Lab 1 - Implementing a reference monitor with AppArmor

DV2543 - Computer security

Enugurthi Madhukar 9506077537 maen16@student.bth.se

Siddhartha Srinadhuni 9508286854 sisr16@student.bth.se

Introduction:

LSM (Linux Security modules) is a framework that allows Linux Kernel to adhere an array of diversified computer security modules and to avoid bias in selecting a particular implementation model. It is licensed under the terms and regulations of GNU General Public License and it is a part of Linux kernel since Linux 2.6. AppArmor, SElinux, Smack etc are the current modules in the kernel.[1]

Apparmor ("Application armor") is a security software for Linux released under the terms of GNU General Public License. It is a Linux kernel security module which allows the system administrator to restrict program's or application's capabilities with per-program profiles. The profiles generated have abilities such as network access, raw socket access, and the permission to read, write or execute files on matching paths. It allows system administrator to associate each program with a security profile to restrict its capabilities.[2]

The following are the Apparmor command line tools:[3]

aa-genprof: It generates or updates a profile through various functionalities

aa-logprof: It manages Apparmor profiles. Apparmor syslog entries can be viewed through this command

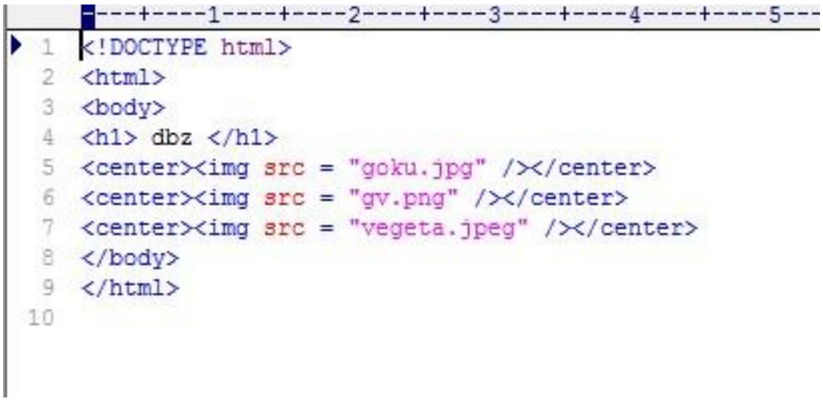
aa-complain: It sets Apparmor to complain mode

aa-enforce: It sets Apparmor from complain to enforce mode

The above mentioned commands were tried and used for the task completion. It's functionalities and the effects of using it were adhered from the videos mentioned in the lab manual to get a clear idea on its implementation.

The aim of this task is to generate static html pages in thttpd server(Tiny/Throtlling/Turbo) through which images in the format JPEG, PNG, or JPG etc that are embedded in the desired html page should be allowed to display in the server. Images of any other html page other than the desired page shouldn't be displayed after implementation of apparmor.

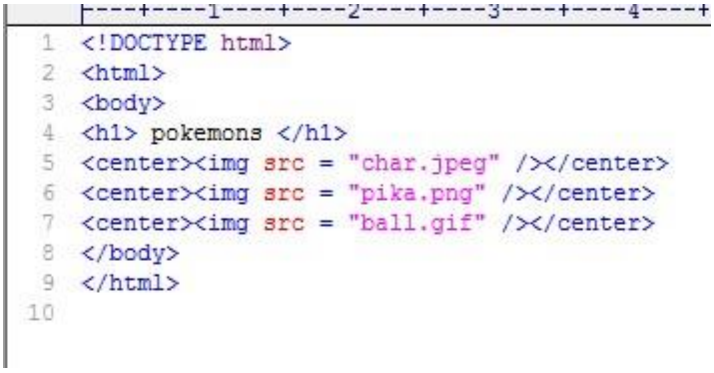
The source code for the desired HTML page is mentioned in figure 1.a



```
<!DOCTYPE html>
<html>
<body>
<h1> dbz </h1>
<center><img src = "goku.jpg" /></center>
<center><img src = "gv.png" /></center>
<center><img src = "vegeta.jpeg" /></center>
</body>
</html>
```

Figure 1.a

The source code for another html page embedded into the server is mentioned in figure 1.b



```
<!DOCTYPE html>
<html>
<body>
<h1> pokemons </h1>
<center><img src = "char.jpeg" /></center>
<center><img src = "pika.png" /></center>
<center><img src = "ball.gif" /></center>
</body>
</html>
```

