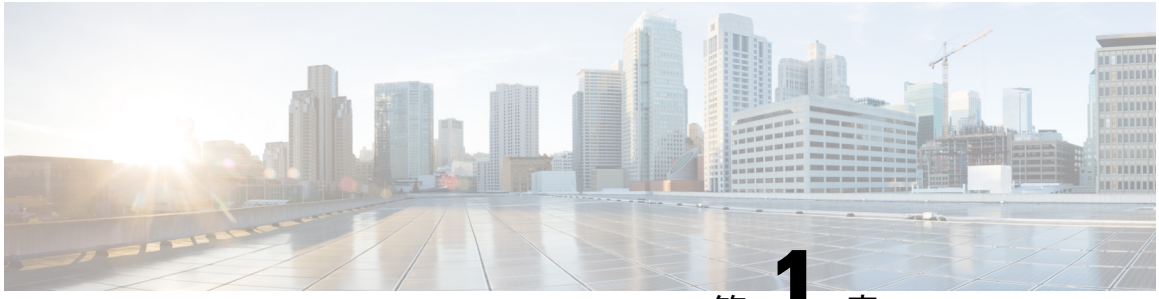
Cisco Secure Firewall ASA コンテナ 9.22 スタートアップガイド

最終更新：2025 年 12 月 24 日

シスコシステムズ合同会社
〒107-6227 東京都港区赤坂9-7-1　ミッドタウン・タワー
http://www.cisco.com/jp
お問い合わせ先：シスコ コンタクトセンター
0120-092-255 （フリーコール、携帯・PHS含む）
電話受付時間：平日 10:00～12:00、13:00～17:00
http://www.cisco.com/jp/go/contactcenter/



第 1 章

Docker 環境での ASA コンテナの展開

任意のクラウドプラットフォームで実行されるオープンソースの Docker 環境で ASA コンテナ（ASAc）を展開できます。

概要

コンテナは、コンピューティング環境でアプリケーションが正常に実行されるようにするためのコードと関連する要件（システムライブラリ、システムツール、デフォルト設定、ランタイムなど）をバンドルしたソフトウェアパッケージです。Cisco Secure Firewall ASA バージョン 9.22 以降では、オープンソースの Docker 環境で ASA コンテナ（ASAc）を展開できます。

Docker環境でASAコンテナを展開するためのガイドラインと制限事項

- ASA コンテナソリューションは、オープンソースの Kubernetes および Docker 環境でのみ検証されます。
- EKS、GKE、AKS、OpenShift などの他の Kubernetes フレームワークは、まだ検証されていません。

- アップグレードは、新しいコンテナイメージを使用してローリングアップグレードとして実行されます。
- ASA コンテナの再起動はサポートされていません。
- 次の機能は検証されていません。
  - クラスタ
  - トランスペアレント モード
  - サブインターフェイス

