ABSTRACT

Penetration Testing Report

KUSHAL K. RAHATKAR

2113202

CMM518 SECURITY TESTING

COURSEWORK PART 2

TABLE OF CONTENT

Executive Summary

The Penetration Testing on the box was conducted by Kushal Rahatkar in order to figure out vulnerabilities and damage a black-hat hacker can do. All activities were conducted to determine the following things :

1. What a remote penetration tester can bring out of this system
2. Precautions to take to mitigate the damage

Efforts were placed for identifying and exploiting security weaknesses which could lead a black-hat hacker a full remote access of the system which will be unauthorised and can be misused to get unauthorized data. The attacks were conducted from the level of access a normal to advance black-hat hacker or a internet user would have.

Summary of the Result

Initially, NMAP scan revealed all the systems running with versions of the applications running on the system. As most of the versions of the applications were outdated and not correctly patched it was very straight forward to exploit them. While searching through the different web applications, improper sanitization was observed on one of the systems which lead me to exploit Local File Inclusion. It allowed me to watch some critical files which should not be allowed to get exposed on a browser.

I found multiple ways to exploit. A stable reverse shell was created using Remote Command Execution, Command Line Injection and File Upload Execution.

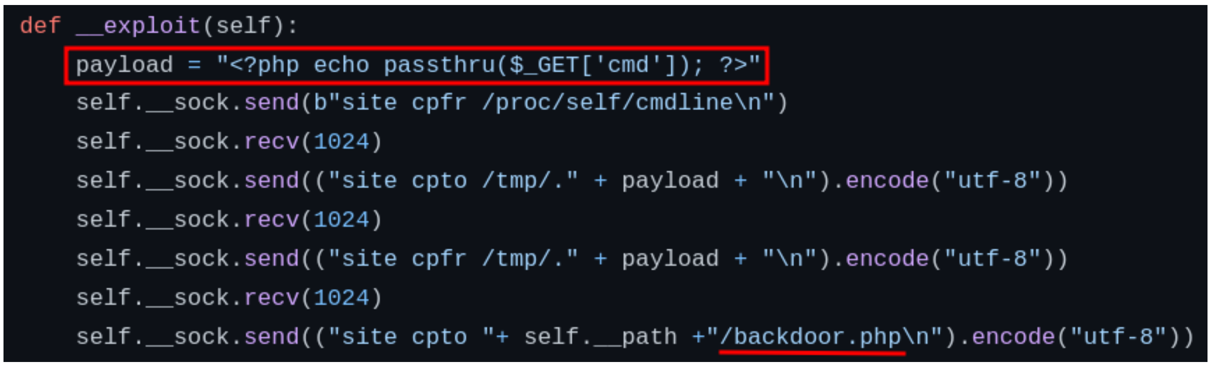
Exploitations of different applications landed me to the different users privilege. As the all users were with limited accessibilities, Root privilege was gained by exploiting the Kernal. The Kernal Version is not updated. Multiple compilers installed on the Victim Box made it really easy. Poor sanitization in modification of the system is observed as I was able to create a new user with root permissions. It was possible because, no password was asked while creating the new user having root permission.

Attack Narratives

COMMAND LINE INJECTION

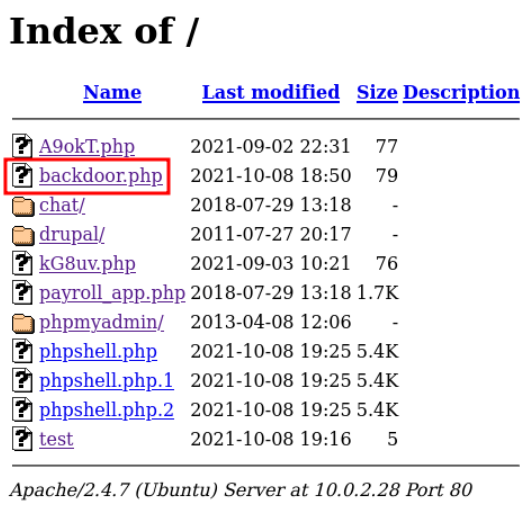
The service running on port 21 was captured by NMAP. The service was ProFTPD 1.3.5. This service was exploited using the combination of Command Line Execution and File Upload Vulnerability. A exploit was triggered which exploited a File Upload Vulnerability which added a Command Execution File. Using that file, I was able to do Command Line Injection to upload a reverse shell and execute it. This lead to the stable reverse shell.