Figure 1.b

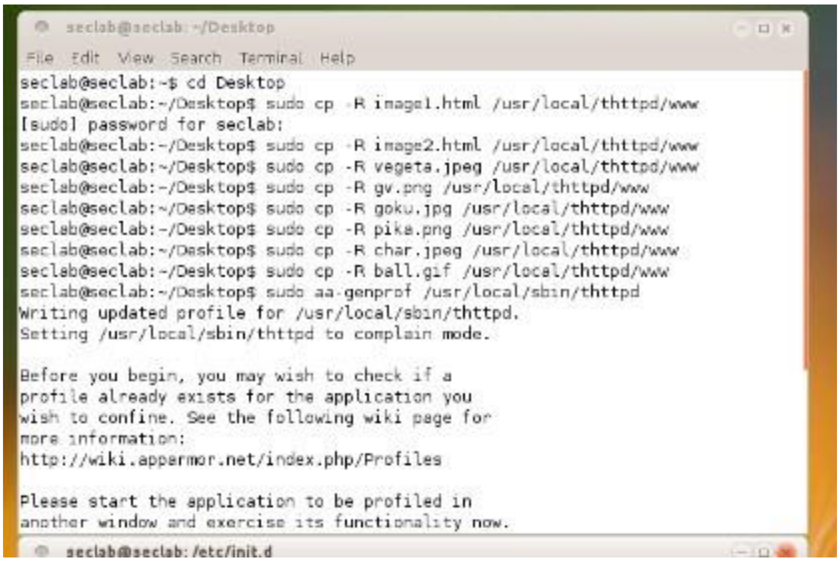
The HTML page with source code mentioned in Figure 1.a is taken as our desired html page where the images are displayed on the web page in the server and the another html page source code mentioned in Figure 1.b is taken for testing purpose.

The implementation of Apparmor for the task and the detailed description of the steps to test the results are mentioned in the next section

IMPLEMENTATION OF APPARMOR(STEPS):

The following section shows the implementation of apparmor for testing the static html page mentioned in the above section

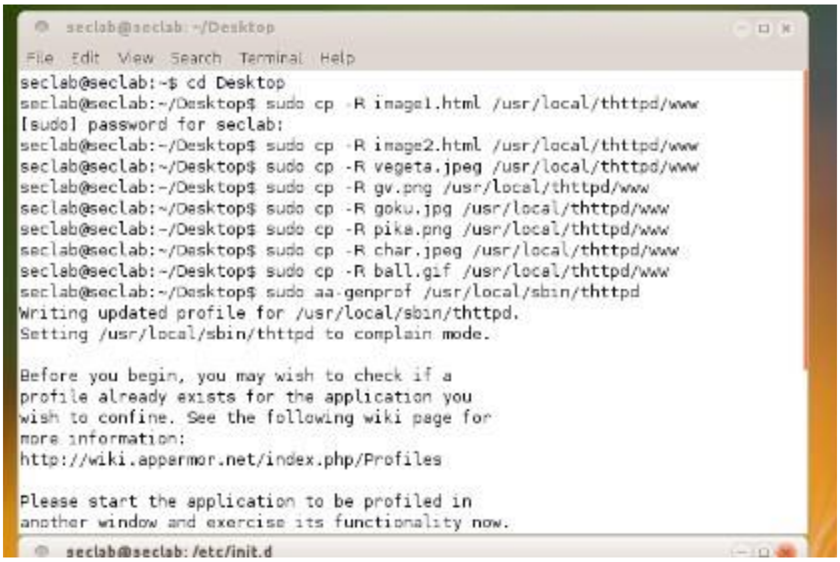
STEP 1: Creating a desired html page with images embedded as follows goku.jpg(JPG format), gv.png(PNG format), vegeta.jpeg(JPEG format) and another HTML page with images embedded as pika.png(PNG format), char.JPEG(JPEG format), ball.gif(GIF format) and moving it www folder in thttpd directory where the html page has to run in the thttpd server with the command sudo cp -R (filename to be copied) (path to be copied) as mentioned in figure 2


seclab@seclab: ~/Desktop
File Edit View Search Terminal Help
seclab@seclab: /etc/init.d

```
seclab@seclab:~$ cd Desktop
seclab@seclab:~/Desktop$ sudo cp -R image1.html /usr/local/thttpd/www
[sudo] password for seclab:
seclab@seclab:~/Desktop$ sudo cp -R image2.html /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R vegeta.jpeg /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R gv.png /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R goku.jpg /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R pika.png /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R char.jpeg /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R ball.gif /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.
```

