

Deploy the ASA Container in a Docker Environment

You can deploy the ASA container (ASAc) in an open source Docker environment running on any cloud platform.

- Overview, on page 1
- Guidelines and Limitations to Deploy ASA Container in Docker Environment, on page 1
- Licenses to Deploy ASA Container in Docker Environment, on page 2
- Components of Solution to Deploy ASA Container in Docker Environment, on page 2
- Sample Topology to Deploy ASA Container in Docker Environment, on page 3
- Prerequisites to Deploy ASA Container in Docker Environment, on page 4
- Deploy ASA Container in Docker Environment, on page 4
- High Availability (HA) for ASAc in Docker Environment, on page 6
- SR-IOV Configuration for ASAc in Docker Environment, on page 6
- Validate ASA Container Deployment in Docker Environment, on page 7
- Access ASA Container Deployment Logs in Docker Environment, on page 7
- Access ASA Container in Docker Environment, on page 7

Overview

A container is a software package that bundles up code and associated requirements such as system libraries, system tools, default settings, runtime, and so on, to ensure that the application runs successfully in a computing environment. From Secure Firewall ASA version 9.22, you can deploy the ASA container (ASAc) in an open-source Docker environment.

Guidelines and Limitations to Deploy ASA Container in Docker Environment

- The ASA container solution is validated on open-source Kubernetes and Docker environments only.
- Upgrade will be performed as a rolling upgrade using a new container image.
- Rebooting the ASA container is not supported.

- The following features are not validated:
  - Cluster
  - Transparent mode
  - Subinterfaces

Licenses to Deploy ASA Container in Docker Environment

Use one of the following licenses to enable deployment of ASA container on Docker:

Note ASA Virtual license entitlement can also be used for ASAc licensing.

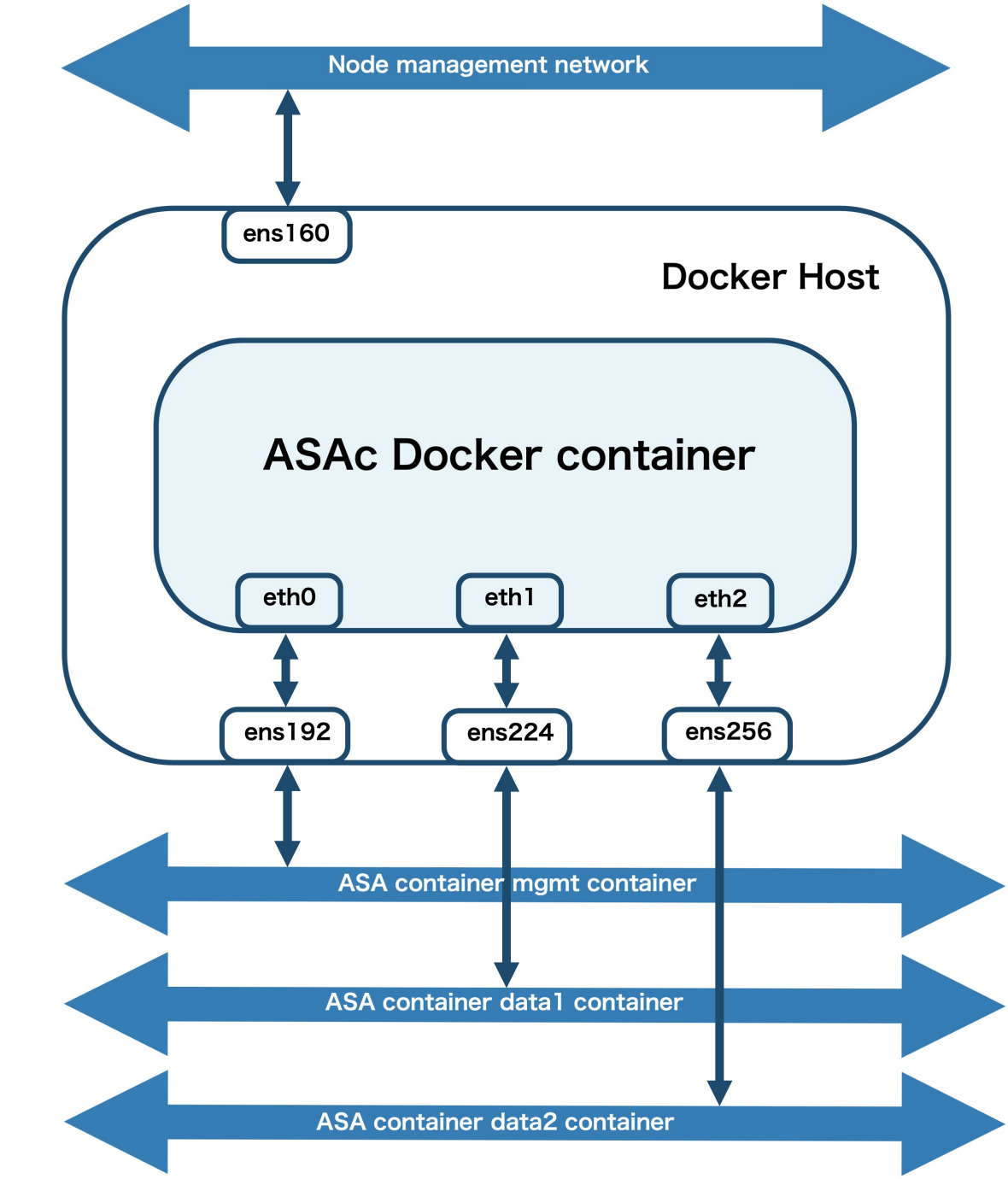
- ASAc5 - 1 vCPU, 2 GB RAM, and 100 Mbps rate limit
- ASAc10 - 1 vCPU, 2 GB RAM, and 1 Gbps rate limit
- ASAc30 - 4 vCPU, 4 GB RAM, and 2 Gbps rate limit
- ASAc50 - 8 vCPU, 16 GB RAM, and 10 Gbps rate limit
- ASAc100 - 16 vCPU, 32 GB RAM, and 20 Gbps rate limit