Docker環境でASAコンテナを展開するためのライセンス

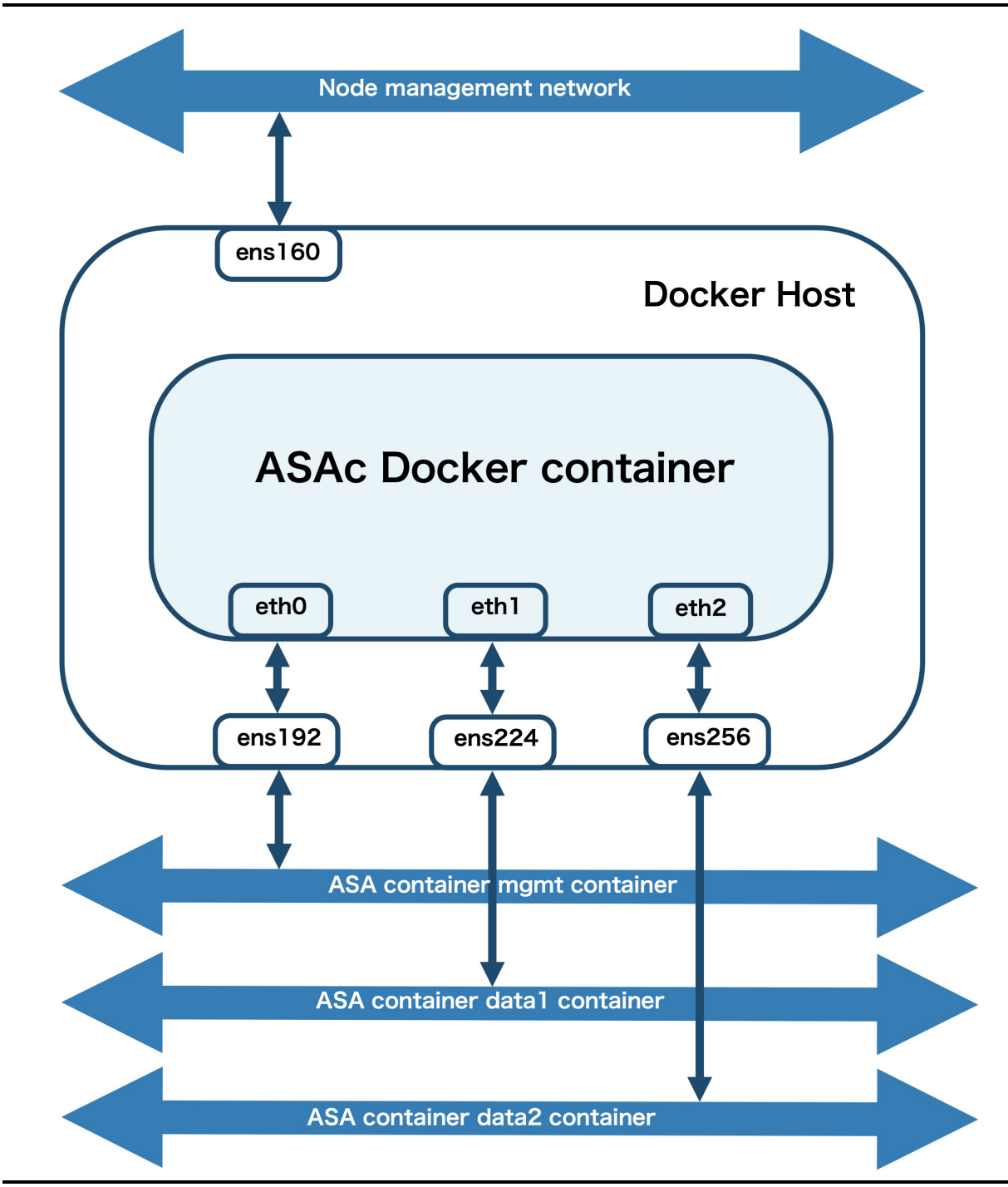
次のいずれかのライセンスを使用すると、Docker で ASA コンテナを展開できます。

- ASAc5：1 vCPU、2 GB RAM、および 100 Mbps のレート制限
- ASAc10：1 vCPU、2 GB RAM、および 1 Gbps のレート制限

Docker 環境で ASA コンテナを展開するためのソリューションのコンポーネント

- オペレーティング システム
  - Docker ホスト上の Ubuntu 20.04.6 LTS
- 設定検証用の Macvlan ネットワーク

Docker環境でASAコンテナを展開するためのサンプルトポロジ



このサンプルトポロジでは、ASA Docker コンテナに 3 つの仮想ネットワーク インターフェイス（eth0、eth1、eth2）があり、インターフェイス ens192、ens224、および ens256 に接続されています。これらのインターフェイスは、ASAc mgmt、data1、および data2 ネットワークにマッピングされます。インターフェイス ens160 は、ノード管理インターフェイスです。

Docker 環境で ASA コンテナを展開するための前提条件

- Ubuntu 20.04.6 LTS が Docker ホストにインストールされていることを確認します。
- ASA コンテナの操作のために、Docker ホストに 3 つの仮想インターフェイスを割り当てます。
- Docker ホストへの SSH アクセスに使用する Docker ホストの管理インターフェイスをセットアップします。
- Docker ホストで Hugepages を有効にします。
- 設定検証用の macvlan ネットワークを使用して Docker バージョン 24.0.5 をセットアップします。

