

# Web Application and Server Hacking

Chapter 7

- Web Server Architecture
- Web Server Defacement
- Web Server Attack
- Web Application Attack
- Defend and Countermeasures

# Introduction

# Web server market share

Apache  
IIS

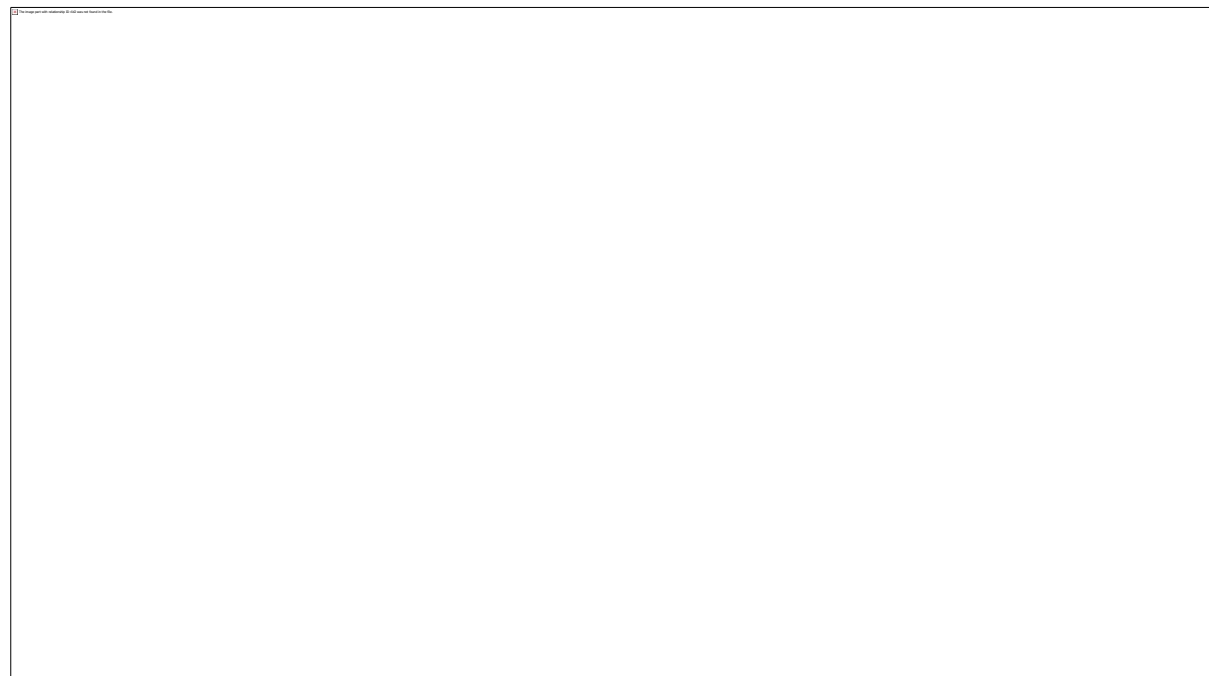


# Web Server Architecture

- Open Source web Server
  - Linux - the server's operating system
  - Apache - the web server component
  - MySQL - a relational database
  - PHP/JSP/ASP - the application layer



- IIS WebServer



# Web Server Defacement

# Web Server Defacement

- Web defacement occurs when an intruder maliciously alters visual appearance of a web page by inserting or substituting provocative and frequently offending data
- Defaced pages exposes visitors to some propaganda or misleading information until the unauthorized change is discovered and corrected



# Why kena deface

Cause	Consequences
Installing the server with default settings	Unnecessary default, backup, or sample files
Improper file and directory permissions	Security conflicts with business ease-of-use case
Default accounts with their default passwords	Misconfigurations in web server, operating systems and networks
Misconfigured SSL certificates and encryption settings	Bugs in server software, OS, and web applications
Use of self-signed certificates and default certificates	Improper authentication with external systems
Unnecessary services enabled, including content management and remote administration	Administrative or debugging functions that are enabled or accessible
Lack of proper security policy, procedures, and maintenance	Security flaws in the server software, OS and applications

# Impact of webserver attack

- Compromise of user accounts:
  - If the attacker is able to compromise a user account, then the attacker can gain a lot of useful information.
  - Attacker can use the compromised user account to launch further attacks on the web server.
- Data tampering :
  - Attacker can alter or delete the data.
- Secondary attacks from the website :
  - can use the server to launch further attacks on various websites or client systems
- Data theft :
  - Attackers can get access to sensitive data of the company like source code of a particular program.
- Root access to other applications or server
  - Attackers can perform any action once they get root access to the source.