Figure 2

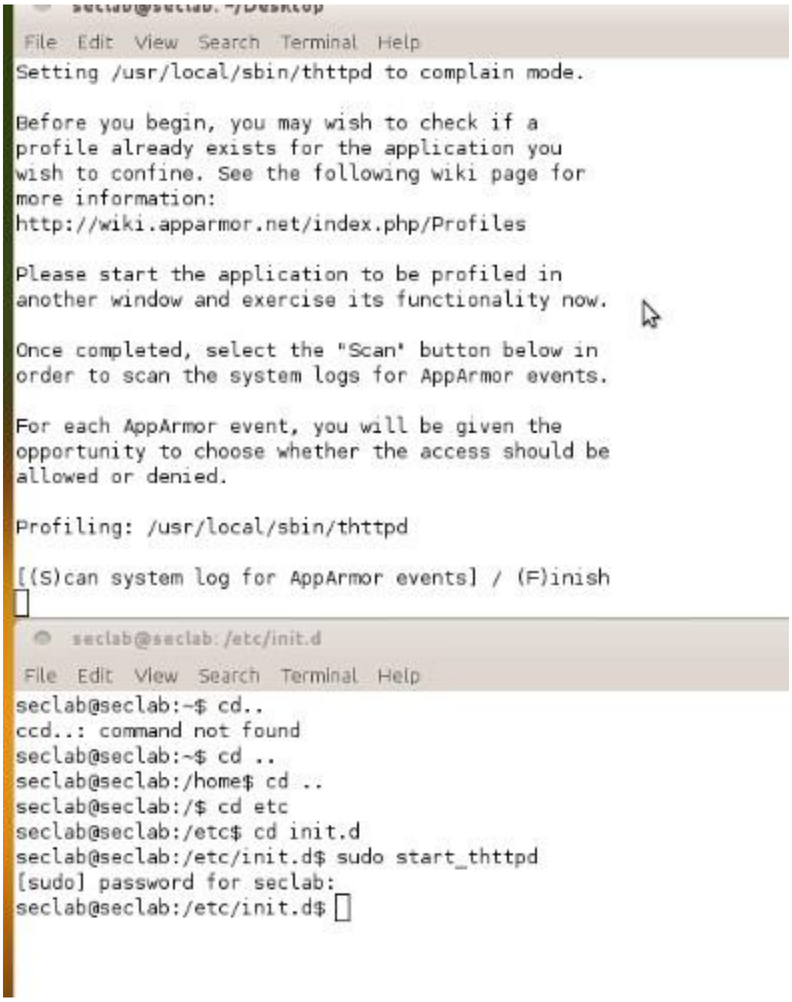
STEP 2: After copying the files into the directory apparmor profile has to be generated for the thttpd server using this command sudo aa-genprof /usr/local/sbin/thttpd as shown in figure 3


seclab@seclab: ~/Desktop
File Edit View Search Terminal Help
seclab@seclab: /etc/init.d

```
seclab@seclab:~$ cd Desktop
seclab@seclab:~/Desktop$ sudo cp -R image1.html /usr/local/thttpd/www
[sudo] password for seclab:
seclab@seclab:~/Desktop$ sudo cp -R image2.html /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R vegeta.jpeg /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R gv.png /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R goku.jpg /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R pika.png /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R char.jpeg /usr/local/thttpd/www
seclab@seclab:~/Desktop$ sudo cp -R ball.gif /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.
```

Figure 3

STEP 3: The next step is to open another terminal and start the thttpd server as shown in figure 4 using command start_thttpd from the directory /etc/init.d as init is the root/parent of all processes executing in linux and we should check whether our html page in working on the server or not as shown in figure image. We should note that localhost= 127.0.0.1 where thttpd is working



```
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..
ccd..: command not found
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$
```

Figure 4

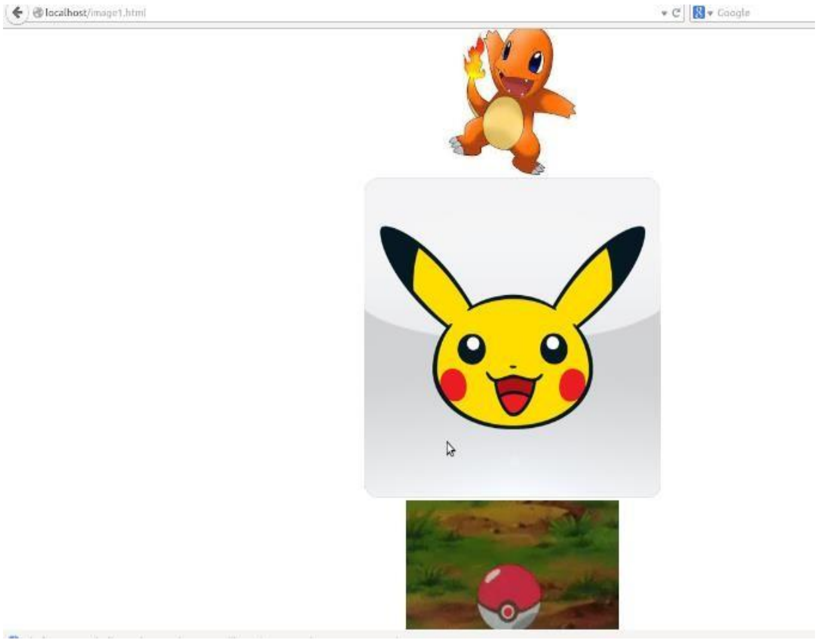


Figure HTML image (4.a)