これらの前提条件に記載されている一般的な Docker 操作の詳細については、Docker のドキュメントを参照してください。

Docker 環境での ASA コンテナの展開

Docker 環境で ASA コンテナ（ASAc）を展開するには、次の手順を実行します。

手順

ステップ 1 前提条件に記載されている要件をセットアップします。

ステップ 2 **route -n** コマンドを実行し、ネットワーク インターフェイス構成を確認します。この例では、ens160 はノードの管理インターフェイスです。ノード ens192、ens224、および ens256 は、ASAc インターフェイスにマッピングされます。

（注）
以下に示す出力は、サンプル出力のみです。

Docker 環境での ASA コンテナの展開

```
ubuntu@k8s-worker:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.10.4.1       0.0.0.0         UG    100    0        0 ens160
10.10.4.0       0.0.0.0         255.255.255.224 U     0      0        0 ens160
10.10.4.1       0.0.0.0         255.255.255.255 UH    100    0        0 ens160
10.10.4.32      0.0.0.0         255.255.255.224 U     0      0        0 ens192
10.10.4.64      0.0.0.0         255.255.255.224 U     0      0        0 ens224
10.10.4.96      0.0.0.0         255.255.255.224 U     0      0        0 ens256
10.244.235.192  10.244.235.192  255.255.255.192 UG    0      0        0 vxlan.calico
10.244.254.128  0.0.0.0         255.255.255.192 U     0      0        0 *
172.17.0.0      0.0.0.0         255.255.0.0     U     0      0        0 docker0
```

ステップ 3 次に示す **cat** コマンドを実行し、hugepage 設定を確認します。

```
ubuntu@k8s-worker:~$ cat /proc/meminfo | grep -E 'HugePages_Total|HugePages_Free'
HugePages_Total:    2048
HugePages_Free:     2048
```

ステップ 4 ASA コンテナイメージを含む ASA Docker tar バンドルを、software.cisco.com からダウンロードします。

ステップ 5 ホストで Docker tar バンドルをロードします。

```
$ docker load < asac9-22-1-1.tar
$ docker images
   REPOSITORY                                  TAG         IMAGE ID
   dockerhub.cisco.com/asac-dev-docker/asac    9.22.1.1    55f5dbc5f3aa
```

ステップ 6 ASAc GitHub リポジトリの **docker** フォルダからテンプレートなどをダウンロードします。

ステップ 7 **docker network create** コマンドを実行して、Docker ネットワークを作成します。ASAc には、内部と外部のネットワーク用に 1 つの管理インターフェイスと 2 つのデータインターフェイスが必要です。Docker を起動すると、Docker ネットワークがアルファベット順に Docker に接続されます。管理インターフェイスには、Docker に接続される最初のインターフェイスになるように名前を付けることをお勧めします。

```
$ 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
```

ステップ 8 **docker network ls** コマンドを実行して、ネットワークが正常に作成されたことを確認します。

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

ステップ 9 **day0-config** ファイルに存在するデフォルトのパラメータ値を確認します。必要に応じて、これらの値を更新することもできます。

ステップ 10 必要に応じて、**start_docker_asac.sh** スクリプトを開き、CPU、メモリ、コンテナ名、およびイメージリポ名の設定値を更新します。

（注）

start_docker_asac.sh スクリプトのパラメータには、デフォルトの設定値が指定されています。必要な場合にのみ変更してください。

ステップ 11 次のコマンドを実行して、Docker 環境で ASAc を起動します。

```
$ ./<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:9.22.1.1 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:9.22.1.1

    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
```

Docker 環境での ASA コンテナ展開の検証

Docker ホストで実行されているコンテナのリストを確認することで、ASA コンテナが正常に展開されているかどうかを検証します。

```
$ docker ps -a
CONTAINER ID IMAGE                                                      COMMAND
    CREATED       STATUS        PORTS  NAMES
6e5bff4dbcaf dockerhub.cisco.com/asac-dev-docker/asac:9.22.x.x   "/asa/bin/lina_launc…"
 3 minutes ago Up 3 minutes         asac
```

