# Lab 1 - Implementing a reference monitor with AppArmor

Vinay Kumar Vennu

9408042910

Vive16@student.bth.se

Neeraj Redddy Avutu

9411053375

neav16@student.bth.se

**INTRODUCTION:**

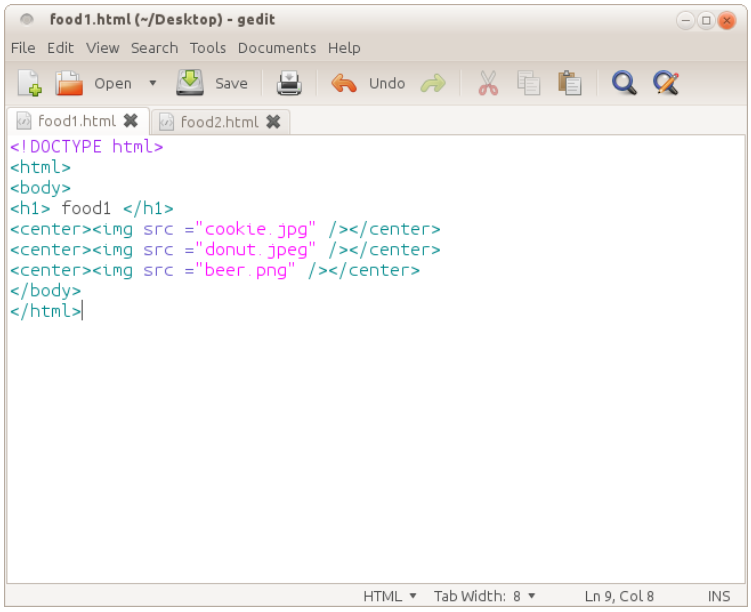
As a part of the assignment, we have generated embedded HTML pages on thttpd server with different image files and are allowed to be displayed after the implementation of AppArmor. We have broken it down from the introduction to the tasks that have been performed as a part of this assignment in the following.

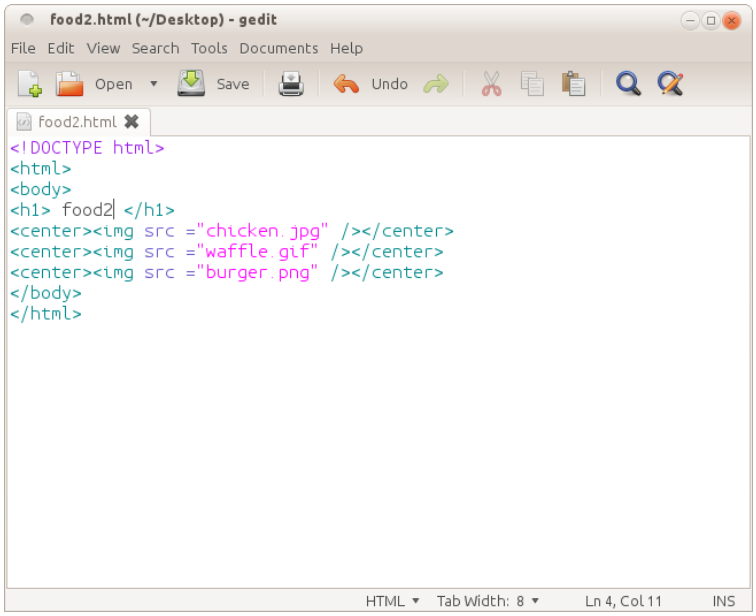
Linux kernels have been able to access different control models that are to be implemented as loadable kernel modules. This has an efficient framework that is general purpose, light weighted and is known as Linux Security Modules **[1].** LSM is defined as a framework that allows the kernel to support a variety of other security modules **[2].**

AppArmor is a Mandatory Access Control (MAC) system which roles as an enhancement that filters the programs to a limited set of resources. This segregation can be done by loading profiles onto the boot. The profiles are of two types of which 'enforce' is one and 'complain' is the other. We use the given two modes as a part of this assignment. These two modes are to be used with commands, aa-enforce and aa-complain respectively. Further, AppArmor allows the system administrator to assigns program with a security profile to restrict the kernel's capabilities.

Additionally, the following commands have been a part of our task completion to update and manage the profiles that have been created. aa-genprof command generates a stand-alone profile and aa-logprof command scans the system log produced in complain and enforce modes. Our approach has been fairly straight-forward. We have created two html pages with different images corresponding to different types (jpg, jpeg, png) and have placed the user in control for allowing and denying the pages to be executed on the thttpd server after the implementation of AppArmor.

The following are the **steps** for the task completion. We have kicked off with the HTML code for the static pages which have image files embedded on them. Below is the image that shows the code that has been used.




