# Lab 03 – File Integrity

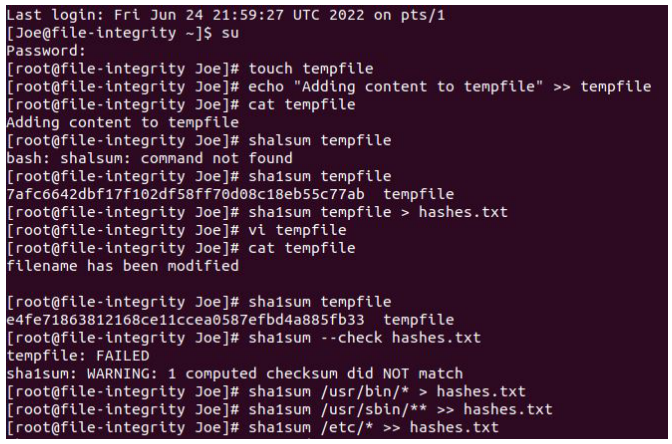
Raymond Ng: JQG999
IS 3033-CY1 – Summer 2019
June 24, 2022

## INTRODUCTION

The purpose of this lab is to provide students an opportunity and a safe, controlled environment to experiment with tools that may help with exploitation of compromised systems. Moreover, it will help students realize and appreciate the concept of system integrity.

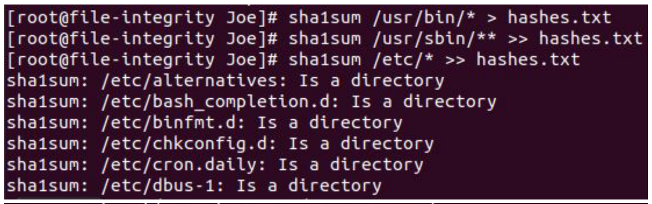
## PROCEDURE

### Poor Mans' Integrity

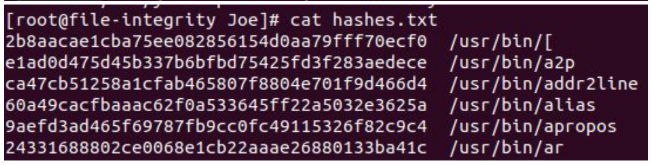


```
Last login: Fri Jun 24 21:59:27 UTC 2022 on pts/1
[Joe@file-integrity ~]$ su
Password:
[root@file-integrity Joe]# touch tempfile
[root@file-integrity Joe]# echo "Adding content to tempfile" >> tempfile
[root@file-integrity Joe]# cat tempfile
Adding content to tempfile
[root@file-integrity Joe]# shalsum tempfile
bash: shalsum: command not found
[root@file-integrity Joe]# sha1sum tempfile
7afc6642dbf17f102df58ff70d08c18eb55c77ab  tempfile
[root@file-integrity Joe]# sha1sum tempfile > hashes.txt
[root@file-integrity Joe]# vi tempfile
[root@file-integrity Joe]# cat tempfile
filename has been modified

[root@file-integrity Joe]# sha1sum tempfile
e4fe71863812168ce11ccea0587efbd4a885fb33  tempfile
[root@file-integrity Joe]# sha1sum --check hashes.txt
tempfile: FAILED
sha1sum: WARNING: 1 computed checksum did NOT match
[root@file-integrity Joe]# sha1sum /usr/bin/* > hashes.txt
[root@file-integrity Joe]# sha1sum /usr/sbin/** >> hashes.txt
[root@file-integrity Joe]# sha1sum /etc/* >> hashes.txt
```

Using `touch` command to create `tempfile`. Using `echo`, I added ***Adding content to tempfile*** to include some content to the new file. Per the instructions, learned about calculating digests for a file using `sha1sum`. It produced an error when I first used the command, but then I realized I had mistake the `1` for the letter `l`. I was a little confused on the step instructing me to modify the content in `tempfile` so I used Vi editor—which I learned about from *javapoint.com*—to edit the existing content *Adding content to tempfile* to *filename has been modified*.

