6. Sending Split packet out of order with delay is an example of: A. Insertion Attack
B. Fragmentation Attack
C. Obfuscating
D. Session Splicing

# Chapter 13: Hacking Web Servers

## Technology Brief

Web Servers are programs that are used for hosting websites. Web servers can be deployed on separate web server hardware or installed on a host as a program. Use of web applications has increased over the last few years. New web applications are flexible and capable of supporting larger clients. In this chapter, we will discuss web server vulnerabilities, techniques and tools for attacking them, and mitigation methods.

## Web Server Concepts

A Web Server is a program that hosts websites, based on both hardware and software. It delivers files and other content on the website over Hyper Text Transfer Protocol (HTTP). As use of internet and intranet has increased, web services have become a major part of the internet. They are used for delivering files, email communication, and other purposes. Whereas all of web servers support HTML for basic content delivery, they support different types of application extensions. Web servers differ regarding security models, Operating Systems, and other factors.

## Web Server Security Issues

Security Issues for web servers may include network-level attacks and Operating System-level attacks. Usually, an attacker will target any vulnerability or error in web server configuration and exploit these loopholes. These vulnerabilities may include:

- Improper permission of file directories
- Default configuration
- Enabling unnecessary services
- Lack of security
- Bugs
- Misconfigured SSL Certificates
- Enabling debugging

Server administrators must make sure to eliminate all vulnerabilities and deploy network security measures such as IPS/IDS and firewalls. Threats and attacks to a web server are described later in this chapter. Once a web server is compromised, it can result in the compromise of all user accounts, denial of server services, defacement, and the launch of further attacks through the compromised website, accessing resources and data theft.

## Open Source Web Server Architecture

Open Source Web Server Architecture is the web server model in which an open source web server is hosted, either on a web server or a third-party host over the internet. The most popular and widely used open source web servers are:

- Apache HTTP Server
- NGINX
- Apache Tomcat
- Lighttpd
- Node.js

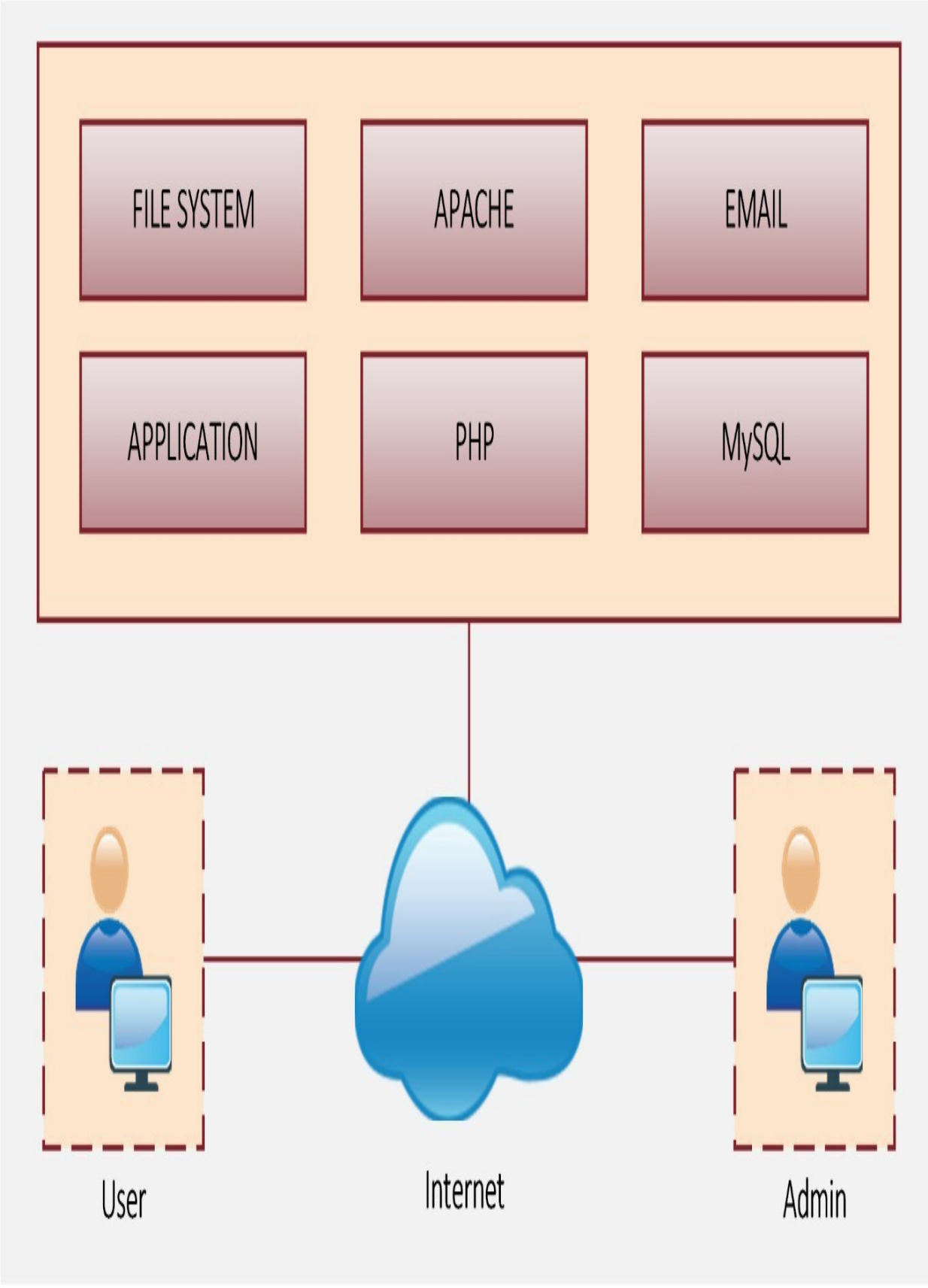



*Figure 13. 01: Open Web Server Architecture*

*Figure 13-01: Open Web Server Architecture*

## IIS Web Server Architecture

Internet Information Services (IIS) is a Windows-based service that provides a request processing architecture. IIS latest version is 7.x. The architecture includes Windows Process Activation Services (WAS), Processing Pipelines. IIS contains

Web Server Engine, multiple components and Integrated Request

responsible for several functions such as listening to a request, managing processes, reading configuration files, etc.

### *Components of IIS*

Components of IIS include:

- *Protocol Listeners*

Protocol Listeners are responsible for receiving protocol-specific requests. They forward these requests to IIS for processing and then return responses to requestors.

- *HTTP.sys*

HTTP listener is implemented as a kernel-mode device driver called the HTTP protocol stack (HTTP.sys). HTTP.sys is responsible for listening to HTTP requests, forwarding these requests to IIS for processing, and then returning processed responses to client browsers.

- *World Wide Web Publishing Service (WWW Service)*
- *Windows Process Activation Service (WAS)*

In the previous version of IIS, World Wide Web Publishing Service (WWW Service) handled functionality, whereas in version 7 and later, WWW Service and WAS Service are used. These services run svchost.exe on the local system and share same binaries.

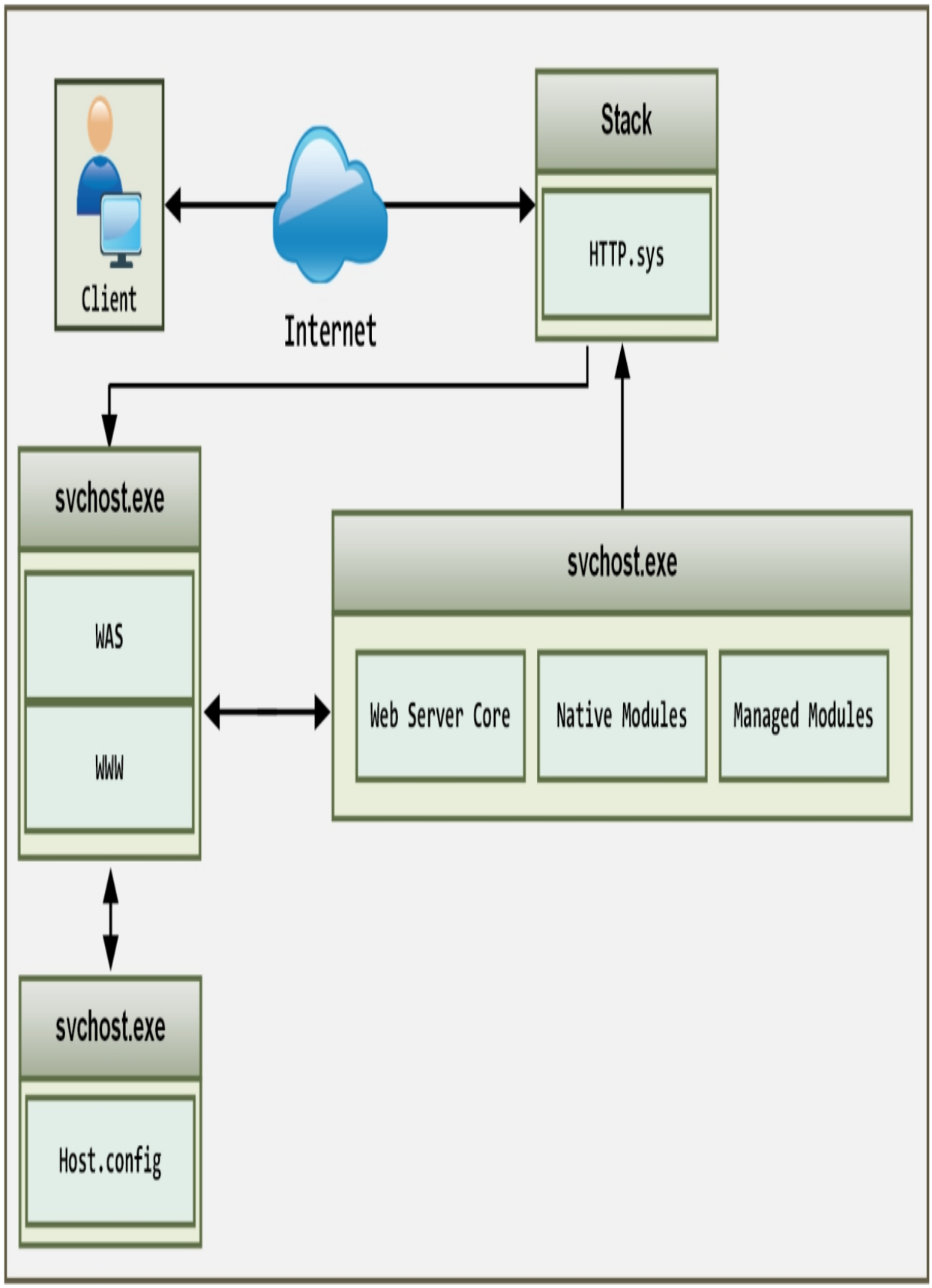



*Figure 13-02: IIS Web Server Architecture*

*Figure 13-02: IIS Web Server Architecture*

## Web Server Attacks

There are several Web Server Attacking techniques, some of which were defined earlier in this workbook. The remaining techniques are defined below:

### *DoS/DDoS Attacks*

DOS and DDOS attack techniques are defined in detail in Chapter 9. These DOS/DDOS attacks are used to flood fake requests towards the web server resulting in crashing, unavailability, or denial of service for all users.

### *DNS Server Hijacking*

By compromising the DNS server, an attacker modifies the DNS configuration. Modification results in redirecting requests meant for the target web server to the malicious server owned or controlled by the attacker.

### *DNS Amplification Attack*

A DNS Amplification Attack is performed with the help of the DNS recursive method. An attacker takes advantage of this feature and spoofs the lookup request to the DNS server. The DNS server sends the request to the spoofed address, i.e., the address of the target. Amplifying the size of the request and using botnets results in a distributed denial-of-service attack.

### *Directory Traversal Attacks*

In this type of attack, attackers use the trial and error method to access restricted directories using dots and slash sequences. By accessing the directories outside the root directory, attackers can reveal sensitive information about the system.

### *Man-in-the-Middle/Sniffing Attack*

As defined in previous chapters, by using a Man-in-the-Middle Attack, an attacker places him/herself between the client and server and sniffs

the packets. He/she extracts sensitive information from the communication by intercepting and altering the packets.

### *Phishing Attacks*

By using Phishing Attacks, an attacker attempts to extract login details from a fake website that appears to be a legitimate website. The attacker tries to impersonate a legitimate user on the actual target server, using stolen information, usually credentials.

### *Website Defacement*

Website Defacement is a process in which attackers, after successful intrusion into a legitimate website, alter, modify, and change the appearance of the website. Accessing and defacing a website can be performed with several techniques such as SQL injection.

### *Web Server Misconfiguration*

Another method of attack is finding vulnerabilities in a website and exploiting them. An attacker may look for misconfigurations and vulnerabilities in the system and web server components. The attacker may identify weaknesses in terms of default configuration, remote functioning, misconfigurations, default certification, and debug in order to exploit them.

### *HTTP Response Splitting Attack*

HTTP Response Splitting Attacks are techniques in which an attacker sends responsesplitting requests to the server. In this way, an attacker can add a header response, resulting in the server splitting the response into two. The second response comes under the control of the attacker so the user can be redirected to the malicious website.

### *Web Cache Poisoning Attack*

A Web Cache Poisoning Attack is a technique in which an attacker wipes the actual cache of the web server and stores fake entries by

sending a crafted request into the cache. This will redirect the users to the malicious web pages.

### *SSH Brute-Force Attack*

Brute-Forcing the SSH tunnel allows an attacker to use an encrypted tunnel. This encrypted tunnel is used for the communication between hosts. By brute-forcing the SSH login credentials, an attacker can gain unauthorized access to the SSH tunnel.

### Web Application Attacks

Other web application related attacks include:

- Cookie Tampering
- DoS Attack
- SQL Injection
- Session Hijacking
- Cross-Site Request Forgery (CSRF) Attack
- Cross-Site Scripting (XSS) Attack
- Buffer Overflow

## Attack Methodology

### Information Gathering

Information gathering involves collecting information about a target using different platforms, either through social engineering or internet surfing. An attacker may use different tools and networking commands to extract information. They may also navigate to the robot.txt file to extract information about internal files.

*Figure 13-03: Robots.txt File*

### Web Server Footprinting

This includes footprinting focused on the web server using different tools such as Netcraft, Maltego, and httprecon, etc. The results of web server footprinting can include the server name, type, Operating System, running application, and other information about the target website.

## Lab 13- 1: Web Server Footprinting Tool

Download and install the ID Server tool.

1. Enter the URL or IP address of the target server.

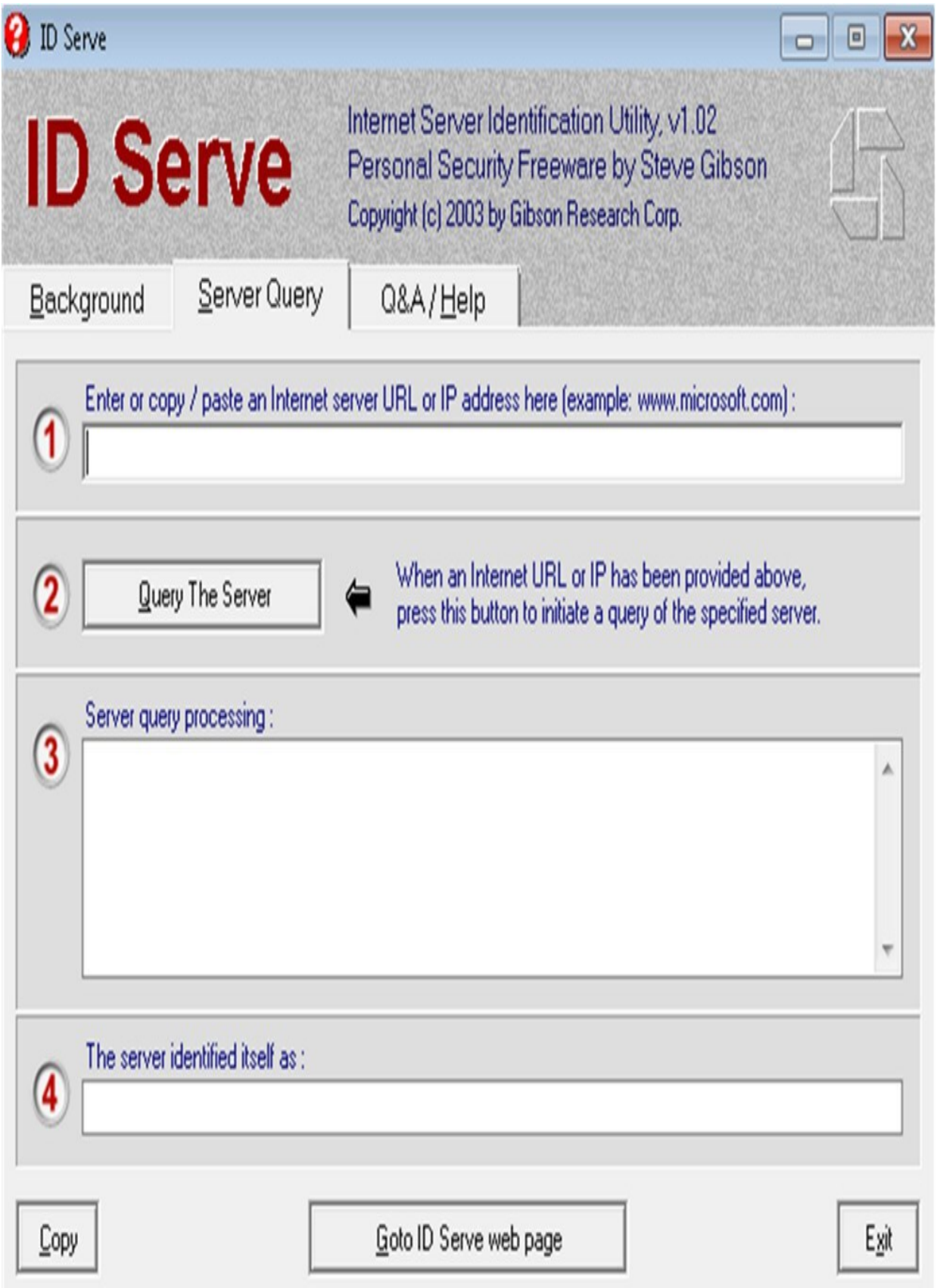

ID Serve
ID Serve
Internet Server Identification Utility, v1.02
Personal Security Freeware by Steve Gibson
Copyright (c) 2003 by Gibson Research Corp.
Background
Server Query
Q&A / Help
1
Enter or copy / paste an Internet server URL or IP address here (example: www.microsoft.com) :
2
Query The Server
When an Internet URL or IP has been provided above, press this button to initiate a query of the specified server.
3
Server query processing :
4
The server identified itself as :
Copy
Goto ID Serve web page
Exit

*Figure 13-04: ID Serve Application*

2. Click the "Query the Server" button.

*Figure 13-05: Generating Query*

3. Copy the extracted information.

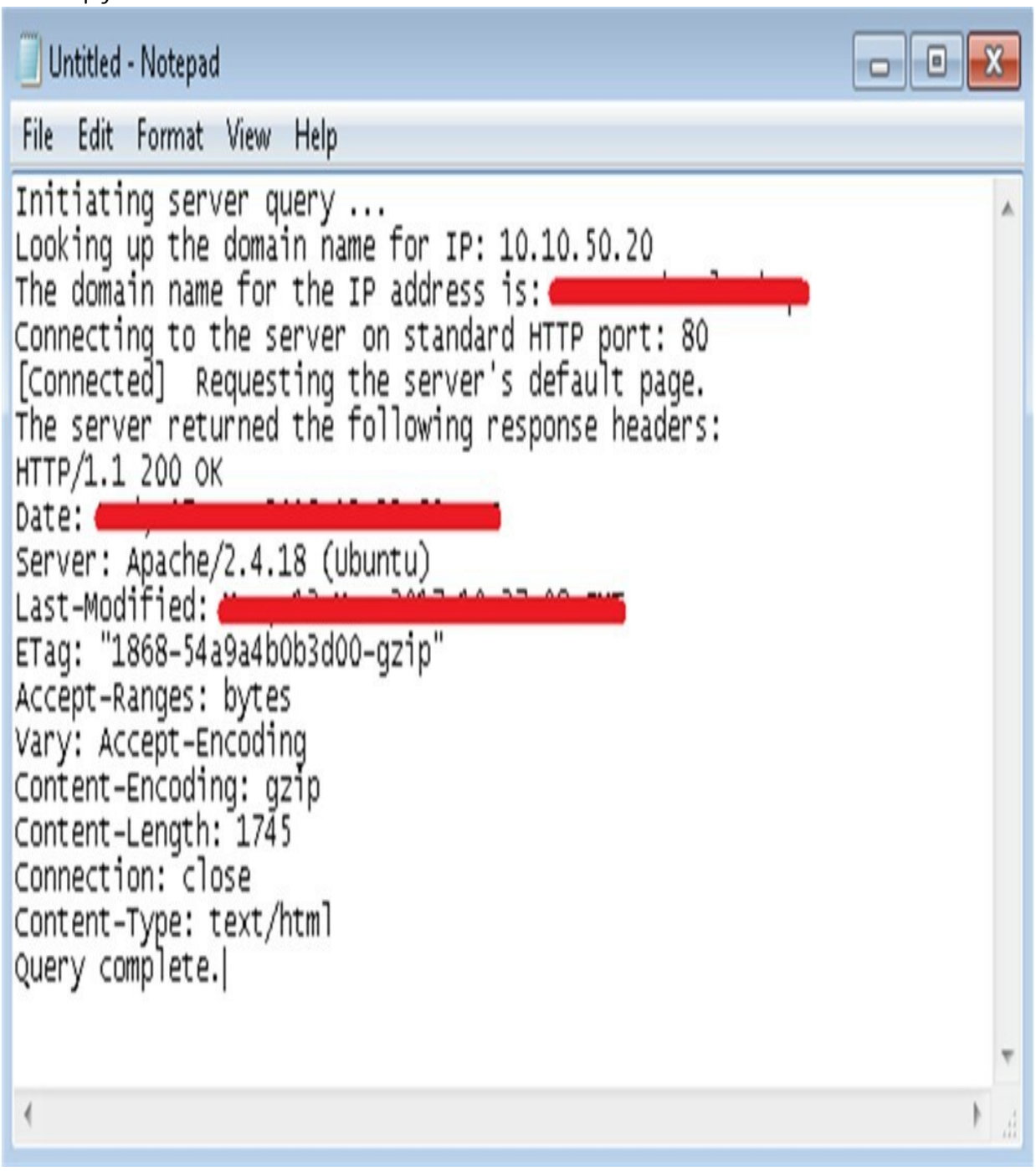



*Figure 13-06: Extracted Information*

Information such as domain name, open ports, and server type are

extracted.

