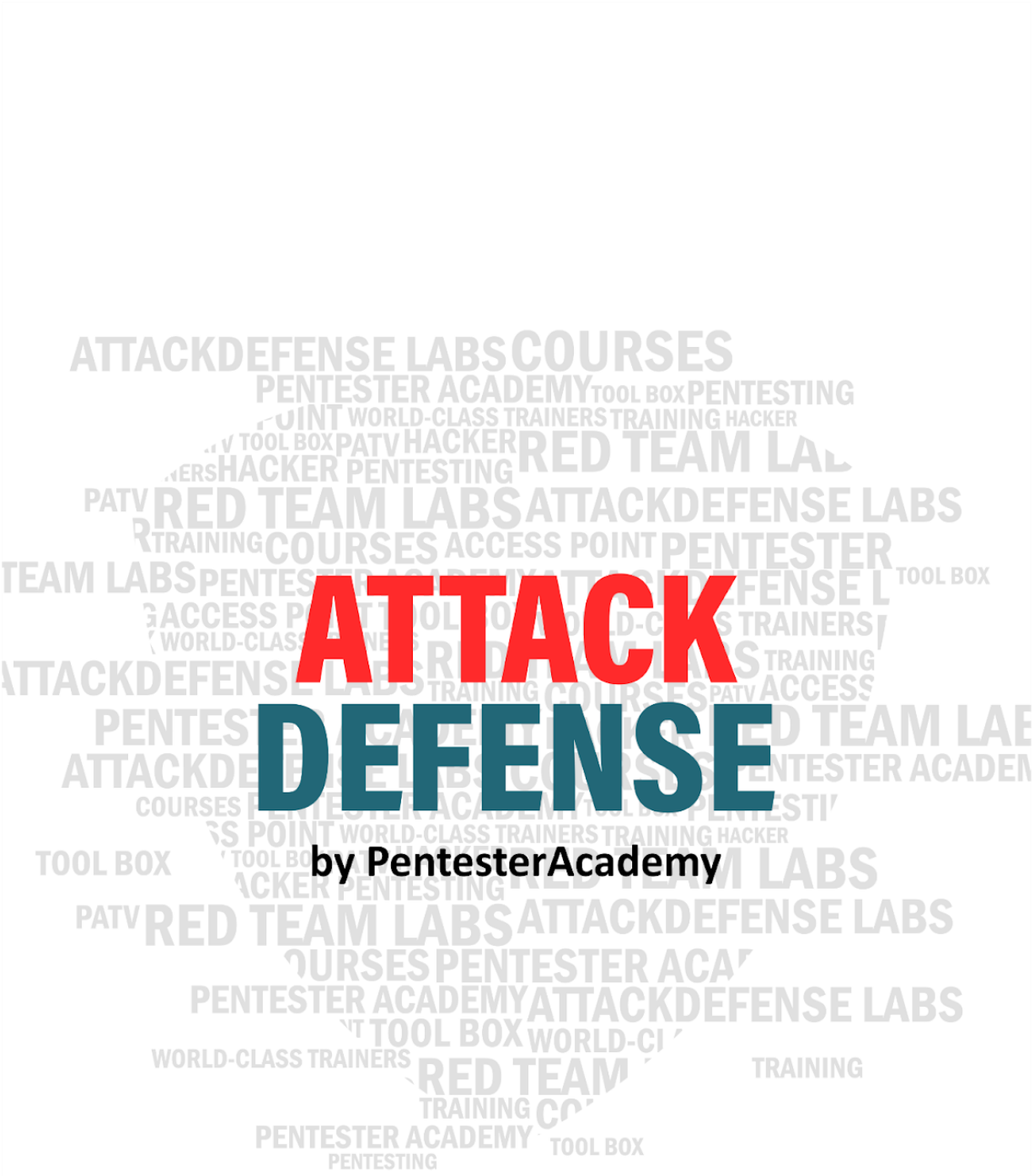
ATTACK
DEFENSE
by PentesterAcademy

Name	Confining Containers with AppArmor
URL	https://attackdefense.com/challengedetails?cid=1835
Type	Privilege Escalation : AppArmor

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Objective: Learn how to use AppArmor profiles with Docker for confining the containers.

Solution:

Step 1: Check if the apparmor support is present in the Docker.