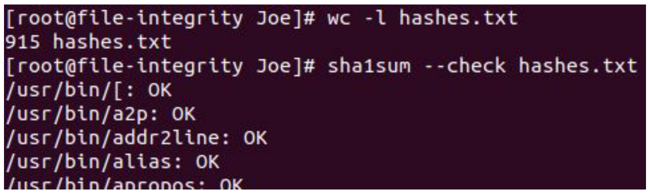


```
[root@file-integrity Joe]# sha1sum /usr/bin/* > hashes.txt
[root@file-integrity Joe]# sha1sum /usr/sbin/** >> hashes.txt
[root@file-integrity Joe]# sha1sum /etc/* >> hashes.txt
sha1sum: /etc/alternatives: Is a directory
sha1sum: /etc/bash_completion.d: Is a directory
sha1sum: /etc/binfmt.d: Is a directory
sha1sum: /etc/chkconfig.d: Is a directory
sha1sum: /etc/cron.daily: Is a directory
sha1sum: /etc/dbus-1: Is a directory
```



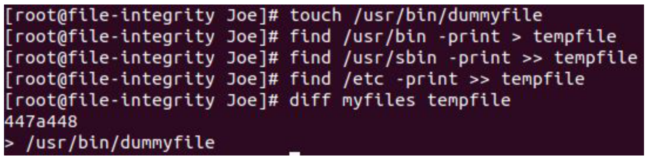
```
[root@file-integrity Joe]# cat hashes.txt
2b8aacae1cba75ee082856154d0aa79fff70ecf0  /usr/bin/[
e1ad0d475d45b337b6bfbd75425fd3f283aedece  /usr/bin/a2p
ca47cb51258a1cfab465807f8804e701f9d466d4  /usr/bin/addr2line
60a49cacfbaaac62f0a533645ff22a5032e3625a  /usr/bin/alias
9aefd3ad465f69787fb9cc0fc49115326f82c9c4  /usr/bin/apropos
24331688802ce0068e1cb22aaae26880133ba41c  /usr/bin/ar
```

After redirecting the output of sha1sum to a filed called hashes.txt, a recalculation of tempfile was executed. Further, I learned how to get a bigger view of the system by calculating digests for critical files by using the commands seen in the screenshots to the left.



```
[root@file-integrity Joe]# wc -l hashes.txt
915 hashes.txt
[root@file-integrity Joe]# sha1sum --check hashes.txt
/usr/bin/[: OK
/usr/bin/a2p: OK
/usr/bin/addr2line: OK
/usr/bin/alias: OK
/usr/bin/apropos: OK
```

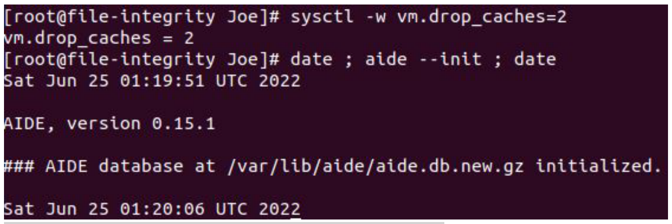
To determine how may files were hashed, used `wc -l hashes.txt`, output indicated 915. Executed `sha1sum --check hashes.txt` to verify that the digests calculated properly.



```
[root@file-integrity Joe]# touch /usr/bin/dummyfile
[root@file-integrity Joe]# find /usr/bin -print > tempfile
[root@file-integrity Joe]# find /usr/sbin -print >> tempfile
[root@file-integrity Joe]# find /etc -print >> tempfile
[root@file-integrity Joe]# diff myfiles tempfile
447a448
> /usr/bin/dummyfile
```

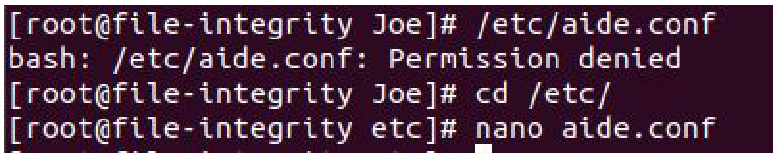
Simulated a simple hacker scenario executed `touch /usr/bin/dummyfile` to add an empty file into the `/bin` directory using the `touch` command. Recreated the hierarchy of files using the commands seen in the screenshot to the left. Then, executed `diff myfiles tempfile` to compare old list with the new list and identified the newly added empty file.

