Gaara Proving Grounds Penetration Test Report

Table of Contents

1.0– High-Level Summary

An internal penetration test was performed on the gaara network in the Offensive Security Proving Ground Labs. An internal test simulates an attacker that is directly connected into the network, in this case through a VPN tunnel.

The purpose of this test was to simulate an attack where the attacker had access to the network, with attempts made to break into a system and then elevate privileges on the machine.

Over-all, the intent was to enumerate the services on the exposed network, determine an attack vector to get access, and then exploit any flaw found within the system.

During the test it was found that a weak password for the gaara account allowed access into the device remotely through ssh, and then privilege escalation was performed through a command given SUID privilege. This allowed for root access to the device.

1.1- Recommendations

It is recommended that a password policy be instituted that disallows passwords that are on the rockyou.txt password list. Any account with a password from this list, can easily be brute forced. It is also recommended to apply SUID values to files only for those that absolutely need it, and that do not have an entry under https://gtfobins.github.io/gtfobins that shows a SUID exploit.

2.0-Methodologies

Below is the methods that were undertaken to break into the device, and ultimately achieve root access on the device.

2.1-Information Gathering

The information gathering portion was mostly null, as the network address of 192.168.191.142 was provided ahead of the pentest commencing.

2.2-Service Enumeration

This was mainly accomplished with nmap scan of the base 1000, followed by a scan of all tcp ports on the device. The UDP top port only returned 3 ports open, none looked available for an exploit. This left it with the following ports as possible exploit vectors:

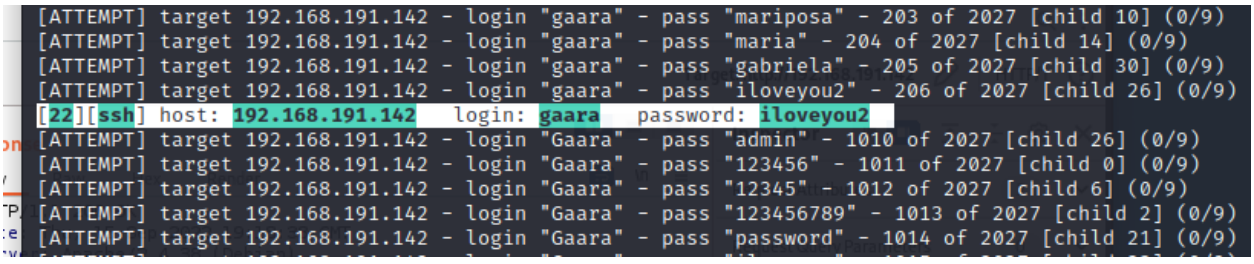
22 secure shell remote access

80 http web port

2.3-Penetration Testing

The methodology for breaking into the network began with determining ports open through enumeration, then testing for known exploits on the web portal. Nikto, ZAP, Dirbuster, and nmap .nse scripts were run against the portal; no valid exploit showed that would allow for access or remote code injection. This appears due to the barebones nature of the site.

With no luck on the port 80 webpage, ssh on port 22 was the next target for the attack. Utilizing a 1K wordlist showed no results for users root or admin. On testing against gaara, the password for the user was found through brute forcing with hydra.