Command: docker info | grep apparmor

```
root@localhost:~# docker info | grep apparmor
WARNING: No swap limit support
  apparmor
root@localhost:~#
```

Docker default profile

Step 2: Check the AppArmor status.

Command: sudo aa-status

```
root@localhost:~# aa-status
apparmor module is loaded.
50 profiles are loaded.
13 profiles are in enforce mode.
   /sbin/dhclient
   /usr/bin/lxc-start
   /usr/lib/NetworkManager/nm-dhcp-client.action
   /usr/lib/NetworkManager/nm-dhcp-helper
   /usr/lib/chromium-browser/chromium-browser//browser_java
   /usr/lib/chromium-browser/chromium-browser//browser_openjdk
   /usr/lib/chromium-browser/chromium-browser//sanitized_helper
   /usr/lib/connman/scripts/dhclient-script
   docker-default
   lxc-container-default
37 profiles are in complain mode.
   /usr/lib/chromium-browser/chromium-browser
   /usr/lib/chromium-browser/chromium-browser//chromium_browser_sandbox
   /usr/lib/chromium-browser/chromium-browser//lsb_release
   /usr/lib/chromium-browser/chromium-browser//xdgsettings
   /usr/lib/dovecot/anvil
   /usr/lib/dovecot/auth
   /usr/lib/dovecot/config
   /usr/lib/dovecot/deliver
   /usr/lib/dovecot/dict
   /usr/lib/dovecot/dovecot-auth
0 processes have profiles defined.
0 processes are in enforce mode.
0 processes are in complain mode.
0 processes are unconfined but have a profile defined.
root@localhost:~#
```

The default apparmor profile for docker “docker-default” is present in the enforce mode.

Step 3: On Docker versions 1.13.0 and later, the Docker binary generates this profile in tmpfs and then loads it into the kernel. On Docker versions earlier than 1.13.0, this profile is generated in /etc/apparmor.d/docker instead.

Also this profile is for the containers and NOT for the Docker daemon.
Reference: https://docs.docker.com/engine/security/apparmor/

However, docker-default.profile is provided in the lab (by AttackDefense lab team) so the user can check it.

Command: cat docker-default.profile

```
root@localhost:~# cat docker-default.profile
profile docker-default flags=(attach_disconnected,mediate_deleted) {
  #include <abstractions/base>
  network,
  capability,
  file,
  umount,
  deny @{PROC}/* w,   # deny write for all files directly in /proc (not in a subdir)
  # deny write to files not in /proc/<number>/** or /proc/sys/**
  deny @{PROC}/{[^1-9],[^1-9][^0-9],[^1-9s][^0-9y][^0-9s],[^1-9][^0-9][^0-9][^0-9]*}/** w,
  deny @{PROC}/sys/[^k]** w,  # deny /proc/sys except /proc/sys/k* (effectively /proc/sys/kernel)
  deny @{PROC}/sys/kernel/{?,??,[^s][^h][^m]**} w,  # deny everything except shm* in /proc/sys/kernel/
  deny @{PROC}/sysrq-trigger rwklx,
  deny @{PROC}/mem rwklx,
  deny @{PROC}/kmem rwklx,
  deny @{PROC}/kcore rwklx,
  deny mount,
  deny /sys/[^f]*/** wklx,
  deny /sys/f[^s]*/** wklx,
  deny /sys/fs/[^c]*/** wklx,
  deny /sys/fs/c[^g]*/** wklx,
  deny /sys/fs/cg[^r]*/** wklx,
  deny /sys/firmware/efi/efivars/** rwklx,
  deny /sys/kernel/security/** rwklx,
  # suppress ptrace denials when using 'docker ps' or using 'ps' inside a container
  ptrace (trace,read) peer=docker-default,
}
```

Code to generate it: https://github.com/moby/moby/blob/master/profiles/apparmor/template.go

To summarise the rules mentioned in this profile:

- Access to all networking
- No capability is defined (However, some capabilities will come from including basic base rules i.e. #include <abstractions/base>)
- Writing to any /proc file is not allowed
- Other subdirectories/files of /proc and /sys are denied read/write/lock/link/execute access
- Mount is not allowed
- Ptrace can only be run on a process that is confined by same apparmor profile

Running container with docker-default profile

Step 4: List the docker images present on the machine.