# Web Server Attack

# Web Server Attack

- Directory Traversal Attacks
  - attackers are able to access restricted directories and execute commands outside of the web server root directory by manipulating a URL.
  - Attackers can use the trial-and-error method to navigate outside of the root directory and access sensitive information in the system.

```
?php $template = 'red.php'; if (  
($_COOKIE['TEMPLATE'])) $template =  
$_COOKIE['TEMPLATE'];  
include ("/home/users/phpguru/templates/" .  
$template); ?>
```

An attack against this system could be to send the following HTTP request:

```
GET  
/vulnerable.php HTTP 1.0 Cookie:  
TEMPLATE=../../../../../../../../../../../../etc/passwd
```

Generating a server response such as:


```
HTTP1.0 200 OK Content-Type: text/html Server:  
Apache root:fi3sED95ibqR6:0:1:System  
Operator:/:/bin/ksh daemon*:1:1::/tmp:  
phpguru:f8fk3j10If31.:182:100:Developer:/home/  
users/phpguru/./bin/csh
```

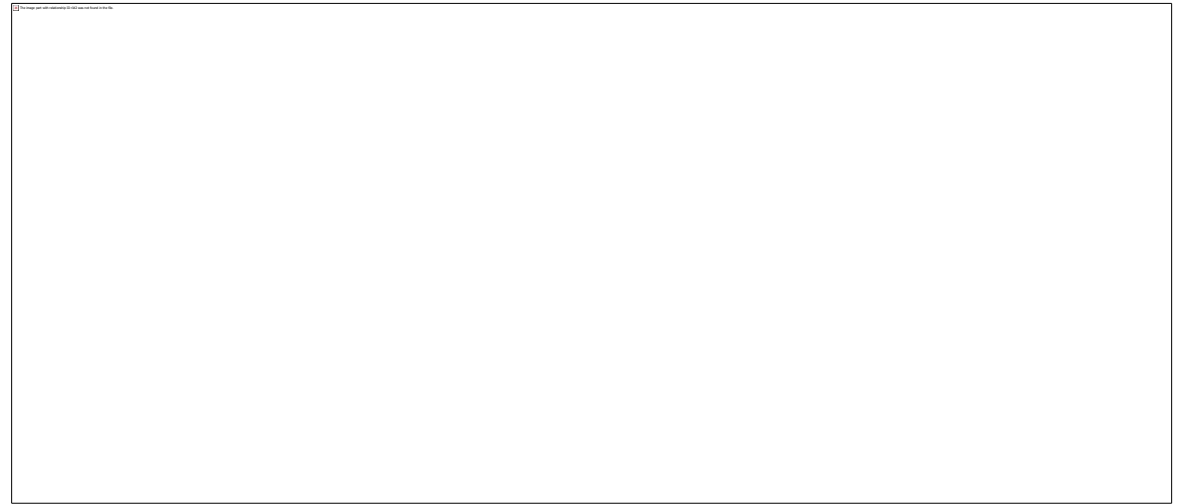
- HTTP Response Splitting Attack

- An HTTP response attack is a web-based attack where a server is tricked by injecting new lines in to response headers along with arbitrary code
- Cross-Site Scripting (XSS), Cross Site Request Forgery (CSRF), and SQL Injection are some of the examples for this type of attacks

- Web Cache Poisoning Attack

- Web cache poisoning is an attack that is carried out in contrast to the reliability of an intermediate web cache source, in which honest content cached for a random URL is swapped with infected content.
- Users of the web cache source can unknowingly use the poisoned content instead of true and secured content when demanding the required URL through the web cache.

