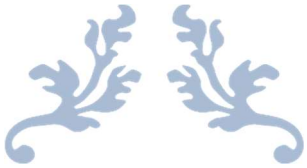


RAVEN SECURITY

Vulnerability Assessment

AUGUST 8, 2019

BEN HALPERN

Executive Summary

The Raven security server proved vulnerable in many accounts. By stringing together the different vulnerabilities an attacker is able to achieve a root shell. A basic enumeration of the box shows the server has 3 ports open, port 80,22, 111. Port 22 running SSH login, showing an exposed login server, Port 80 running an outdated apache server, and 111 running RPCbind. The server has 2 high risk, 1 low, and one medium risk vulnerabilities. By chaining the exposed login server and the outdated apache server with a WordPress vulnerability, a shell is possible to be achieved, and from that shell able to escalate to a root shell due to poor permission.

With a root shell an attacker could set up many persistent threats Denial Of Service (DOS), Supply-Chain Attacks (SCA) , as well open up more back doors to potentially more attacks. These attacks can affect the reputability, integrity, as well as the confidentiality of the business, costing them upwards to millions of dollars depending on their daily revenue.

In the following pages, it lists in more detail the actions taken to discover the vulnerabilities as well as infiltrate the system through escalation from “michael” to “steven” to root. However it is possible to escalate from “michael” through MySQL since it is running as root. That isn’t included in this writeup however it is another vulnerability that is possible to be exploited. Mitigating this vulnerability is included in this write up, since the method of obtaining a root shell wasn’t done so through exploiting MySQL however stealing credentials from MySQL. This report includes a list of the proper mitigation to the found vulnerabilities.

Attack Narrative

This assessment involved the attempted compromise of Web Server, and privilege escalation.

Reconnaissance

General Reconnaissance

After running an Aggressive and intensive Nmap scan we discover that there are three TCP ports open. The Raven server is running a Debian based Linux 3.2/4.9, with ports 22,80,111 open unfiltered. Port 22 hosts SSH, a secure login shell using SSL to host a secure connection over potential non secure networks. Port 80 hosts http, the server hosts a website called Raven security hosted with Apache/2.4.10 (Debian), an outdated webserver service. Port 111 hosts RPCbind. RPCbind is a service that communicates with programs to identify and communicate what services are running on which ports.

Enumeration and Vulnerability Analysis

This section summarizes the most critical vulnerabilities affecting the target network.

IP Address	Operating System	Vulnerabilities	Risk (Low/Med/High)
192.168.234.129	Debian Linux 3.2-4.9	Exposed login server	Medium
192.168.234.129	Debian Linux 3.2-4.9 Apache 2.4.10	Exposed Web Server	Low
192.168.234.129	Debian Linux 3.2-4.9	Weak Credentials	High
192.168.234.129	Debian Linux 3.2-4.9	File System Weak	High

		permissions	

Web Server Analysis

Port 80 is hosting a web server running apache 2.4.10

Once discovering that port 80 was open and hosting a webserver, it is possible to enumerate directories by brute forcing them to discover what pages we are able to access. A tool such as Dirb, Dirbuster, and GoBuster can be used to brute force the http server directories. It uses wordlists to try different directories and see if they are front facing, by analyzing the error response codes. By doing so, WordPress is discovered, http:///{Raven'sIP}/wordpress . WordPress is a commonly vulnerable blog, using a WordPress enumeration software to enumerate and fuzz credentials from the WordPress Directories and posts, Wpscan. Using Wpscan and a common wordlist the usernames Michael and Steven are discovered.

Network Analysis

Port 22 is open on the Raven system is running an SSH login.

Putting the username credentials discovered in Wpscan in a wordlist, and using a common password wordlist, a login bruteforcer called hydra can be used to bruteforce the login. After about 30 seconds the password to Michael is discovered, "michael". It continues to run in the background to attempt to discover Steven's password.