Command: docker images

```
root@localhost:~# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
modified-ubuntu     latest              54ee2a71bdef        5 months ago        855MB
ubuntu              18.04               775349758637        5 months ago        64.2MB
alpine              latest              965ea09ff2eb        6 months ago        5.55MB
root@localhost:~#
```

Step 5: Run alpine image in interactive mode.

Command: docker run -it alpine sh

```
root@localhost:~# docker run -it alpine sh
/ #
/ #
```

Step 6: Open another terminal T2 and check apparmor status.

Command: aa-status

```
1 processes have profiles defined.
1 processes are in enforce mode.
   docker-default (749)
0 processes are in complain mode.
0 processes are unconfined but have a profile defined.
root@localhost:~#
```

One can observe that the container is running confined with docker-default profile in enforce mode.

Running container with unconfined profile

Step 7: Run alpine image in interactive mode with unconfined profile.

Command: docker run -it --security-opt apparmor=unconfined alpine sh

```
root@localhost:~# docker run -it --security-opt apparmor=unconfined alpine sh
/ #
```

Step 8: Switch to terminal T2 and check apparmor status.

Command: aa-status

```
0 processes have profiles defined.
0 processes are in enforce mode.
0 processes are in complain mode.
0 processes are unconfined but have a profile defined.
root@localhost:~#
```

This time, the docker-default profile is not listed here. Please note that it will also not show up in unconfined processes because the profile is not defined for it.

Apparmor even blocks capabilities

Step 9: Run modified-ubuntu image in interactive mode with seccomp unconfined and default docker profile. And, give it SYS_ADMIN capability.

Command: docker run -it --cap-add SYS_ADMIN --security-opt seccomp=unconfined modified-ubuntu sh

```
root@localhost:~# docker run -it --cap-add SYS_ADMIN --security-opt seccomp=unconfined modified-ubuntu bash
root@d5317648fde2:~#
root@d5317648fde2:~#
```

Step 10: Try to overwrite /proc/stat file

Command: echo "" > /proc/stat

```
root@d5317648fde2:~# echo "" > /proc/stat
bash: /proc/stat: Permission denied
root@d5317648fde2:~#
```

Attempt failed even when SYS_ADMIN capability is given because the docker-default profile has a rule which disallows write on all /proc files.

Step 11: Run modified-ubuntu image in interactive mode with seccomp unconfined and unconfined apparmor profile. And, give it SYS_ADMIN capability.

Command: docker run -it --cap-add SYS_ADMIN --security-opt seccomp=unconfined --security-opt apparmor=unconfined modified-ubuntu sh

```
root@localhost:~# docker run -it --cap-add SYS_ADMIN --security-opt seccomp=unconfined --security-opt apparmor=unconfined modified-ubuntu bash
root@a7d43755dd09:~#
root@a7d43755dd09:~#
```

Step 12: Try to overwrite /proc/stat file

Command: echo "" > /proc/stat

```
root@a7d43755dd09:~# echo "" > /proc/stat
bash: echo: write error: Input/output error
root@a7d43755dd09:~#
```

This time the write permission was granted (write however caused some issue which can be ignored here) because the apparmor docker-default profile was not enforced on this container.

Creating custom profile

Step 13: Create a copy of the given docker-default profile and modify it.

Commands:
cp docker-default.profile /etc/apparmor.d/docker.mod
vim /etc/apparmor.d/docker.mod

```
root@localhost:~# cp docker-default.profile /etc/apparmor.d/docker.mod
root@localhost:~#
root@localhost:~# vim /etc/apparmor.d/docker.mod
root@localhost:~#
```

After modification the profile should look like the following:

```
root@localhost:~# cat /etc/apparmor.d/docker.mod
profile docker flags=(attach_disconnected,mediate_deleted) {
  network,
  capability,
  file,
  umount,
  deny mount,
  deny /sys/[^f]*/** wklx,
  deny /sys/f[^s]*/** wklx,
  deny /sys/fs/[^c]*/** wklx,
  deny /sys/fs/c[^g]*/** wklx,
  deny /sys/fs/cg[^r]*/** wklx,
  deny /sys/firmware/efi/efivars/** rwklx,
  deny /sys/kernel/security/** rwklx,
}
root@localhost:~#
```

To summarise the changes:

- All rules related to /proc are removed.
- Ptrace access is also removed.
- The name of the profile is changed to “docker”.

Step 14: Enforce this profile.

