Oracle Linux 7 – How to audit changes to a trusted file such as /etc/passwd or /etc/shadow

Article Number: 325 | Rating: Unrated | Last Updated: Wed, Jul 25, 2018 1:42 PM

Linux auditing is quite powerful and a lot of different use cases might be handled via the auditing framework. However, in this blog I would like to show you, how to audit changes on trusted files, like /etc/passwd or /etc/shadow. Of course, you are not limited to these files. You can audit whatever you want. Maybe the sqlnet.ora, the /etc/oratab or Oracle wallets are of more interest in your environment.

Before we start, we got to make sure that the the auditd deamon is enabled and running.

```
1
                                                [root@dbidg03 ~]# systemctl list-unit-files | grep
2                                               audit
3                                               auditd.service                disabled
4
5                                               [root@dbidg03 ~]# systemctl enable auditd
6                                               Created symlink from /etc/systemd/system/multi-
                                                user.target.wants/auditd.service to
7                                               /usr/lib/systemd/system/auditd.service.
8
9                                               [root@dbidg03 ~]# service auditd start
10                                              Redirecting to /bin/systemctl start  auditd.service
11
```

```
[root@dbidg03 ~]# ps -ef | grep auditd | grep -v grep
root       107     2  0 07:50 ?        00:00:00 [kauditd]
root      6772     1  0 13:03 ?        00:00:00 /sbin/auditd -n


[root@dbidg03 ~]# auditctl -s
enabled 1
failure 1
pid 6772
rate_limit 0
backlog_limit 64
lost 0
backlog 0
loginuid_immutable 0 unlocked
```

Now we can start implementing our first rules. One for auditing the passwd and on for shadow.

```
[root@dbidg03 ~]# auditctl -w /etc/passwd -p rwa -k audit_passwd

[root@dbidg03 ~]# auditctl -w /etc/shadow -p rwa
```

```
4                                       -k audit_shadow
5                                       [root@dbidg03 ~]# auditctl -l
                                        -w /etc/passwd -p rwa -k audit_passwd
                                        -w /etc/shadow -p rwa -k audit_shadow
```

The options that I used are, -w, which is the path_to_file. In other words, the file or directory that is audited. The next one is -p. These are the permissions that are logged, which can be:

- r — read access to a file or a directory
- w — write access to a file or a directory
- x — execute access to a file or a directory
- a — change in the file's or directory's attribute

Last but not least, -k. This is the key_name which is an optional string. That one is a quite important one. The key_name is a tag that you can assign to your audit rule. Especially when your audit logs are huge, it can help you enormously to identify which rule or set of rules generated a particular log entry. We will see it later, when it comes to audit search, how beneficial the tagging is.

Be aware that audit rules defined by auditctl are not persistent across reboots. You have to include them in the /etc/audit/audit.rules file, in case you want to make them persistent. The beauty of the audit.rule file is, that is uses the same auditctl command line syntax to specify the rules. E.g. you could simply pipe the auditctl output into your audit.rules file, and it works. A good practice, is of course to backup your current audit.rules file.

```
1
                                        [root@dbidg03 rules.d]# pwd
2
                                        /etc/audit/rules.d
3
                                        [root@dbidg03 rules.d]# auditctl -l > audit.rules
```

Now we start doing our first test, by simulating a read on the /etc/passwd file by the oracle user.

```
1
                                                oracle@dbidg03:/home/oracle/ [rdbms112] cat
                                                /etc/passwd
```

The read created immediately a log entry in the audit.log file.

```
1
                                                [root@dbidg03 audit]# tail -50f
2                                               /var/log/audit/audit.log

3                                               ...

4                                               type=SYSCALL msg=audit(1495451317.823:32):
                                                arch=c000003e syscall=2 success=yes exit=3
5                                               a0=7ffc0e042dad

6                                               a1=0 a2=1fffffffffff0000 a3=7ffc0e041e30
                                                items=1 ppid=6863 pid=7066
7
                                                auid=54321 uid=54321 gid=54321 euid=54321
8                                               suid=54321 fsuid=54321 egid=54321 sgid=54321

9                                               fsgid=54321 tty=pts2 ses=61 comm="cat"
                                                exe="/usr/bin/cat" key="audit_passwd"
10
                                                type=CWD msg=audit(1495451317.823:32):
                                                cwd="/home/oracle"
```

```
type=PATH msg=audit(1495451317.823:32): item=0 name="/etc/passwd" inode=39423106 dev=fb:00

mode=0100644 ouid=0 ogid=0 rdev=00:00 nametype=NORMAL

type=PROCTITLE msg=audit(1495451317.823:32): proctitle=636174002F6574632F706173737764
```

The audit.log is kinda cryptic to read. This is where the ausearch and aureport come into play. E.g. we can combine the tag, which we have created beforehand with a start time.

```
1
                                                    [root@dbidg03 rules.d]# ausearch -k audit_passwd
2                                                   --start today | aureport -f
3
4                                                   File Report
5                                                   ===============================================
6                                                   # date time file syscall success exe auid event
7                                                   ===============================================
8                                                   
9                                                   1. 05/22/2017 13:07:53 /etc/passwd 2 yes /usr/sbin/sshd -1 15
10                                                  
                                                    2. 05/22/2017 13:08:37 /etc/passwd 2 yes
```

```
/usr/bin/cat 54321 32

3. 05/22/2017 13:10:01 /etc/passwd 2 yes
/usr/sbin/crond -1 34

4. 05/22/2017 13:20:01 /etc/passwd 2 yes
/usr/sbin/crond -1 43
```

Maybe, you want to limit the audit search, to display only data about user oracle. To do so, use the –uid switch.

```
1
                                    [root@dbidg03 ~]# getent passwd | grep 54321
2
                                    oracle:x:54321:54321::/home/oracle:/bin/bash
3
4
                                    [root@dbidg03 rules.d]# ausearch -k audit_passwd
5                                   --start today --uid 54321 | aureport -f
6
7                                   File Report
8                                   ===========================================
                                    ==========
9
                                    # date time file syscall success exe auid event
10
                                    ===========================================
                                    ==========

                                    1. 05/22/2017 13:08:37 /etc/passwd 2 yes
```

```
/usr/bin/cat 54321 32
```

Now we can see clearly, that user oracle run the cat command on the /etc/passwd this midday.