Docker 環境での ASA コンテナ展開ログへのアクセス

何らかの問題が発生した場合は、トラブルシュートのために **docker logs asac** コマンドを実行して Docker ログを確認してください。

```
$ docker logs asac
Skip NVMe Device for ASAc mode
cdrom device /dev/sr0 found
mount: /mnt/cdrom: WARNING: source write-protected, mounted read-only.
```

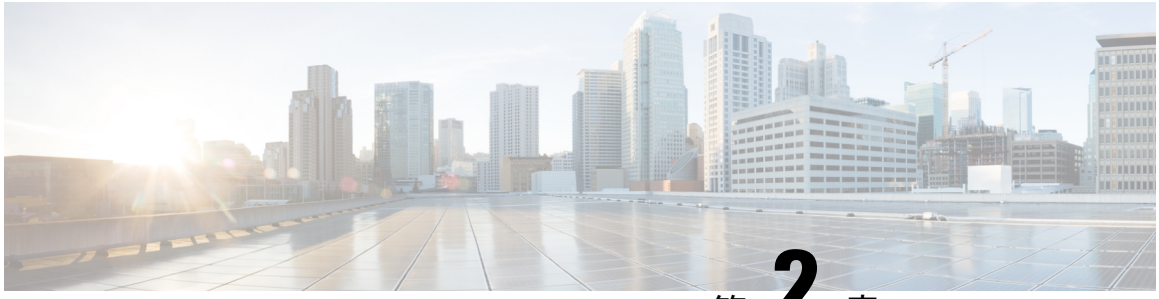
```
Error: Encrypted file system support not in Linux kernel.
nr_overcommit_hugepages set to 128 for virtual platform
info: ASAc SSHd Directory Created
No interface-config file found at /interface-config, using default shared
file: /mnt/disk0/interface-config/interface-config
No day0-config file found 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>
```

Docker 環境での ASA コンテナへのアクセス

docker attach asac コマンドを実行して ASA コンテナ（ASAc）の CLI にアクセスし、必要な出力を取得します。この例では、ASAc の CLI にアクセスして **show version** コマンドを実行しています。

（注） ASDM を使用して Docker 環境で ASAc にアクセスすることもできます。

```
ciscoasa> enable
Password: *********
ciscoasa# sh version
Cisco Adaptive Security Appliance Software Version 9.22
SSP Operating System Version 82.16(0.216i)
Device Manager Version 7.22
Compiled on Tue 28-Nov-23 14:37 GMT by builders
System image file is "Unknown, monitor mode tftp booted image"
Config file at boot was "startup-config"
ciscoasa up 9 mins 50 secs
Start-up time 36 secs
Hardware: ASAc, 2048 MB RAM, CPU Xeon E5 series 2100 MHz, 1 CPU (1
core)
BIOS Flash Firmware Hub @ 0x1, 0KB
0: Ext: Management0/0 : address is 0242.ac12.0002, irq 0
1: Ext: GigabitEthernet0/0 : address is 0242.ac13.0002, irq 0
2: Ext: GigabitEthernet0/1 : address is 0242.ac14.0002, irq 0
3: Int: Internal-Data0/0 : address is 0000.0100.0001, irq 0
```



第 2 章

Kubernetes 環境での ASA コンテナの展開

任意のクラウドプラットフォームで実行されるオープンソースの Kubernetes 環境で ASA コンテナ（ASAc）を展開できます。

概要

コンテナは、コンピューティング環境でアプリケーションが正常に実行されるようにするためのコードと関連する要件（システムライブラリ、システムツール、デフォルト設定、ランタイムなど）をバンドルしたソフトウェアパッケージです。Cisco Secure Firewall ASA バージョン 9.22 以降では、オープンソースの Kubernetes 環境で ASAc を展開できます。このソリューションでは、ASAc はコンテナ ネットワーク インターフェイス（CNI）と統合され、Infrastructure-as-Code（IaC）ソリューションとして展開されます。CNI との統合により、ネットワーク インフラストラクチャの展開の柔軟性が向上します。