Commands: aa-enforce /etc/apparmor.d/docker.mod

Step 15: Check the apparmor status.

Commands: aa-status

```
root@localhost:~# aa-status
apparmor module is loaded.
51 profiles are loaded.
14 profiles are in enforce mode.
   /sbin/dhclient
   /usr/bin/lxc-start
   /usr/lib/NetworkManager/nm-dhcp-client.action
   /usr/lib/NetworkManager/nm-dhcp-helper
   /usr/lib/chromium-browser/chromium-browser//browser_java
   /usr/lib/chromium-browser/chromium-browser//browser_openjdk
   /usr/lib/chromium-browser/chromium-browser//sanitized_helper
   /usr/lib/connman/scripts/dhclient-script
   docker
   docker-default
   lxc-container-default
```

The newly created docker profile is listed under enforce mode.

Step 16: Run modified-ubuntu image in interactive mode with default profile with SYS_ADMIN capability. And, try to run strace on process with ID 1.

Command: docker run -it --cap-add SYS_PTRACE modified-ubuntu sh

```
root@localhost:~# docker run -it --cap-add SYS_PTRACE  modified-ubuntu bash
root@a1dc2196b4f5:~#
root@a1dc2196b4f5:~#
root@a1dc2196b4f5:~# strace -p 1
strace: Process 1 attached
wait4(-1, ^Cstrace: Process 1 detached
 <detached ...>
root@a1dc2196b4f5:~#
```

The strace was able to run on PID 1.

Step 16: Run modified-ubuntu image in interactive mode with newly created apparmor profile. And, give it SYS_PTRACE capability.

Command: docker run -it --cap-add SYS_PTRACE --security-opt apparmor=docker modified-ubuntu bash

```
root@localhost:~# docker run -it --cap-add SYS_PTRACE --security-opt apparmor=docker modified-ubuntu bash
root@ca33f4049cf2:~#
root@ca33f4049cf2:~#
root@ca33f4049cf2:~# strace -p 1
^C
root@ca33f4049cf2:~#
```

And as the ptrace capability is removed from the profile, the strace is not able to run.

Step 17: The same can be verified from the logs present in the audit.log file.

Command: cat /var/log/audit/audit.log | grep apparmor

```
type=AVC msg=audit(1587886102.588:1537): apparmor="AUDIT" operation="getattr" profile="docker" name="/lib/x86_64-linux-gnu/libpthread-2.27.so" p
id=2105 comm="strace" requested_mask="r" fsuid=0 ouid=0
type=AVC msg=audit(1587886102.588:1538): apparmor="AUDIT" operation="file_mmap" profile="docker" name="/lib/x86_64-linux-gnu/libpthread-2.27.so"
 pid=2105 comm="strace" requested_mask="r" fsuid=0 ouid=0
type=AVC msg=audit(1587886102.612:1539): apparmor="DENIED" operation="ptrace" profile="docker" pid=2105 comm="strace" requested_mask="trace" den
ied_mask="trace" peer="docker"
type=AVC msg=audit(1587886102.612:1539): apparmor="DENIED" operation="ptrace" profile="docker" pid=2105 comm="strace" requested_mask="tracedby"
denied_mask="tracedby" peer="docker"
type=AVC msg=audit(1587886102.612:1540): apparmor="DENIED" operation="signal" profile="docker" pid=2105 comm="strace" requested_mask="send" denie
d_mask="send" signal=kill peer="docker"
type=AVC msg=audit(1587886102.612:1540): apparmor="DENIED" operation="signal" profile="docker" pid=2105 comm="strace" requested_mask="receive" d
enied_mask="receive" signal=kill peer="docker"
type=AVC msg=audit(1587886146.436:1541): apparmor="AUDIT" operation="getattr" profile="docker" name="/dev/pts/0" pid=2031 comm="bash" requested_
mask="r" fsuid=0 ouid=0
```

Learning:

- AppArmor can help us to confine a binary only to one task (i.e. read a specific file).
- Similarly, by extrapolating the same, such task specific binaries/scripts/processes can be created for the low-privilege user.
-

References:

- AppArmor man page (http://manpages.ubuntu.com/manpages/bionic/man7/apparmor.7.html) (http://manpages.ubuntu.com/manpages/bionic/man5/apparmor.d.5.html)
- Beginning AppArmor profile development (https://ubuntu.com/tutorials/beginning-apparmor-profile-development)