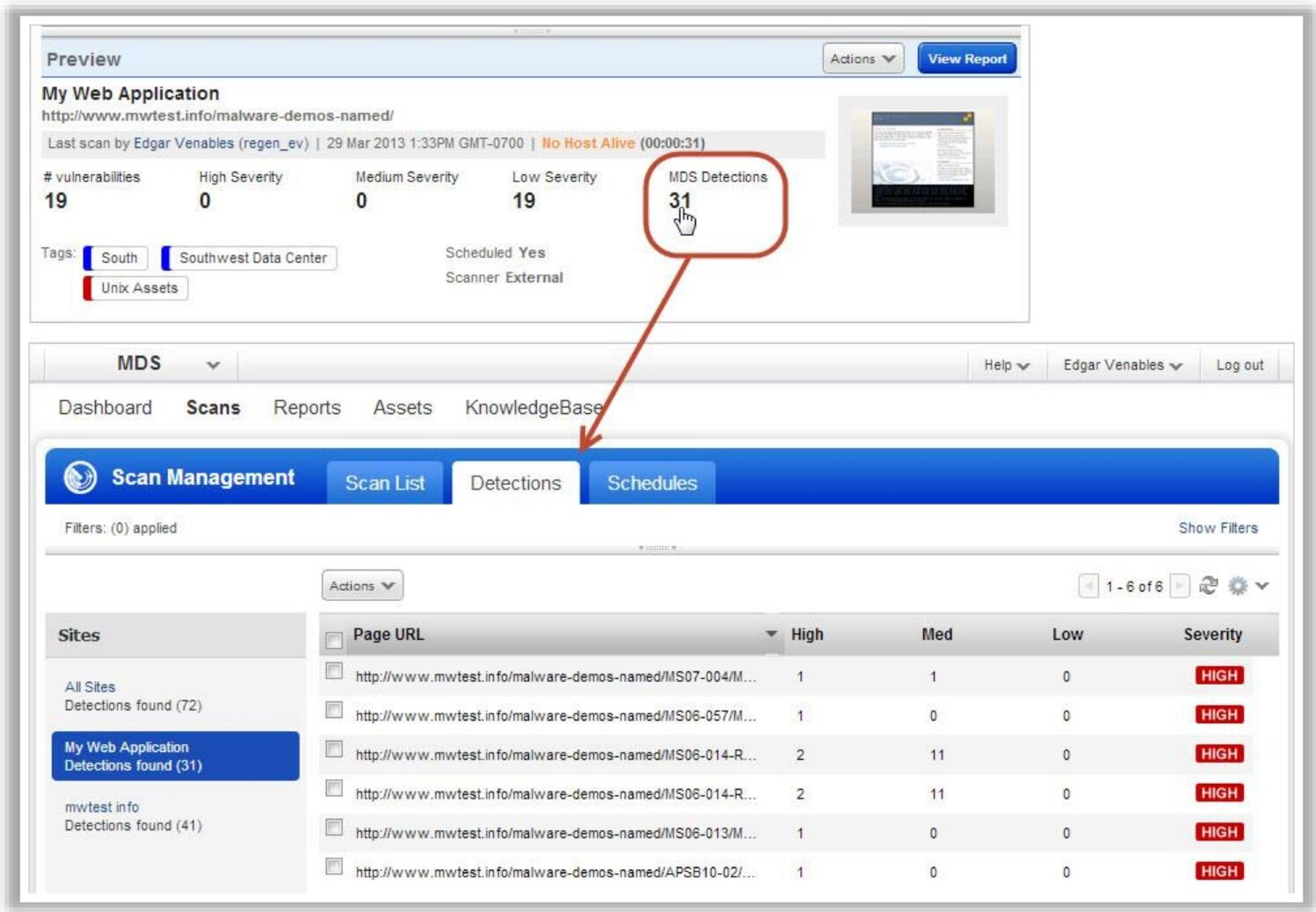
