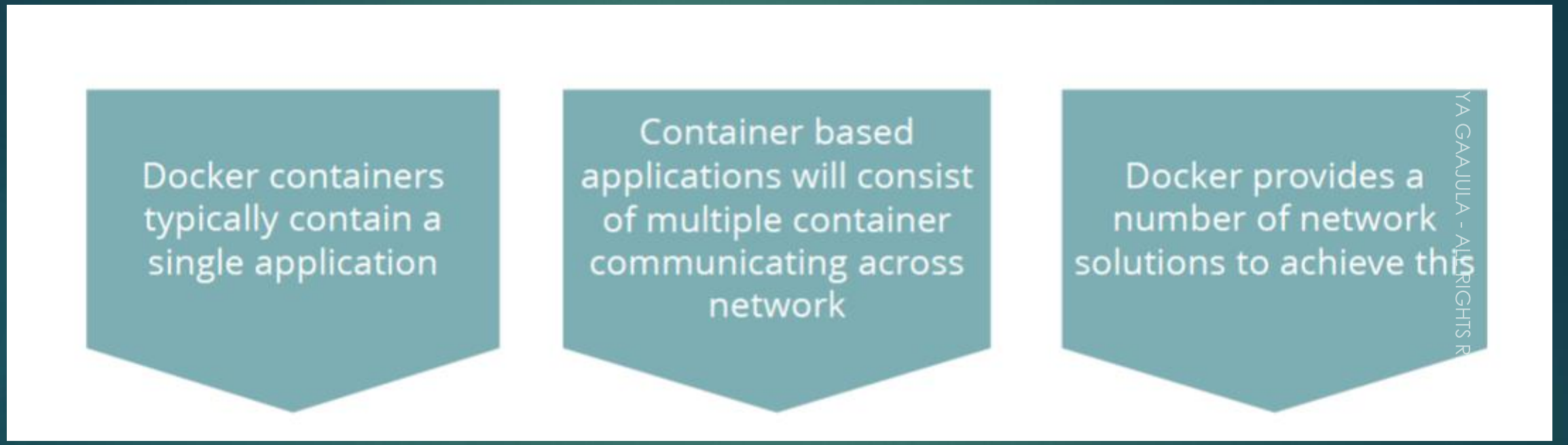
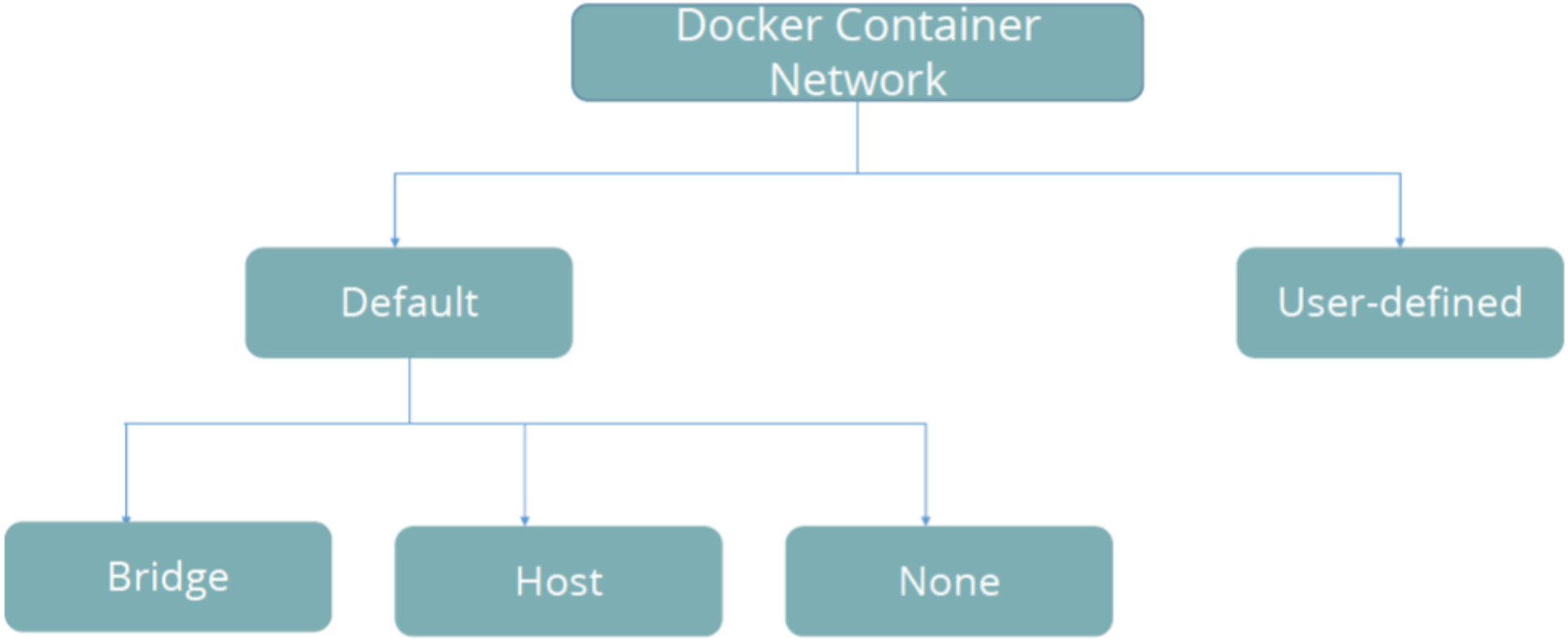
Docker Networking