STEP 1 The former is the source code is considered to display the page that we have intended to and the latter is the source code of the display page that we consider to deny the display after the implementation of AppArmor.

STEP 2 Now, the experimenting was not a tough task since the idea was very straight-forward and we have carried out the first step with creating files of different formats and making it run on the thttpd server using the command sudo cp -R followed by the filename to be copied and the path to be copied. Refer to the image below.

```
seclab@seclab: ~/Desktop
File  Edit  View  Search  Terminal  Help
seclab@seclab:~$ cd Desktop
seclab@seclab:~/Desktop$ sudo cp -R food1.html /usr/local/thttpd/www
[sudo] password for seclab:
seclab@seclab:~/Desktop$ sudo cp -R food2.html /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R cookie.jpg /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R beer.png /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R donut.jpeg /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R waffle.gif /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R burger.png /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R chicken.jpg /usr/local/thttpd/www
```

STEP 3 Like we have mentioned in the report aa-genprof is executed in this phase to generate an AppArmor profile for the thttpd server [3].

```
seclab@seclab: ~/Desktop
File  Edit  View  Search  Terminal  Help
seclab@seclab:~$ cd Desktop
seclab@seclab:~/Desktop$ sudo cp -R food1.html /usr/local/thttpd/www
[sudo] password for seclab:
seclab@seclab:~/Desktop$ sudo cp -R food2.html /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R cookie.jpg /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R beer.png /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R donut.jpeg /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R waffle.gif /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R burger.png /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R chicken.jpg /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo aa-genprof /usr/local/sbin/thttpd
Writing updated profile for /usr/local/sbin/thttpd.
Setting /usr/local/sbin/thttpd to complain mode.

Before you begin, you may wish to check if a
profile already exists for the application you
wish to confine. See the following wiki page for
more information:
http://wiki.apparmor.net/index.php/Profiles

Please start the application to be profiled in
another window and exercise its functionality now.
```

STEP 4 Starting the thttpd server in a new terminal window using start_thttpd from init.d. Now, thttpd should be up and working. If that is not the case, then check has to be taken if the localhost has a valid address where thttpd should be working.

```
seclab@seclab: ~/Desktop
File  Edit  View  Search  Terminal  Help
Setting /usr/local/sbin/thttpd to complain mode.

Before you begin, you may wish to check if a
profile already exists for the application you
wish to confine. See the following wiki page for
more information:
http://wiki.apparmor.net/index.php/Profiles

Please start the application to be profiled in
another window and exercise its functionality now.

Once completed, select the "Scan" button below in
order to scan the system logs for AppArmor events.

For each AppArmor event, you will be given the
opportunity to choose whether the access should be
allowed or denied.

Profiling: /usr/local/sbin/thttpd

[(S)can system log for AppArmor events] / (F)inish
```

```
seclab@seclab: /etc/init.d
File  Edit  View  Search  Terminal  Help
seclab@seclab:~$ cd ..
seclab@seclab:/home$ cd ..
seclab@seclab:/$ cd etc
seclab@seclab:/etc$ cd init.d
seclab@seclab:/etc/init.d$ sudo start_thttpd
[sudo] password for seclab:
seclab@seclab:/etc/init.d$
```

STEP 5 Scanning the various profiles would yield out different access modifiers such as allow, deny, finish and abort. We planned to allow the intended HTML page and deny the one which we have developed for testing purpose. Hence, allow and deny access modifiers come into play.

```
seclab@seclab: ~/Desktop
File  Edit  View  Search  Terminal  Help
order to scan the system logs for AppArmor events.

For each AppArmor event, you will be given the
opportunity to choose whether the access should be
allowed or denied.

Profiling: /usr/local/sbin/thttpd

[(S)can system log for AppArmor events] / (F)inish
Reading log entries from /var/log/syslog.
Updating AppArmor profiles in /etc/apparmor.d.
Complain-mode changes:

Profile:    /usr/local/sbin/thttpd
Capability: net_bind_service
Severity:   8

 [1 - #include <abstractions/nis>]
  2 - capability net_bind_service

[(A)llow] / (D)eny / Audi(t) / Abo(r)t / (F)inish
```

```
seclab@seclab: /etc/init.d
File  Edit  View  Search  Terminal  Help
seclab@seclab:~$ cd ..
seclab@seclab:/home$ cd ..
seclab@seclab:/$ cd etc
seclab@seclab:/etc$ cd init.d
seclab@seclab:/etc/init.d$ sudo start_thttpd
[sudo] password for seclab:
seclab@seclab:/etc/init.d$
```