## Mirroring a Website

As defined earlier in this workbook, mirroring a website is the process of replicating an entire website on the local system. By downloaded the entire website onto the system, the attacker is able to use and inspect the websites, its directories and structures and is able to find its vulnerabilities. This is easiest way to find a website's vulnerabilities, and easier than sending multiple copies to a web server,.

## Vulnerability Scanning

Vulnerability Scanners are automated utilities specially developed to detect vulnerabilities, weaknesses, problems, and holes in an Operating System, network, software, and in applications. These scanning tools perform deep inspection of scripts, open ports, banners, running services, configuration errors, and other areas.

## Session Hijacking

Session Hijacking is also known as TCP Session Hijacking. It is a technique for taking control of a user's web session by manipulating the session ID. The attacker steals a legitimate user's authenticated session without initiating a new session with the target server.

## Hacking Web Passwords

Password Cracking is the method of extracting a password to gain authorized access to a target system in the guise of a legitimate user. Password cracking may be performed through a social engineering attack or cracking through tempering the communication and stealing the stored information.
Password attacks are classified as the following:

- Non-Electronic Attacks
- Active Online Attacks
- Passive Online Attacks
- Default Password
- Offline Attack

## Countermeasures

The basic recommendation for securing a web server from internal and external attacks and other threats is to place the web server in a secure zone where security devices such as firewalls, IPS, and IDS are deployed to constantly filter and inspect the traffic destined to the web server. Placing the web server in an isolated environment such as a DMZ protects it from threats.

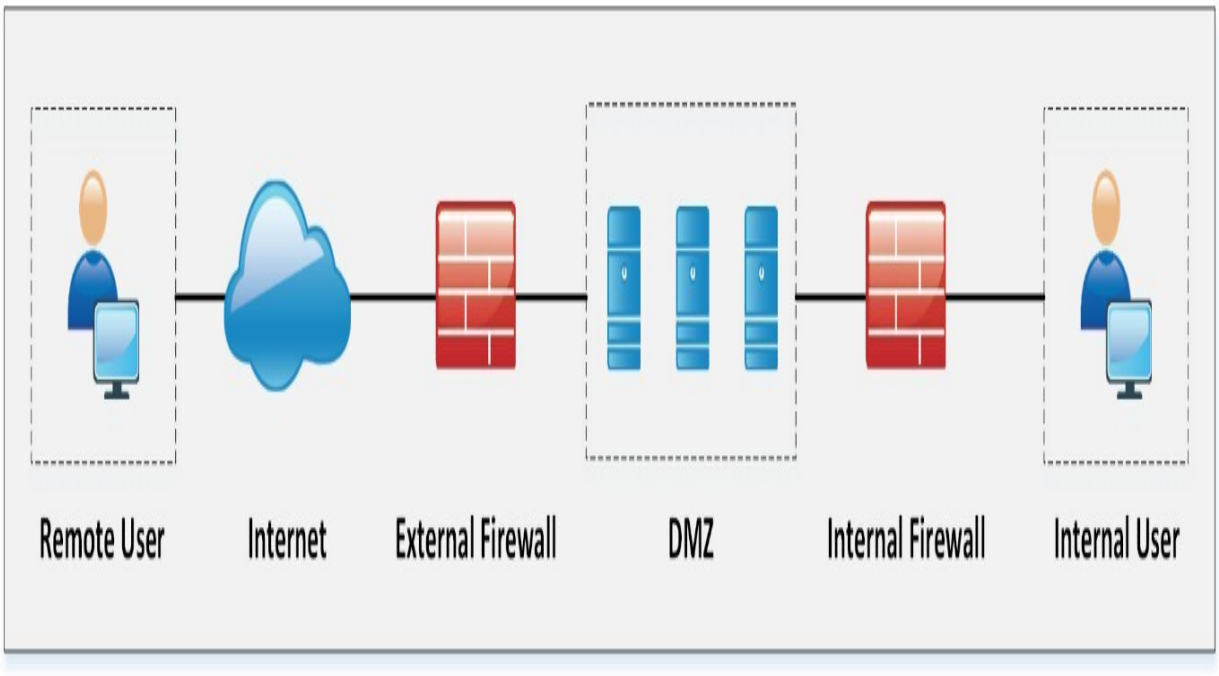



*Figure 13-07: Web Server Deployment*

## Detecting Web Server Hacking Attempts

The are several techniques for detecting intrusions or unexpected activity on a web server, for example, a website change detection system that uses scripting to detect hacking attempts and focuses on inspecting changes made by executable files. Similarly, hashes are periodically compared to detect modification.

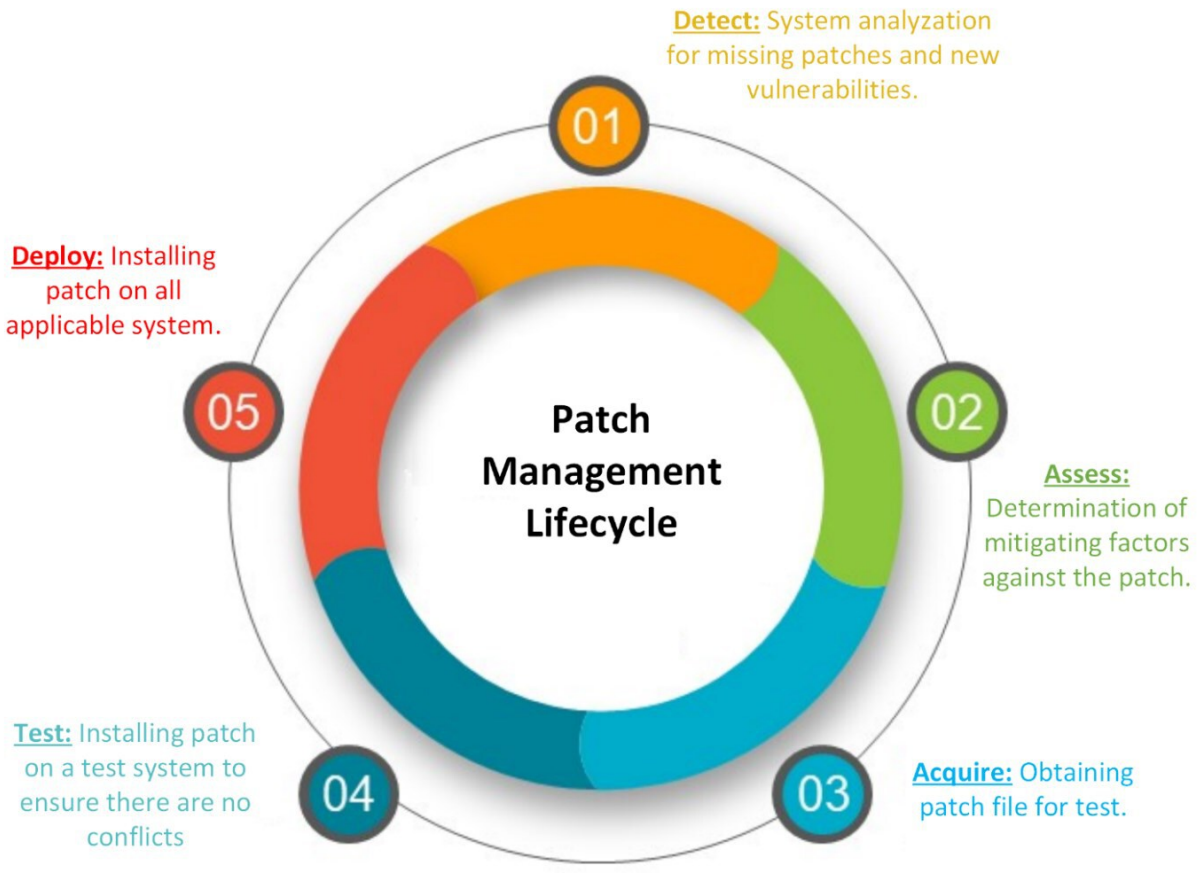
## Defending Against Web Server Attacks

- Auditing Ports
- Disabling Insecure and Unnecessary Ports
- Using Port 443 HTTPS over Port 80 HTTP

- Encrypted Traffic
- Server Certificate
- Code Access Security Policy
- Disable Tracing

## Disable Debug Compiles Patch Management

Patches and Hotfixes are used to remove vulnerabilities, bugs, and issues in a software release. Hotfixes are updates that fix these issues, whereas patches are pieces of software specially designed for fixing an issue. A hotfix is referred to as a hot system, specially designed for a live production environment where fixes are made outside normal development and testing is done to address the issue.




Patches must be downloaded from official websites, home sites, and application and Operating System vendors. Registering or subscribing is recommended so as to receive alerts about the latest issues and

patches.
These patches can be downloaded in the following ways:

- Manual Download from a Vendor
- Auto-Update

Patch management is an automated process that ensures the installation of necessary patches on a system. The patch management process detects the missing security patches, finds a solution, downloads the patch, tests the patch in an isolated environment, i.e., testing machine, and then deploys the patch onto the system.

Mind Map


Lab 13-2: Microsoft Baseline Security Analyzer (MBSA)

The Microsoft Baseline Security Analyzer is a Windows-based patch management tool powered by Microsoft. MBSA identifies any missing security updates and common security misconfigurations. The MBSA 2.3 release adds support for Windows 8. 1, Windows 8, Windows Server 20 12 R2, and Windows Server 20 12. Windows 2000 will no longer be supported with this release.

## Procedure:

MBSA is capable of scanning a local system, remote system, and range of the computers.

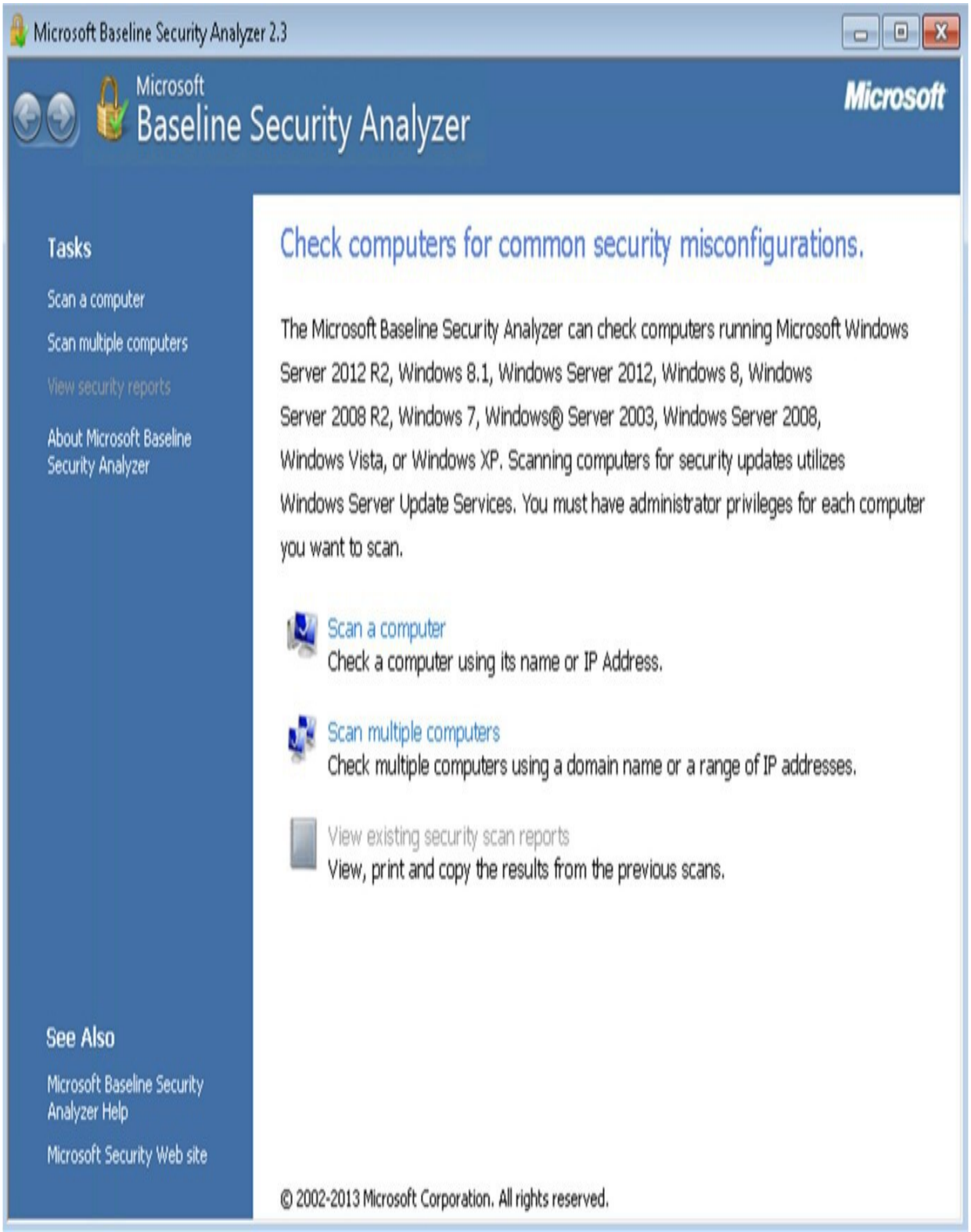



*Figure 13-08: Microsoft Baseline Security Analyzer*

Select the scanning options as required.


Microsoft Baseline Security Analyzer 2.3
Microsoft
Baseline Security Analyzer
Microsoft
Which computer do you want to scan?
Enter the name of the computer or its IP address.
Computer name:
WORKGROUP\WIN7-1-PC
(this computer)
IP address:
Security report name:
%D% - %C% (%T%)
%D% = domain, %C% = computer, %T% = date and time, %IP% = IP address
Options:
Check for Windows administrative vulnerabilities
Check for weak passwords
Check for IIS administrative vulnerabilities
Check for SQL administrative vulnerabilities
Check for security updates
Configure computers for Microsoft Update and scanning prerequisites
Advanced Update Services options:
Scan using assigned Windows Server Update Services(WSUS) servers only
Scan using Microsoft Update only
Scan using offline catalog only
Learn more about Scanning Options
Start Scan
Cancel

*Figure 13-09: Scanning Local System Using MBSA*

MBSA will first get updates from Microsoft, scan them, and then download the security updates.

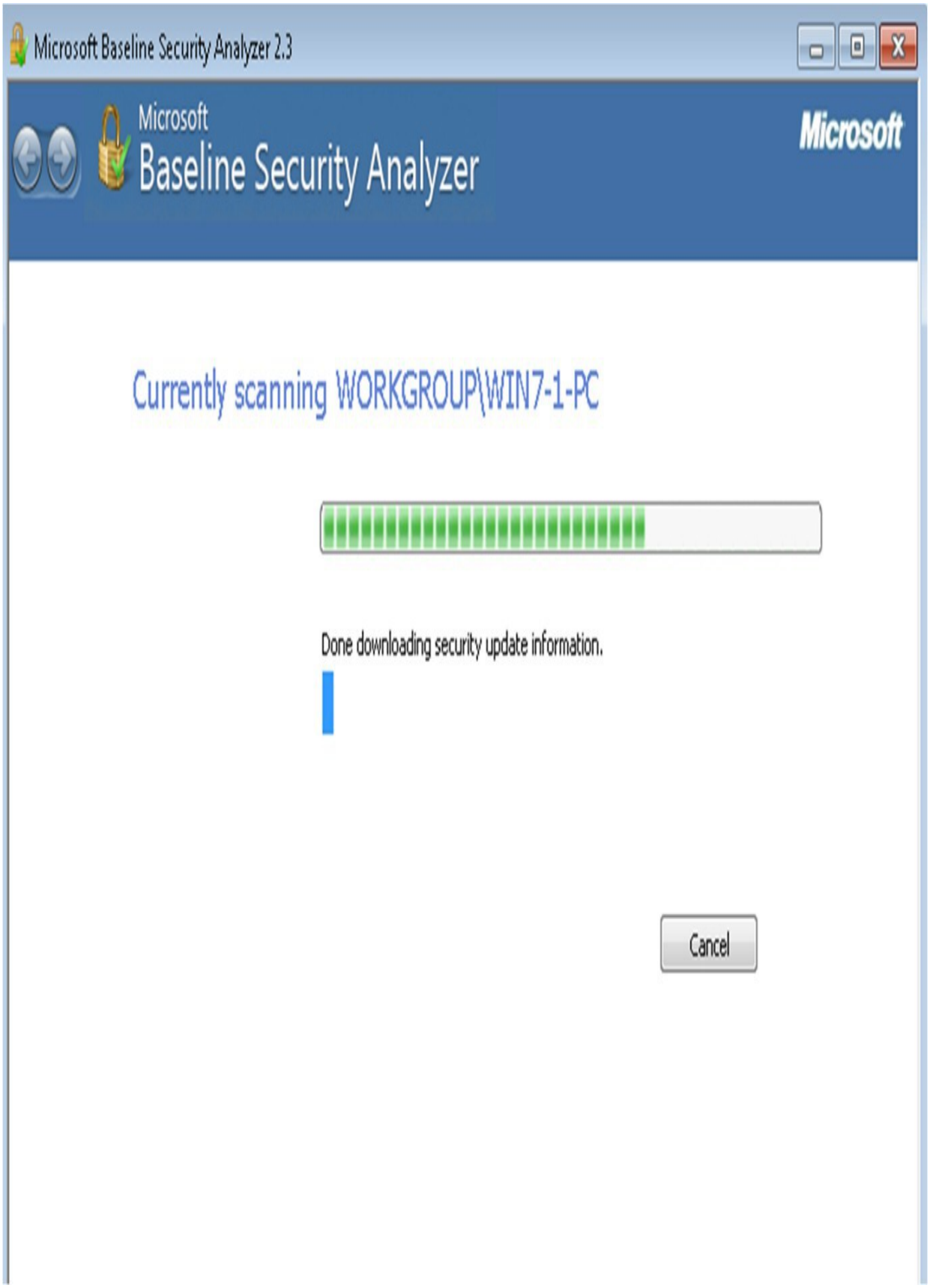

Microsoft Baseline Security Analyzer 2.3
Microsoft
Baseline Security Analyzer
Microsoft
Currently scanning WORKGROUP\WIN7-1-PC
Done downloading security update information.
Cancel