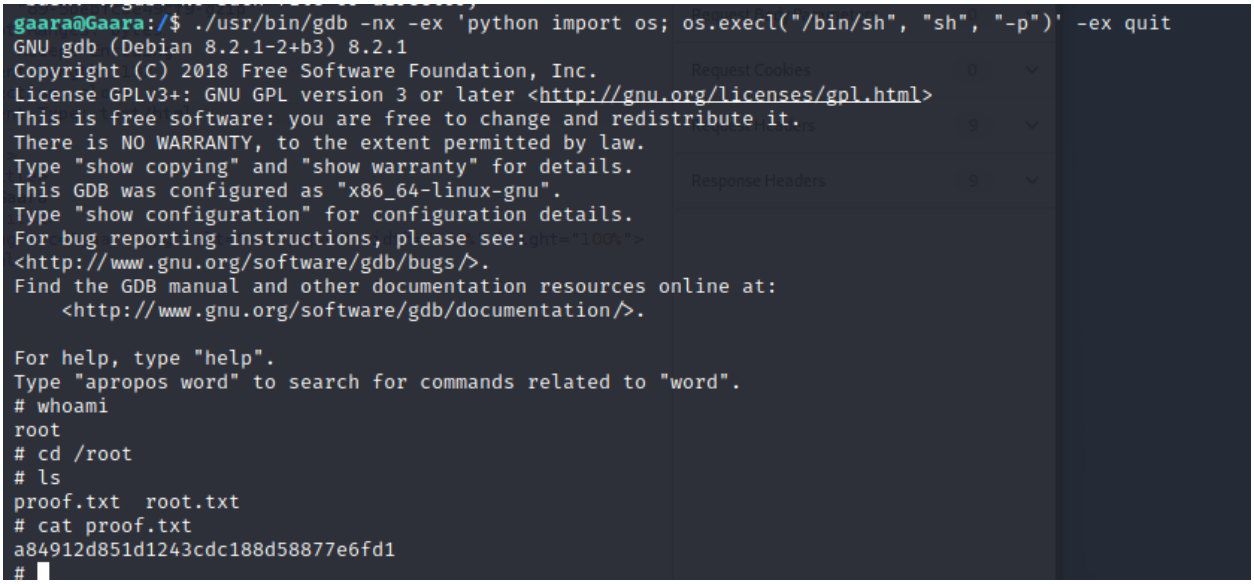
With this information, it was possible to log into the device, and collect the information for the flag stored locally on it.

```
└─$ ssh gaara@192.168.191.142
The authenticity of host '192.168.191.142 (192.168.191.142)' can't be established.
ED25519 key fingerprint is SHA256:XpX1VX2RtX8OaktJHdq89ZkpLlYvr88cebZ0tPZMI0I.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.191.142' (ED25519) to the list of known hosts.
gaara@192.168.191.142's password:
Linux Gaara 4.19.0-13-amd64 #1 SMP Debian 4.19.160-2 (2020-11-28) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
gaara@Gaara:~$ ls
flag.txt  local.txt
gaara@Gaara:~$ cat local.txt
edc388d31541f09ccb04bb0d9cad9d66
```

After this, it was checking for a method to get root access to have complete control of the box. This was accomplished by checking for SUID enabled files that were exploitable.

```
gaara@Gaara:/$ find / -perm -u+s -type f 2>/dev/null

/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/eject/dmcrypt-get-device
/usr/lib/openssh/ssh-keysign
/usr/bin/gdb
/usr/bin/sudo
/usr/bin/gimp-2.10
/usr/bin/fusermount
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/gpasswd
/usr/bin/newgrp
/usr/bin/su
/usr/bin/passwd
/usr/bin/mount
/usr/bin/umount
gaara@Gaara:/$
```

Upon checking for any that were a possible vector, the gdb file stuck out as a possibility. Utilizing the python that was on the box, a root shell was spawned utilizing the gdb SUID permissions for root access.



System Vulnerable 192.168.191.142:

Vulnerabilities Exploited:

Weak password

SUID enabled file privilege escalation

Vulnerability Fixes:

Change password policy to disallow common passwords found on rockyou.txt

Disallow SUID enabled files, if they are on the GTFOBins site

Severity: Critical

Proof of Privilege Escalation:

Local.txt: edc388d31541f09ccb04bb0d9cad9d66

Proof.txt: a84912d851d1243cdc188d58877e6fd1

2.4-Report: Clean-up

Clean up was not required during this exploit, as it was using normal credentials to log in, and a temporary spawned root shell to accomplish all these actions. No files were modified or added during this process