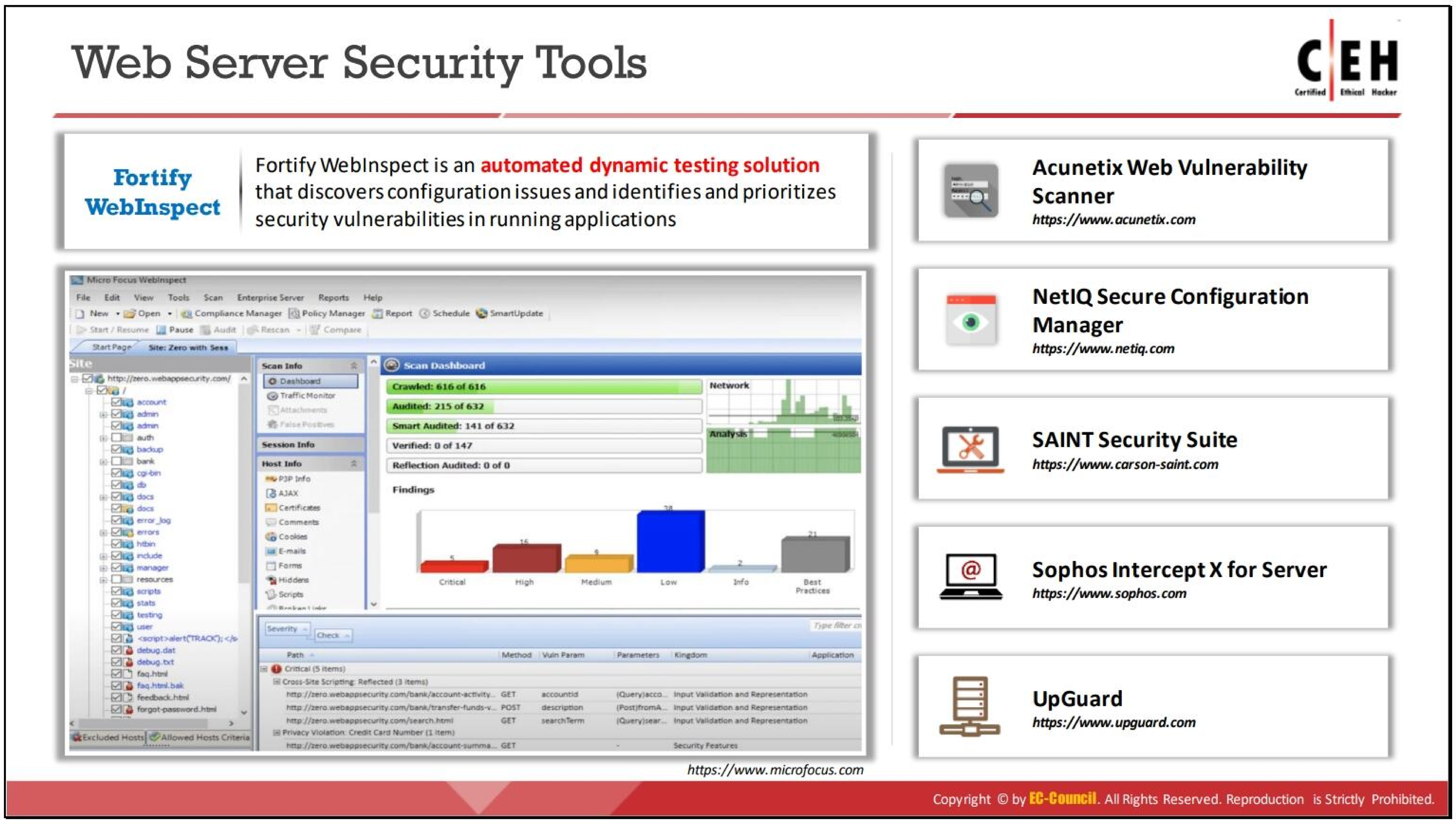