*Figure 13-10: MBSA Scanning*

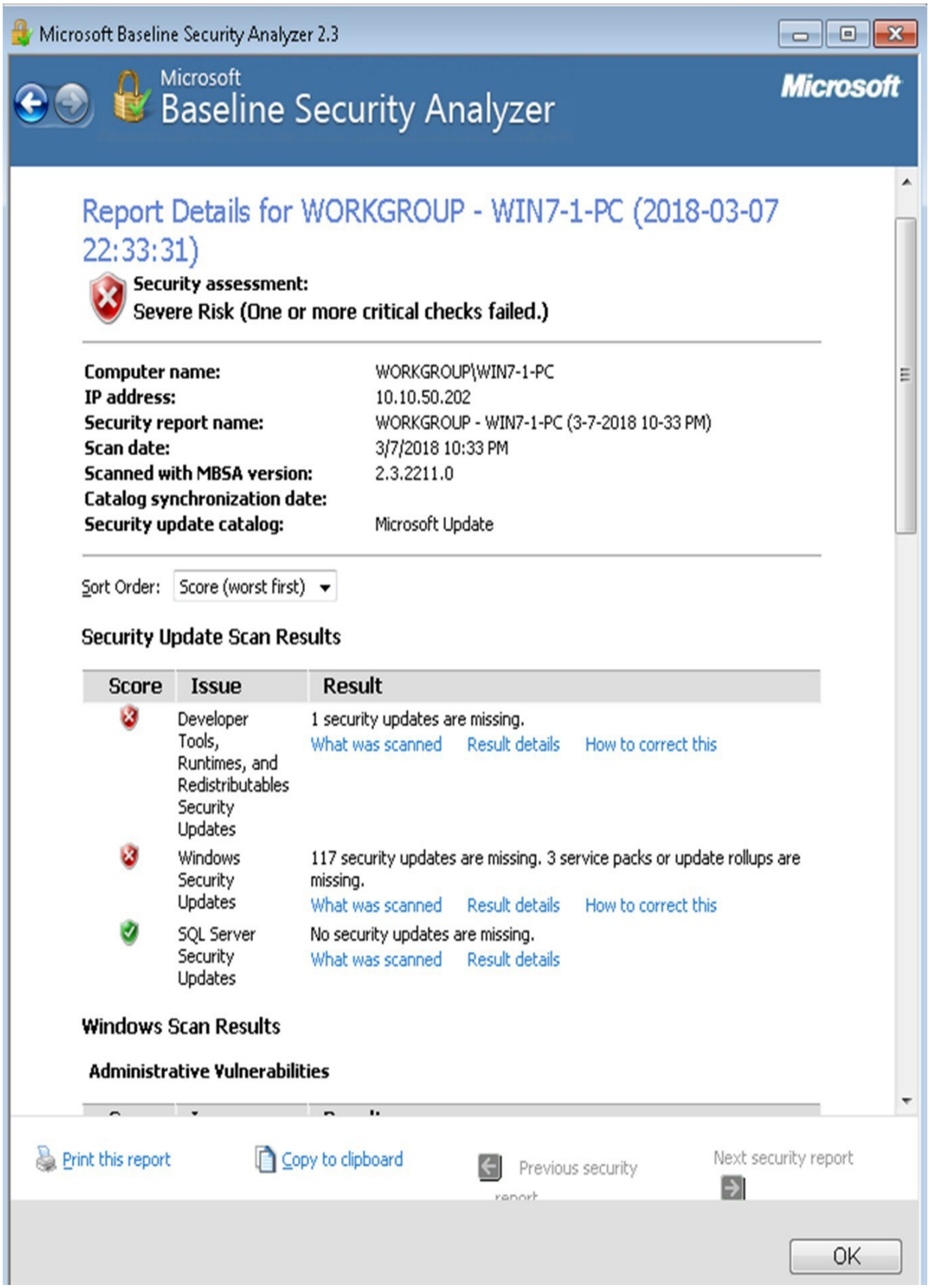

Microsoft Baseline Security Analyzer 2.3
Microsoft
Baseline Security Analyzer
Microsoft
Report Details for WORKGROUP - WIN7-1-PC (2018-03-07 22:33:31)
Security assessment:
Severe Risk (One or more critical checks failed.)
Computer name: WORKGROUP\WIN7-1-PC
IP address: 10.10.50.202
Security report name: WORKGROUP - WIN7-1-PC (3-7-2018 10-33 PM)
Scan date: 3/7/2018 10:33 PM
Scanned with MBSA version: 2.3.2211.0
Catalog synchronization date:
Security update catalog: Microsoft Update
Sort Order: Score (worst first)
Security Update Scan Results
Score
Issue
Result
Developer Tools, Runtimes, and Redistributables Security Updates
1 security updates are missing.
What was scanned
Result details
How to correct this
Windows Security Updates
117 security updates are missing. 3 service packs or update rollups are missing.
What was scanned
Result details
How to correct this
SQL Server Security Updates
No security updates are missing.
What was scanned
Result details
Windows Scan Results
Administrative Vulnerabilities
Print this report
Copy to clipboard
Previous security report
Next security report
OK

*Figure 13-11: MBSA Scanning Results*

In the above figure, the MBSA scanning shows **Security Update Scan Results** . Security update scan results are categorized by issue and the results show a number of missing updates.

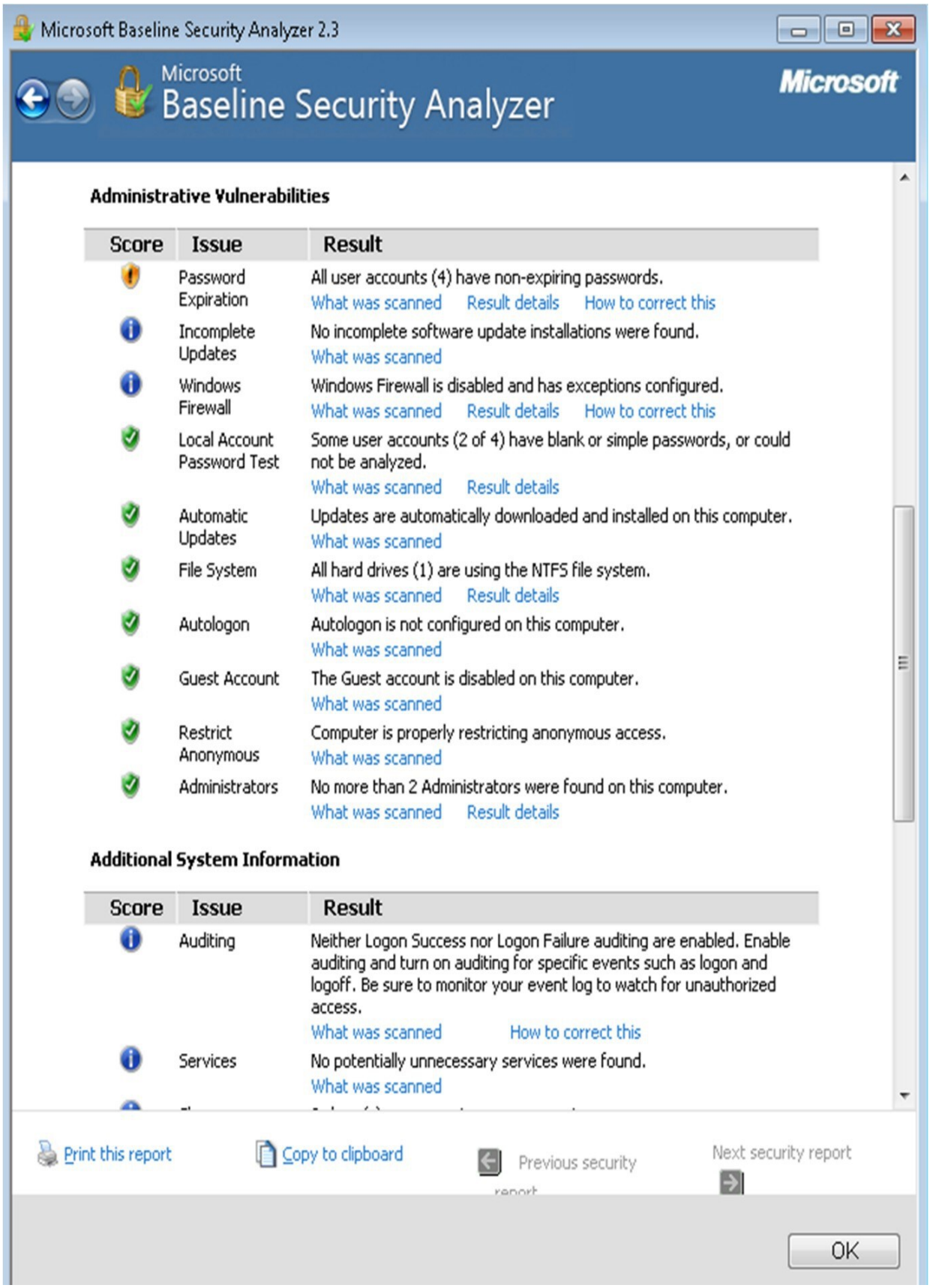



Figure 12–12: MBSA Scanning Results

*Figure 13-12: MBSA Scanning Results*

In the figure above, the MBSA Scanning results shows **Administrative Vulnerabilities** . Vulnerabilities can be password expiry, updates, firewall issues, accounts, etc.

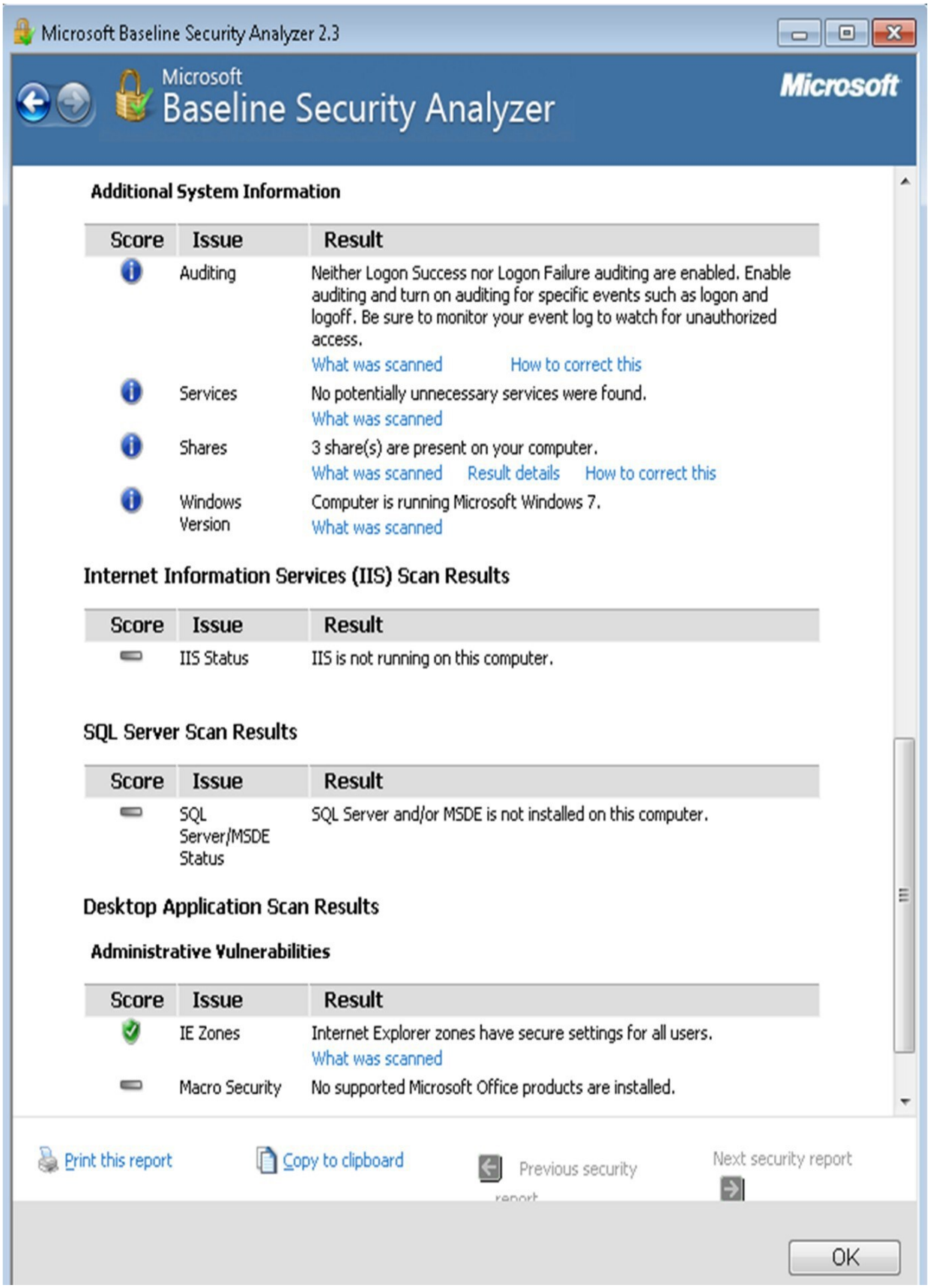



*Figure 12-12: MBSA Scanning Results*

*Figure 13-13: MBSA Scanning Results*

In the above figure, the MBSA scanning results show System Information; IIS Scan Results, SQL Server Results, and Desktop Application Results .

## Lab 13-3: Web Server Security Tool

### Procedure:

Using Syhunt Hybrid, go to "Dynamic Scanning". This package also supports Code Scanning and Log Scanning.

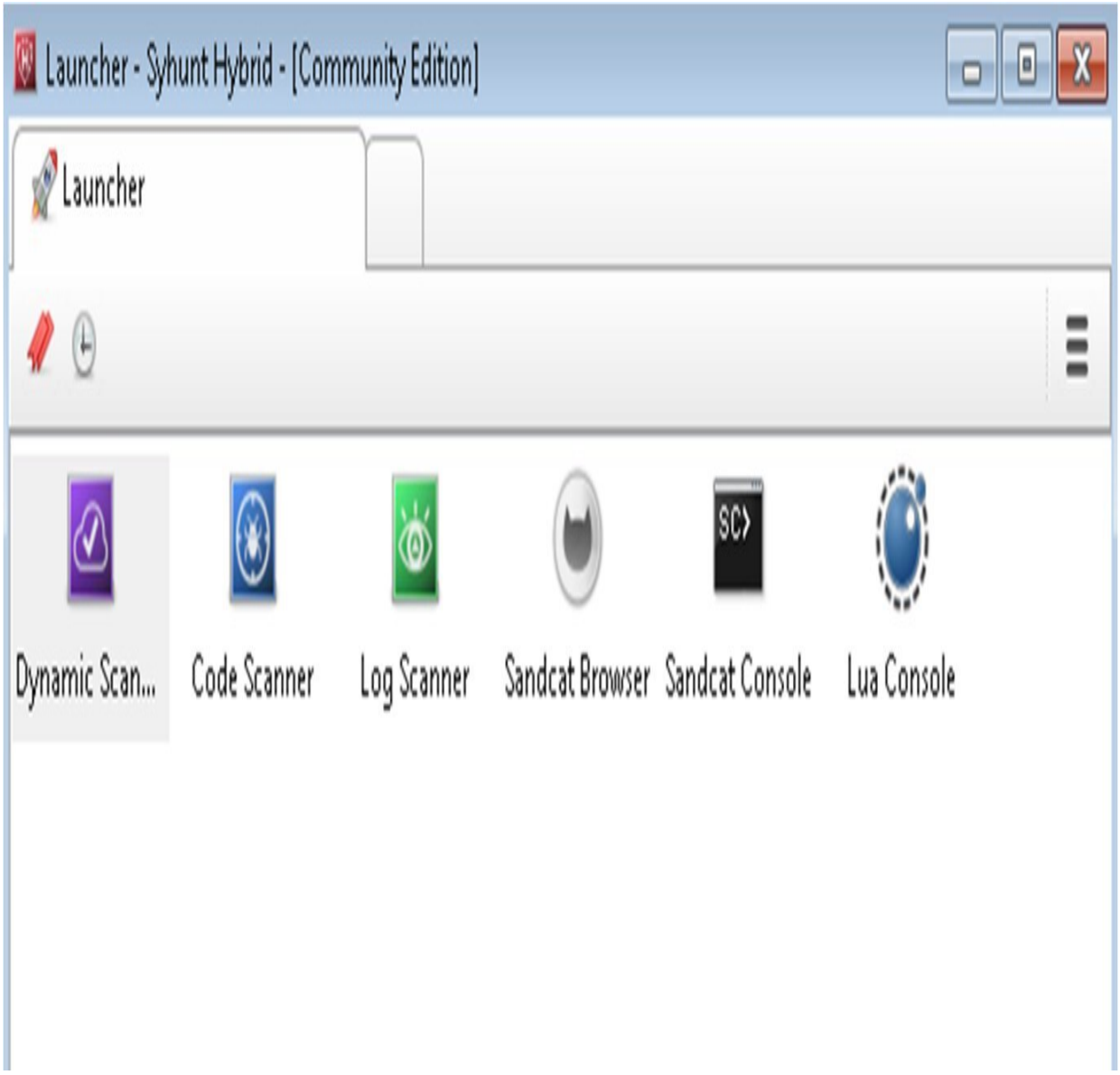



*Figure 13-14: Syshunt Dynamic Scanning*

Enter the URL or IP address.

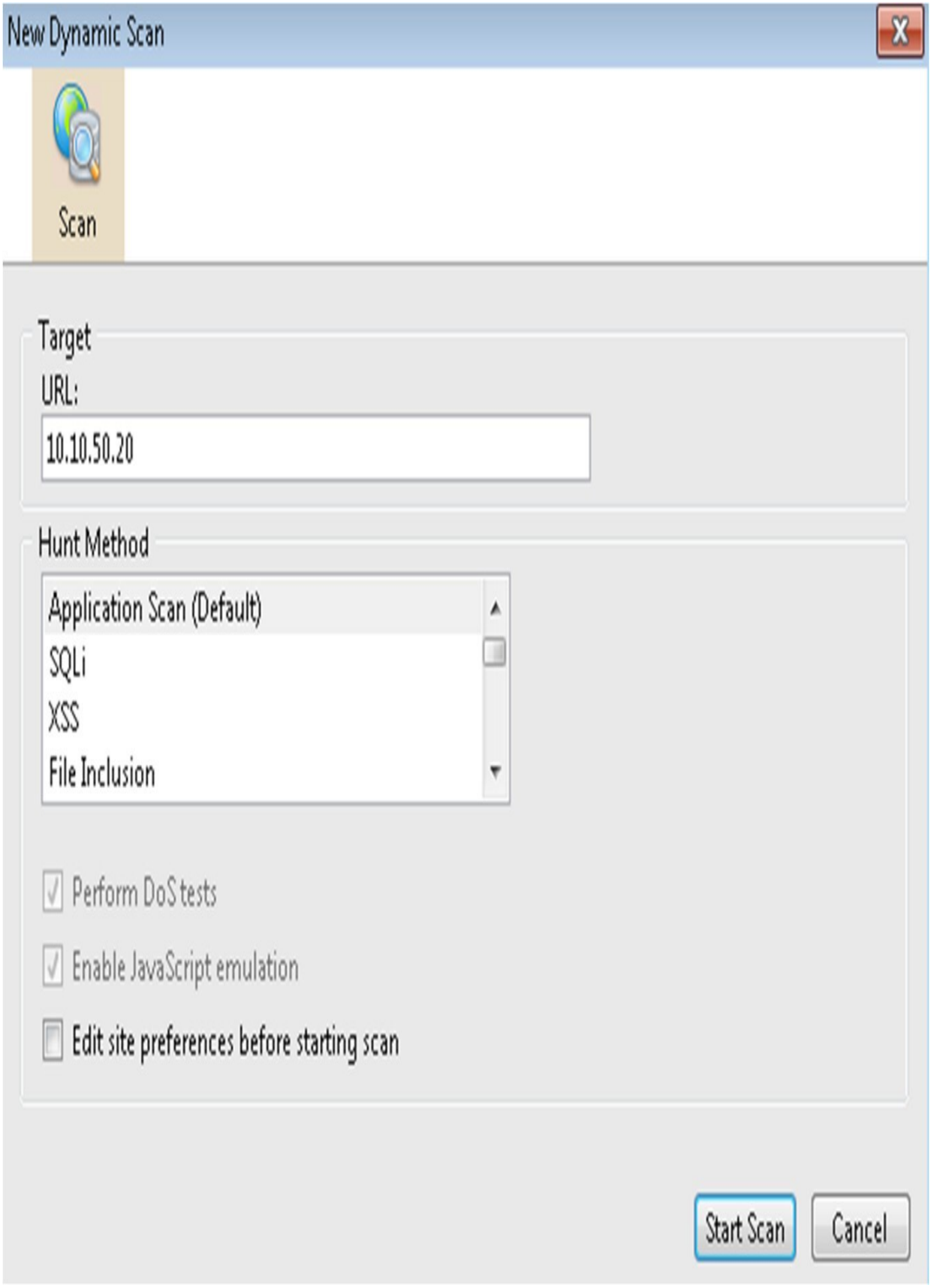

New Dynamic Scan
Scan
Target
URL:
10.10.50.20
Hunt Method
Application Scan (Default)
SQLi
XSS
File Inclusion
Perform DoS tests
Enable JavaScript emulation
Edit site preferences before starting scan
Start Scan
Cancel

*Figure 13-15: Syshunt Dynamic Scanning*

When you see Scanning Results, click on the vulnerability to check the issue and its solution.