Note The vfio-pci driver is required to achieve throughput greater than 1 Gbps.

Components of Solution to Deploy ASA Container in Docker Environment

- Operating system
  - Ubuntu 20.04.6 LTS on docker host
- Macvlan network for configuration validation

Sample Topology to Deploy ASA Container in Docker Environment



In this sample topology, the ASA docker container has three virtual network interfaces –eth0, eth1, and eth2, that are connected to the following interfaces – ens192, ens224, and ens256. These interfaces are mapped to the ASAc mgmt, data1, and data2 networks. The interface ens160 is the node management interface.

Prerequisites to Deploy ASA Container in Docker Environment

- Ensure that Ubuntu 20.04.6 LTS is installed on the docket host.
- Allocate three virtual interfaces on the docker host for ASA container operations.
- Set up the docker host's management interface to be used for ssh access to the docker host.
- Enable Hugepages on the docker host.
- Set up Docker version 24.0.5 with macvlan network for configuration validation.

For more information on general Docker operations mentioned in these prerequisites, see Docker documentation.

Deploy ASA Container in Docker Environment

Perform the procedure given below to deploy ASA container (ASAc) in Docker environment.

Procedure

Step 1 Set up the requirements mentioned in the Prerequisites.

Step 2 Run the **ifconfig** command to verify the network interface configuration. In this example, ens160 is the node's management interface. The nodes ens192, ens224, and ens256, are mapped to the ASAc interfaces.

Note
The outputs given below are sample outputs only.

```
$ ifconfig
ens160: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::250:56ff:fe9d:6125  prefixlen 64  scopeid 0x20<link>
        ether 00:50:56:9d:61:25  txqueuelen 1000  (Ethernet)
        RX packets 317297807  bytes 447854277676 (447.8 GB)
        RX errors 0  dropped 2100  overruns 0  frame 0
        TX packets 5517880  bytes 378756756 (378.7 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens192: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.3.4  netmask 255.255.255.224  broadcast 10.10.3.31
        inet6 fe80::250:56ff:fe9d:fa1c  prefixlen 64  scopeid 0x20<link>
        ether 00:50:56:9d:fa:1c  txqueuelen 1000  (Ethernet)
        RX packets 70324790  bytes 30189381762 (30.1 GB)
        RX errors 0  dropped 2437  overruns 0  frame 0
        TX packets 60676399  bytes 16108954006 (16.1 GB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens224: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::250:56ff:fe9d:2cbe  prefixlen 64  scopeid 0x20<link>
        ether 00:50:56:9d:2c:be  txqueuelen 1000  (Ethernet)
        RX packets 489699  bytes 41669463 (41.6 MB)
```

```
        RX errors 0  dropped 1969  overruns 0  frame 0
        TX packets 285031  bytes 23421780 (23.4 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens256: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::250:56ff:fe9d:92ba  prefixlen 64  scopeid 0x20<link>
        ether 00:50:56:9d:92:ba  txqueuelen 1000  (Ethernet)
        RX packets 7023252  bytes 8223100366 (8.2 GB)
        RX errors 0  dropped 2145  overruns 0  frame 0
        TX packets 31481074  bytes 44913129384 (44.9 GB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Step 3 Run the **cat** command given below to verify hugepage configuration.