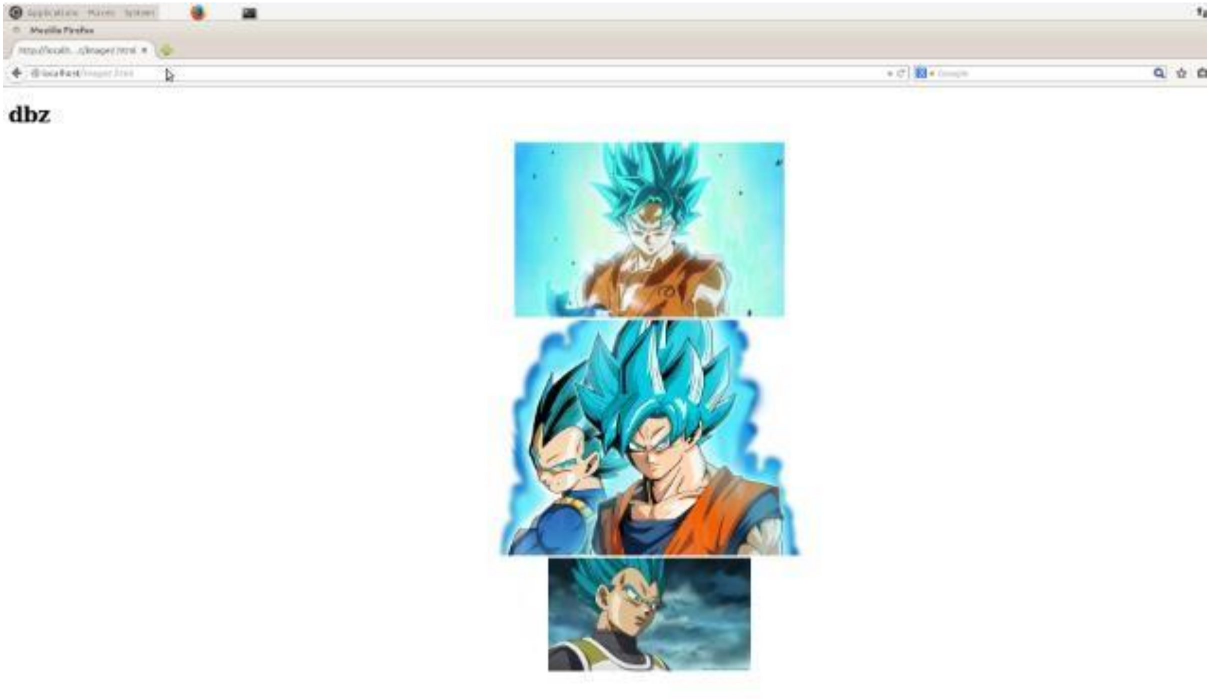
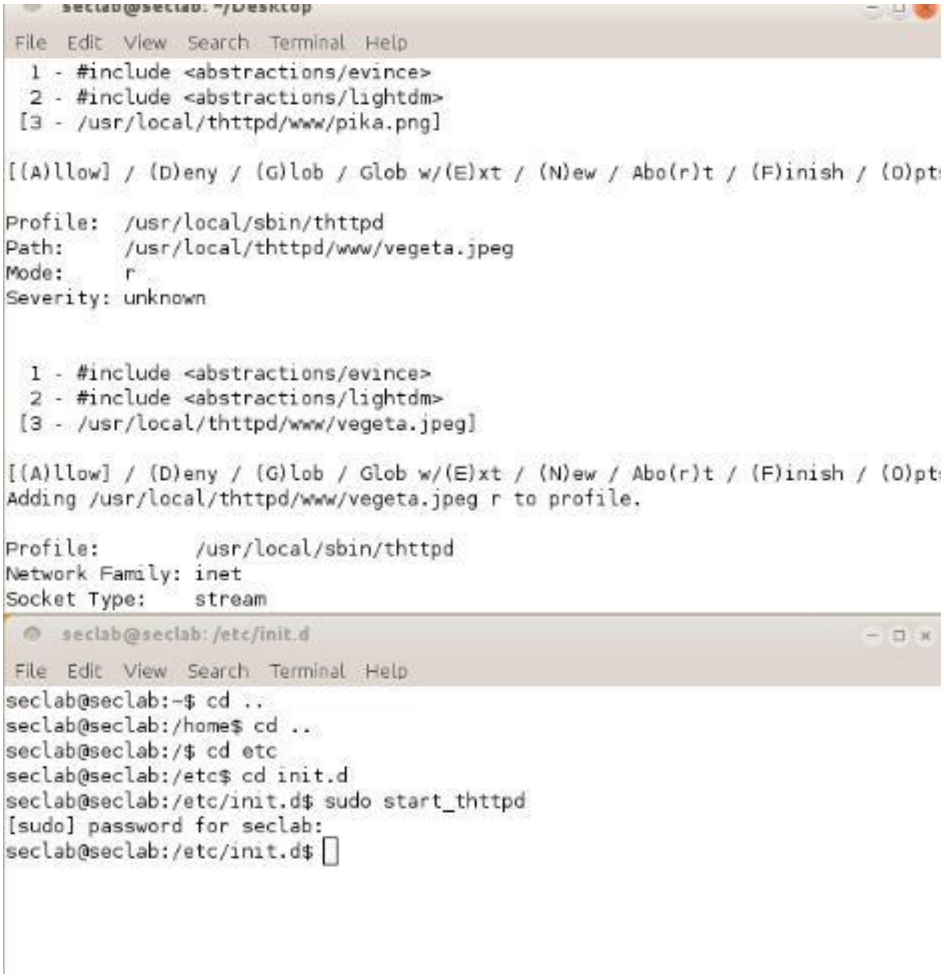


Figure desired HTML page(4.b)

STEP 4: As the server is started, the next step is to check whether the created HTML pages is running on the thttpd server. Once its running , as we have already executed the aa-genprof in another terminal we should scan the generated profile by pressing ‘S’ button where we get several access modifiers such as allow,deny,audit,abort and finish where we can change the behaviour of the existing files in the server as shown in figure 5. So in our case as we are allowing only one html page with embedded images (figure 4.b) to be displayed in the server, deny modifier should be applied to the images of another HTML page (figure 4.b).