Figure 13.48: Screenshot of QualysGuard Malware Detection

The following are some additional web server malware infection monitoring tools:

- Sucuri SiteCheck (*https://sucuri.net*)
- SiteLock Malware Removal (*https://www.sitelock.com*)
- Quttera (*https://quttera.com*)
- Web Inspector (*https://www.webinspector.com*)
- SiteGuarding (*https://www.siteguarding.com*)



Web Server Security Tools

- **Fortify WebInspect**

  Source: *https://www.microfocus.com*

  Fortify WebInspect is an automated dynamic testing solution that discovers configuration issues as well as identifies and prioritizes security vulnerabilities in running applications. It mimics real-world hacking techniques and provides a comprehensive dynamic analysis of complex web applications and services. WebInspect dashboards and reports provide organizations with visibility and an accurate risk posture of its applications.

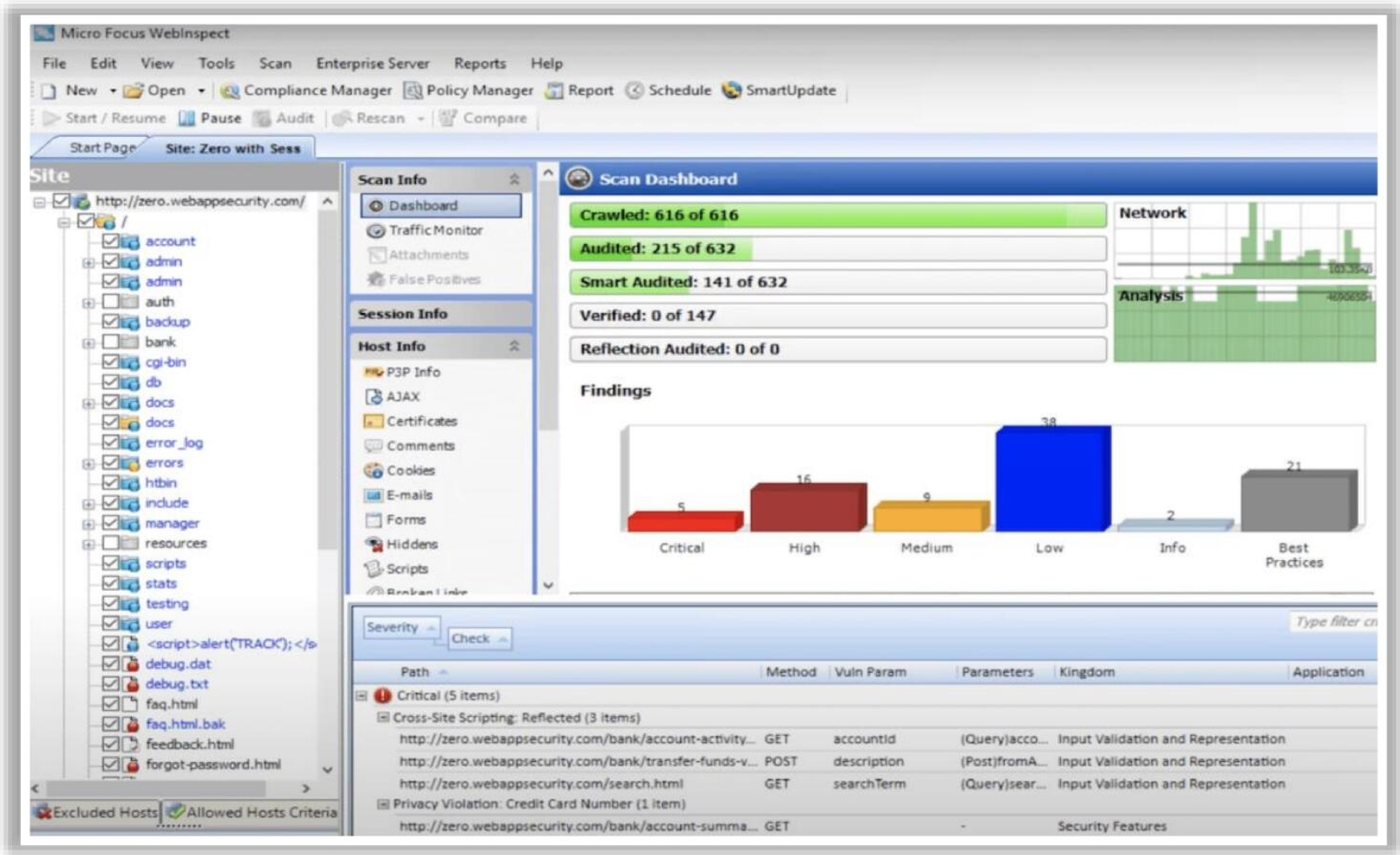


Figure 13.49: Screenshot of Fortify WebInspect

The following are some additional web server security tools:

- Acunetix Web Vulnerability Scanner (*https://www.acunetix.com*)
- NetIQ Secure Configuration Manager (*https://www.netiq.com*)
- SAINT Security Suite (*https://www.carson-saint.com*)
- Sophos Intercept X for Server (*https://www.sophos.com*)
- UpGuard (*https://www.upguard.com*)