Kubernetes 環境で ASA コンテナを展開するためのガイドラインと制限事項

- ASA コンテナソリューションは、オープンソースの Kubernetes および Docker 環境でのみ検証されます。
- EKS、GKE、AKS、OpenShift などの他の Kubernetes フレームワークは、まだ検証されていません。
- アップグレードは、新しいコンテナイメージを使用してローリングアップグレードとして実行されます。
- ASA コンテナの再起動はサポートされていません。
- 次の機能は検証されていません。
  - クラスタ
  - トランスペアレント モード
  - サブインターフェイス

Kubernetes 環境で ASA コンテナを展開するためのライセンス

次のいずれかのライセンスを使用すると、Kubernetes で ASA コンテナを展開できます。

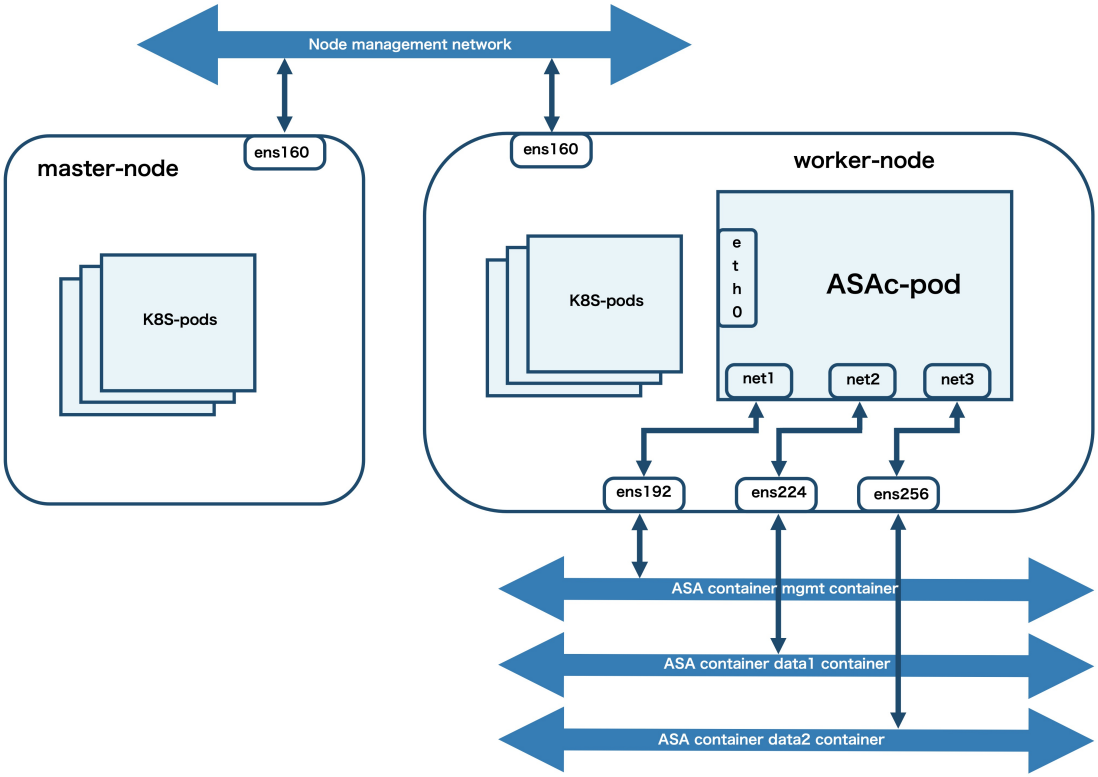
- ASAc5：1 vCPU、2 GB RAM、および 100 Mbps のレート制限
- ASAc10：1 vCPU、2 GB RAM、および 1 Gbps のレート制限