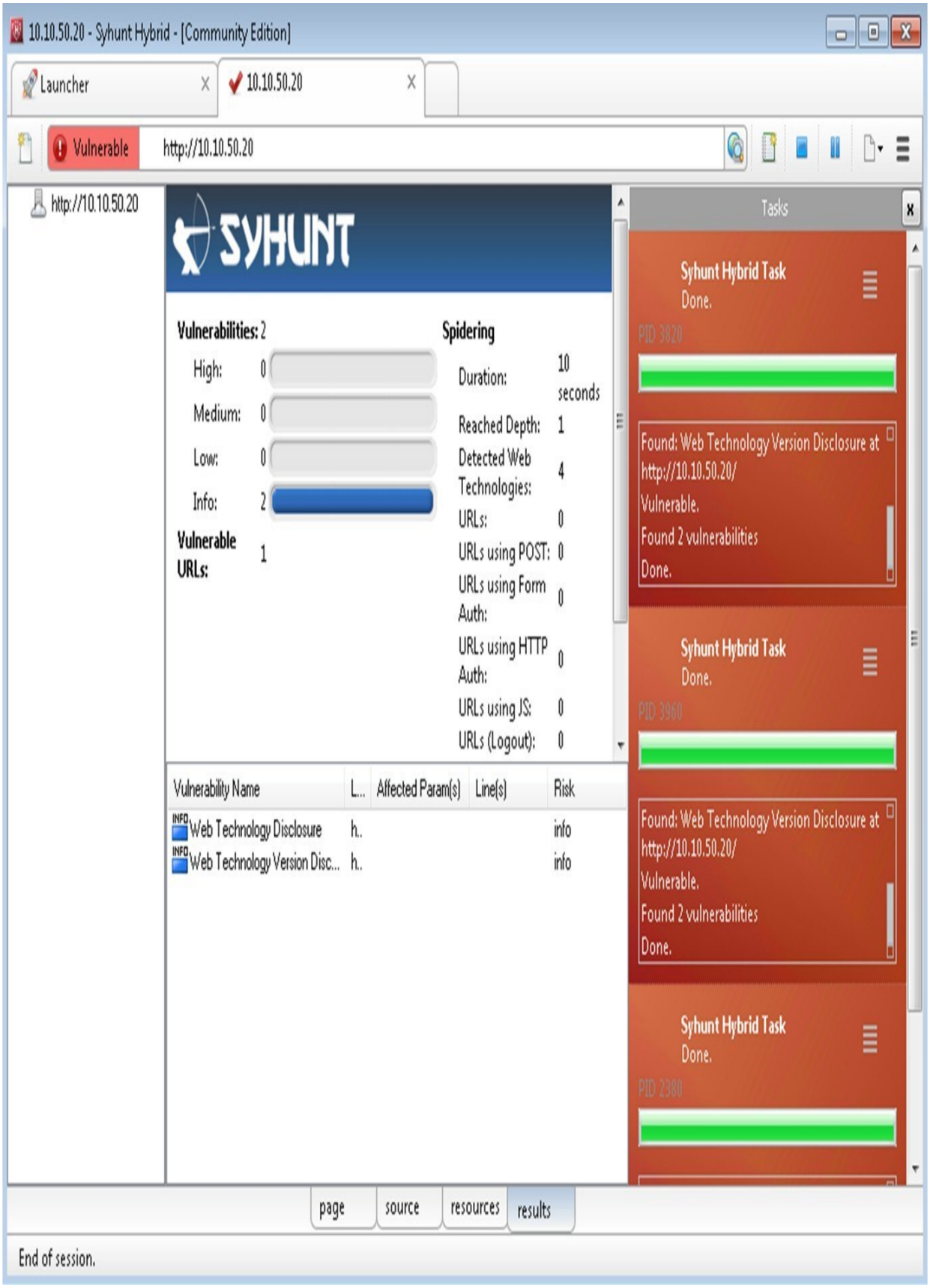

10.10.50.20 - Syhunt Hybrid - [Community Edition]
Launcher
10.10.50.20
Vulnerable
http://10.10.50.20
http://10.10.50.20
SYHUNT
Vulnerabilities: 2
High: 0
Medium: 0
Low: 0
Info: 2
Vulnerable URLs: 1
Spidering
Duration: 10 seconds
Reached Depth: 1
Detected Web Technologies: 4
URLs: 0
URLs using POST: 0
URLs using Form Auth: 0
URLs using HTTP Auth: 0
URLs using JS: 0
URLs (Logout): 0
Vulnerability Name
L...
Affected Param(s)
Line(s)
Risk
Web Technology Disclosure
h..
info
Web Technology Version Disc...
h..
info
Tasks
Syhunt Hybrid Task
Done.
PID 3820
Found: Web Technology Version Disclosure at http://10.10.50.20/
Vulnerable.
Found 2 vulnerabilities
Done.
Syhunt Hybrid Task
Done.
PID 3960
Found: Web Technology Version Disclosure at http://10.10.50.20/
Vulnerable.
Found 2 vulnerabilities
Done.
Syhunt Hybrid Task
Done.
PID 2380
page
source
resources
results
End of session.

*Figure 13-16: Syshunt Dynamic Scanning*

The figure above shows the description of a vulnerability that was detected. The solution tool will provide a recommendation to resolve the issue.

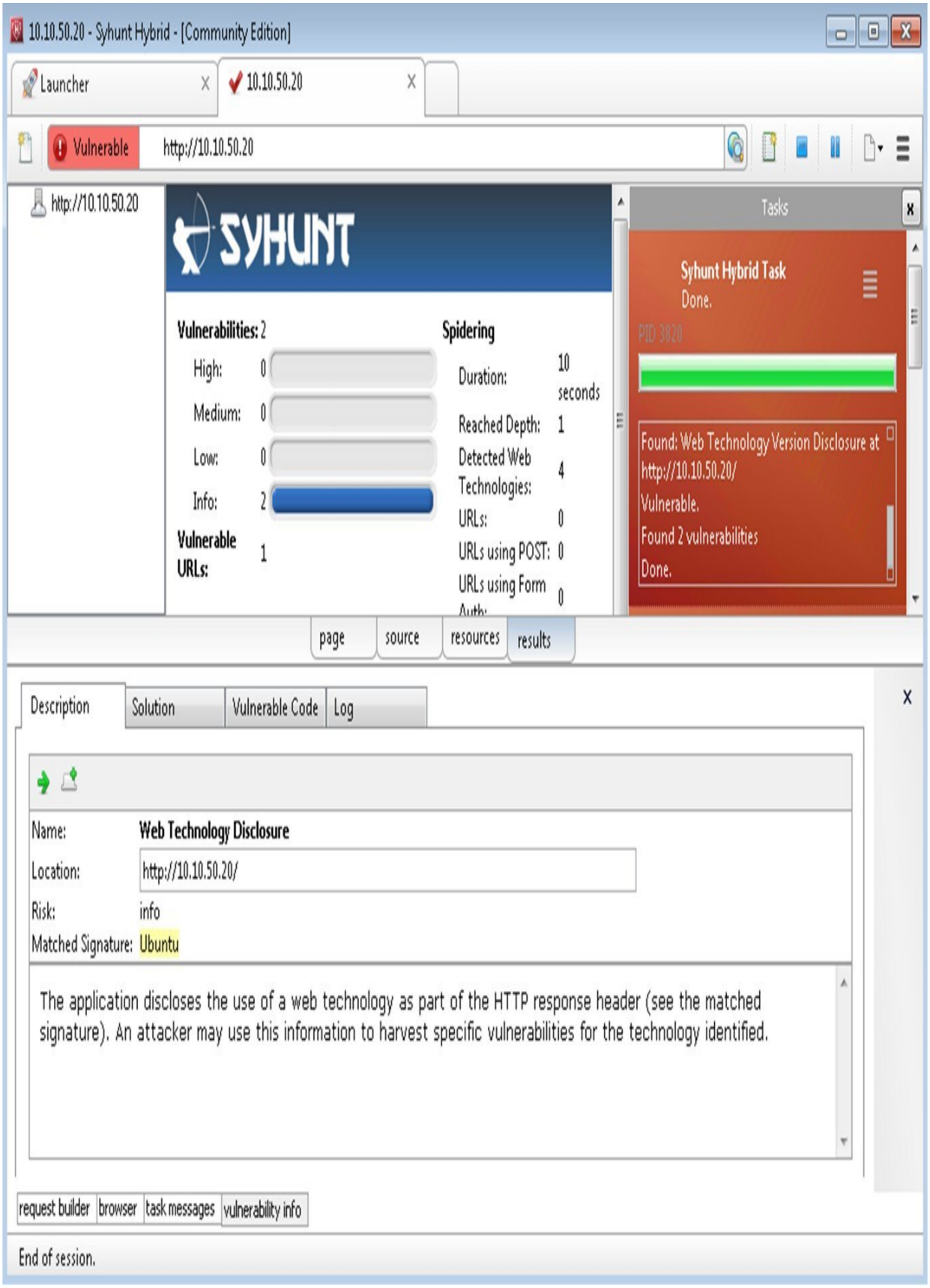

10.10.50.20 - Syhunt Hybrid - [Community Edition]
Launcher
10.10.50.20
Vulnerable
http://10.10.50.20
http://10.10.50.20
SYHUNT
Vulnerabilities: 2
High: 0
Medium: 0
Low: 0
Info: 2
Vulnerable URLs: 1
Spidering
Duration: 10 seconds
Reached Depth: 1
Detected Web Technologies: 4
URLs: 0
URLs using POST: 0
URLs using Form Auth: 0
Tasks
Syhunt Hybrid Task
Done.
PID 3820
Found: Web Technology Version Disclosure at http://10.10.50.20/
Vulnerable.
Found 2 vulnerabilities
Done.
page
source
resources
results
Description
Solution
Vulnerable Code
Log
Name: Web Technology Disclosure
Location: http://10.10.50.20/
Risk: info
Matched Signature: Ubuntu
The application discloses the use of a web technology as part of the HTTP response header (see the matched signature). An attacker may use this information to harvest specific vulnerabilities for the technology identified.
request builder
browser
task messages
vulnerability info
End of session.

*Figure 13-17: Syshunt Dynamic Scanning*

Note: Fuzzy testing is an automated software testing technique that involves providing invalid, unexpected, or random data as input to a computer program and is monitored for exceptions such as crashes, failing built-in code assertions, or potential memory leaks.

Security Identifier (SID) is a user's unique immutable identifier. It consists of a 6-byte identifier authority that is followed by one to fourteen 32-bit sub authority values and ends on a single 32-bit relative identifier (RID). All windows accounts with an RID of 500 are considered as a built-in Administrator accounts in their respective authority.

## Practice Questions

1. Which of the following is not a type of Open Source Web Server architecture?
A. Apache
B. NGINX
C. Lighttpd
D. IIS Web Server
2. An attacker is attempting trial and error method to access restricted directories using dots and slash sequences. Which type of web server attack is this?
A. LDAP Attack
B. AD Attack
C. Directory Traversal Attack
D. SQL Injection

3. An attacker sends a request, which allows him to add header response; now he redirects the user to a malicious website. Which type of attack is this?
A. Web Cache Poisoning
B. HTTP Response Splitting Attack
C. Session Hijacking
D. SQL Injection

4. Update that is specially designed to fix the issue for a live production environment is called:
A. Hotfix
B. Patch
C. Bugs
D. Patch Management

5. A piece of software developed to fix an issue is called: A. Hotfix
B. Patch
C. Bugs
D. Update

6. Which of the following is a Patch Management Tool? A. Microsoft Baseline Security Analyzer
B. Microsoft Network Monitor
C. Syshunt Hybrid
D. SolarWinds SIEM tool

# Chapter 14: Hacking Web Applications

## Technology Brief

A significant increase in the usage of a web application requires it to have high availability and extreme performance. In this modern era, as well as being used globally for social purposes, web applications are popularly used in the corporate sector for carrying out important tasks. It has become a serious challenge for web server and application server administrators to ensure security measures and eliminate vulnerabilities in order to provide high availability and smooth performance.

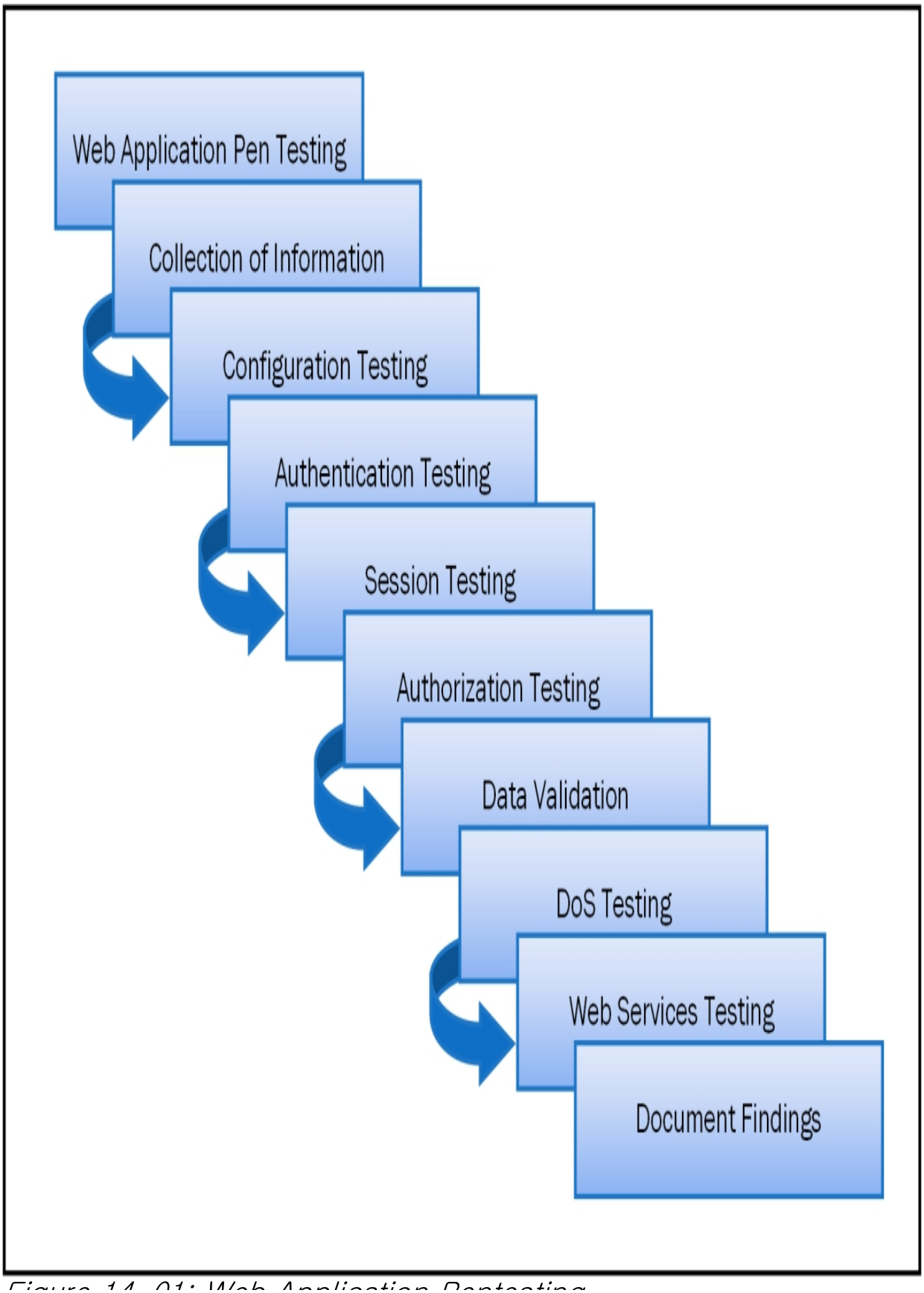



*Figure 14_01: Web Application Pentesting*

*Figure 14-01: Web Application Pentesting*

## Web Application Concepts

Web Applications run on a remote application server and are available for clients over the internet. A web application can be available on different platforms, for example, browsers and software. Use of web applications has increased enormously in the last few years. They are dependent on a Client-Server relationship, and provide an interface to clients to use web services. Web pages may be generated on the server or might contain scripts for dynamic execution on the client web browser.

### *Server Administrator*

The Server Administrator takes care of the safety, security, functionality, and performance of the web server. It is responsible for estimating security measures, deploying security models, and finding and eliminating vulnerabilities.

### *Application Administrator*

The Application Administrator is responsible for the management and configuration required for the web application. It ensures the availability and high performance of the web application.

### *Client*

Clients are those endpoints that interact with the web server or application server to make use of the services offered by the server. These clients require a highly available service from the server at any given time. When the clients access the resources, they use various web browsers that might be risky in terms of security.

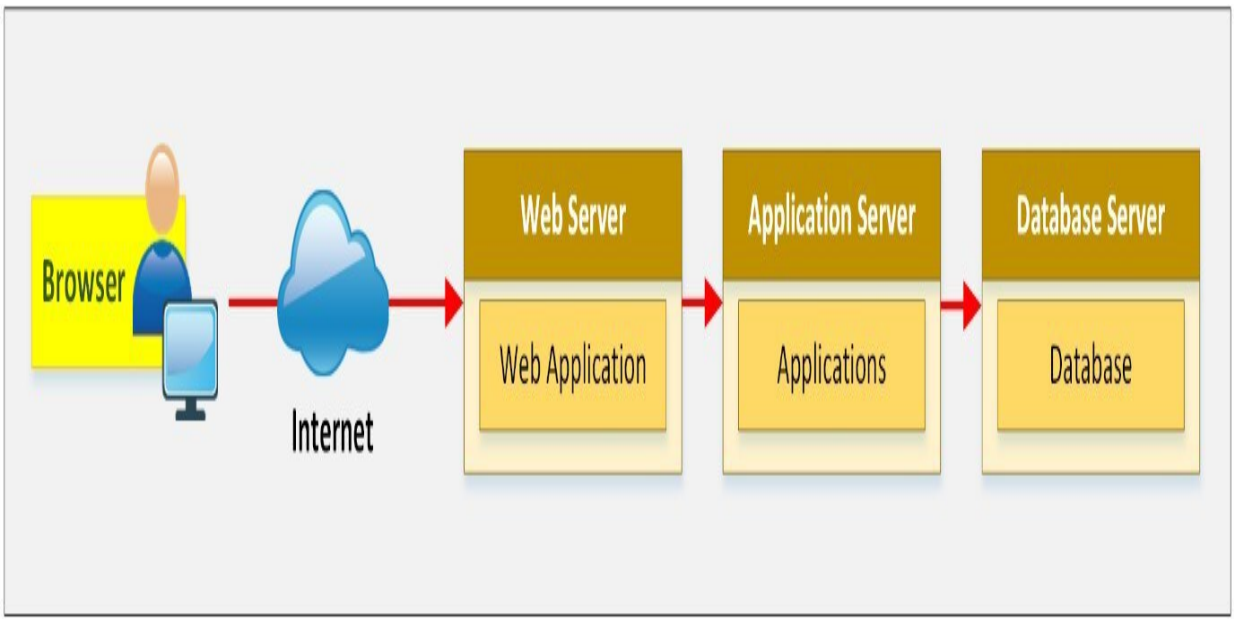



*Figure 14-02: Web Application Architecture*

## *How do Web Applications Work?*

A Web Application functions in two steps, i.e., Front end and Back end. User requests are handled by the front end, where the user interacts with the webpages. Services are communicated to the user from the server through buttons and other controls on the webpage. All processing is controlled and processed on the back end.

Server-side languages include:

- Ruby on Rails
- PHP
- C#
- Java
- Python
- JavaScript

Client-side languages include:

- CSS
- JavaScript
- HTML

A web application works on the following layers:
• *Presentation Layer* : The Presentation Layer is responsible for displaying and presenting information to the user on the client end
• *Logic Layer* : The Logic Layer is used to transform, query, edit, and otherwise manipulate information to and from the forms
• *Data Layer* : The Data Layer is responsible for holding data and information for the application as a whole

## Web 2.0

Web 2.0 is the World Wide Web website generation that provides dynamic and flexible user interaction. It provides ease of use and interoperability between other products, systems, and devices. Web 2.0 allows users to interact and collaborate with social platforms such as social media and social networking sites. The previous generation, i.e. web 1.0, was limited to passive viewing of static content. Web 2.0 offers almost all users the same freedom to contribute. The characteristics of Web 2.0 are rich in user experience and participation, dynamic content, metadata, web standards, and scalability.