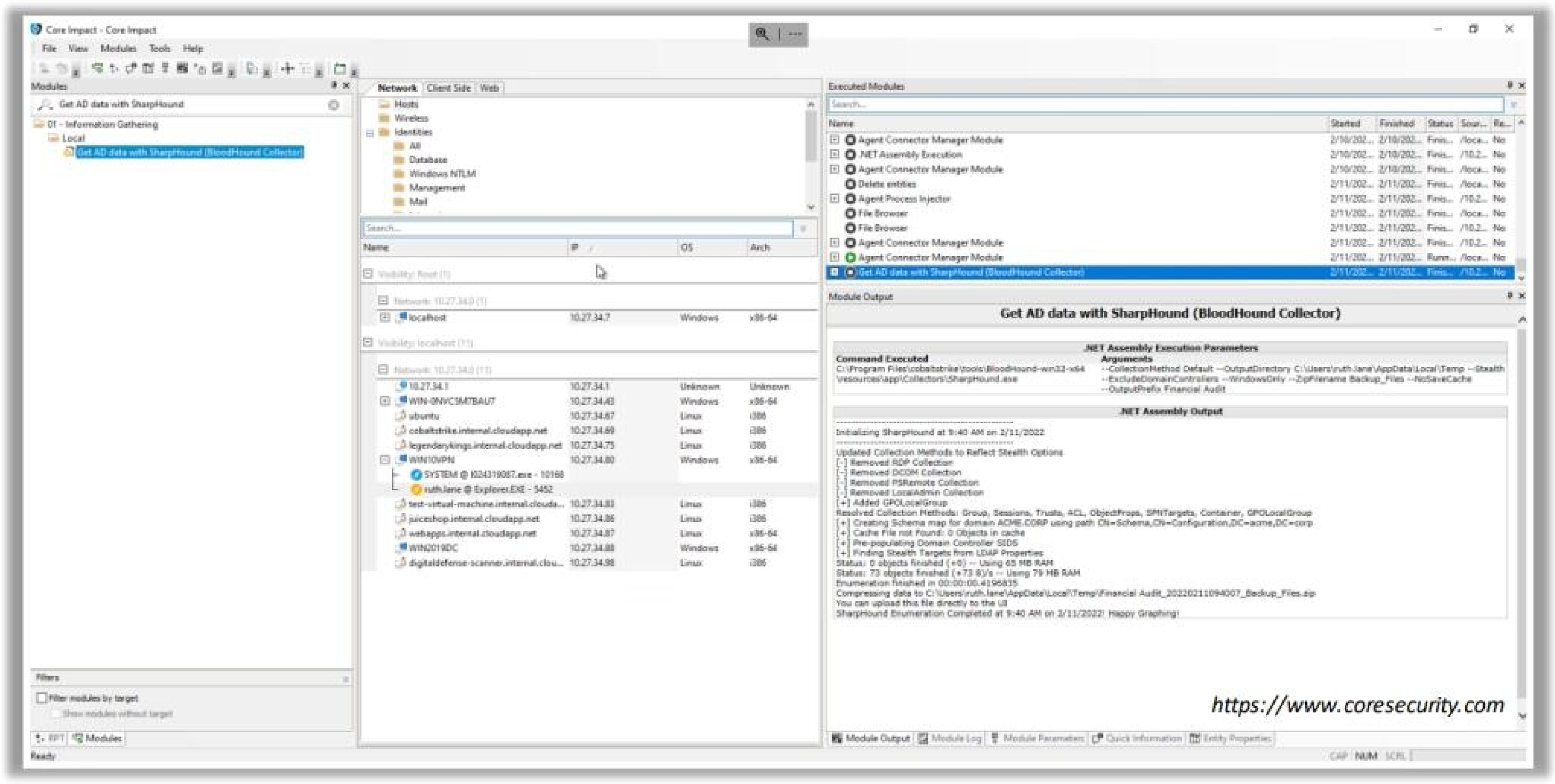
Web Server Pen Testing Tools

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 present-day cybercrime techniques

Web Server Pen Testing Tools

- Immunity CANVAS (*https://www.immunityinc.com*)
- Arachni (*https://www.arachni-scanner.com*)
- WebSurgery (*https://sunrisetech.gr*)
- Mitmprox (*https://mitmproxy.org*)
- Webalizer (*https://webalizer.net*)



Web Server Pen Testing Tools

- **CORE Impact**

  Source: *https://www.coresecurity.com*

  CORE Impact finds vulnerabilities in an organization's web server. This tool allows a user to evaluate the security posture of a web server by using the same techniques currently employed by cyber criminals. It scans for possible vulnerabilities in the web server, imports scan results, and runs exploits to test the identified vulnerabilities. It can also scan network servers, workstations, firewalls, routers, and various applications for vulnerabilities; identify which vulnerabilities pose real threats to the network; determine the potential impact of exploited vulnerabilities; and prioritize and execute remediation efforts.