Need for Docker Container Network



Types of Docker Container Network



Docker Container Networks—Default

- Docker creates three networks by default, which can't be removed
- The none network is local to the container – it has localhost
- The host network gives the container the same network as the host
- The bridge network is the default
- A docker0 or bridge0 virtual interface is created on the host

```
$ docker network ls
  NETWORK ID    NAME    DRIVER  SCOPE
   9d6a9ab487ba     bridge   bridge   local
   c7956146a031hosthostlocal
   115642b21a91       none     null local
```

Default—Bridge Network

The bridge network creates a subnet and a subnet mask.

```
$ docker network inspect bridge
[ {
"Name": "bridge",
"Id": "9d6a9ab487ba1d00715bfa60833a9cf5daa564d9a02918424ca3d38e26b2b5f8",
"Scope": "local",
"Driver": "bridge",
"EnableIPv6": false,
"IPAM": {
"Driver": "default",
"Options": null,
"Config": [
{
"Subnet": "172.17.0.0/16",
"Gateway": "172.17.0.1"
}
]},
```

Default—Bridge Network

- The bridge network assigns MAC and IP addresses to each container.

```
$ docker network inspect bridge
...
"Containers": {
"eb6dc24ff73fff0da60e98b02aa28e76f92f316aeed73f774ef7b3b0220b5b69": {
"Name": "centos",
"EndpointID":
"5b4437f5c54b0923f4558ecc01f9326fafa0fbb4d1f8564131d6dac9bd47e0de",
"MacAddress": "02:42:ac:11:00:02",
"IPv4Address": "172.17.0.2/16",
"IPv6Address": ""
}
},
```

Default—Bridge Network Hosts File

- The bridge network supplies a /etc/hosts file for each container.

```
127.0.0.1 localhost
::1 localhost ip6-
localhost ip6-loopback
fe00::0    ip6-localnet
ff00::0    ip6-mcastprefix
ff02::1    ip6-allnodes
ff02::2    ip6-allrouters
172.17.0.2eb6dc24ff73f
```

Default—Host Network

- A container attached to a host network has the same network as the host
- It has the same network configuration as the host
- It is not used much any more

Default—None Network

- A container attached to a none network has no network
- It has only a localhost interface
- It can't communicate with other networked containers

Docker Container Networks—User-defined

- User defined networks can be created
- Docker provides drivers including bridge
- Containers can only communicate with other containers on the same network
- Multiple networks can be created
- Containers can be connected to multiple networks
- Can communicate with any container on any connected network

Creating User-defined Networks

- New networks can be created
- The default driver is the bridge network
- A new subnet is created unless addresses are specified

```
$ docker network create isolated_bridge
b58db4ec8887a9187151c46850d69b58276a95d780e7465a24aaffb014f6ad8
```

```
$ docker network ls
    NETWORK ID   NAME          DRIVER    SCOPE
    9d6a9ab487ba      bridge          bridge      local
    c7956146a031      host  host  local
    db58db4ec888      isolated_bridge   bridge  local
    115642b21a91      none          null   local
```

Using Networks

- Networks can be specified only when a container is run
- A network can be added to an existing container
- A new network interface is added
- A network can be disconnected from a container
- A user-defined network can be removed

docker run -it --network isolated_bridge --name java centos-java docker network connect isolated_bridge centos docker network disconnect isolated_bridge centos

docker network rm isolated_bridge

Remote Access

- A container can be accessed remotely by port binding
- The listener port on the container is bound to a port on the host
- The container can then be accessed through the host IP
- User name and password are required to be set up in the container for remote ssh access

docker run -it --name ssh –p 2222:22 centos-ssh

ssh –p 2222 root@localhost

Firewall Rules

- The host will need to have firewall rules set up to limit the access
- Any container with exposed ports will need protection o Docker creates a DOCKER iptables chain on Linux hosts
- Firewall rules on the host can prevent remote access
- It can also prevent communication between docker-machine and Docker nodes
- It checks the iptables INPUT chain which may block Docker ports
- SELinux can always cause problems

THANK YOU