## Web App Threats

Threats to Web Application include:

• Cookie Poisoning
• Insecure Storage
• Information Leakage
• Directory Traversal
• Parameter/Form Tampering • DOS Attack
• Buffer Overflow
• Log Tampering
• SQL Injection
• Cross-Site (XSS)
• Cross-Site Request Forgery • Security Misconfiguration • Broken Session Management • DMZ Attacks
• Session Hijacking
• Network Access Attacks

## *UInvalidated Inputs*

UInvalidated Input refers to the processing of non-validated input from the client to a web application or back-end servers. This vulnerability can be exploited to perform XSS, buffer overflow, and injection attacks.

## *Parameter/Form Tampering*

Parameter Tampering refers to an attack in which parameters are manipulated while the client and server are communicating with each other. An attacker modifies parameters such as the Uniform Resource Locator (URL) or web page form fields. In this way, a user may either be redirected to another website, which may exactly look like the legitimate site, or attacker can modify the fields, for example, cookies, form fields, and HTTP Headers.

## *Injection Flaws*

Injection attacks work because of web application vulnerabilities. If a web application is vulnerable enough to allow untrusted input to be executed, then following injection attacks can be performed:

- SQL Injection
- Command Injection
- LDAP Injection

## *SQL Injection:*

SQL Injection is the injection of malicious SQL queries. Using SQL queries, an unauthorized user interrupts the processes, manipulates the database, and executes commands and queries by injection, resulting in data leakage or loss. These vulnerabilities can be detected by using application vulnerability scanners. SQL injection is often executed using the address bar. Attackers bypass the vulnerable application's security and extract the valuable information from its database using SQL injection.

## *Command Injection:*

Command injection can be done with any of the following methods:

- Shell Injection
- File Injection
- HTML Embedding

### *LDAP Injection*

LDAP injection is another technique that takes advantage of a non-validated input vulnerability. An attacker may access the database using a LDAP filter to search the information.

### *Denial-of-Service DoS Attack*

An attacker may perform a DoS attack in the following ways:

#### 1. User Registration DoS

An attacker automates a process of constantly registering with fake accounts

#### 2. Login DoS

An attacker sends repeated login requests

#### 3. User Enumeration

An attacker tries to use different username and password combinations from a dictionary file

#### 4. Account Lockout

An attacker attempts to lock a legitimate account with invalid passwords

## Web App Hacking Methodology

### Analyze Web Applications

Analyzing Web Applications includes observing the functionality and other parameters to identify vulnerabilities, entry points, and server technologies that can be exploited. HTTP requests and HTTP fingerprinting techniques are used to diagnose these parameters.

### Attack Authentication Mechanism

By exploiting the Authentication Mechanism using different techniques, an attacker may bypass the authentication or steal information. Attacking on authentication mechanism includes:

- Username Enumeration
- Cookie Exploitation
- Session Attacks
- Password Attacks

## Authorization Attack Schemes

By accessing the web application using a low privilege account, an attacker can escalate privileges to access sensitive information. Different techniques are used, for example, URL, POST data, Query string, cookies, parameter tampering, HTTP header, etc., to escalate privileges.

## Session Management Attack

As defined earlier, a Session Management Attack is performed by bypassing authentication in order to impersonate a legitimate authorized user. This can be done using different session hijacking techniques such as:

- Session Token Prediction
- Session Token Tampering
- Man-in-the-Middle Attack
- Session Replay

## Perform Injection Attacks

An Injection Attack is the injection of malicious code, commands, and files by exploiting vulnerabilities in a web application. An injection attack may be performed in a different forms, for example:

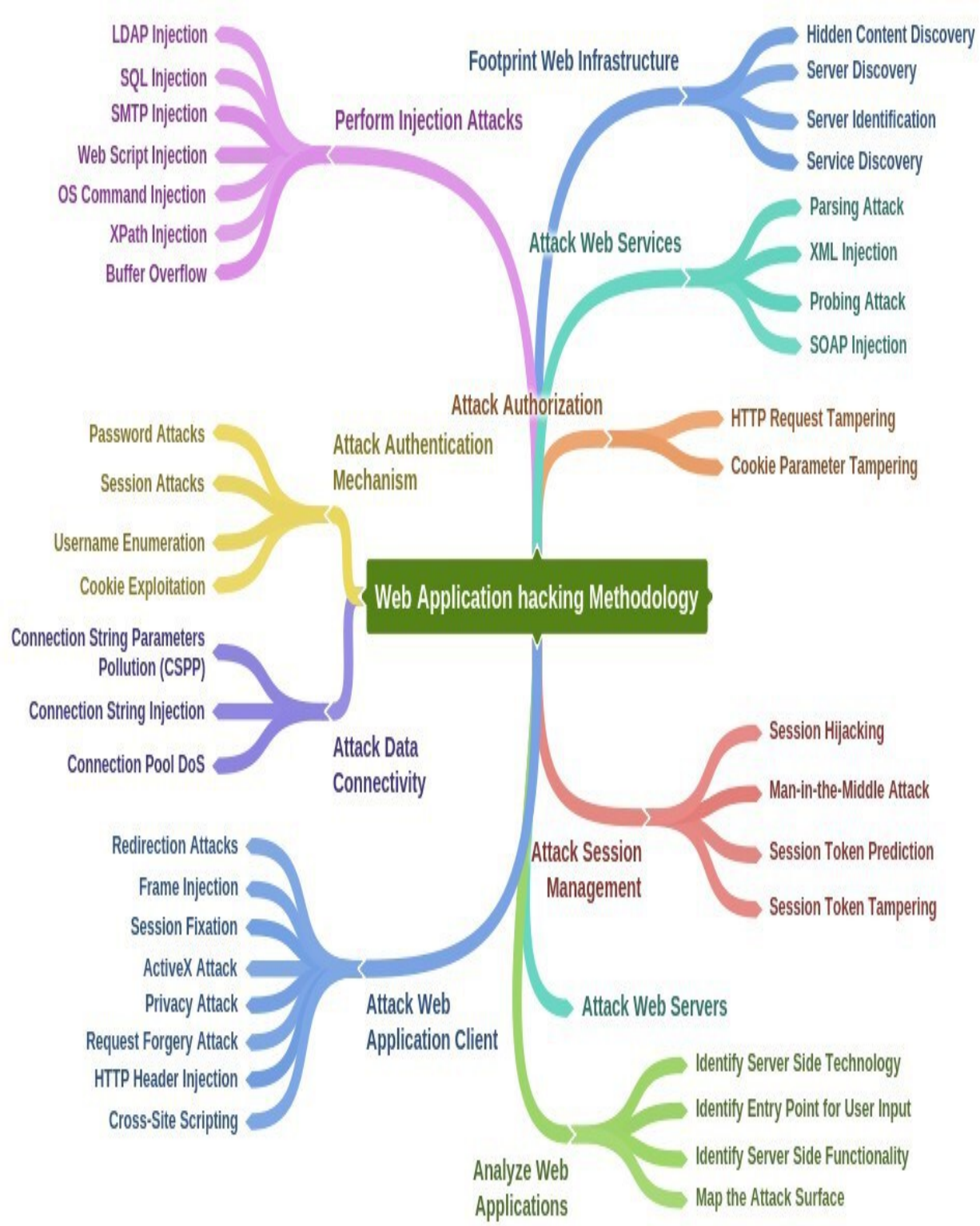
- Web Script Injection
- OS Command Injection
- SMTP Injection
- SQL Injection
- LDAP Injection
- XPath Injection
- Buffer Overflow
- Canonicalization

## Attack Data Connectivity

A Database Connectivity Attack focuses on exploiting the data connectivity between an application and its database. Initiating a connection to the database requires a connection string. A data connectivity attack includes:

1. Connection String Injection
2. Connection String Parameters Pollution (CSPP)
3. Connection Pool DoS

Mind Map


Web Application hacking Methodology
Perform Injection Attacks
LDAP Injection
SQL Injection
SMTP Injection
Web Script Injection
OS Command Injection
XPath Injection
Buffer Overflow
Footprint Web Infrastructure
Hidden Content Discovery
Server Discovery
Server Identification
Service Discovery
Attack Web Services
Parsing Attack
XML Injection
Probing Attack
SOAP Injection
Attack Authorization
HTTP Request Tampering
Cookie Parameter Tampering
Attack Authentication Mechanism
Password Attacks
Session Attacks
Username Enumeration
Cookie Exploitation
Attack Data Connectivity
Connection String Parameters Pollution (CSPP)
Connection String Injection
Connection Pool DoS
Attack Session Management
Session Hijacking
Man-in-the-Middle Attack
Session Token Prediction
Session Token Tampering
Attack Web Application Client
Redirection Attacks
Frame Injection
Session Fixation
ActiveX Attack
Privacy Attack
Request Forgery Attack
HTTP Header Injection
Cross-Site Scripting
Attack Web Servers
Analyze Web Applications
Identify Server Side Technology
Identify Entry Point for User Input
Identify Server Side Functionality
Map the Attack Surface

Note: The Open Web Application Security Project (OWASP) is an online community that produces freely available articles, documentation, methodologies, tools, and technologies in the field of web application security.
WebGoat is an insecure web application maintained by OWASP designed to teach web application security. This program is a demonstration of common server-side application flaws. The exercises are intended to be used by people to learn about application security and penetration testing techniques.

## Secure Application Development and Deployment
## Development of Life Cycle Models

Software production is the result of processes that involve tasks such as requirement gathering, planning, designing, coding, testing, and supporting. These tasks are performed according to the process model enabled by the team members.

Two of these are discussed below.

### *Waterfall Model*

One of the frameworks of application development is the Waterfall Model, which is a "sequential design process". In this process each step is taken sequentially, that is, the second step follows completion of the first, the third step follows completion of the second, and so forth. The Waterfall model can be implemented in multiple ways, but they all follow similar steps.

Some of the most common advantages and disadvantages of the Waterfall model are as follows:

Pros Cons

It is a sequential approach

Emphasizes methodical record keeping and documentation Clients know what is expected at every step

Strong documentation results in less hassle

Developers cannot go back to the previous step to make changes, i.e. every step is final

A fault in instructions can result in havoc as the project depends upon the initial input and instruction
Only at the end of the sequence is the test performed
Change implementation can be a nightmare for developers
*Table 14-01: Pros and Cons of the Waterfall Model*
A common framework for application development:

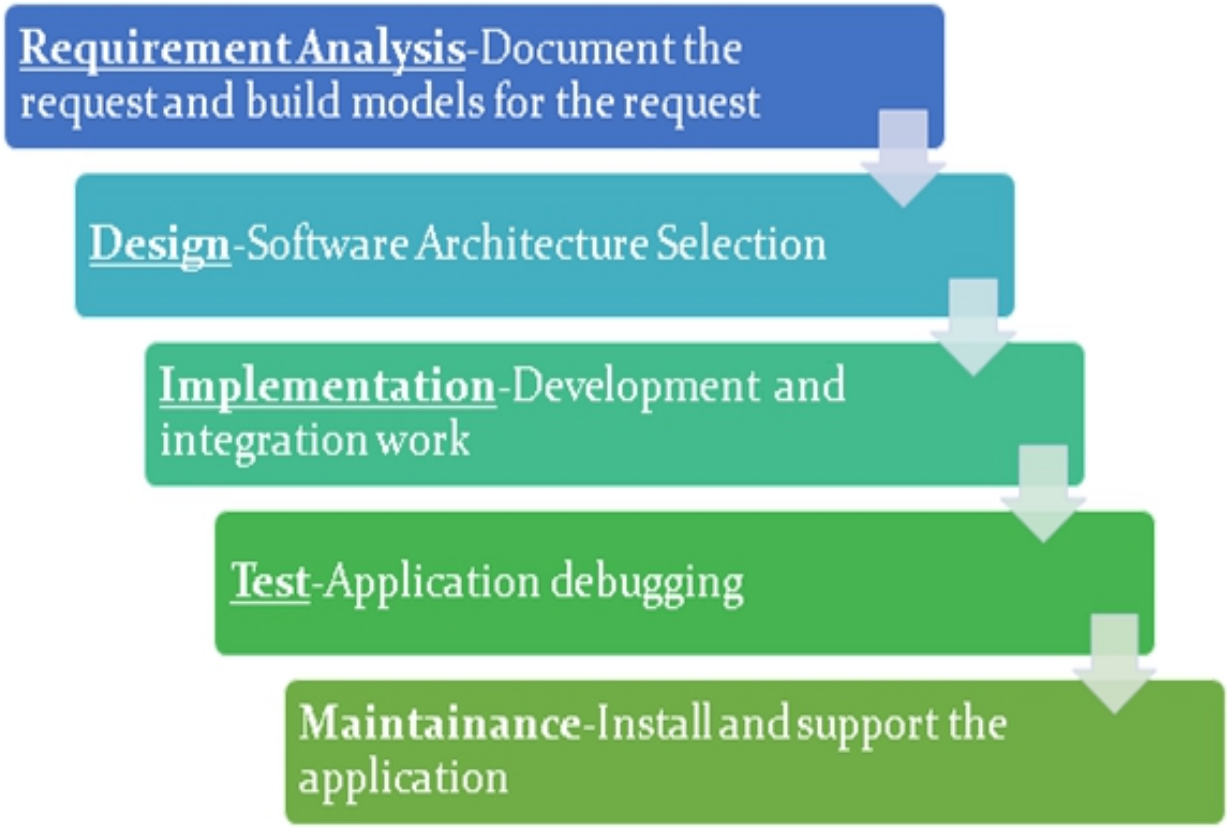



*Figure 14-03: The Waterfall Model*

### *Agile Model*

In the Agile Model, no sequential path is followed. Instead, multiple tasks are performed simultaneously in development. An advantage of the Agile model is that making changes is easy, i.e. the development process in the Agile model is continuous.

The two major forms of Agile development are as follows:
- Scrum
- Extreme Programming (XP)

Some of the most common advantages and disadvantages of the Agile model are as follows:

Pros Cons
It is a team-based approach

It allows us to make changes Mismanagement could lead toward code sprints with no ends

The final project could be completely different from a planned project
At every step, testing can be performed
Simultaneous testing helps in launching a project quickly Impossible for an outsider to tell who is working on what
Lack of emphasis on documentation
*Table 14-02: Pros and Cons of the Agile Model*

## Secure DevOps

### *Security Automation*

Automation is the key element of DevOps and it relies on automation for most of its efficiencies. Security automation, as the name refers, automatically handles security related tasks.

### *Continuous Integration*

Continuous Integration (CI) in DevOps refers to the continuous upgrading and improvement of the production code base. Through high-level automation and safety nets, CI permits DevOps team members to update and test minor changes without much overhead.

### *Baselining*

Standardizing performance and functionality at a certain level is known as Baselining. This provides a reference point when changes are made, which is why it is so important to DevOps and security. Reference points are used to represent the improvements with each change. At the time of a major change or development, it is important for the development team to baseline the system.

### *Immutable System*

A system that is never patched or upgraded once it is deployed is known as an Immutable System. If upgrading is needed, the system is simply replaced with a new patched or upgraded system. In a typical system (changeable system), it is difficult to perform authorized software and system updates and lock down directories at the same time. This is because updating the system creates temporary files in the directories and these directories contain some files that should never be modified. The immutable system resolves this problem.

### *Infrastructure as Code*

Infrastructure as Code, or programmable infrastructure, refers to the usage of code to build a system, although a normal configuration mechanism is used to manually configure the code.

Infrastructure as code is a way of using automation to build out a system that is reproducible and efficient. It is considered a key attribute of enabling the best practices in DevOps.

## Version Control and Change Management

Changes like bug fixes, security patches, the addition of new features, etc. in an application are guaranteed by the vendor. During the application development process, multiple changes need to be implemented, and that requires version control.

### *Version Control*

Version Control tracks changes and can also revert back to see what changes have been made. This version control feature is used in multiple software, as well as in the Operating System, cloud-based files, and wiki software. It is also important from a security perspective because it identifies required modification with respect to time.

## Provisioning and De-Provisioning

Provisioning refers to “making something available”, for example, deploying an application. Necessary provisioning includes the web

server, database server, certificate updates, user workstation configuration, etc.

De-provisioning is the process of removing an application. An important factor related to the de-provisioning of an application is that every instance of the application needs to be removed and verified.

## Secure Coding Techniques

### *The Basic Concept of Secure Coding*

The security of an application starts with code that is secure and free from any vulnerability. However, all codes have vulnerabilities and weaknesses, thus the goal is to create code that maintains a desired level of security and possesses an effective defense against vulnerability exploitation.

A secure application can be created if configuration, errors, and exceptions are handled properly. The security risk profile of a system can be determined if an application is tested throughout the Software Development Life Cycle (SDLC).
Software Development Life Cycle Methodology (SDLM) possesses elements that can assist in secure code development. Some of the SDLM processes that can improve code security are as follows:

- Cross-Site Scripting
- Cross-Site Request Forgery
- Input Validation
- Error and Exceptional Handling

### *Proper Error Handling*

Encountering errors and exceptions in an application is common and needs to be handled in a secure manner. One attack methodology forces an error to move applications from normal to exceptional handling. If the exception handling is incorrect, it can lead to a wide range of disclosures. For example, SQL errors disclose data elements and structures, RPC (Remote Procedure Call) errors can disclose sensitive information such as server, filename, and path, and

programmatic errors can disclose information such as stack element or the line number on which an exception occurred.

## *Proper Input Validation*

As we move toward web-based applications, errors have shifted from buffer overflow to input handling issues. In order to prevent malicious attacks, it is a developer's duty to handle input properly. A buffer overflow may be considered improper input, but recent attacks include arithmetic and canonicalization attacks. Input Validation is the most important mechanism that can be employed for defense.

Many attacks based on common vulnerabilities can be mitigated if all the inputs are hostile before validation. The following are vulnerabilities that require input validation as a defense mechanism:

- Cross-Site Scripting
- Cross-Site Forgery Attack
- Buffer Overflow
- Incorrect Calculation of Buffer Size
- Path Traversal
- In Security Decision, Reliance on Untrusted Inputs

## *Stored Procedure*

Stored Procedure is a method in which developer prepares SQL query and saves the query to be reused repeatedly in the program. To understand the purpose of stored procedure, consider a scenario where you have to run a SQL query multiple times. It is better to store the query or queries as a stored procedure and just use it where required. The security also improves; this way the user input is isolated from execution of SQL query. In other words, it is the primary mechanism of defense against an SQL injection attack. The stored procedure has better performance than other data access forms, and that is why many major database engines support it.
Example: The following SQL statement creates a stored procedure named "FinanceRecords" that selects all records from the "Clients" table:

```
CREATE PROCEDURE FinanceRecords
AS
SELECT * FROM Clients;
```

Execute the stored procedure above as follows:

```
EXEC FinanceRecords;
```

## *Code Signing*

A mechanism performed by the end user to verify code integrity is Code Signing. It applies a digital signature to the code for code integrity verification. In addition, it provides evidence as to the source of the software. It relies on established (Public Key Infrastructure) PKI, and the developer needs a pair of keys to decrypt the data. The public key is recognized by the end user and needs to be signed by the certification authority.

## *Encryption*

To have secure and usable encryption in an application, it is necessary to adopt and utilize a proven algorithm and code bases.

## *Obfuscation*

Obfuscation is also known as Camouflage, meaning *"to hide the obvious meaning of observation".* Obfuscation is added to the system so that it becomes difficult for an attacker to understand and exploit it.

Obfuscation works well for data names or other such exposed elements, but it does not work well for code construction. Obfuscated code is not just hard but almost impossible to read; an example of such code is the ticking time bomb. These are some of the reasons the construction of code is considered inconvenient.

| Original Source Code Before Rename Obfuscation | Reverse-Engineered Source Code After Rename Obfuscation |
| --- | --- |
| `private void CalculatePayroll(SpecialList employeeGroup) {`<br>`  while(employeeGroup.HasMore()) {`<br>`  employee = employeeGroup.GetNext(true);`<br>`  employee.UpdateSalary();`<br>`  DistributeCheck(employee);`<br>`  }`<br>`}` | `private void a(a b) {`<br>`  while (b.a()) {`<br>`  a = b.a(true);`<br>`  a.a();`<br>`  a(a);`<br>`  }`<br>`}` |

*Figure 14-04: Example of Code Obfuscation*

## *Code Re-Use/ Dead Code*

### 1. Code Re-use

Code Re-use, or use of old code or components, such as libraries or common functions etc., reduces development costs and time. However, massive re-use of code also results in a ripple effect across the application. Therefore, it is necessary for the development team to decide about the appropriate level of code re-use. Code re-use is preferred for a complex function like cryptography.