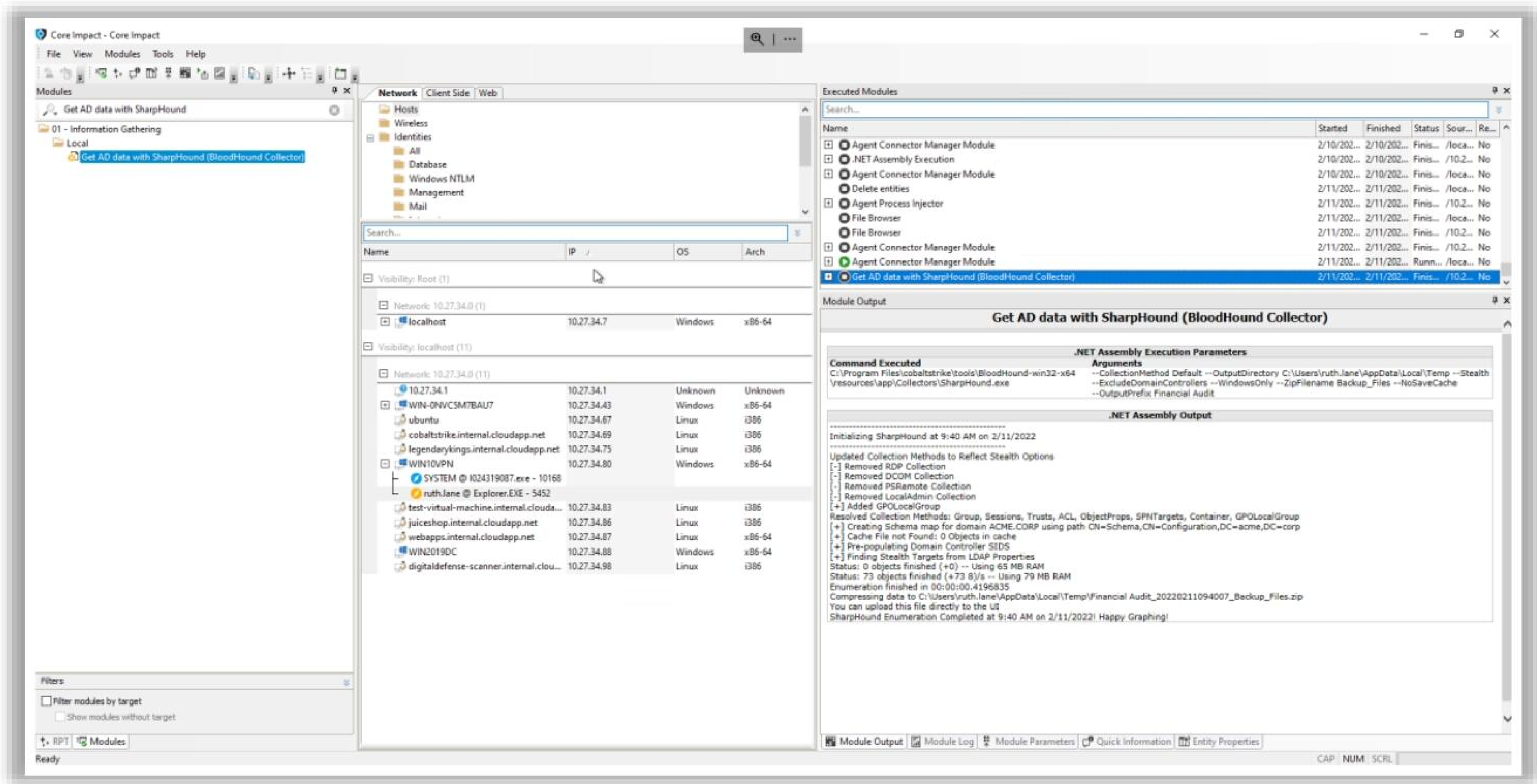


Figure 13.50: Screenshot of CORE Impact

The following are some additional web server pen testing tools:

- Immunity CANVAS (*https://www.immunityinc.com*)
- Arachni (https:*//www.arachni-scanner.com*)
- WebSurgery (*https://sunrisetech.gr*)
- Mitmprox (*https://mitmproxy.org*)
- Webalizer (*https://webalizer.net*)

LO#05: Summarize Patch Management Concepts

Patch Management

Developers always attempt to find bugs in a web server and fix them. Bug fixes are distributed in the form of patches, which provide protection against known vulnerabilities. Unpatched or vulnerable patches can create a security loophole in the web server. This section describes the role of patches, upgrades, and hotfixes in securing web servers. This section also provides guidance for choosing proper patches, upgrades, hotfixes, and their appropriate sources for secure patch management.