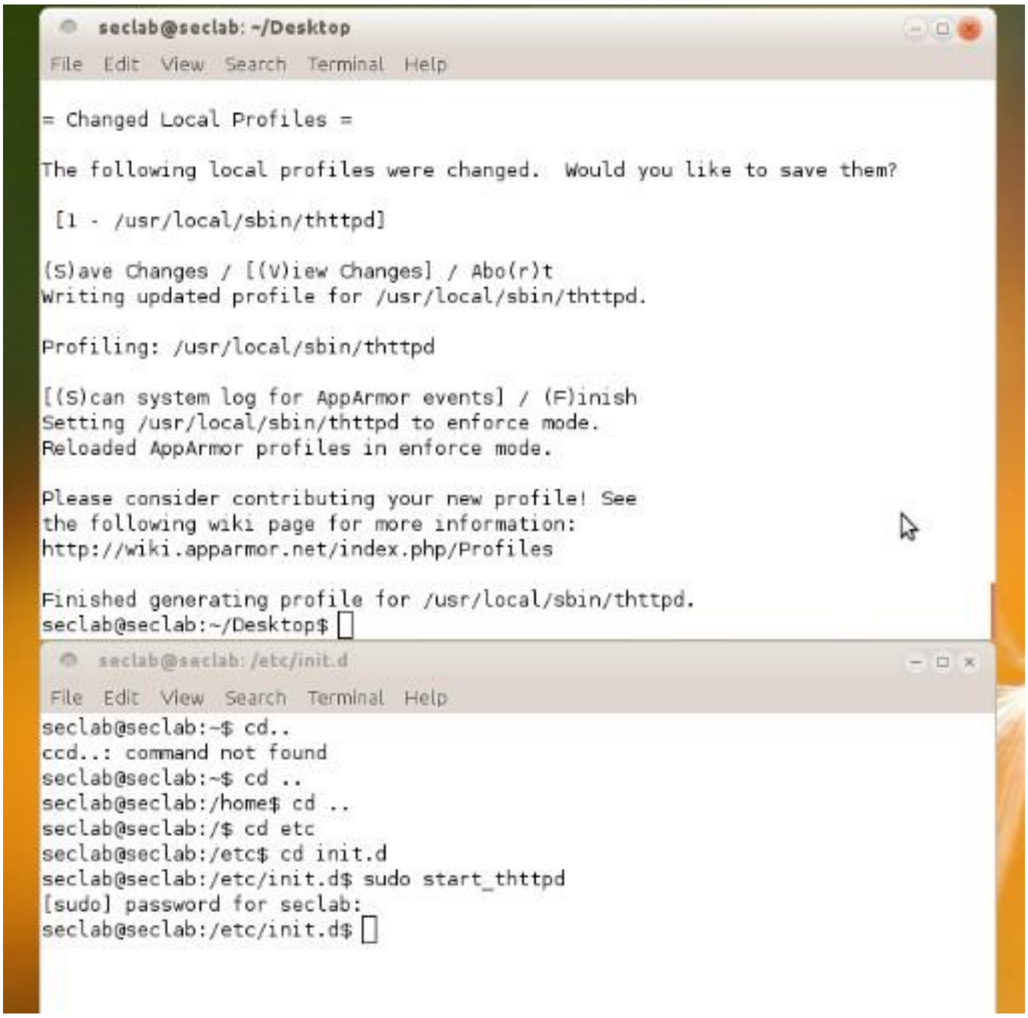
```
  1 - #include <abstractions/evince>
  2 - #include <abstractions/lightdm>
 [3 - /usr/local/thttpd/www/pika.png]

[(A)llow] / (D)eny / (G)lob / Glob w/(E)xt / (N)ew / Abo(r)t / (F)inish / (O)pt

Profile:  /usr/local/sbin/thttpd
Path:     /usr/local/thttpd/www/vegeta.jpeg
Mode:     r
Severity: unknown


  1 - #include <abstractions/evince>
  2 - #include <abstractions/lightdm>
 [3 - /usr/local/thttpd/www/vegeta.jpeg]

[(A)llow] / (D)eny / (G)lob / Glob w/(E)xt / (N)ew / Abo(r)t / (F)inish / (O)pt
Adding /usr/local/thttpd/www/vegeta.jpeg r to profile.

Profile:        /usr/local/sbin/thttpd
Network Family: inet
Socket Type:    stream
```