The challenge with the re-use of code is that if the old code contains vulnerabilities, reusing the code will transfer those vulnerabilities to other applications. Another challenge with code re-use is the symptoms of dead code.

### 2. Dead Code

The result produced by dead code is never used anywhere in an application while it may be executed, which simply means that the machine runs the executables (code is executed), thereby making it a

dead code. Almost every code has security problems. Therefore, an application can be made more secure by removing dead code.

## *Validation*

### 1. Server-Side Validation

Data validation can be done at multiple places, for example, on the server. This is known as server-side validation. In server-side validation, all checks occur on the server itself.

### 2. Client-Side Validation

As the name implies, this validation process occurs at the front end of the application, that is, the client side. It helps in filtering legitimate input from a genuine user and also benefits the user by providing additional speed.

Note : Validation on the server side is always needed, but it is best if both the validation, i.e. Server-Side Validation and Client-Side Validation are used.

## *Memory Management*

Memory Management refers to those actions required to coordinate and control computer memory, assigning memory to variables, and reclaiming it when no longer needed. Memory management errors lead to the memory leak problem. The process of clearing memory that is no longer in use is called Garbage Collection. Programming languages such as Java, Python, C#, and Ruby provide automatic garbage collection, but where there is no automatic garbage collector, for example, in C programming, the programmer has to allocate free memory.

## *Use of Third-Party Libraries and SDKs*

To extend the functionality of a programming language, third-party libraries, and Software Development Kits (SDKs) are used.

## *Data Exposure*

During operation, loss of data control is known as Data Exposure. Protection of data is very important so it must always be protected at

every step of a process, for example, during communication or transmission, during use, and when at rest, that is, during storage.

It is the responsibility of the programming team to chart the data flow and to ensure protection from data exposure. Exposed data can result in confidentiality failure (data can be lost to an unauthorized person) and integrity failure (data can be changed by an unauthorized person).

## Code Quality and Testing

Application developers use tools and techniques to assist them in testing and checking the security level of code. Code analysis is performed to find weaknesses and vulnerabilities. This analysis can be performed either dynamically or statically.

### *Code Analysis*

Code Analysis is the process of inspecting vulnerabilities and weaknesses in code. It is divided into two types, i.e., Static and Dynamic. Static analysis examines code without executing the program, whereas dynamic analysis examines code during execution.

### *Code Testing*

Code Testing is the process verifying that the code meets the functional requirements as laid out in the business requirement process.

### *Static Code Analyzer*

Static Code Analysis can be performed on both source and object code. It is used when the code is examined without executing the program. It can be performed both by tools and manually. However, it is usually performed with tools because tools can be used against any form of code base. Various names are given to these tools, for example, static code analyzer, source code analyzer, or sometimes binary code scanner or bytecode scanner.

### *Dynamic Analysis*

Dynamic Analysis is performed on an emulated or target system while executing the software. Dynamic analysis requires specialized

automation to perform specific testing. A brute-force method that addresses vulnerabilities and input validation issues is known as Fuzzing (Fuzz Testing).

### *Stress Testing*

Finding bugs is not the only objective of the performance testing. It also includes finding performance factors and tailbacks. Stress Testing basically increases the load of an application to see what happens. This can lead to unintended results such as error messages, kernel or memory dumps, and exposure of application details not intended to be shown to users. Options stress testing are:

■ Automate Individual Workstation
■ Simulate Large Workstation Loads
And in both the cases, extensive reports, response times and results are generated describing how the application is affected by the stress test.

### *Sandboxing*

Executing code in an environment that isolates the target system and code from direct contact is called Sandboxing. A sandbox is used for the execution of unverified and untrusted code. A sandbox works just like a virtual machine and can mediate a number of system interactions, for example, accessing memory, network access, and accessing other programs, devices, and file systems. A sandbox offers protection at a level depending on the mediation offered and the isolation level.

### *Model Verification*

Model verification ensures that code is doing what it is supposed to do. In model verification, the program results are matched with the desired design model. This testing process consists of two steps, validation and verification.

#### Verification
Verification is a process that checks whether the software is working

properly, whether there are any bugs to address, or whether the product meets the model specifications. Validation

Validation refers to the process of determining whether an application meets certain requirements, including high-level requirements, secure software building requirements, security requirements, and compatibility. It also investigates whether or not the product is right for an organization.

## Compiled vs Runtime Code

When the source code is compiled into an executable, it is called Compiled Code. Once the code is compiled, the source code becomes hidden (you do not see it). During the process of compilation, any bugs and errors that can be resolved by recompiling the code are identified by the compiler. After fixing the bugs, an error-free application can be developed.
Many software applications that we use are runtime code, for example, the PHP code of PHP-based applications. In runtime code, the source code is viewable and it executes at the time an application initially runs. This means that there is no compiler to check for bugs; they are only found when the code is executing. It differs from compiled code because in a compiled code, the errors and bugs are identified before providing the application to the end user.

## An Overview of Federated Identities
## Server-based Authentication

Web communication is stateless communication because every command or request is unique, which means that it has no link to the preceding request or command. This is why authentication through the web is a challenge. So the question here is, how can we extend the authentication of a previous request ?

Conventionally, this is achieved through Server-based Authentication. In server-based authentication, the server has a record of the login. A session ID is granted to every user during login, and when the user sends a request, the server checks the session validity. This process

adds overhead and ends in scalability issues to the server as the users increase.

The process of Server-based Authentication is as follows:

1. When the client logs in to the session, information is received by the server.
2. The server checks the session information when the client sends an application request.
3. If the session information is authentic, the feedback is sent to the client.

## Token-based Authentication

Like web communication (HTTP), this is also a stateless-based authentication. In this authentication process, session information is not saved on the server. Instead, the server sends a token to the client and the client stores that token. The token is moved with the request when the client makes a subsequent request. The server checks the validity of the token, and, if valid, the server then responds accordingly to the client. This process is secure because the token expires after a certain amount of time. It is also scalable because now the session information is kept by the client and not by the server.

The process of token-based authentication is as follows:

1. The client logs in to the server.
2. After investigating the validity of the authentication process, a token is sent to the client.
3. The client sends that token along with the application request.
4. If the token is valid, the server responds to the client.

## Federation

Federation is a system that grants access to other users who may not have local login. It means a single token is given to the user who is entrusted or authenticated across various systems, just like in SSO (Single Sign-On). A federated network is created by third parties so that users can log in with separate credentials, for example, Facebook

credentials, Twitter credentials, etc. Before establishing a federated network, the third party has to create a trust-based relationship.

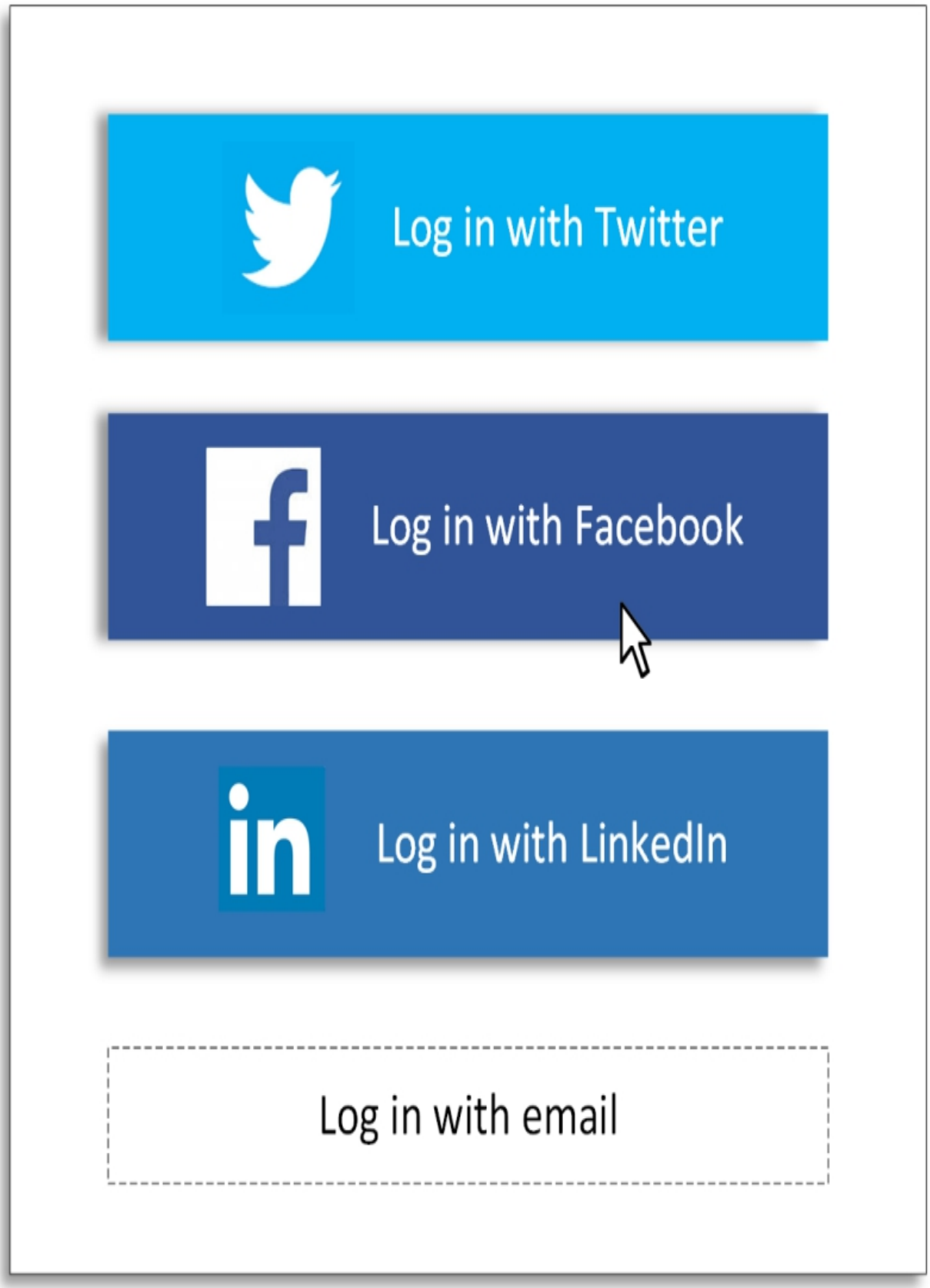



*Figure 14.05: Example of Federation*

*Figure 14-05: Example of Federation*

## Security Assertion Mark-up Language (SAML)

SAML is an open standard authentication and authorization method. The user is authenticated for achieving entry to local sources through a third party. Shibboleth software is an example of SAML. It is a security concern that modern mobile networks do not have SAML support.

## OAuth

This was introduced by Google, Twitter, and other parties. It serves as an authorization to the resources a user can gain access to. OAuth is usually used by Facebook, Google, etc. It is not a protocol for authentication and just provides authorization between applications. OAuth is combined with OpenID Connect (handles SSO) and then OAuth decides what resources a user may gain access to.

## Important Considerations for Best Practices

### Encoding Schemes

Web applications use different encoding schemes for securing their data. There are two categories of encoding scheme.

### URL Encoding

URL Encoding is an encoding technique for secure handling of a URL. In URL encoding, the URL is converted into ASCII format for secure transportation over HTTP. Unusual ASCII characters are replaced by ASCII code, and a "%" is followed by two hexadecimal digits. The default character set in HTML5 is UTF8. Table 14-0 1 shows some symbols and their codes.

### *HTML Encoding*

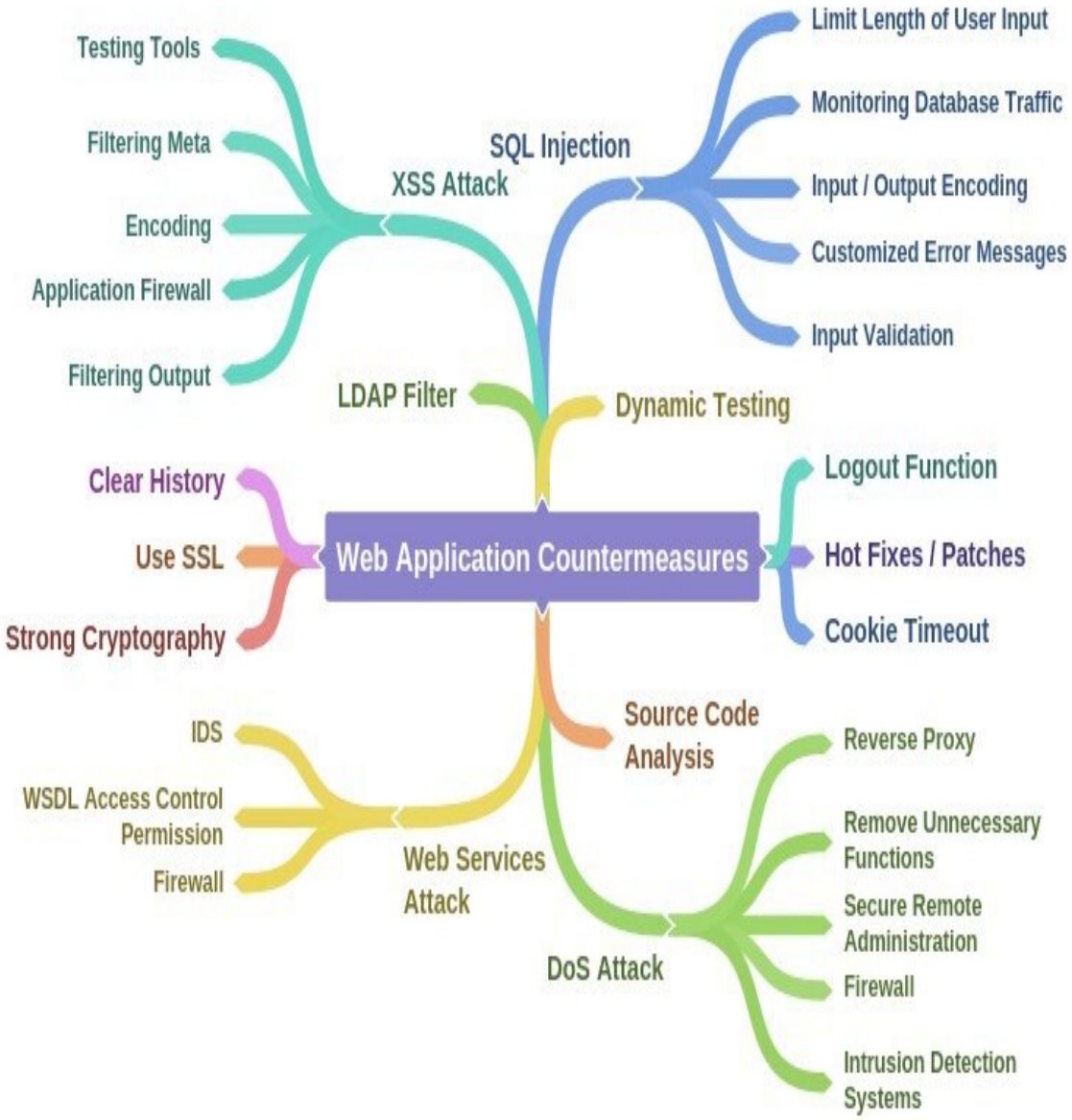
Similar to URL encoding, HTML Encoding is a technique representing unusual characters with an HTML code. ASCII, which was the first character-encoding standard, supports 128 different alphanumeric characters. Other techniques such as ANSI and ISO8859- 1 support 256. UTF8 (Unicode) covers almost every character and symbol.

For HTML 4:
<meta http-equiv="Content-Type"
content="text/html;charset=ISO8859- 1"> For HTML5: <meta
charset="UTF8">

Character From Windows 1252 From UTF8

space !
"
#
$
%
&
%20 %20 %21 %21 %22 %22 %23 %23 %24 %24 %25 %25 %26
%26

*Table 14-03: Encoding Schemes*
Mind Map




## Practice Questions

1. An individual who is responsible for the management and configuration required for the web application is called:
A. Server Administrator
B. Network Administrator
C. Application Administrator
D. DC Administrator

2. Which of the following is not a Back-end Programming Language? A. PHP

B. CSS
C. JavaScript
D. Python

3. Which of the following is not a Front-end Programming Language? A. HTML
B. JavaScript
C. CSS
D. C#

4. Web Application's working is categorized into the following three basic layers: A. Presentation layer
B. Logic Layer
C. Data Layer
D. Transport Layer

5. An attacker has accessed the web application. Now, he is escalating privileges to access sensitive information. Which type of web application attack is this? A. An Attack on the Authentication Mechanism
B. Authorization Attack
C. Session Management Attack
D. Injection Attack

6. Which of the following is not appropriate for Data Connectivity Attack between application and its database?
A. Connection String Injection
B. Connection String Parameters Pollution
C. Connection pool DoS
D. Canonicalization

# Chapter 15: SQL Injection

## Technology Brief

This chapter covers Structured Query Language (SQL) Injection. SQL Injection is a popular and complex method of attack on web services, applications, and databases. It requires deep knowledge about web

application processes and its components such as databases and SQL. SQL injection is the insertion of malicious code or scripts by exploiting vulnerabilities to launch an attack powered by back-end components. This chapter gives information about SQL injection, types, methodology and defence techniques.

## SQL Injection Concepts

SQL Injection Attack uses SQL websites or web applications. It relies on the strategic injection of malicious code or scripts into existing queries. This malicious code is drafted with the intention of revealing or manipulating data stored in the tables within a database.

SQL injection is a powerful and dangerous attack. It identifies the flaws and vulnerabilities in a website or application. The fundamental concept of SQL injection is to inject commands to reveal sensitive information from the database. Hence, it can result in a high profile attack.

## The scope of SQL Injection

SQL Injection can be a serious threat to a website or application. The impact of an SQL injection can be measured by observing the following parameters that an attacker attempts to affect:

- Bypassing Authentication
- Revealing Sensitive Information
- Compromising Data integrity
- Erasing the Database
- Remote Code Execution

## How SQL Query Works

An attacker executes an SQL injection query the server, which sends a response. For example, an attacker requests the following SQL query to the server.

```
SELECT * FROM [Orders]
```

These commands will reveal all information stored in the database Orders table. If an organization maintains records of their orders in a

database, an attacker can download all the information kept in this database table using this command.