Patches and Hotfixes

1. Hotfixes are an **update to fix a specific customer issue** and not always distributed outside the customer organization
2. 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
3. Users may be notified through **emails** or through the **vendor's website**
4. A patch can be considered as a **repair job for a programming problem**
5. Hotfixes are sometimes packaged as a set of fixes called a **combined hotfix** or **service pack**

Patches and Hotfixes

A patch is a small piece of software designed to fix problems, security vulnerabilities, and bugs as well as improve the usability or performance of a computer program or its supporting data. A patch can be considered a repair job for a programming problem. A software vulnerability is the weakness of a software program that makes it susceptible to malware attacks. Software vendors provide patches that prevent exploitations and reduce the probability of threats exploiting a specific vulnerability. Patches include fixes and updates for multiple known bugs or issues. A patch is a publicly released update that is available for all customers. A system without patches is much more vulnerable to attacks than a regularly patched system. If an attacker can identify a vulnerability before it is fixed, then the system might be susceptible to malware attacks.

A hotfix is a package used to address a critical defect in a live environment and contains a fix for a single issue. It updates a specific product version. Hotfixes provide quick solutions and ensure that the issues are resolved. Apply hotfixes to software patches on production systems.

Vendors update users about the latest hotfixes through email or make them available on their official website. Hotfixes are updates that fix a specific customer issue and are not always distributed outside the customer organization. Vendors occasionally deliver hotfixes as a set of fixes called a combined hotfix or service pack.

What is Patch Management?

- “Patch management is a process used to fix known vulnerabilities by ensuring that the **appropriate patches** are installed on a system”

An automated patch management process

Detect	Use tools to detect missing security patches
Assess	Asses the issue(s) and associated severities 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 ensure that the applications are not affected
Maintain	Subscribe to get notifications about vulnerabilities as they get detected

What is Patch Management?

According to https://www.techtarget.com/searchenterprisedesktop/, patch management is an area of systems management that involves acquiring, testing, and installing multiple patches (code changes) in an administered computer system. Patch management is a method of defense against vulnerabilities that cause security weaknesses or corrupt data. It is a process of scanning for network vulnerabilities, detecting missed security patches and hotfixes, and then deploying the relevant patches as soon as they are available to secure the network. It involves the following tasks:

- Choosing, verifying, testing, and applying patches
- Updating previously applied patches with current patches
- Listing patches applied previously to the current software
- Recording repositories or depots of patches for easy selection
- Assigning and deploying the applied patches

An automated patch management process includes the following steps.

- **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 test machine to verify the consequences of the update.
- **Deploy**: Deploy the patch to computers and ensure that applications are not affected.
- **Maintain**: Subscribe to receive notifications about vulnerabilities when 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 **downloads the patch** from the vendor and installs 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 a proper **patch management program** to validate file 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

Installation of a Patch

The installation of a patch entails the following tasks.

- **Identifying Appropriate Sources for Updates and Patches**

  It is important to identify appropriate sources for updates and patches. Patches and updates that are not installed from trusted sources can render the target server even more vulnerable to attacks, instead of hardening its security. Thus, the selection of appropriate sources for updates and patches plays a vital role in securing web servers.

  The following are some methods for identifying appropriate sources for updates and patches.

  - Create 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 OS vendors.
  - The recommended method of tracking issues relevant to proactive patching is to register to 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.