STEP 6 After saving the changes to the profile, we locate the generated profile for thttpd that is in the folder apparmor.d extension where editing and viewing are done. Using the command nano.usr.local.sbin.thttpd we made it possible to view and make changes.

```
seclab@seclab: /etc/apparmor.d
File  Edit  View  Search  Terminal  Help
Profiling: /usr/local/sbin/thttpd

[(S)can system log for AppArmor events] / (F)inish
Setting /usr/local/sbin/thttpd to enforce mode.
Reloaded AppArmor profiles in enforce mode.

Please consider contributing your new profile! See
the following wiki page for more information:
http://wiki.apparmor.net/index.php/Profiles

Finished generating profile for /usr/local/sbin/thttpd.
seclab@seclab:~/Desktop$ cd
seclab@seclab:~$ cd ..
seclab@seclab:/home$ cd ..
seclab@seclab:/$ cd etc
seclab@seclab:/etc$ cd apparmor.d
seclab@seclab:/etc/apparmor.d$ ls
abstractions    lightdm-guest-session  usr.bin.evince         usr.sbin.cupsd
cache           local                  usr.bin.firefox        usr.sbin.ntpd
disable         sbin.dhclient          usr.lib.telepathy      usr.sbin.rsyslogd
force-complain  tunables               usr.local.sbin.thttpd  usr.sbin.tcpdump
seclab@seclab:/etc/apparmor.d$
```

```
seclab@seclab: /etc/init.d
File  Edit  View  Search  Terminal  Help
seclab@seclab:~$ cd ..
seclab@seclab:/home$ cd ..
seclab@seclab:/$ cd etc
seclab@seclab:/etc$ cd init.d
seclab@seclab:/etc/init.d$ sudo start_thttpd
[sudo] password for seclab:
seclab@seclab:/etc/init.d$
```

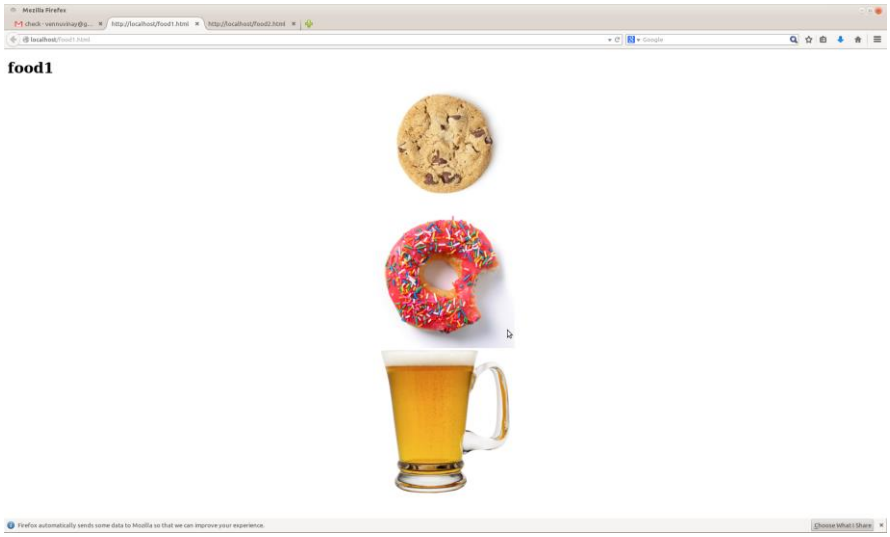
STEP 7 Hence, from the image below, we putforth the results that the server has the ability to allow and deny the HTML pages.