After triggering an Exploit, a file with the name backdoor.php was uploaded in the directory which was linked to the port 80. The path to which backdoor.php should have been uploaded was not properly sanitized and it was captured from the directory called payroll_app.php. It is seen that this path has been leaked in other files as well. Which are A9okT.php, kG8uv.php.



```
def __exploit(self):
    payload = "<?php echo passthru($_GET['cmd']); ?>"
    self.__sock.send(b"site cpfr /proc/self/cmdline\n")
    self.__sock.recv(1024)
    self.__sock.send(("site cpto /tmp/." + payload + "\n").encode("utf-8"))
    self.__sock.recv(1024)
    self.__sock.send(("site cpfr /tmp/." + payload + "\n").encode("utf-8"))
    self.__sock.recv(1024)
    self.__sock.send(("site cpto "+ self.__path +"/backdoor.php\n").encode("utf-8"))
```

After successfully uploading a file, command line execution was possible which was totally done from the browser itself.



Index of /

```
   Name              Last modified     Size Description
------------------------------------------------------
   A9okT.php         2021-09-02 22:31   77
   backdoor.php      2021-10-08 18:50   79
   chat/             2018-07-29 13:18    -
   drupal/           2011-07-27 20:17    -
   kG8uv.php         2021-09-03 10:21   76
   payroll_app.php   2018-07-29 13:18 1.7K
   phpmyadmin/       2013-04-08 12:06    -
   phpshell.php      2021-10-08 19:25 5.4K
   phpshell.php.1    2021-10-08 19:25 5.4K
   phpshell.php.2    2021-10-08 19:25 5.4K
   test              2021-10-08 19:16    5
------------------------------------------------------
Apache/2.4.7 (Ubuntu) Server at 10.0.2.28 Port 80
```

As one file was uploaded, uploading reverse shell and running it with exploiting Command Line Execution was done. Which resulted in a stable Reverse Shell.

To know more about how the reverse shell was accessed please refer to the 'Course work1' writeup.

SHELLSHOCK

As NMAP declared that the system is running Apache web server, WebDAV is a process which is run by the Apache Web Server which allows user to upload unauthenticated data on the directory called /uploads. If this is not restricted, it can be exploited to upload malicious files and get a stable reverse shell. The reverse shell gives attacker User and in worst case Administrator privilege where commands can run.

The attack was conducted with the help of Metasploit, but there are exploits available on the internet which can be executed without the use of Metasploit.

Once the attacker gets fully functioning reverse shell, serious damage can be done. And if the attacker manages to get Administrator Privilege access, then he can modify the settings and access almost everything in the network.

The mitigation of this vulnerability can be achieved by updating the version of the Apache Web Server. Also, this can be avoided by applying sanitization to the user input data. Mostly, insertion of special characters must be provided. Monitoring network logs and network traffic are one of the most important ways.

To understand how the attack was conducted please refer to the course work 1 writeup.

XSS

XSS stands for Cross Site scripting. On port 80, in /chat directory, a chatting application was discovered vulnerable to XSS.

I identified this vulnerability using HTML tags which gave the output once submitted to the application. For example <b> test </b> is the input I gave and the output I received was **test.** Which clearly indicates the application is vulnerable to the Cross Site Scripting. (For proof, please refer to the coursework 1 writeup)

This can be exploited to get the stable reverse shell and can impact badly on the company.

To avoid this, a good sanitisation to the input data is required. If a developer avoids the use of special character parsing through the input, the XSS is not possible to be executed.

Due to time constraint, I was unable to dig in this attack and failed to provide the proof of reverse shell. But this is vulnerability must be avoided.

BACKDOOR PLANTATION

Backdoor plantation is an attack in which a file is uploaded on victim machine and executed to perform multiple attacks. It can include DDoS, it can used to get a stable reverse shell and many other.

Here I uploaded a reverse shell file with the name phpshell.php which was executed on the server side resulting in the stable reverse shell on attacker machine.

To avoid this, the directory of web pages should not have permission to edit or run a file. There should be antivirus which will recognise unauthenticated network traffic and immediately block the access. Another way is to keep up to date patches of the server.

In this case, the backdoor was able to be delivered on the victim machine by the exploitation of the service running on port 21 ProFTPD 1.3.5.

To understand how the attacks was conducted please refer to the coursework 1 writeup with the Port 21.

REMOTE COMMAND EXECUTION

Remote Command Execution (RCE) is a way of exploitation of systems in which an attacker is able to execute commands from his machine on the victim machine.

Having this type of access can result in stable reverse shell. And can be exploited further to get Administrator Privilege.

This can be avoided using antivirus, applying strong firewall and directory having the webpage with no rights to make modification.

In this case, the exploitation was possible using the service running on Port 21 ProFTPD.

To check how the attack was exploited, refer to the coursework 1 writeup.

REMOTE FILE INCLUSION

Remote File Inclusion is a vulnerability which allows attacker to run commands.

The vulnerability was found on the /readme directory. Here a URL /readme?os=../../../../../../etc/passwd which resulted me in the /etc/passwd file on the victim's system.