**AIDE**



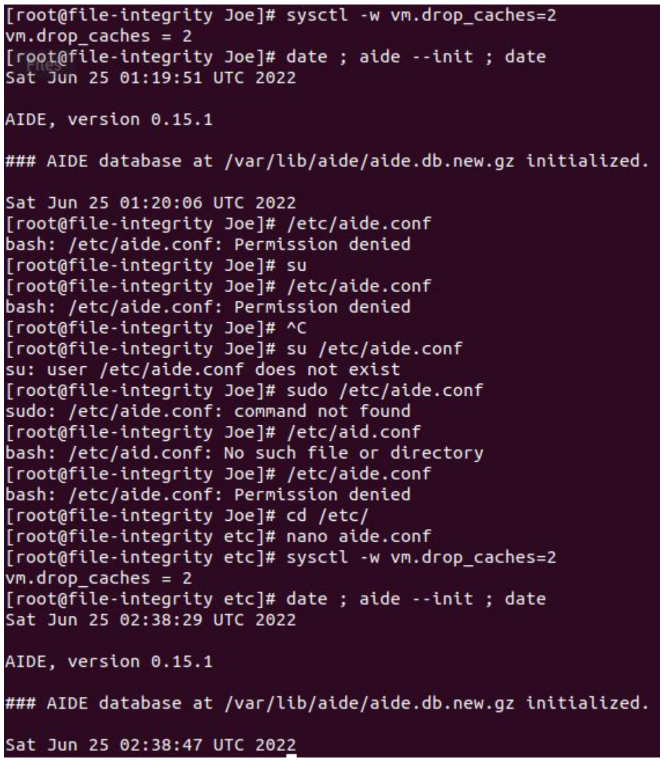
```
[root@file-integrity Joe]# sysctl -w vm.drop_caches=2
vm.drop_caches = 2
[root@file-integrity Joe]# date ; aide --init ; date
Sat Jun 25 01:19:51 UTC 2022

AIDE, version 0.15.1

### AIDE database at /var/lib/aide/aide.db.new.gz initialized.

Sat Jun 25 01:20:06 UTC 2022
```

In this section I experimented with Advanced Intrusion Detection Environment (AIDE). Executed `sysctl -w vm.drop_caches=2` to drop the caches in the OS. Then used default AIDE configuration to build an integrity database by executing `date ; aide --init ; date`. There was an approximate 5-7 second wait time between the initial AIDE database with the default configuration. You can see in the screenshot the difference in the date/time group.



```
[root@file-integrity Joe]# /etc/aide.conf
bash: /etc/aide.conf: Permission denied
[root@file-integrity Joe]# cd /etc/
[root@file-integrity etc]# nano aide.conf
```

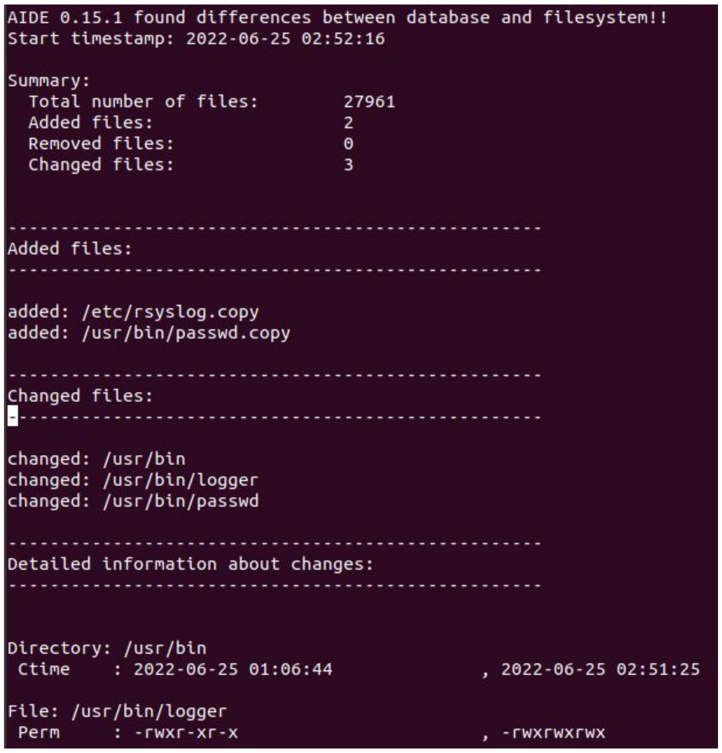
I did run into an issue with `/etc/aide.conf`. Instead I had to execute change directory `cd /etc/` then use `nano aide.conf` to access the AIDE configuration file



```
[root@file-integrity Joe]# sysctl -w vm.drop_caches=2
vm.drop_caches = 2
[root@file-integrity Joe]# date ; aide --init ; date
Sat Jun 25 01:19:51 UTC 2022

AIDE, version 0.15.1

### AIDE database at /var/lib/aide/aide.db.new.gz initialized.

Sat Jun 25 01:20:06 UTC 2022
[root@file-integrity Joe]# /etc/aide.conf
bash: /etc/aide.conf: Permission denied
[root@file-integrity Joe]# su
[root@file-integrity Joe]# /etc/aide.conf
bash: /etc/aide.conf: Permission denied
[root@file-integrity Joe]# ^C
[root@file-integrity Joe]# su /etc/aide.conf
su: user /etc/aide.conf does not exist
[root@file-integrity Joe]# sudo /etc/aide.conf
sudo: /etc/aide.conf: command not found
[root@file-integrity Joe]# /etc/aid.conf
bash: /etc/aid.conf: No such file or directory
[root@file-integrity Joe]# /etc/aide.conf
bash: /etc/aide.conf: Permission denied
[root@file-integrity Joe]# cd /etc/
[root@file-integrity etc]# nano aide.conf
[root@file-integrity etc]# sysctl -w vm.drop_caches=2
vm.drop_caches = 2
[root@file-integrity etc]# date ; aide --init ; date
Sat Jun 25 02:38:29 UTC 2022

AIDE, version 0.15.1

### AIDE database at /var/lib/aide/aide.db.new.gz initialized.

Sat Jun 25 02:38:47 UTC 2022
```