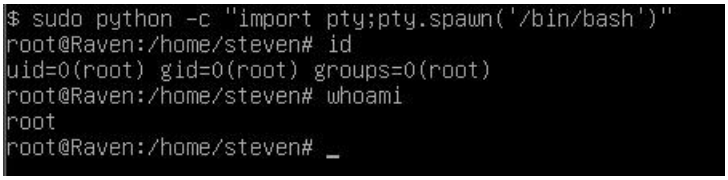
Post-Exploitation Exploration and Privilege Escalation

Post-exploitation involves enumerating and gathering as much information as one can from the session. This can be done through various methods, and various enumeration scripts. In this case no enumeration scripts were used, simply traversing through the directories.

Once on the SSH connection, enumerate the processes running and scan through the directories. Look at the sudoers file or use the command "sudo -l" to see what privileges Michael has. MySQL is discovered to be running, a easy opensource database. Looking over the wp-config.php configuration file, the username and password to MySQL can be found, "R@v3nSecurity. Logging into to MySQL we scan through the databases and tables, dumping all the information from the database. In the database WordPress, in the tables wp_users, the hashes for Steven and Michael can be found. Running through a hash cracking software, such

as “John the Ripper” which brute forces hashes against a wordlist, until there is a match, the hash to Steven is cracked, “Pink84”.

Assuming that the username and password to the WordPress login are the same for the SSH connection, the username and password are attempted on the SSH login. Steven is logged in and running “/bin/sh“. Running “sudo -l” to see what privileges Steven has as root, /bin/python is listed with “nologin” meaning that steven can run commands as root through python without any credentials. Using “python -c” it is possible to spawn a shell from a terminal using python, because Steven has sudo permissions to use python, prefixing the command with sudo allows for a root shell to be spawned: sudo python -c “import pty; pty.spawn(‘/bin/bash’)”



The flags in the raven server can be found in the following locations:
Flag 1: flag1{b9bbcb33e11b80be759c4e844862482d}
Location: http://192.168.234.129/service in the source code
Flag 2: flag2{fc3fd58dcdad9ab23faca6e9a36e581c}
Location: /var/www/html/flag2.txt
Flag 3: flag3{afc01ab56b50591e7dccf93122770cd2}
Location: in the MySQL database wordpress in table wp_posts
Flag 4: flag4{715dea6c055b9fe3337544932f2941ce}
Location: in the MySQL database wordpress in table wp_posts

Persistence

Once inside a system, an attacker wants to set up persistence. As a root user there are many ways to set up persistence. One way to set up persistence is by grabbing the shadow and passwd file and cracking the hashes within. By analyzing the root hash in /etc/shadow, the hash signature identifies it as sha512-crypto, a very difficult hash to crack. In the root folder an attacker could place their ssh keys to be able to connect through ssh to root. The root directory currently doesn’t have a .ssh and no ssh keys stored within it. Alternatively a bind shell could be set up opening up a back door into the system allowing an attacker to connect to it whenever they choose, or it could be tied to the boot sequence, allowing the system to run the bind shell on boot.

There are many options for persistence. An attacker chooses the right persistence depending on the system, for example, if they need to hide the persistent threat they may set a bootkit or a rootkit hidden to start on the system start up. To defend against this is to prevent the attacker from getting a root shell, as is mentioned in mitigation.

Conclusion and Recommendations

Based on the results documented above, we recommend the client take the following steps to remediate the vulnerabilities identified on the target machine.

Web Server

- *Update their apache web server*
- *Change their WordPress credentials to more secure credentials*
- *Change the permissions on directories that the public doesn't need to be able to access*

Network Services

- *Change the passwords to more secure credentials.*
- *Possibly require ssh key to gain access to the server / dual factor authentication*

- *Close ports/ filter ports with a firewall if they don't need to be accessed by the outside world.*

Hardening the Server

- *Strictly abide by Rule of Least Permissions*
    - *Giving users the least amount of permissions so that they can do their job in their scope.*
    - *Don't let basic users have access to configuration files unless they are working on them.*
- *Only allow users to use sudo/root permissions only when they require them, then disable those permissions after they no longer require them.*
- *Don't allow MySQL to be run as root in non-root users, and don't leave root credentials in configuration files in plain text.*
- *Set up IPChain/ firewall rules only allowing the necessary processes in and out from the allowed and to the proper IP Blocks.*