Kubernetes 環境で ASA コンテナを展開するためのソリューションのコンポーネント

- オペレーティング システム
  - Ubuntu 20.04.6
  - Kubernetes バージョン v1.26
  - Helm バージョン v3.13.1
- Kubernetes クラスターノード：マスターノードおよびワーカーノード

- Kubernetes CNI
  - POD 管理 CNI：Calico
  - ASAc ネットワーク CNI：Multus macvlan
- Helm チャートは yaml ファイルとして提供され、Infrastructure-as-Code（IaC）のセットアップに使用されます。

Kubernetes 環境で ASA コンテナを展開するためのサンプルトポロジ



このサンプルトポロジでは、ASA コンテナ（ASAc）ポッドに 3 つの仮想ネットワーク インターフェイス（net1、net2、net3）があり、ワーカー ノード インターフェイス ens192、ens224、および ens256 に接続されています。これらのワーカー ノード インターフェイスは、ASAc mgmt、data1、および data2 ネットワークにマッピングされます。インターフェイス ens160 は、ノード管理インターフェイスです。インターフェイス eth0 は、Calico CNI から派生しています。インターフェイス net1、net2、および net3 は、Multus macvlan CNI から派生しています。

Kubernetes 環境で ASA コンテナを展開するための前提条件

- マスターノードとワーカーノードの両方に Ubuntu 20.04.6 LTS がインストールされていることを確認します。
- ASA コンテナ（ASAc）の操作のためにワーカーノードに 3 つの仮想インターフェイスを割り当てます。
- ワーカーノードへの SSH アクセスに使用するワーカーノードの管理インターフェイスをセットアップします。
- ワーカーノードで Hugepages を有効にします。
- POD 管理として使用する Calico CNI をセットアップします。
- ASAc インターフェイスの管理に使用する Multus with macvlan CNI をセットアップします。

これらの前提条件に記載されている一般的な Kubernetes 操作の詳細については、Kubernetes のドキュメントを参照してください。

Kubernetes 環境での ASA コンテナの展開

Kubernetes 環境で ASA コンテナ（ASAc）を展開するには、次の手順を実行します。

手順

ステップ 1 前提条件に記載されている要件をセットアップします。

ステップ 2 **kubectl get nodes**、**kubectl get pods**、および **kubectl get all** コマンドを実行し、それぞれ、すべてのノード、ポッド、およびすべてのリソースのステータスを表示します。Kubernetes のポッドとノードが準備完了状態であることを確認します。

（注）
以下に示す出力は、サンプル出力のみです。

```
ubuntu@k8s-master:~$ kubectl get nodes -o wide
NAME         STATUS   ROLES           AGE   VERSION   INTERNAL-IP   EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION      CONTAINER-RUNTIME
k8s-master   Ready    control-plane   94d   v1.26.9   10.10.4.17    <none>        Ubuntu 20.04.6 LTS   5.4.0-164-generic   containerd://1.7.2
k8s-worker   Ready    <none>          94d   v1.26.9   10.10.4.14    <none>        Ubuntu 20.04.6 LTS   5.4.0-169-generic   containerd://1.7.2
```