The vulnerability can be mitigated by applying good sanitization on the URL locator. This can also be achieved by disabling Remote Inclusion option from the configuration. As the service is running PHP, setting allow_url_include to '0' will avoid this vulnerability.

KERNAL EXPLOIT

Kernal is the bridge between Hardware and Software of the system. It is the most important part of the OS on which the Operating System is build

Exploitation of the Kernal gives an attacker Higher Level Privilege access which can be abused to change configuration or get the Administrator Authentication required data.

To avoid this vulnerability, there is only best solution which is to keep up to date patches of the system kernel. And side support can be taken by a strong antivirus.

To understand how I exploited the Kernal, please refer to the coursework 1 writeup.

WEAK SECURITY TO ADD NEW USER

This cannot be dictated as a vulnerability but I felt to mention it here. This can be abuse to create a new user having administrator rights and an attacker can keep accessing the system with the administrator right whenever he wants.

To avoid this, there should not be a user shell for a Root user. Generally Linux systems use Bash shell for the Command Line Interface. Here if the /bin/bash shell is restricted and only the /sbin/nologin shell is allowed to the root then there are no chances that the new root will be added and misuse. The changes should be done in /etc/passwd file. Another way can be implemented is disabling SSH access. SSH is called as Secure Shell. This is a service which is used to get the access of the system by providing correct username and password with the system IP address which results in the Command Line Interface access to the user. By disabling this, even if the new user is created, it becomes hard for an attacker to abuse the new user against the victim as the most of the times, connectivity is done via SSH.

Conclusion

Most of the part was able to exploit because of outdated versions of the applications which were running on the system. A stable reverse shell from Low Level Privilege to the Higher-Level Privilege was achieved only by exploiting the outdated versions of the systems. Apart from this, few vulnerabilities were exploited because of poor user input sanitization at the server end. At the end, only one configuration was misused which was the result of the poor configuration.

RECOMMENDATION

1. Applications should be up to date.
2. Good sanitization should be provided for the data when it is being accepted by the user or any third party user.
3. Operating System must be running the latest versions.
4. Unused ports must be blocked.
5. No port should have root access entry.
6. Root password must be asked when creating new root user.
7. /sbin/nologin should be applied in /etc/shadow to prevent bash shell access.

RISK RATING AS PER THE VERSION OF THE APPLICATION

Vulnerability / version	Risk Rating
ProFTPD 1.3.5	High
Apache httpd 2.4.7	High
UnrealIRCD	High
Webrick HTTPD 1.3.1	High
Kernal Exploit	Extremely High
XSS	Low
Local File Inclusion	Low
Creating user as root	Moderate

Appendix A: Vulnerability Details and Mitigation

Vulnerability / version	**Solution**
ProFTPD 1.3.5	1. Update to the latest version 1.3.8rc2, 1.3.7c
Apache httpd 2.4.7	1. Update to the latest version 2.4.51
UnrealIRCD	1. Update to 6.0.0-rc2
Webrick HTTPD 1.3.1	1. Update to 1.7.0/11 2. Run service on one port only
Kernal Exploit	1. Keep Kernal up to date. 2. Keep operating system with the late
XSS	1. Keep application up to date 2. Sanitize and validate Input 3. Use web application firewall
Local File Inclusion	1. Disable File Inclusion option 2. Sanitizing URL Locator 3. Set ‘allow_url_encode to ‘0’
Creating user as root	1. Change /bin/bash to /sbin/nologin in /etc/passwd 2. SSH root login shall not be allowed

Reference

1. NeuraLegion. (2021). *Local File Inclusion: Understanding and Preventing Attacks*. [online] Available at: https://www.neuralegion.com/blog/local-file-inclusion-lfi/.
2. eSecurityPlanet. (2021). *How to Prevent Cross-Site Scripting (XSS) Attacks*. [online] Available at: https://www.esecurityplanet.com/endpoint/prevent-xss-attacks/.
3. webdesign@politeia.in (2020). *UnrealIRCd - The most widely deployed IRC server - UnrealIRCd*. [online] Unrealircd.org. Available at: https://www.unrealircd.org/.
4. www.proftpd.org. (n.d.). *The ProFTPD Project: Home*. [online] Available at: http://www.proftpd.org/.
5. Kili, A. (n.d.). *4 Ways to Disable Root Account in Linux*. [online] www.tecmint.com. Available at: https://www.tecmint.com/disable-root-login-in-linux/.
6. Team, N.S. (2019). *What is the Remote File Inclusion vulnerability?* [online] www.netsparker.com. Available at: https://www.netsparker.com/blog/web-security/remote-file-inclusion-vulnerability/.
7. SafetyDetectives. (2018). *What is a Backdoor and How to Protect Against it*. [online] Available at: https://www.safetydetectives.com/blog/what-is-a-backdoor-and-how-to-protect-against-it/.