KPLABS Course

Docker Certified Associate 2020

Security

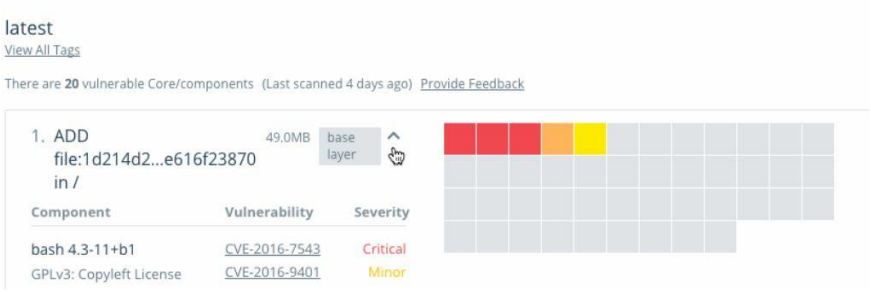
ISSUED BY
Zeal Vora

REPRESENTATIVE
instructors@kplabs.in

Module 1: Overview of Container Security Scanning

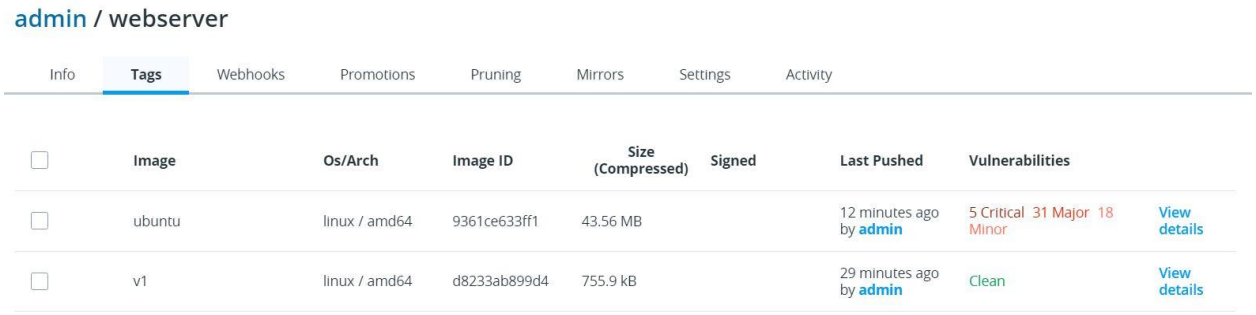
Docker Containers can have security vulnerabilities.

If blindly pulled and if containers are running in production, it can result in a breach.



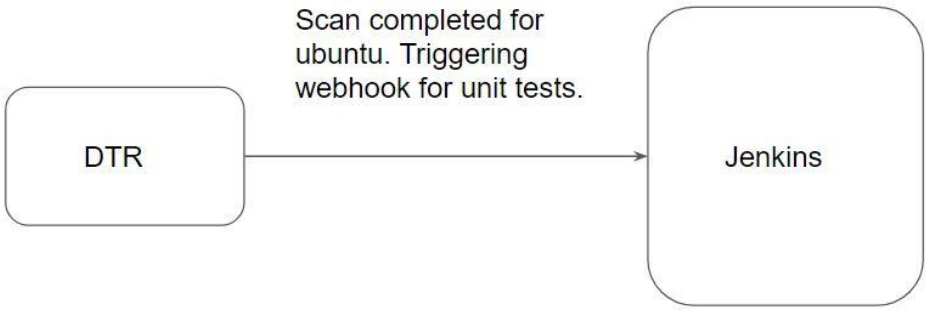
DTR allows us to perform a security scan for the containers.

These scans can perform “On Push” or even manually.



Module 2: DTR Webhooks

You can configure DTR to automatically post-event notifications to a webhook URL of your choosing




```
{
  "type": "TAG_PUSH",
  "createdAt": "2019-05-15T19:39:40.607337713Z",
  "contents": {
    "namespace": "foo",
    "repository": "bar",
    "tag": "latest",
    "digest": "sha256:b5bb9d8014a0f9b1d61e21e796d78dccdf1352f23cd32812f4850b878ae4944c",
    "imageName": "foo/bar:latest",
    "os": "linux",
    "architecture": "amd64",
    "author": "",
    "pushedAt": "2015-01-02T15:04:05Z"
  },
  "location": "/repositories/foo/bar/tags/latest"
}
```

Module 3: UCP Client Bundles

A client bundle is a group of certificates downloadable directly from the Docker Universal Control Plane (UCP).