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$ 
```

Figure 5

STEP 5: Once the modifiers are assigned we should save the changes to the profile as mentioned in figure 6



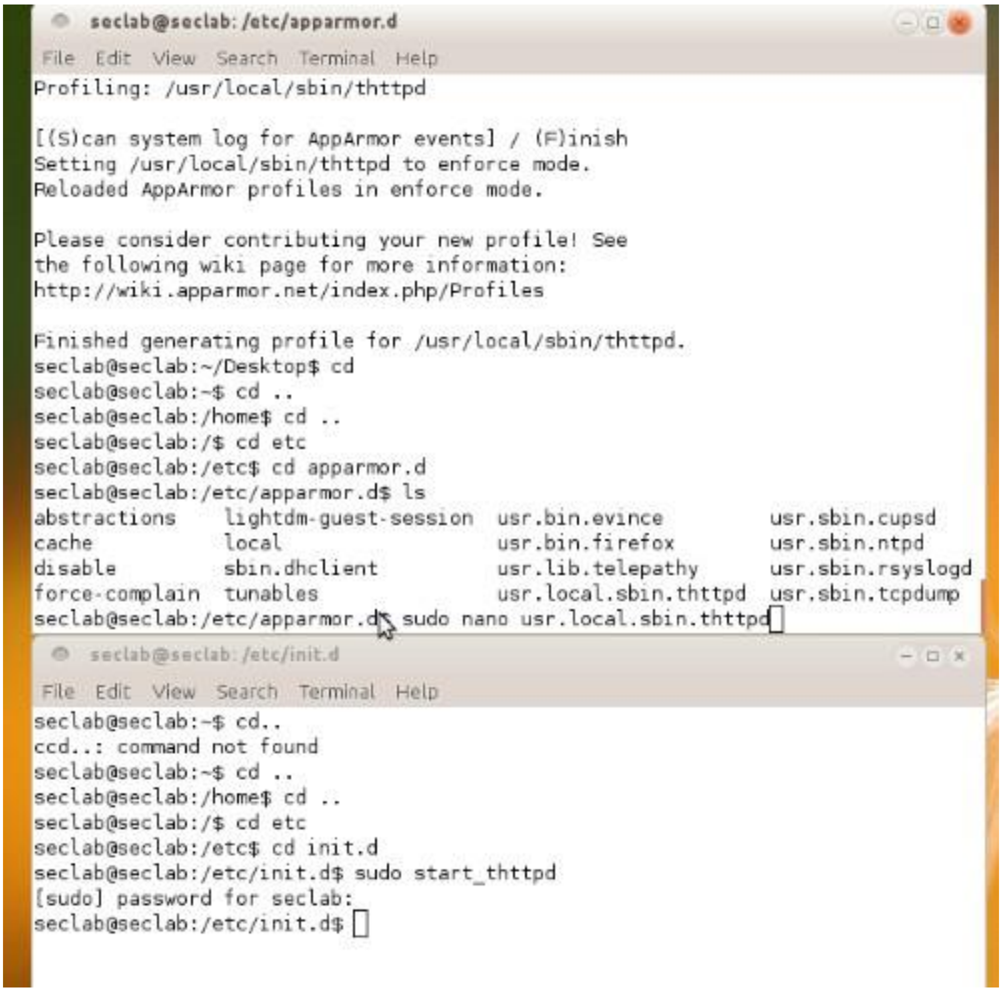
```
= Changed Local Profiles =

The following local profiles were changed.  Would you like to save them?

 [1 - /usr/local/sbin/thttpd]

(S)ave Changes / [(V)iew Changes] / Abo(r)t
Writing updated profile for /usr/local/sbin/thttpd.

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$ 
```

seclab@seclab: /etc/init.d
File Edit View Search Terminal Help

```
seclab@seclab:~$ cd..
ccd..: command not found
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$ 
```

Figure 6

STEP 6: The generated profile for thttpd lies in etc folder apparmor.d extension where we can view and make changes accordingly suitable for the aim of the task as shown in figure 7. It can be opened by executing the command sudo nano usr.local.sbin.thttpd.



seclab@seclab: /etc/apparmor.d
File Edit View Search Terminal Help

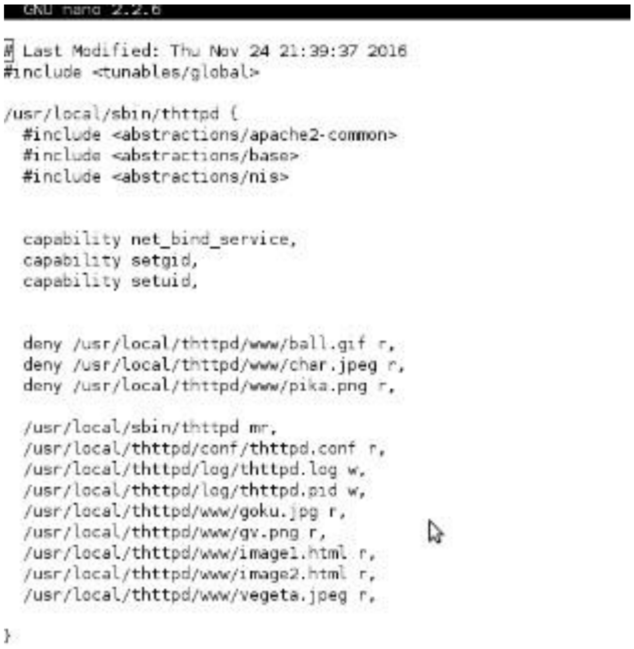
```
Profiling: /usr/local/sbin/thttpd

[(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$ sudo nano usr.local.sbin.thttpd
```

seclab@seclab: /etc/init.d
File Edit View Search Terminal Help

```
seclab@seclab:~$ cd..
ccd..: command not found
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$ 
```

Figure 7

STEP 7: From the generated profile as shown in figure 8, we can conclude that the server now allows only the images to be displayed in our desired HTML page(figure 4.b) and denies the images of the other HTML page(4.a) created for testing purpose.