```
$ cat /proc/meminfo | grep -E 'HugePages_Total|HugePages_Free'
HugePages_Total:    1024
HugePages_Free:      1024
```

Step 4 Download the ASA docker tar bundle that includes the ASA container image from software.cisco.com.

Step 5 Load the docker tar bundle on the host.

```
$ docker load < asac9-22-1-115.tar
$ docker images
   REPOSITORY                                    TAG         IMAGE ID
   dockerhub.cisco.com/asac-dev-docker/asac    9.22.1.115    55f5dbc5f3aa
```

Step 6 Download the templates and other files from the **docker** folder in the ASAc GitHub repository.

Step 7 Run the **docker network create** command to create docker networks. The ASAc needs one management interface and two date interfaces for inside and outside networks. When docker starts, the docker networks are attached to the docker in alphabetical order. We recommend that you name the management interface in such a way that it is the first interface that is attached to the docker.

```
$ docker network create -d macvlan -o parent=ens192 asac_nw1
$ docker network create -d macvlan -o parent=ens224 asac_nw2
$ docker network create -d macvlan -o parent=ens256 asac_nw3
```

Step 8 Run the **docker network ls** command to verify that the networks have been created successfully.

```
$ docker network ls
NETWORK ID    NAME      DRIVER   SCOPE
06f5320016f8  asac_nw1  macvlan  local
258954fa5611  asac_nw2  macvlan  local
3a3cd7254087  asac_nw3  macvlan  local
```

Step 9 Verify the default parameter values present in the **day0-config** file. You can also update these values as per your requirement.

Step 10 Open the **start_docker_asac.sh** script to update configuration values for CPU, memory, container-name, and image repo name, as per your requirement.

Note
Default configuration values are provided for the parameters in the start_docker_asac.sh script. Modify them only if required.

Step 11 Run the command given below to start ASAc in the docker environment.

```
$ ./<script-name> <asac-image-path-and-version> <asac-mgmt-nw> <asac-data1-nw> <asac-data2-nw>
```

```
$ ./start_docker_asac.sh dockerhub.cisco.com/asac-dev-docker/asac:9.22.1.115 asac_nw1 asac_nw2
asac_nw3
   Docker networks are provided..
   Starting ASA Build Container...
   docker create -it --privileged --cap-add=NET_RAW --network asac_nw1 --name asac -e ASAC_CPUS=1
-e ASAC_MEMORY=2048M -v /dev:/dev -v /home/ubuntu/standalone-asac/docker/day0-config:/asacday0-
config/day0-config:Z -v /home/ubuntu/standalone-asac/docker/interface-config:/mnt/disk0/
interface-config/interface-config:Z -e CORE_SIZE_LIMIT=200MB -e COREDUMP_PATH=/mnt/coredump_repo/
-e ASA_DOCKER=1 -e ASAC_STANDALONE_MODE=1 -e ASAC_ROOT_PRIVILEGE=1 --entrypoint /asa/bin/
lina_launcher.sh dockerhub.cisco.com/asac-dev-docker/asac:9.22.1.115