Depending on the permission associated with the user, you can now execute docker swarm commands from your remote machine that take effect on the remote cluster.

For example:

You can create a new service in UCP from your laptop.
Login to remote container from your laptop without SSH (via API)

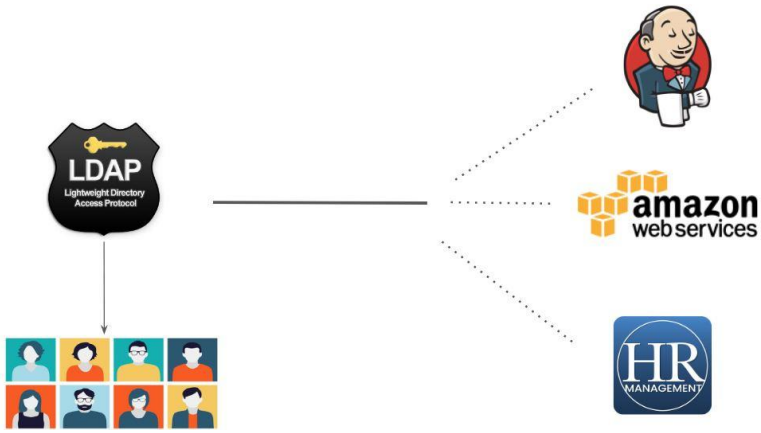
Module 4: Overview of LDAP

4.1 Need for LDAP

Let's assume there are 500 users within an organization. Your organization are using 3 services:-

- AWS (Infrastructure)
- Jenkins (CI / CD)
- HR Activator (Payroll)

You have been assigned a role to give users access to all 3 services.



There are various solutions available which can store users centrally:-

Microsoft Active Directory
RedHat Identity Management / freeIPA

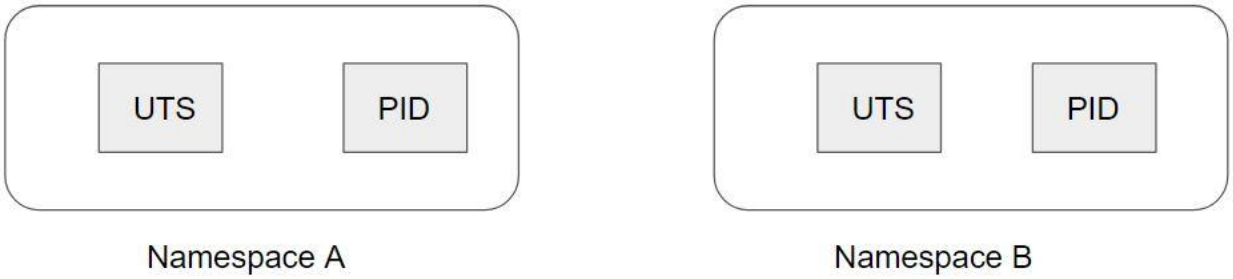
Microsoft Active Directory

freeIPA
identity | policy | audit

Module 5: Linux Namespaces

Docker uses a technology called namespaces to provide the isolated workspace called the container

These namespaces provide a layer of isolation. Each aspect of a container runs in a separate namespace and its access is limited to that namespace.

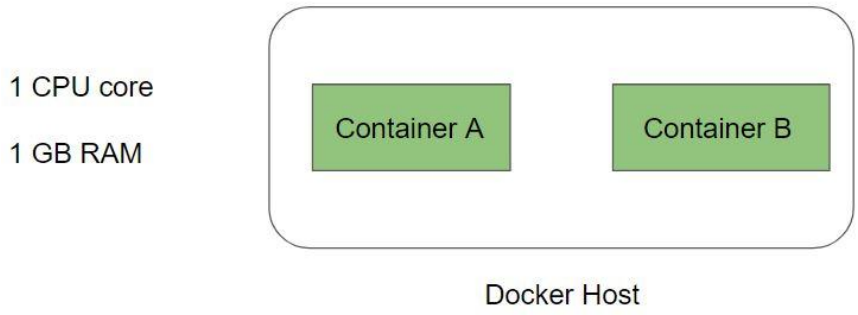


Currently, Linux provides six different types of namespaces as follows:

- Inter-Process Communication (IPC)
- Process (pid)
- Network (net)
- User Id (user)
- Mount (mnt)
- UTS

Module 6: Control Groups (cgroups)

Control Groups (cgroups) is a Linux kernel feature that limits, accounts for, and isolates the resource usage (CPU, memory, disk I/O, network, etc.) of a collection of processes.