It took approximately 2 minutes to build the AIDE database when the two hash functions were included.

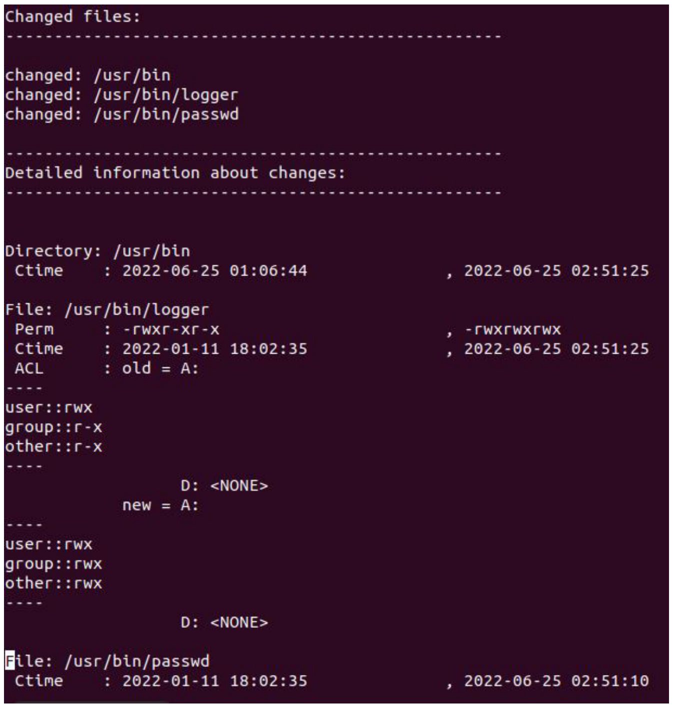
I noticed a significant time difference, from 01:19:51 to 02:38:29, over an hour difference. This likely resulted from adding the "sha512+" hash to the existing "sha256" hash. From what I can glean via *security.stackexchange.com*, sha512 is not very practical because sha256 by itself is smaller, requiring less bandwidth to store and transit, less memory, and in some cases less processing power. Further, sha256 seems to be the universal choice.



```
AIDE 0.15.1 found differences between database and filesystem!!
Start timestamp: 2022-06-25 02:52:16

Summary:
  Total number of files:        27961
  Added files:                  2
  Removed files:                0
  Changed files:                3


---------------------------------------------------
Added files:
---------------------------------------------------

added: /etc/rsyslog.copy
added: /usr/bin/passwd.copy

---------------------------------------------------
Changed files:
---------------------------------------------------

changed: /usr/bin
changed: /usr/bin/logger
changed: /usr/bin/passwd

---------------------------------------------------
Detailed information about changes:
---------------------------------------------------


Directory: /usr/bin
 Ctime    : 2022-06-25 01:06:44              , 2022-06-25 02:51:25

File: /usr/bin/logger
 Perm     : -rwxr-xr-x                       , -rwxrwxrwx
```

I ran into a permissions issue again attempting to open `/var/log/aide/aide.log`. So I executed the same commands that I used back in III.AIDE, Step 3 to open `/etc/aide.conf,` executed `cd /var/log/aide/` and `nano aide.log` to open the AIDE report.