```
  GNU nano 2.2.6

# Last Modified: Thu Nov 24 21:39:37 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/ball.gif r,
  deny /usr/local/thttpd/www/char.jpeg r,
  deny /usr/local/thttpd/www/pika.png 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/goku.jpg r,
  /usr/local/thttpd/www/gv.png r,
  /usr/local/thttpd/www/image1.html  r,
  /usr/local/thttpd/www/image2.html  r,
  /usr/local/thttpd/www/vegeta.jpeg r,

}
```

Figure 8

STEP 8: After generation of profile has to be in complain mode and enforce mode to foresee the changes , it is done by executing the following commands

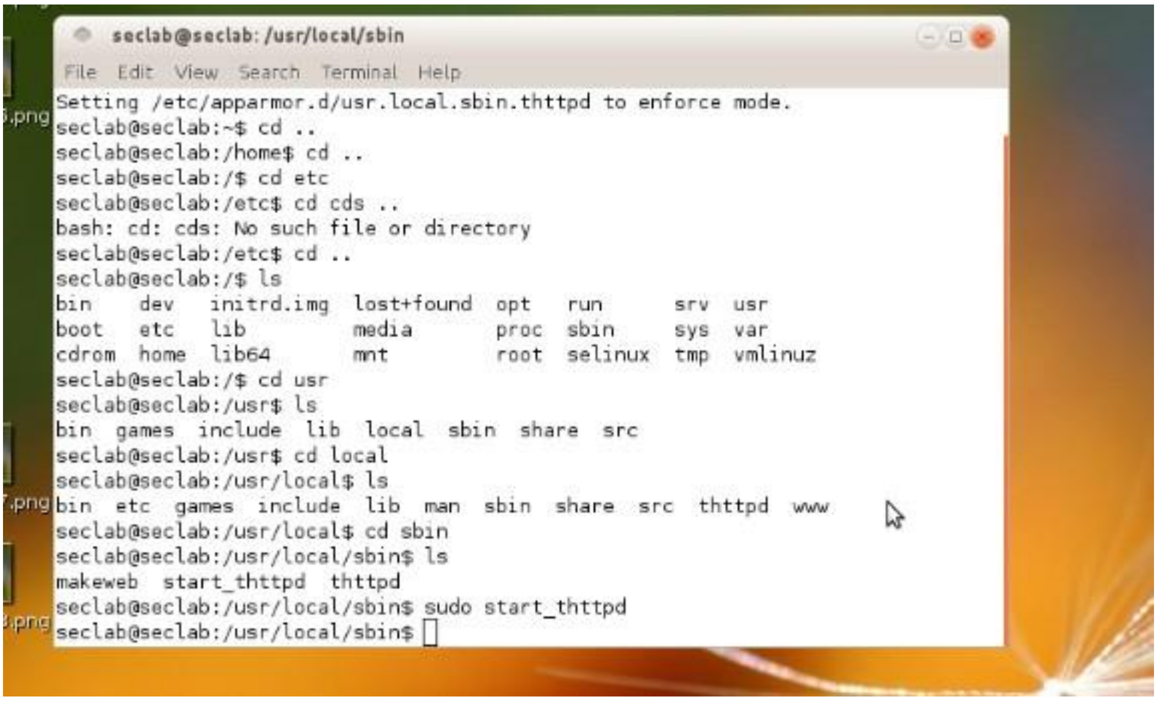
sudo aa-complain thttpd

sudo aa-enforce thttpd

once we enter into the complain mode we can see the changes made to it by executing

cd var/log -> cd sudo nano tail -f syslog . After executing this command we can find the log file of the profile. aa-enforce enables us to run the HTML page to see the desired changes after changing the behaviour of the generated profile by making changes.

STEP 9 : After the above step thttpd server did not show any changes made to the HTML page and it crashed several times , so we restarted the system, cleared cookies in the browser and the executed the above commands again and started the thttpd server through sbin folder as mentioned in the figure 9


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$ 
```

Figure 9

STEP 10: We opened the HTML pages after the completion of the above step and found out that our desired HTML page was intact (figure 10.a) and the another one was displayed with broken images as shown in figure 10.b, hence our test was successful.