There are various ways in which you can limit the CPU for containers, these include:

Approaches	Description
--cpus=<value>	If host has 2 CPUs and if you set --cpus=1, than container is guaranteed at most one CPU. You can even specify --cpus=0.5
--cpuset-cpus	Limit the specific CPUs or cores a container can use.A comma-separated list or hyphen-separated range of CPUs a container can use. 1st CPU is numbered 0 2nd CPU is numbered 1. Value of 0-3 means usage of first, second, third and fourth CPU. Value of 1,3 means second and fourth CPU.

Module 7: Reservation vs Limits

By default, a container has no resource constraints and can use as much of a given resource as the host's kernel scheduler allows.

It is important not to allow a running container to consume too much of the host machine's memory.

On Linux hosts, if the kernel detects that there is not enough memory to perform important system functions, it throws an Out Of Memory Exception and starts killing processes to free up memory.

Limit imposes an upper limit to the amount of memory that can potentially be used by a Docker container.

Reservations, on the other hand, are less rigid.

When the system is running low on memory and there is contention, reservation tries to bring the container's memory consumption at or below the reservation limit.

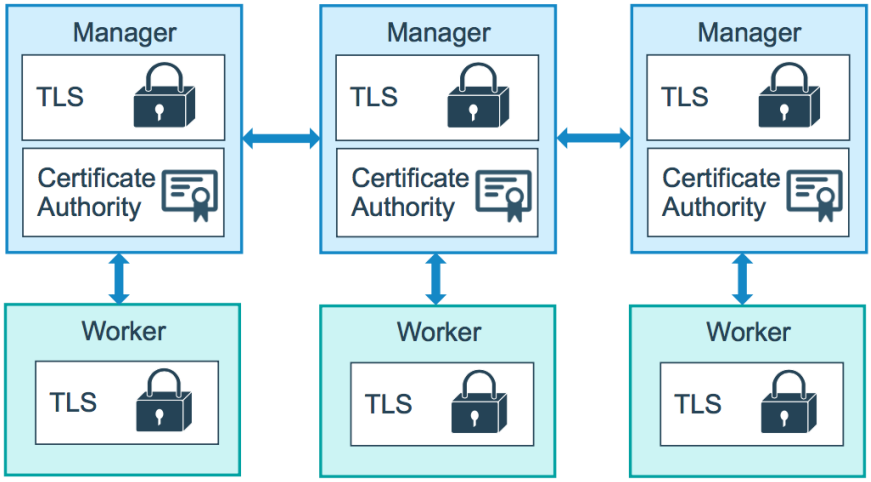
Module 8: Swarm MTLS

The nodes in a swarm use mutual Transport Layer Security (TLS) to authenticate, authorize, and encrypt the communications with other nodes in the swarm.

By default, the manager node generates a new root Certificate Authority (CA) along with a key pair, which is used to secure communications with other nodes that join the swarm.

You can specify your own externally-generated root CA, using the --external-ca flag of the docker swarm init command.

Each time a new node joins the swarm, the manager issues a certificate to the node.



8.1 Overview of Tokens

The manager node also generates two tokens to use when you join additional nodes to the swarm: one worker token and one manager token

Each token includes the digest of the root CA's certificate and a randomly generated secret.

When a node joins the swarm, the joining node uses the digest to validate the root CA certificate from the remote manager.

8.2 Certificate Details

By default, each node in the swarm renews its certificate every three months.

You can configure this interval by running the docker swarm update --cert-expiry <TIME PERIOD>

In the event that a cluster CA key or a manager node is compromised, you can rotate the swarm root CA so that none of the nodes trust certificates signed by the old root CA anymore.

8.3 Rotating the CA Certificate

Run docker swarm ca --rotate to generate a new CA certificate and key.

In the above command, the docker generated a cross-signed certificate signed by the previous old CA.

After every node in the swarm has a new TLS certificate signed by the new CA, Docker forgets about the old CA certificate and key material and tells all the nodes to trust the new CA certificate only.

Join tokens also changes accordingly.