The screenshot to the left depicts the changes made in III.AIDE, step 6. `/etc/ryslog.conf` was copied into `/etc/ryslog.copy` and `/usr/bin/passwd` was copied into `/usr/bin/passwd.copy`; adding two files. Lastly, `echo " "` was appended to /`usr/bin/passwd`, changing 3 files. `echo "# another comment" >> /etc/rsyslog.conf` was not detected. I'm assessing it wasn't detected because it was copied into another file which might have hid it.



```
Changed files:
---------------------------------------------------

changed: /usr/bin
changed: /usr/bin/logger
changed: /usr/bin/passwd

---------------------------------------------------
Detailed information about changes:
---------------------------------------------------


Directory: /usr/bin
 Ctime    : 2022-06-25 01:06:44             , 2022-06-25 02:51:25

File: /usr/bin/logger
 Perm     : -rwxr-xr-x                      , -rwxrwxrwx
 Ctime    : 2022-01-11 18:02:35             , 2022-06-25 02:51:25
 ACL      : old = A:
----
user::rwx
group::r-x
other::r-x
----
                  D: <NONE>
            new = A:
----
user::rwx
group::rwx
other::rwx
----
                  D: <NONE>

File: /usr/bin/passwd
 Ctime    : 2022-01-11 18:02:35             , 2022-06-25 02:51:10
```

Screenshot to the left depicts detailed information regarding the changes made to the logger command `/usr/bin/logger`. Looking at `Perm:` line, an administrator will likely see that a permissions change occurred to user, group and other, allowing all groups to read (`r`), write (`w`) and execute (`x`) in the file. This would make sense because in III.AIDE, step 6 I executed a change mode (`chmod`) command, `chmod ugo=rwx /bin/logger`, to change the permissions of `/logger`.

From reading about `/etc/shadow` file from linuxize.com, I understand it is a text file that contains information about the system's user's passwords. I needed to gain a better understanding regarding digests so I visited *datacamadia.com* and interpreted digests simply as the output of hash function (or known as checksum. From reading more about AIDE via *aid.github.io*, I learned that AIDE creates checksum or has using one or more combinations of message digest algorithms, such `sha256` and `sha512`, the ones used in this lab. So simply I would open the `/etc/shadow` file –assuming root privilege is gained—and incorporate the appropriate hashes to generate several digests for the file.

## Conclusion

The goals of the techniques were to allow the student to experiment with tools to detect irregularities that may indicate compromised system and to develop an appreciation of system integrity. Specifically, students experimented with sha1sum, a tool used to calculate digests in CentOS and AIDE, an open source integrity product not found in CentOS. As first timer, accessing these tools was very interesting, learned a lot.  They seem pretty to be pretty useful tool and I can see how administrator might leverage them to exploit potential threats and/or conduct routine checks in the real world environment. I'm not really sure what other aspects of the lab could be explored more, but in terms of difficulty I though it was good way for a novice user to practice and experiment in a controlled environment.

## References

**Internet Resources**

https://security.stackexchange.com/questions/165559/why-would-i-choose-sha-256-over-sha-512-for-a-ssl-tls-certificate https://linuxize.com/post/etc-shadow-file/

https://datacadamia.com/crypto/hash/message_digest#:~:text=A%20message%20digest%20is%20the,when%20the%20input%20has%20changed.

https://aide.github.io/doc/

https://linuxize.com/post/how-to-use-nano-text-editor/

https://shapeshed.com/unix-sha1sum/

**Collaboration**

Most of the lab was executed independently. I did run into permission issues in section III.Aide, Steps 3a and 8. I learned from my peers at work that executing a change command and leveraging the nano command I could overcome the permissions issues. I referenced Lab01 to review basic Unix commands, specifically the cd command to change directories. Then I came across *linuxize.com* and learned about the nano command and applied it to get pass the permissions issues I was encountered. Further, I wanted to learn more about `sha1sum` command so I visited *shapeshed.com* to gain a better understanding.