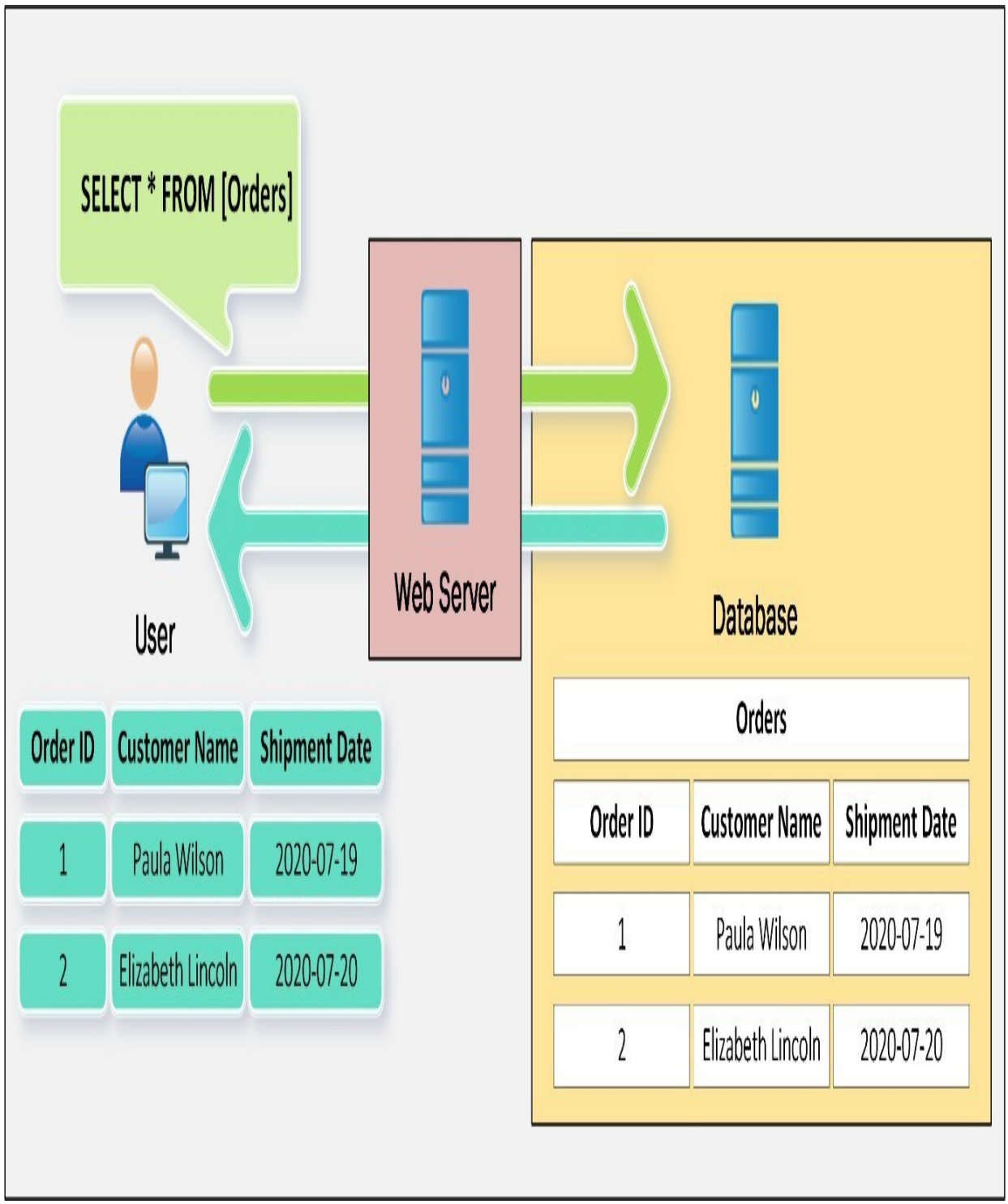



*Figure 15-01: SQL Query Working*

## *SQL Delete Query*

The DELETE statement is used to delete existing records in a table. To understand this further, consider the table Customers in a database as shown below:

Customer ID Customer Name City
1 Maria Anders London
2 Alfreds Futterkiste Prague
3 Elizabeth Brown Paris
4 Ana Trujillo New York
5 Thomas Hardy Boston
*Table 15-01: Database Before a Delete Query*

Execution of the delete command will erase the record.
DELETE FROM Customers
WHERE CustomerName='Alfreds Futterkiste';
Now the database table will be like this:

CustomerID CustomerName City 1 Maria Anders London
3 Elizabeth Brown Paris
4 Ana Trujillo New York
5 Thomas Hardy Boston *Table 15-02: Database After a Delete Query*

## *SQL Update Query*
The UPDATE statement is used to modify existing records in a table. For example, consider the following command:

UPDATE Customers
SET ContactName = 'IPSpecialist, City= 'Frankfurt' WHERE CustomerID = 1;

Now the database will be:

CustomerID CustomerName 1 IP Specialist
3 Elizabeth Brown
4 Ana Trujillo
5 Thomas Hardy

City Frankfurt Paris New York Boston

*Table 15-03: Database After an Update Query*

### *SQL Injection Tools*

There are several tools available for SQL injection, for example:

- BSQL Hacker
- Marathon Tool
- SQL Power Injector
- Havij

## Types of SQL Injection

SQL injection is classified in three major categories:

1. In-band SQLi
2. Inferential SQLi
3. Out-of-band SQLi

## In-band SQL Injection

In-band SQL Injection includes injection techniques that use the same communication channel to launch an injection attack and to gather information from the response. Inband injection techniques include:

1. Error-based SQL Injection
2. Union-based SQL Injection

### *Error-based SQL Injection*

Error-based SQL Injection is an in-band SQL injection technique. It relies on error messages from the database server to reveal information about the structure of the database. Error-based SQL injection is very useful for an attacker to enumerate an entire database. Error messages are useed during the development phase to troubleshoot issues. These messages should be disabled when an application website is live. Errorbased SQL injection can be performed using the following techniques:

- System Stored Procedure
- End of Line Comment
- Illegal/Logically incorrect Query
- Tautology

## *Union SQL Injection*

Union-based SQL Injection is another in-band SQL injection technique that involves using the UNION SQL operator to combine the results of two or more SELECT statements into a single result.
SELECT <column_name(s)> FROM <table_ 1>
UNION
SELECT <column_name(s)> FROM <table_2>;

## Inferential SQL Injection (Blind Injection)

In an Inferential SQL Injection, no data is transferred from a web application. These are referred to as Blind Injections because the attacker is unable to see the results of an attack; he/she simply observes the behavior of the server. The two types of inferential SQL injection are Boolean-based Blind SQL Injection and Time-based Blind SQL Injection.

## *Boolean Exploitation Technique*

Blind SQL injection is the technique of sending a request to a database. As the response is either true or false, it does not contain any database data. By observing the HTTP response, the attacker can evaluate it and infer whether the injection was successful or unsuccessful.

## Out-of-band SQL Injection

Out-of-band SQL Injection is a technique that uses different channels to launch the injection and to gather the response. It requires some features to be enabled, for example, DNS or HTTP requests on database server, hence, it is not very common.

## SQL Injection Methodology
## Information Gathering and SQL Injection Vulnerability Detection

In the Information Gathering phase, information about the web application, Operating System, database, and the structure of the components is collected. Evaluation of the extracted information is

useful for identifying vulnerabilities that can be exploited. Information can be gathered by using different tools and techniques, such as injecting code into the input fields to observe the response of error messages. Evaluation of the input fields, hidden fields, get and post requests, cookies, string values, and detailed error messages can reveal enough information to initiate an injection attack.

## Launch SQL Injection Attacks

An appropriate SQL injection attack can be initiated just after gathering information about the structure of a database and the vulnerabilities found. An injection succeeds by exploiting them. SQL injection attacks such as union SQL injection, error-based SQL injection, blind SQL injection, and others can be used to extract information from a database such the database name, tables, columns, rows, and fields. The injection can also bypass authentication.

## Advanced SQL Injection

Advanced SQL injection may include an enumeration of databases such as MySQL, MSSQL, MS Access, Oracle, DB2, or Postgre SQL, tables and columns in order to identify users' privilege levels, account information of the database administrator, and database structure disclosure. It can also include password and hash grabbing and transferring the database to a remote machine.

## Evasion Techniques

In order to secure a database, it is recommended that deployment is isolated in a secure network location with an Intrusion Detection System (IDS). IDS continually monitors the network and host traffic as well as database applications. The attacker has to evade IDS to access the database, using different evasion techniques. IDS using Signaturebased Detection System, for example, compares the input strings against the signature to detect intrusion. Now, all an attacker has to do is evade signature-based detection.

## Types of Signature Evasion Techniques

The techniques below are used for evasion:

- Inserting Inline Comments between Keywords • Character Encoding
- String Concatenating
- Obfuscating Codes
- Manipulating White Spaces
- Hex Encoding
- Sophisticated Matches

## Countermeasures

Several detection tools are available to mitigate SQL injection attacks. These tools test websites and applications, report the data and issues, and take remediation action. Some of these advanced tools also offer a technical description of the issue.

## Lab 15- 1: Using IBM Security AppScan Standard

### Procedure:

1. Download and install IBM Security AppScan Standard.
2. Open the application.
3. Select “Create New Scan”.

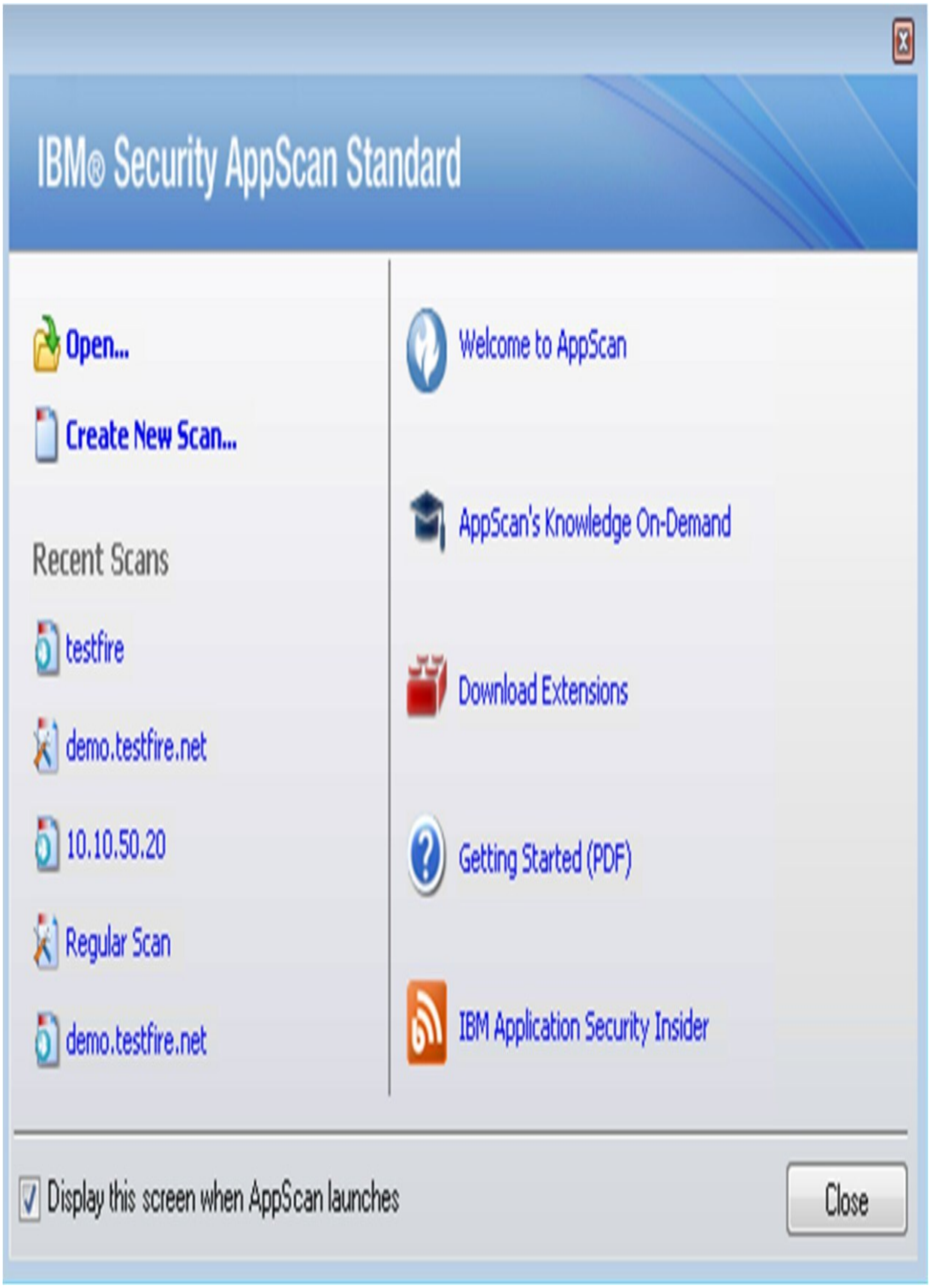



*Figure 15-02: IBM Security AppScan Standard*

*Figure 15-02: IBM Security AppScan Standard*

4. Select scan template and the regular scan will start a new scan. In our case, we are using the pre-defined template demo.testfire.net .

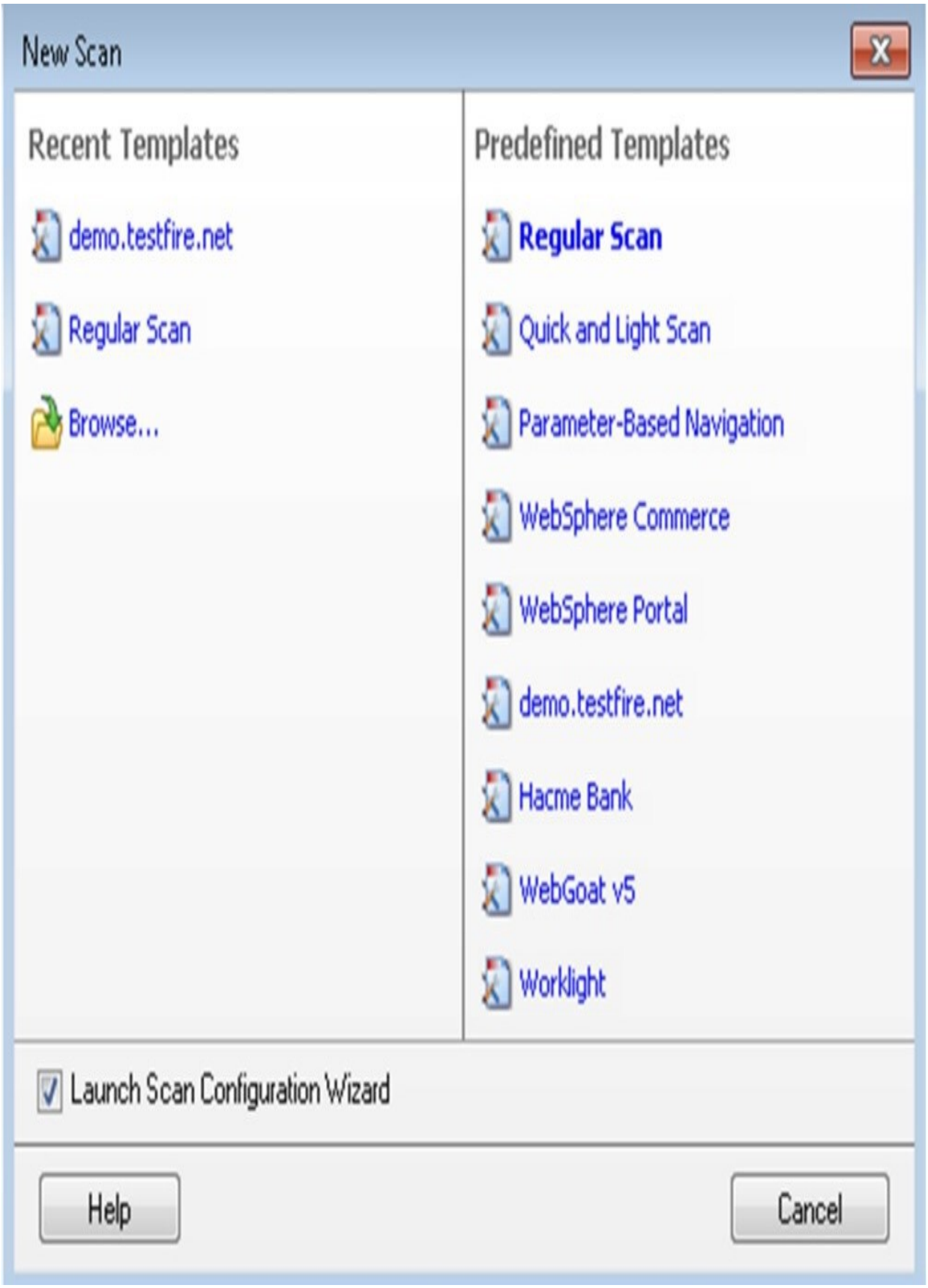

New Scan
Recent Templates
demo.testfire.net
Regular Scan
Browse...
Predefined Templates
Regular Scan
Quick and Light Scan
Parameter-Based Navigation
WebSphere Commerce
WebSphere Portal
demo.testfire.net
Hacme Bank
WebGoat v5
Worklight
Launch Scan Configuration Wizard
Help
Cancel

*Figure 15-03: New Scan*

5. Click “Next”.
6. If you want to edit the configuration, click “Full Scan Configuration”.

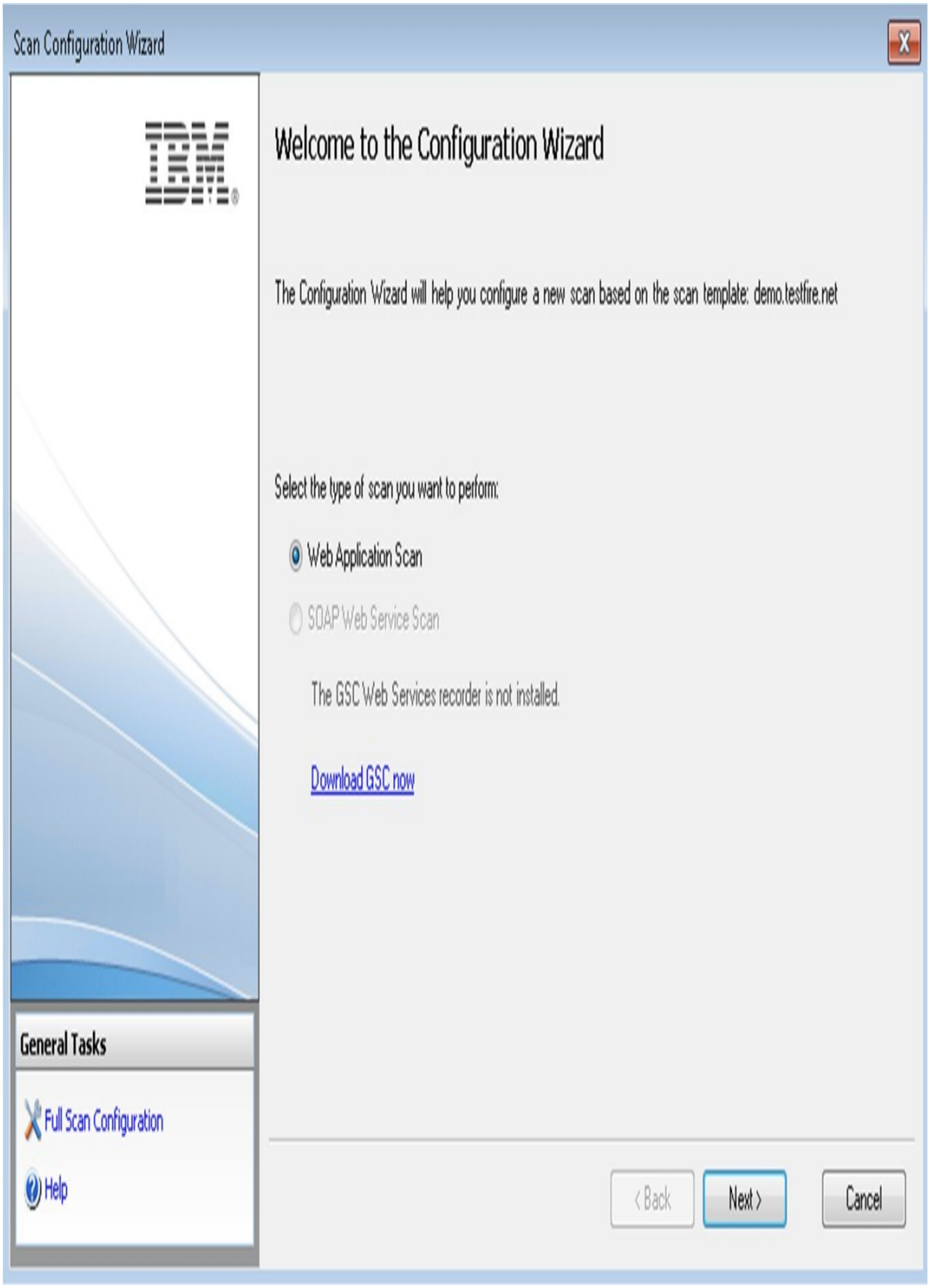

Scan Configuration Wizard
Welcome to the Configuration Wizard
The Configuration Wizard will help you configure a new scan based on the scan template: demo.testfire.net
Select the type of scan you want to perform:
Web Application Scan
SOAP Web Service Scan
The GSC Web Services recorder is not installed.
Download GSC now
General Tasks
Full Scan Configuration
Help
< Back
Next >
Cancel