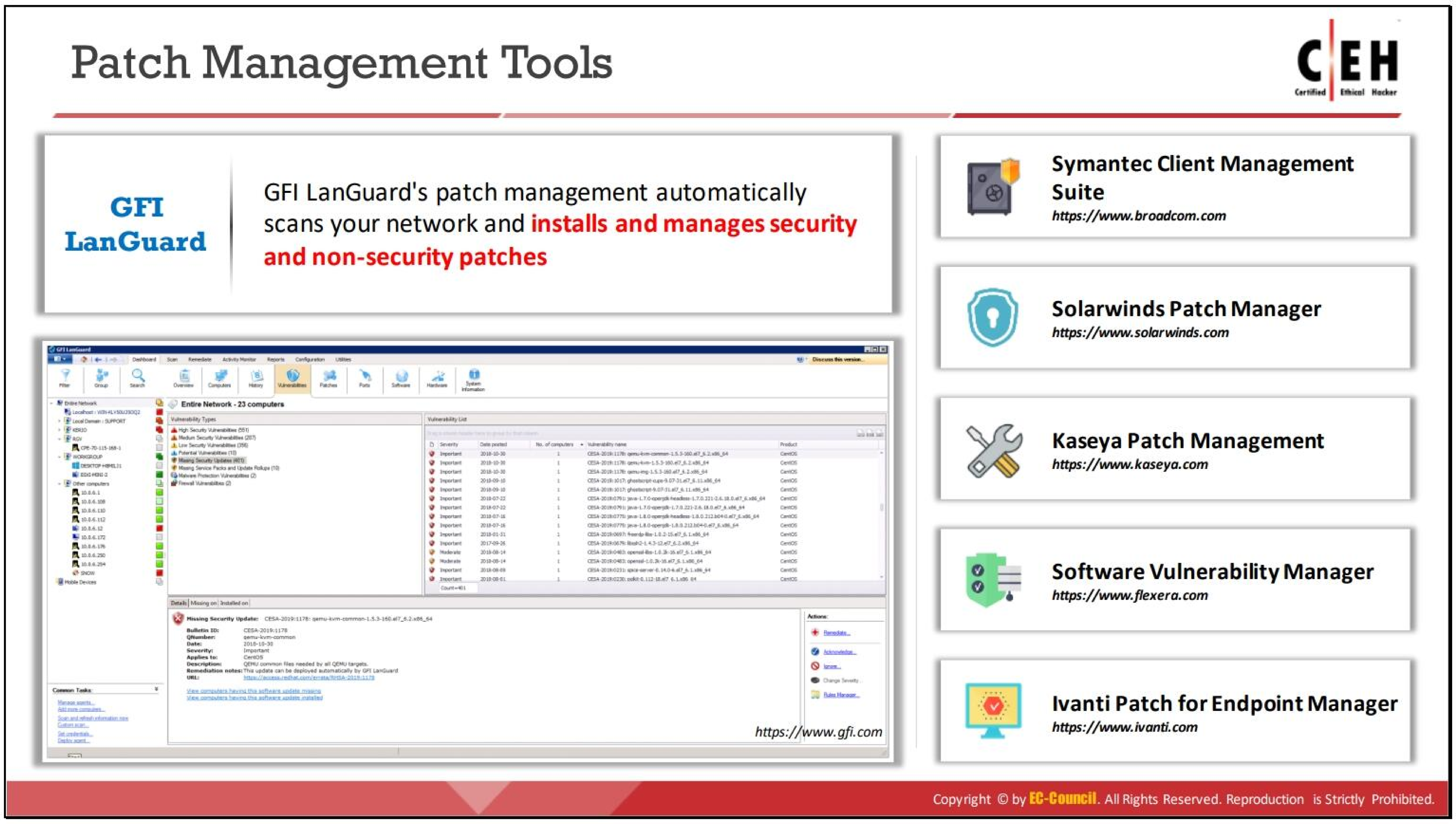
  - **Manual Installation**

    In this method, the user downloads the patch from the vendor and installs it.

- **Automatic Installation**

  In this method, applications use an auto update feature to update themselves.

- **Implementation and Verification of a Security Patch or Upgrade**
  - Before installing any patch, verify the source.
  - Use a proper patch management program to validate file 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**

  Source: *https://www.gfi.com*

  The GFI LanGuard patch management software scans the user's network automatically as well as installs and manages security and non-security patches. It supports machines across Microsoft®, MAC OS X®, and Linux® operating systems, as well as many third-party applications. It allows auto-downloads of missing patches as well as patch rollback, resulting in a consistently configured environment that is protected from threats and vulnerabilities.