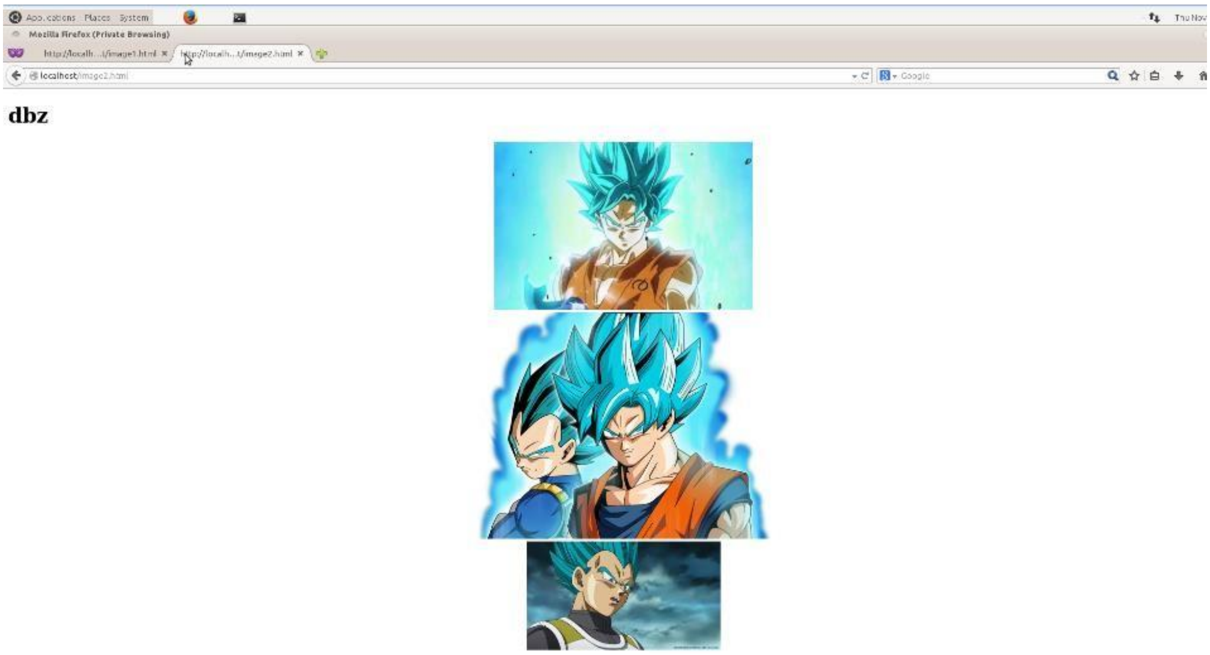
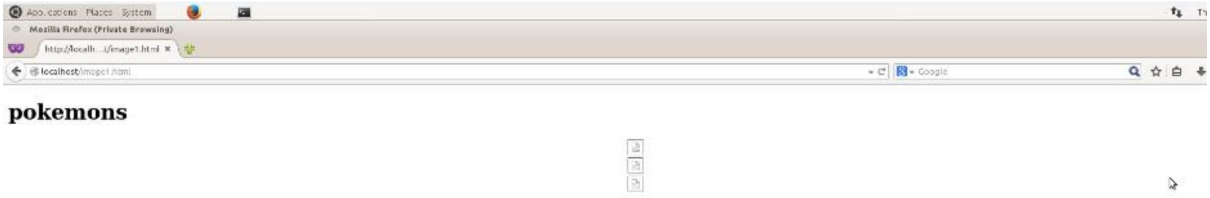

Applications Places System
Mozilla Firefox (Private Browsing)
http://localh...t/image1.html
http://localh...t/image2.html
localhost/image2.html
Google
dbz

Figure 10.a


Applications Places System
Mozilla Firefox (Private Browsing)
http://localh...i/image1.html
localhost/image1.html
Google

pokemons

Figure 10.b

REFLEXION:

This reflexion report includes the tasks that we have broken down, amount of time invested to complete the tasks, complexity of the task and the learnings out of the tasks performed. Since we are naïve in terms of hands on experience with LINUX and the idea on linux security modules and taking the initial steps into security, it took lot of time for us for the above implementation. The time log taken to table below illustrates the further reflexions.

S.no	Tasks	Time Invested	Learnings	Difficulty Level
1.	Thorough overview of Linux commands	12 hours	Understanding the implementation of the commands	Easy
2.	LSM(Linux Security Modules)	1 day	Learning above the modules present and it’s	Medium

			description and implementation	
3.	AppArmor	1 day	Learning the features of Apparmor and its functionality	Medium
4.	thttpd	18 hours	We have wasted a lot of time in configuring thttpd server in our personal computer. After several attempts we decided to do in seclab	Hard
5.	Implementing apparmor	2 days	It took a lot of time for us to generate profile as after several attempts we could start thttpd server from the correct path	Hard
6.	Apparmor profile	2 days	After the generation of profile it took us a while to understand the blueprint of the profile and make changes to it accordingly as per the aim of the given task	hard
7	thttpd	3 hours	As thttpd used to crash often after making changes to the profile , By restarting the system we got the desired output	Medium
8	Report Writing	10 hours	We clearly portrayed the steps involved in	Easy

			completing the task	

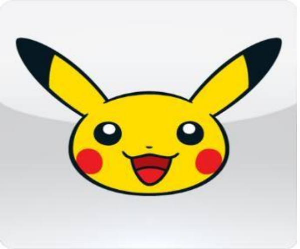
REFERENCES :

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

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

APPENDIX :

The following are the images embedded in the HTML page.



Pika.png

char.jpeg

ball.gif

Goku.jpg

gv.png

vegeta.jpeg