Containerization: A Comprehensive Guide

What is Containerization?

Containerization is a lightweight method of running applications in isolated environments called **containers**. Unlike virtual machines, containers share the host system's kernel, making them more efficient and portable.

Key Benefits

- Lightweight and fast
- Consistent across development, testing, and production
- Scalable and portable
- Secure with proper configuration

Docker

Docker is the most popular containerization platform. It uses container images to package apps and dependencies.

Concepts

- **Containers**: Isolated environments
- **Dockerfile**: Instructions for building images
- **Docker Hub**: Repository for images

Installation on Ubuntu

```
sudo apt update -y
sudo apt install ca-certificates curl gnupg lsb-release -y
sudo mkdir -m 0755 -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --
dearmor -o /etc/apt/keyrings/docker.gpg
echo "deb [arch=$(dpkg --print-architecture) signed-
by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu
$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list >
/dev/null
sudo apt update -y
sudo apt install docker-ce docker-ce-cli containerd.io docker-buildx-plugin
docker-compose-plugin -y
sudo usermod -aG docker htb-student
```

Basic Docker Commands

Command	Description
`docker ps`	List running containers

Command	Description
`docker stop <container>`	Stop container
`docker start <container>`	Start container
`docker run`	Run container from image
`docker rm <container>`	Remove container
`docker rmi <image>`	Remove image
`docker logs <container>`	View logs

Example Dockerfile

```
FROM ubuntu:22.04
RUN apt-get update && apt-get install -y apache2 openssh-server && rm -rf
/var/lib/apt/lists/*
RUN useradd -m docker-user && echo "docker-user:password" | chpasswd
RUN chown -R docker-user:docker-user /var/www/html /var/run/apache2
/var/log/apache2 /var/lock/apache2
RUN usermod -aG sudo docker-user && echo "docker-user ALL=(ALL) NOPASSWD:
ALL" >> /etc/sudoers
EXPOSE 22 80
CMD service ssh start && /usr/sbin/apache2ctl -D FOREGROUND
```

Build and Run

```
docker build -t FS_docker .
docker run -p 8022:22 -p 8080:80 -d FS_docker
```

Linux Containers (LXC)

LXC is a system-level containerization technology that uses the host's kernel to run isolated Linux systems.

Installation

```
sudo apt-get install lxc lxc-utils -y
```

Create LXC Container

```
sudo lxc-create -n linuxcontainer -t ubuntu
```

LXC Management Commands

Command	Description
`lxc-ls`	List containers
`lxc-start -n <name>`	Start container
`lxc-stop -n <name>`	Stop container
`lxc-attach -n <name>`	Connect to container

Security Configuration

```
lxc.cgroup.cpu.shares = 512
lxc.cgroup.memory.limit_in_bytes = 512M
```

Apply Resource Limits

```
sudo vim /usr/share/lxc/config/linuxcontainer.conf
sudo systemctl restart lxc.service
```

Namespaces and Isolation

- **PID namespace**: Process isolation
- **NET namespace**: Network isolation
- **MNT namespace**: Filesystem isolation

Use Cases for Penetration Testers

- Isolated environments for malware/exploit testing
- Lightweight setup for target simulations
- Secure, reproducible test systems

Security Best Practices

- Restrict container access
- Use minimal base images
- Enable AppArmor/SELinux
- Set resource limits via cgroups
- Keep containers up-to-date

Conclusion: Containerization (via Docker or LXC) empowers developers and penetration testers with consistent, secure, and scalable environments.