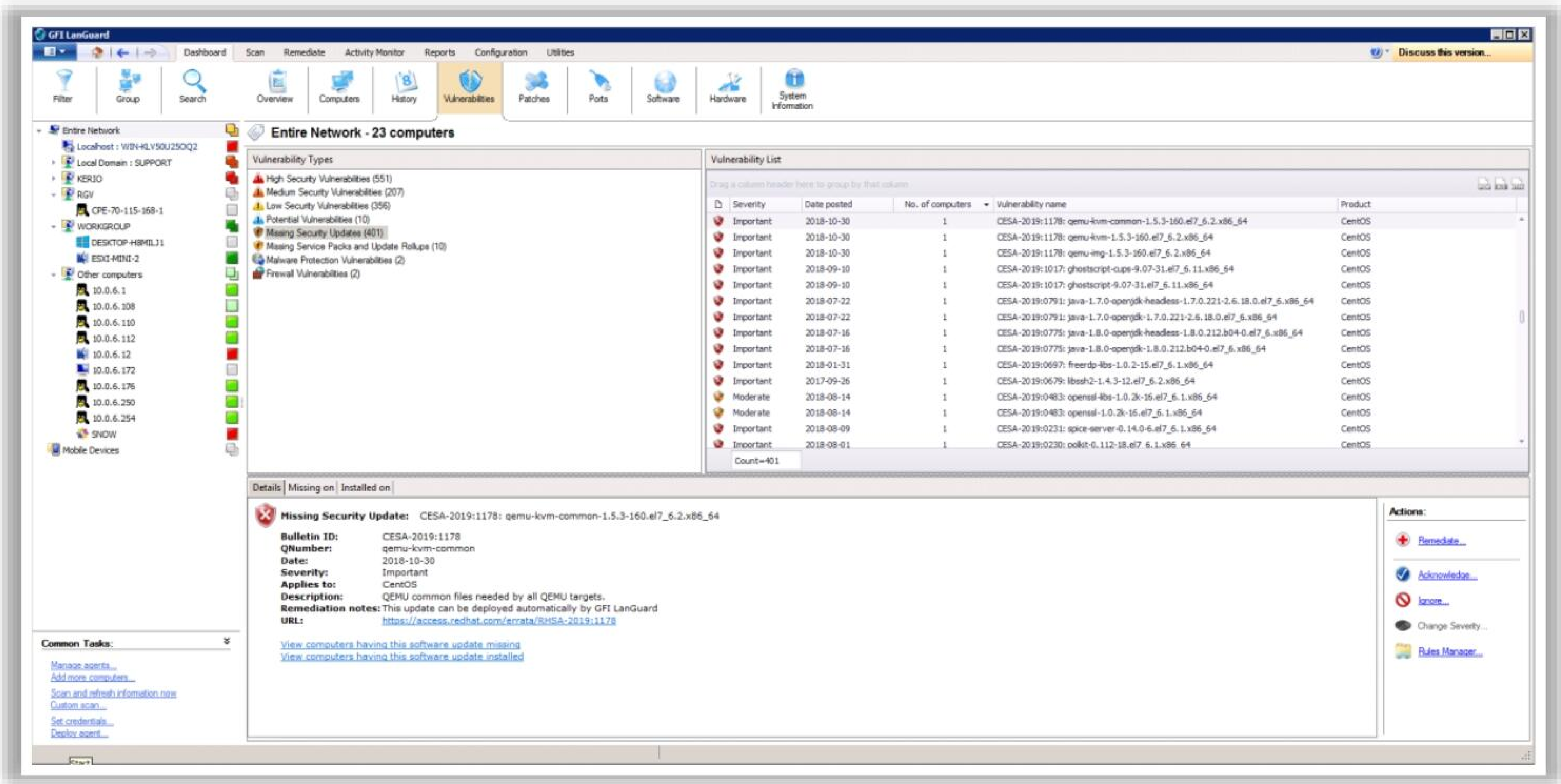


Figure 13.51: Screenshot of GFI LanGuard patch management software

The following are some additional patch management tools:

- Symantec Client Management Suite (*https://www.broadcom.com*)
- Solarwinds Patch Manager (*https://www.solarwinds.com*)
- Kaseya Patch Management (*https://www.kaseya.com*)
- Software Vulnerability Manager (*https://www.flexera.com*)
- Ivanti Patch for Endpoint Manager (*https://www.ivanti.com*)

Module Summary

- In this module, we have discussed the following:
  - Web server concepts
  - Various web server threats and attacks in detail
  - Web server attack methodology in detail, including information gathering, web server footprinting, website mirroring, vulnerability scanning, session hijacking, and web server passwords hacking
  - Various web server hacking tools
  - Various countermeasures that are to be employed to prevent web server hacking attempts by threat actors
  - Detailed discussion on securing web servers using various security tools
  - Patch management concepts
- In the next module, we will discuss in detail how attackers, as well as ethical hackers and pen-testers, hack web applications

Module Summary

In this module, we discussed in detail the general concepts related to web servers; various web server threats and attacks; the web server attack methodology, including information gathering, web server footprinting, website mirroring, vulnerability scanning, session hijacking, and web server passwords hacking; and various web server hacking tools. Additionally, we discussed various countermeasures that can be employed to prevent web server hacking attempts by threat actors. We also discussed how to secure web servers using various security tools. We concluded the module with a detailed discussion on patch management concepts.

In the next module, we will discuss in detail how attackers, including ethical hackers and pen testers, hack web applications.