- Man-in-the-Middle   
Attack(MITM)
  - where an intruder intercepts or modifies the message being exchanged between the user and web server through eavesdropping or intruding into a connection



# Webserver Password Cracking

- An attacker tries to exploit weaknesses to hack well chosen password
- Trying to get into the webserver admin account by cracking the admin page
- The techniques applied is similar to the system hacking password cracking
- Among the target are the applications related to maintaining remote admin access to the webserver such as:-
  - SSH connection
  - FTP server to upload web content
  - SMTP server
  - Web Sharing application

# Vulnerability Assessment Tool



# Webserver Vulnerability Scanning

- determining various vulnerabilities and misconfigurations of a target web server or network. Vulnerability scanning is done with the help of various automated tools known as vulnerable scanners
- Nessus
  - a security scanning tool that scans the system remotely and reports if it detects the vulnerabilities before the attacker actually attacks and compromises them
- BurpSuite
  - an integrated platform for performing security testing of web applications.
- OWASP ZAP

# Web Application Attack

# Web Application Attacks

- 10 Common web attacks

- Cross-site scripting (XSS) flaws
- Command injection flaws
- Malicious file execution
- Unsecured Direct Object Reference
- Cross-site Request Forgery (CSRF)
- Information Leakage and Incorrect Error Handling
- Broken Authentication and Session Management

- Unsecured cryptographic Storage
- Unsecured Communication
- Failure to Restrict URL Access

[Read OWASP Top 10](#)

# Top-10 Web application vulnerabilities

- **Cross-site scripting (XSS) flaws**
  - Attackers inject code into a web page, such as a forum or guestbook
  - When others user view the page, confidential information is stolen
- **Command injection flaws**
  - An attacker can embed malicious code and run a program on the database server
  - Example: SQL Injection

# Top-10 Web application vulnerabilities

- **Malicious file execution**
  - Users allowed to upload or run malicious files
- **Unsecured Direct Object Reference**
  - Information in the URL allows a user to reference files, directories, or records
- **Cross-site Request Forgery (CSRF)**
  - Stealing an authenticated session, by replaying a cookie or other token

# Top-10 Web application vulnerabilities

- **Information Leakage and Incorrect Error Handling**
  - **Error messages that give away too much information**
- **Broken Authentication and Session Management**
  - **Allow attackers to steal cookies or passwords**

# Top-10 Web application vulnerabilities

- **Unsecured cryptographic Storage**
  - Storing keys, certificates, and passwords on a Web server can be dangerous
- **Unsecured Communication**
  - Using HTTP instead of HTTPS
- **Failure to Restrict URL Access**
  - Security through obscurity
  - Hoping users don't find the "secret" URLs

Countermeasures and defence



- Countermeasures are the practice of using multiple security systems or technologies to prevent intrusions.
- These are the key components for protecting and safeguarding the web server against web server intrusions.
- The following components can be essential in providing the countermeasures against web server attack
  - Patches and Update
  - Protocols
  - Accounts
  - Files and directories

# Patches and update

- Scan for existing vulnerabilities and patch and update the server software regularly.
- Ensure that service packs, hotfixes, and security patch levels are consistent on all
- Have a back-out plan that allows the system and enterprise to return to their original state, prior to the failed implementation.
- Test the service packs and hotfixes on a representative non - production environment prior to being deployed to production .
- Ensure that server outages a rescheduled and a complete set of backup tapes and emergency repair disks a reavailable .
- Schedule periodic service pack upgrades as part of operations maintenance and never try to have more than two service packs behind.

# Protocols

- Block all unnecessary ports, Internet Control Message Protocol (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 the system software.
- If using insecure protocols such as Telnet, POP3, SMTP, or FTP, take appropriate measures to provide secure authentication and communication, for example, by using IPSec policies.
- If remote access is needed, make sure that the remote connection is secured properly, by using tunneling and encryption protocols.
- Disable WebDAV if not used by the application or keep secure if it is required.

# Accounts

- Remove all unused modules and application extensions.
- Disable unused default user accounts created during installation of an operating system.
- When creating a new webroot directory, grant the appropriate (least possible) NTFS permissions to the anonymous user being used from the IIS webserver to access the webcontent.
- 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.
- Slowdown brute force and dictionary attacks with strong password policies, and then audit and alert for logon failures.
- Run processes using least privileged accounts as well as least privileged service and user accounts.

# Files and Directories

- 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, Oracle), and OS logs frequently.
- Disable serving of directory listings.
- Eliminate the presence of non-web files such as archive files, backup files, text files, and header/include files.
- Disable serving certain file types by creating a resource mapping
- Ensure the presence of web application or website files and scripts on a separate partition or drive other than that of the operating system, logs, and any other system files

# Web Applications defence mechanism

- For defending the web applications vulnerabilities do read the OWASP Top 10 Web application vulnerabilities document.

# Summary

# Summary

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