Module 9: Managing Secrets in Docker Swarm

Docker secrets to centrally manage this data and securely transmit it to only those containers that need access to it.

Secrets are encrypted during transit and at rest in a Docker swarm.

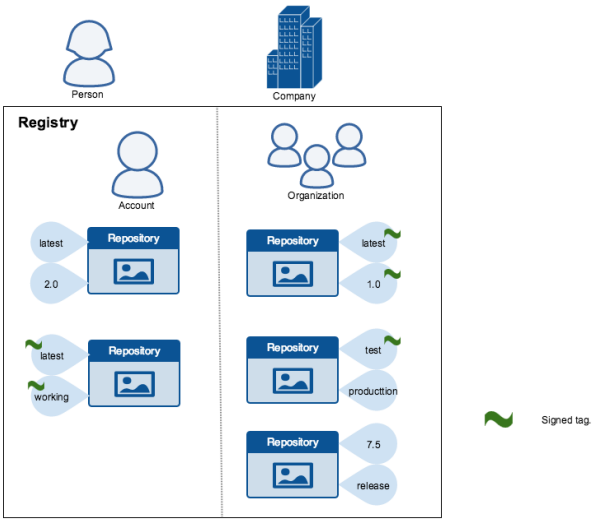
A given secret is only accessible to those services which have been granted explicit access to it, and only while those service tasks are running.

Module 10: Docker Content Trust

When we are downloading images from the network or from the internet, integrity becomes a true concern.

Content trust gives you the ability to verify both the integrity and the publisher of all the data received from a registry over any channel.

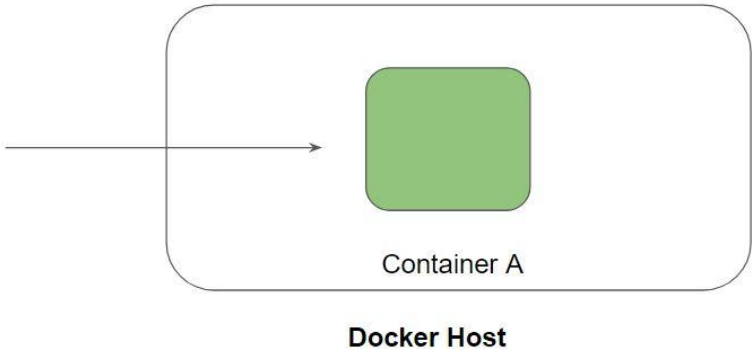
This can be achieved with the help of digital signatures.



Module 11: Overview of Linux Capabilities for Docker

There are several types of capabilities that Linux provides to have granular access at the application level.

Capability	Action
SETPCAP	Yes
CHOWN	Yes
SYS_MODULE	No
LINUX_IMMUTABLE	No



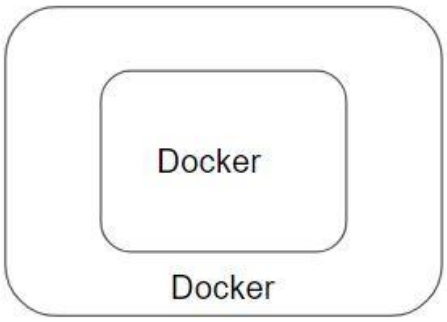
Module 12: Privileged Containers

12.1 Understanding the Challenge

By default, the docker container does not have many capabilities assigned to it.

Docker containers are also not allowed to access any devices.

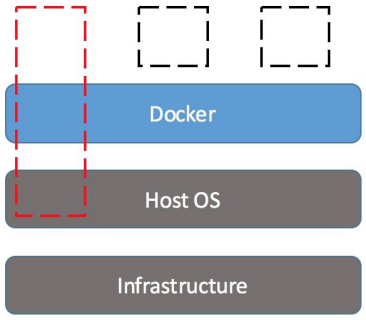
Hence, by default, docker container cannot perform various use-cases like run docker container inside a docker container



12.2 Privileged Containers

Privileged Container can access all the devices on the host as well as have a configuration in AppArmor or SELinux to allow the container nearly all the same access to the host as processes running outside containers on the host.

- Limits that you set for privileged containers will not be followed.
- Running a privileged flag gives all the capabilities to the container



Join Our Discord Community

We invite you to join our Discord community, where you can interact with our support team for any course-based technical queries and connect with other students who are doing the same course.

Joining URL:

http://kplabs.in/chat