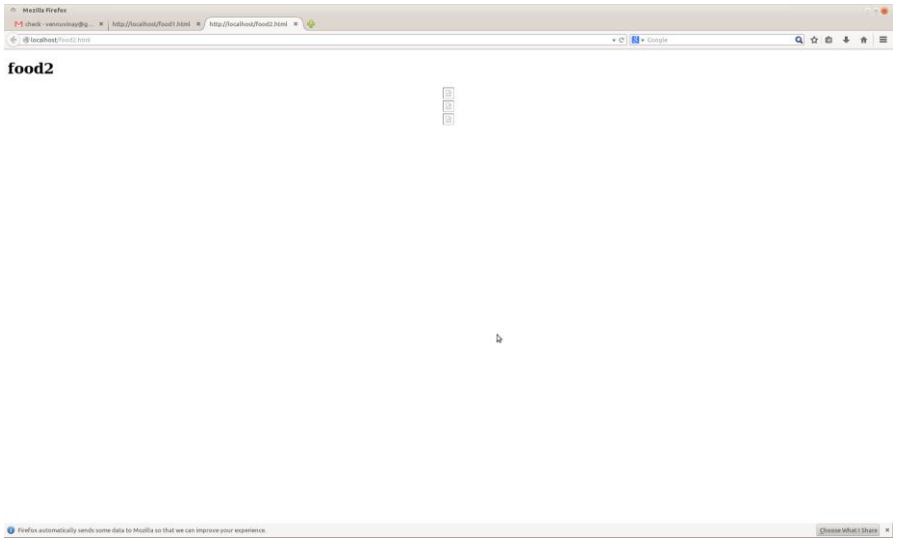
```
seclab@seclab: /etc/apparmor.d
File  Edit  View  Search  Terminal  Help
  GNU nano 2.2.6

# Last Modified: Mon Dec 19 01:31:51 2016
#include <tunables/global>

/usr/local/sbin/thttpd {
  #include <abstractions/apache2-common>
  #include <abstractions/base>
  #include <abstractions/nis>


  capability net_bind_service,
  capability setgid,
  capability setuid,


  deny /usr/local/thttpd/www/burger.png r,
  deny /usr/local/thttpd/www/chicken.jpg r,
  deny /usr/local/thttpd/www/waffle.gif r,

  /usr/local/sbin/thttpd mr,
  /usr/local/thttpd/conf/thttpd.conf r,
  /usr/local/thttpd/log/thttpd.log w,
  /usr/local/thttpd/log/thttpd.pid w,
  /usr/local/thttpd/www/beer.png r,
  /usr/local/thttpd/www/cookie.jpg r,
  /usr/local/thttpd/www/donut.jpeg r,
  /usr/local/thttpd/www/food1.html r,
  /usr/local/thttpd/www/food2.html r,
  /usr/local/thttpd/www/index.html r,

}
```

Note: STEP 8 The changes that are performed can be viewed after entering in complain mode. The complain mode command is sudo aa-complain thttpd. Further, cd var/log ->sudo nano tail -f syslog has to be executed to find the changes that are made which are in the form of a log file of the profiles. Furthermore, aa-enforce mode has the credibility to make the HTML pages run with the desired changes that are enforced.

```
seclab@seclab: /usr/local/sbin
File  Edit  View  Search  Terminal  Help
Setting /etc/apparmor.d/usr.local.sbin.thttpd to enforce mode.
seclab@seclab:~$ cd ..
seclab@seclab:/home$ cd ..
seclab@seclab:/$ cd etc
seclab@seclab:/etc$ cd cds ..
bash: cd: cds: No such file or directory
seclab@seclab:/etc$ cd ..
seclab@seclab:/$ ls
bin    dev   initrd.img  lost+found  opt   run      srv  usr
boot   etc   lib         media       proc  sbin     sys  var
cdrom  home  lib64       mnt         root  selinux  tmp  vmlinuz
seclab@seclab:/$ cd usr
seclab@seclab:/usr$ ls
bin  games  include  lib  local  sbin  share  src
seclab@seclab:/usr$ cd local
seclab@seclab:/usr/local$ ls
bin  etc  games  include  lib  man  sbin  share  src  thttpd  www
seclab@seclab:/usr/local$ cd sbin
seclab@seclab:/usr/local/sbin$ ls
makeweb  start_thttpd  thttpd
seclab@seclab:/usr/local/sbin$ sudo start_thttpd
seclab@seclab:/usr/local/sbin$
```

(STEP10) We conclude at opening HTML finding out that the desired page was only displayed and the page that has been created for testing has not been displayed. Thus, our experimentation has been a success.




**REFLEXION REPORT:**

The learnings as per our understandings have been illustrated in this reflexion report. The report structures as per the difficulty level of the tasks performed as per the time and efforts invested. Kindly refer to the table below for the reflexions.

| S.no | Task Description | Time required to complete each task | Learnings from each task | Level of Difficulty (Level 1-5) |
|---|---|---|---|---|
| 1. | Reviewing various LINUX commands required | 14 hours | Learnings from different implementation of the LINUX commands | Level-1 |
| 2. | Getting the knack of Linux Security Modules | 9 hours | LSM and kernel's interaction to support other security modules | Level-1 |
| 3. | AppArmor and its implementation | 2 days | Generation of the profiles after | Level-4 |

| | | | | |
|---|---|---|---|---|
| | | | understanding the features and reading about the access modifiers. | |
| 4 | AppArmor Profiling | 2 days | Implementing the above. In this phase we have learnt to create profiles and have a hand on experience by creating HTML pages to accessing them. | Level-5 |
| 5. | Document Writing | 7 hours | We have shown the tasks that have been performed in this document. This has been a learning experience for the rookie LINUX users and users who do not have security background. | Level-3 |

**APPENDIX**






**REFERENCES**


**[1]** https://www.usenix.org/legacy/event/sec02/full_papers/wright/wright_html/index.html

**[2]** https://en.wikipedia.org/wiki/Linux_Security_Modules

**[3]** https://www.suse.com/documentation/sled11/singlehtml/apparmor_quickstart/apparmor_quickstart.html