    Mount Points:
    ------------------------------------------------------------------------------------------
    Host                                                    Container
    ----                                                    ---------
    /dev                                                    /dev
    /home/ubuntu/standalone-asac/docker/day0-config        /asac-day0-config/day0-config
    /home/ubuntu/standalone-asac/docker/interface-config
/mnt/disk0/interface-config/interface-config
------------------------------------------------------------------------------------------
docker network connect asac_nw2 asac
docker network connect asac_nw3 asac
docker start asac
```

High Availability (HA) for ASAc in Docker Environment

The steps for ASAc High Availability formation in a Docker environment are the same as deploying two ASAc instances as primary and secondary on two different Docker hosts and configuring High Availability between them.

SR-IOV Configuration for ASAc in Docker Environment

When deploying ASAc in SR-IOV mode:

- The PCI address must be configured as `iface_id`.
- The `vfio-pci` driver must be configured as the `uio_driver` in the `interface-config` file.

Sample `interface-config` file is shown below:

```
[interface0]
  iface_id = 19:02.0;
  uio_driver = vfio-pci;
[interface1]
  iface_id = 19:0a.0;
  uio_driver = vfio-pci;
[interface2]
  iface_id = 19:0b.0;
  uio_driver = vfio-pci;
```

Validate ASA Container Deployment in Docker Environment

Validate successful ASA container deployment by checking the list of containers running on the docker host.

```
$ docker ps -a
CONTAINER ID IMAGE                                                    COMMAND
 CREATED       STATUS        PORTS  NAMES
6e5bff4dbcaf dockerhub.cisco.com/asac-dev-docker/asac:9.22.1.115   "/asa/bin/lina_launc…"
3 minutes ago Up 3 minutes         asac
```

Access ASA Container Deployment Logs in Docker Environment

Run the **docker logs asac** command to check the docker logs for troubleshooting any issues that may occur.

```
$ docker logs asac
Skip NVMe Device for ASAc mode
cdrom device /dev/sr0 found
mount: /mnt/cdrom: WARNING: source write-protected, mounted read-only.
Error: Encrypted file system support not in Linux kernel.
nr_overcommit_hugepages set to 128 for virtual platform
info: ASAc SSHd Directory Created
No interface-config file found at /interface-config, using default shared
file: /mnt/disk0/interface-config/interface-config
No day0-config file found at /day0-config, using default shared file:
/asac-day0-config/day0-config
info: ASAc Day 0 configuration installed.
info: ASAc Primay/backup Key installed
info: Running in vmware virtual environment.
....
INFO: Network Service reload not performed.
INFO: Power-On Self-Test in process.
.....................................
INFO: Power-On Self-Test complete.
INFO: Starting SW-DRBG health test...
INFO: SW-DRBG health test passed.
Creating trustpoint "_SmartCallHome_ServerCA" and installing certificate...
Trustpoint CA certificate accepted.
Creating trustpoint "_SmartCallHome_ServerCA2" and installing
certificate...
Trustpoint CA certificate accepted.
User enable_1 logged in to ciscoasa
Logins over the last 1 days: 1.
Failed logins since the last login: 0.
Type help or '?' for a list of available commands.
ciscoasa>
```

Access ASA Container in Docker Environment

Run the **docker attach asac** command to access the CLI of the ASA container (ASAc) and obtain required outputs. In this example, we access the CLI of the ASAc and run the **show version** command.



Note You can also use ASDM to access ASAc in a Docker environment.

```
$ kubectl attach -it asac-585744f74-fwsh5
If you don't see a command prompt, try pressing enter.
ciscoasa> show version
Cisco Adaptive Security Appliance Software Version 99.25(0)118
SSP Operating System Version 82.19(0.143i)
Compiled on Sun 08-Feb-26 16:53 GMT by builders
System image file is "Unknown, monitor mode tftp booted image"
Config file at boot was "startup-config"
ciscoasa up 4 mins 38 secs
Start-up time 39 secs
Hardware:   ASAc, 4096 MB RAM, CPU Xeon E5 series 2100 MHz, 1 CPU (2 cores)
BIOS Flash Firmware Hub @ 0x1, 0KB

 0: Ext: Management0/0       : address is 9a9d.12b4.0e98, irq 0
 1: Ext: GigabitEthernet0/0  : address is 7a6f.16e5.a500, irq 0
 2: Ext: GigabitEthernet0/1  : address is c6c3.4497.6421, irq 0
 3: Int: Internal-Data0/0    : address is 0000.0100.0001, irq 0

License mode: Smart Licensing
ASAv Platform License State: Unlicensed
No active entitlement: no feature tier and no throughput level configured
Firewall throughput limited to 100 Kbps
```