*Figure 15-04: Configuration Wizard*

7. Click “Next”.

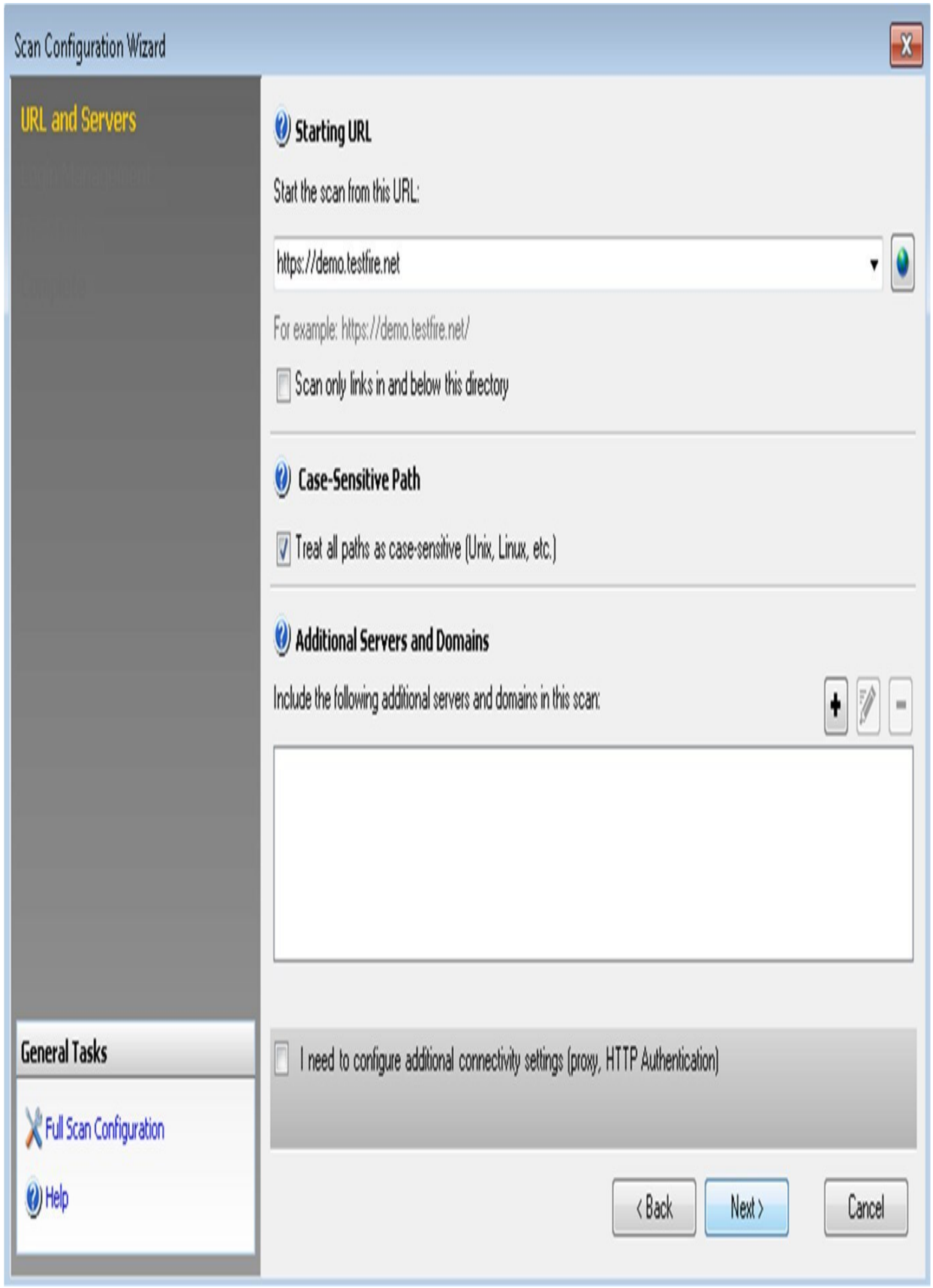

Scan Configuration Wizard
URL and Servers
Starting URL
Start the scan from this URL:
https://demo.testfire.net
For example: https://demo.testfire.net/
Scan only links in and below this directory
Case-Sensitive Path
Treat all paths as case-sensitive (Unix, Linux, etc.)
Additional Servers and Domains
Include the following additional servers and domains in this scan:
General Tasks
Full Scan Configuration
Help
I need to configure additional connectivity settings (proxy, HTTP Authentication)
< Back
Next >
Cancel

*Figure 15-05: Configuration Wizard*

8. Select “Login Method”.

*Figure 15-06: Configuring Login Method*

9. Select “Test Policy”.
10. Click “Next”.


Scan Configuration Wizard
URL and Servers
Login Management
Test Policy
Test Policy
Default
Use this Test Policy for the scan.
This policy includes all tests except invasive and port listener tests.
Policy Files
Recent Policies
Default
Browse...
Predefined Policies
Default
Application-Only
Infrastructure-Only
Third Party-Only
General Tasks
Full Scan Configuration
Help
Send tests on login and logout pages
Do not send session identifiers when testing login pages.
< Back
Next >
Cancel

*Figure 15-07: Configuration Scan Policy*

11. Select how you want to start the scan.
12. Click “Finish”.

*Figure 15-08: Configuration Wizard*

13. You may ask to save the file in the directory. 14. Start the scan.

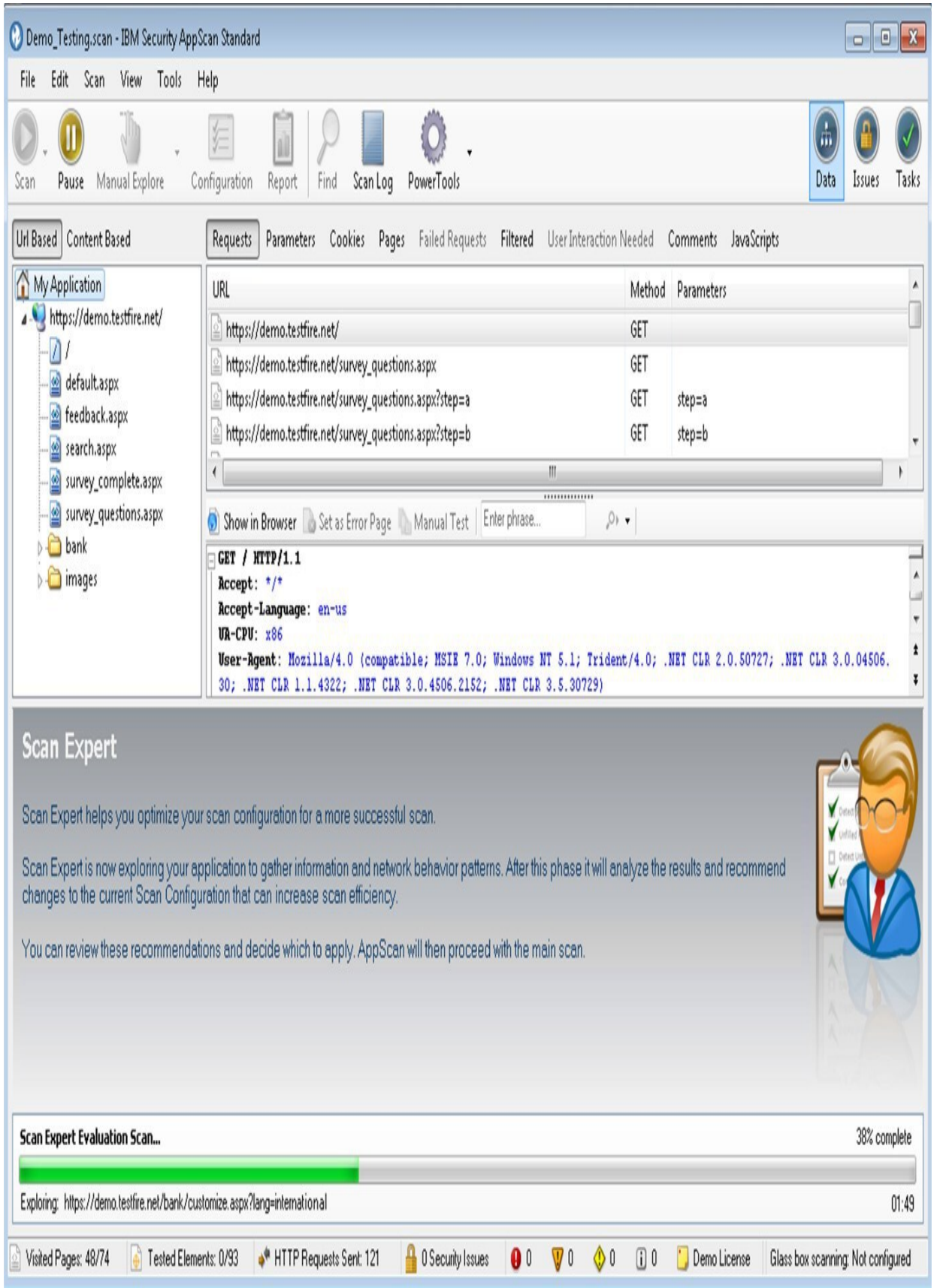

Demo_Testing.scan - IBM Security AppScan Standard
File Edit Scan View Tools Help
Scan Pause Manual Explore Configuration Report Find Scan Log PowerTools
Data Issues Tasks
Url Based Content Based
Requests Parameters Cookies Pages Failed Requests Filtered User Interaction Needed Comments JavaScripts
My Application
https://demo.testfire.net/
/
default.aspx
feedback.aspx
search.aspx
survey_complete.aspx
survey_questions.aspx
bank
images
URL Method Parameters
https://demo.testfire.net/ GET
https://demo.testfire.net/survey_questions.aspx GET
https://demo.testfire.net/survey_questions.aspx?step=a GET step=a
https://demo.testfire.net/survey_questions.aspx?step=b GET step=b
Show in Browser Set as Error Page Manual Test Enter phrase...
GET / HTTP/1.1
Accept: */*
Accept-Language: en-us
UA-CPU: x86
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0; .NET CLR 2.0.50727; .NET CLR 3.0.04506.30; .NET CLR 1.1.4322; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729)
Scan Expert
Scan Expert helps you optimize your scan configuration for a more successful scan.
Scan Expert is now exploring your application to gather information and network behavior patterns. After this phase it will analyze the results and recommend changes to the current Scan Configuration that can increase scan efficiency.
You can review these recommendations and decide which to apply. AppScan will then proceed with the main scan.
Scan Expert Evaluation Scan...
38% complete
Exploring: https://demo.testfire.net/bank/customize.aspx?lang=international
01:49
Visited Pages: 48/74
Tested Elements: 0/93
HTTP Requests Sent: 121
0 Security Issues
0 0 0 0
Demo License
Glass box scanning: Not configured

*Figure 15-09: Scanning*

15. The data pane shows the data scanned during the process.
16. In our case, we are using a demo test, which does not find any issue.

*Figure 15-10: Result - Data Tab*

17. If it does find an issue, the Issue section will show the detected issues list. 18. To explore, click the security issue to reveal the details.

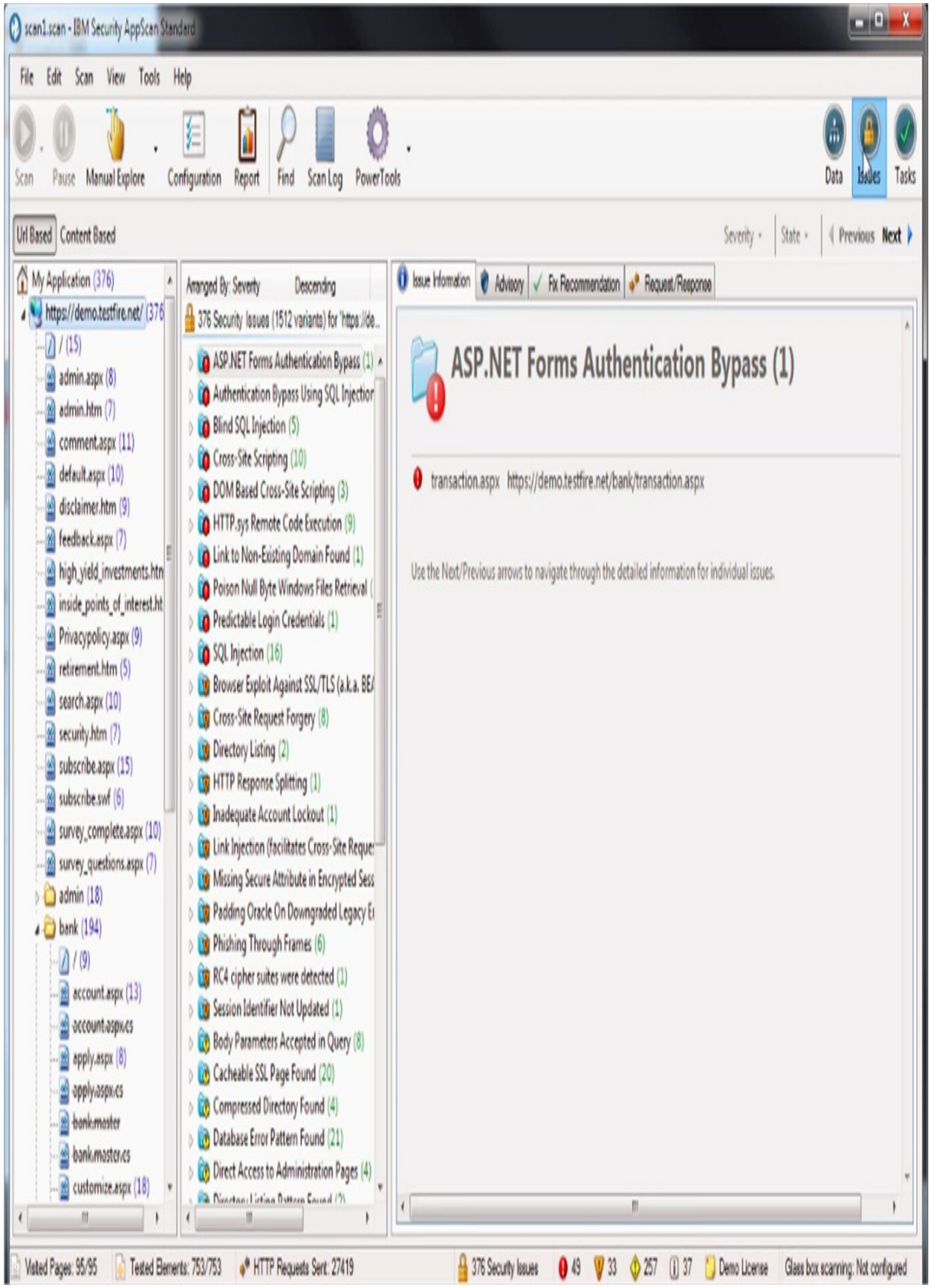
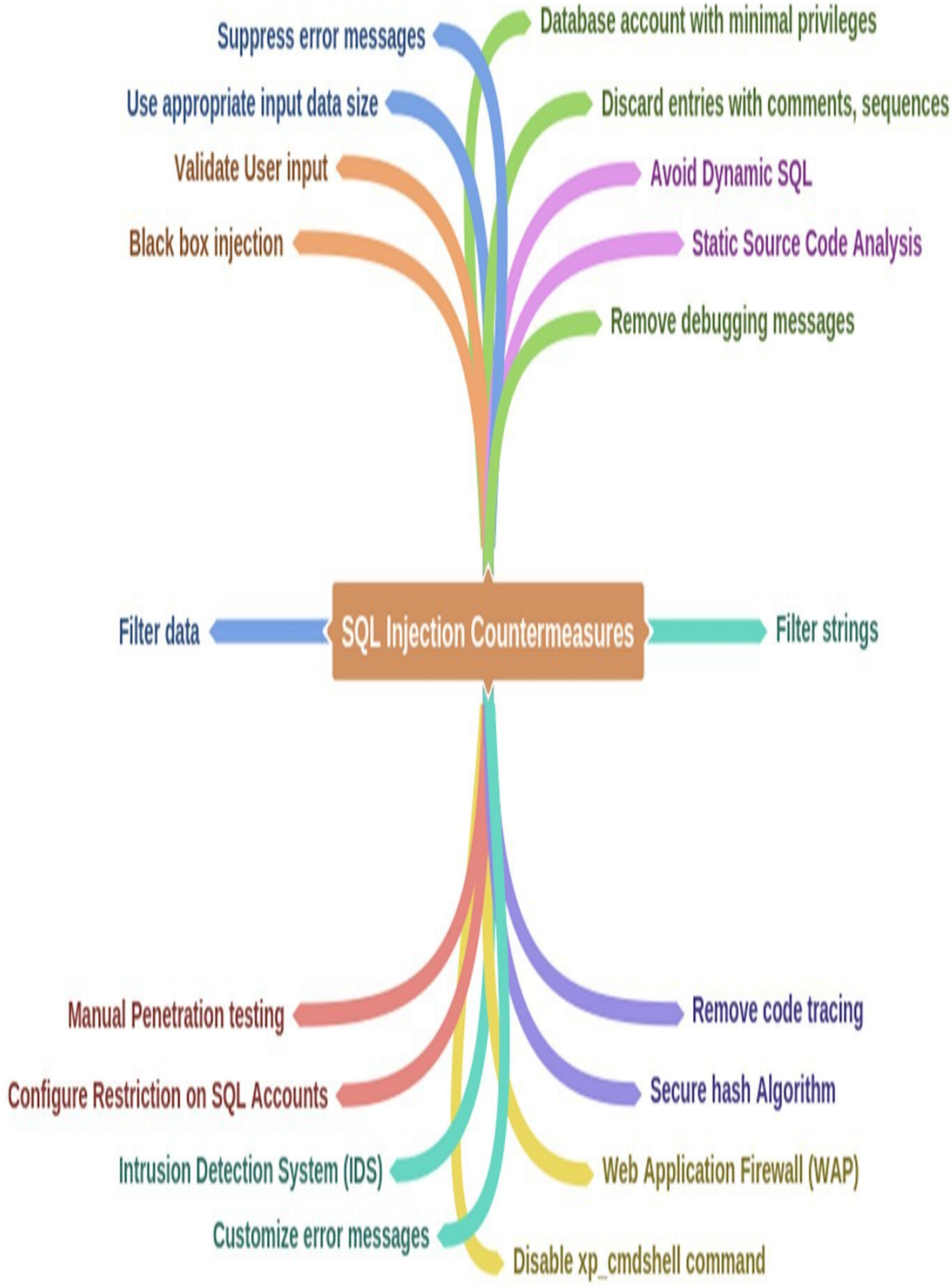

scan1.scan - IBM Security AppScan Standard
File Edit Scan View Tools Help
Scan Pause Manual Explore Configuration Report Find Scan Log PowerTools
Data Issues Tasks
Url Based Content Based
Severity State Previous Next
My Application (376)
https://demo.testfire.net/ (376)
/ (15)
admin.aspx (8)
admin.htm (7)
comment.aspx (11)
default.aspx (10)
disclaimer.htm (9)
feedback.aspx (7)
high_yield_investments.htm
inside_points_of_interest.ht
Privacypolicy.aspx (9)
retirement.htm (5)
search.aspx (10)
security.htm (7)
subscribe.aspx (15)
subscribe.swf (6)
survey_complete.aspx (10)
survey_questions.aspx (7)
admin (18)
bank (194)
/ (9)
account.aspx (13)
account.aspx.cs
apply.aspx (8)
apply.aspx.cs
bank.master
bank.master.cs
customize.aspx (18)
Arranged By: Severity Descending
376 Security Issues (1512 variants) for "https://de...
ASP.NET Forms Authentication Bypass (1)
Authentication Bypass Using SQL Injection
Blind SQL Injection (5)
Cross-Site Scripting (10)
DOM Based Cross-Site Scripting (3)
HTTP.sys Remote Code Execution (9)
Link to Non-Existing Domain Found (1)
Poison Null Byte Windows Files Retrieval
Predictable Login Credentials (1)
SQL Injection (16)
Browser Exploit Against SSL/TLS (a.k.a. BE
Cross-Site Request Forgery (8)
Directory Listing (2)
HTTP Response Splitting (1)
Inadequate Account Lockout (1)
Link Injection (facilitates Cross-Site Reque
Missing Secure Attribute in Encrypted Sess
Padding Oracle On Downgraded Legacy E
Phishing Through Frames (6)
RC4 cipher suites were detected (1)
Session Identifier Not Updated (1)
Body Parameters Accepted in Query (8)
Cacheable SSL Page Found (20)
Compressed Directory Found (4)
Database Error Pattern Found (21)
Direct Access to Administration Pages (4)
Issue Information Advisory Fix Recommendation Request/Response
ASP.NET Forms Authentication Bypass (1)
transaction.aspx https://demo.testfire.net/bank/transaction.aspx
Use the Next/Previous arrows to navigate through the detailed information for individual issues.
Visited Pages: 95/95 Tested Elements: 753/753 HTTP Requests Sent: 27419
376 Security Issues 49 33 257 37 Demo License Glass box scanning: Not configured

*Figure 15-11: Issue Tab*

19. If you have detected an issue, the Task section will show the recommended remediation actions.

*Figure 15-12: Task Tab*

## Mind Map


Suppress error messages
Database account with minimal privileges
Use appropriate input data size
Discard entries with comments, sequences
Validate User input
Avoid Dynamic SQL
Black box injection
Static Source Code Analysis
Remove debugging messages
Filter data
SQL Injection Countermeasures
Filter strings
Manual Penetration testing
Remove code tracing
Configure Restriction on SQL Accounts
Secure hash Algorithm
Intrusion Detection System (IDS)
Web Application Firewall (WAP)
Customize error messages
Disable xp_cmdshell command