Kubernetes 環境での ASA コンテナの展開

```
ubuntu@k8s-master:~$ kubectl get pods -A -o wide
NAMESPACE         NAME                                       READY   STATUS    RESTARTS        AGE   IP              NODE         NOMINATED NODE   READINESS
 GATES
calico-apiserver   calico-apiserver-648b88b9c5-6mlsx          1/1     Running   0               94d   10.244.235.198   k8s-master   <none>           <none>
calico-apiserver   calico-apiserver-648b88b9c5-zd5xz          1/1     Running   0               94d   10.244.235.197   k8s-master   <none>           <none>
calico-system      calico-kube-controllers-6cd4d8dd54-8wtzf   1/1     Running   0               94d   10.244.235.195   k8s-master   <none>           <none>
calico-system      calico-node-2c9bl                          1/1     Running   0               94d   10.10.4.17       k8s-master   <none>           <none>
calico-system      calico-node-fvqpk                          1/1     Running   17 (8m18s ago)  94d   10.10.4.14       k8s-worker   <none>           <none>
calico-system      calico-typha-656cc4f7d4-xwp6m              1/1     Running   0               94d   10.10.4.17       k8s-master   <none>           <none>
calico-system      csi-node-driver-8cdc8                      2/2     Running   34 (8m18s ago)  94d   10.244.254.159   k8s-worker   <none>           <none>
calico-system      csi-node-driver-w6hk9                      2/2     Running   0               94d   10.244.235.193   k8s-master   <none>           <none>
kube-system        coredns-787d4945fb-dxpmp                   1/1     Running   0               94d   10.244.235.196   k8s-master   <none>           <none>
kube-system        coredns-787d4945fb-vnxws                   1/1     Running   0               94d   10.244.235.194   k8s-master   <none>           <none>
kube-system        etcd-k8s-master                            1/1     Running   0               94d   10.10.4.17       k8s-master   <none>           <none>
kube-system        kube-apiserver-k8s-master                  1/1     Running   0               94d   10.10.4.17       k8s-master   <none>           <none>
kube-system        kube-controller-manager-k8s-master         1/1     Running   0               94d   10.10.4.17       k8s-master   <none>           <none>
kube-system        kube-multus-ds-tbjhf                       1/1     Running   0               94d   10.10.4.17       k8s-master   <none>           <none>
kube-system        kube-multus-ds-v5kxm                       1/1     Running   18 (8m18s ago)  94d   10.10.4.14       k8s-worker   <none>           <none>
kube-system        kube-proxy-9qvdc                           1/1     Running   0               94d   10.10.4.17       k8s-master   <none>           <none>
kube-system        kube-proxy-wcj8t                           1/1     Running   17 (8m18s ago)  94d   10.10.4.14       k8s-worker   <none>           <none>
kube-system        kube-scheduler-k8s-master                  1/1     Running   0               94d   10.10.4.17       k8s-master   <none>           <none>
tigera-operator    tigera-operator-776b7d494d-j66m4           1/1     Running   0               94d   10.10.4.17       k8s-master   <none>           <none>
```

```
ubuntu@k8s-master:~$ kubectl get all -A
NAMESPACE          NAME                                                READY   STATUS    RESTARTS       AGE
calico-apiserver   pod/calico-apiserver-648b88b9c5-6mlsx               1/1     Running   0              94d
calico-apiserver   pod/calico-apiserver-648b88b9c5-zd5xz               1/1     Running   0              94d
calico-system      pod/calico-kube-controllers-6cd4d8dd54-8wtzf        1/1     Running   0              94d
calico-system      pod/calico-node-2c9bl                               1/1     Running   0              94d
calico-system      pod/calico-node-fvqpk                               1/1     Running   17 (11m ago)   94d
calico-system      pod/calico-typha-656cc4f7d4-xwp6m                   1/1     Running   0              94d
calico-system      pod/csi-node-driver-8cdc8                           2/2     Running   34 (11m ago)   94d
calico-system      pod/csi-node-driver-w6hk9                           2/2     Running   0              94d
kube-system        pod/coredns-787d4945fb-dxpmp                        1/1     Running   0              94d
kube-system        pod/coredns-787d4945fb-vnxws                        1/1     Running   0              94d
kube-system        pod/etcd-k8s-master                                 1/1     Running   0              94d
kube-system        pod/kube-apiserver-k8s-master                       1/1     Running   0              94d
kube-system        pod/kube-controller-manager-k8s-master              1/1     Running   0              94d
kube-system        pod/kube-multus-ds-tbjhf                            1/1     Running   0              94d
kube-system        pod/kube-multus-ds-v5kxm                            1/1     Running   18 (11m ago)   94d
kube-system        pod/kube-proxy-9qvdc                                1/1     Running   0              94d
kube-system        pod/kube-proxy-wcj8t                                1/1     Running   17 (11m ago)   94d
kube-system        pod/kube-scheduler-k8s-master                       1/1     Running   0              94d
tigera-operator    pod/tigera-operator-776b7d494d-j66m4                1/1     Running   0              94d

NAMESPACE          NAME                                     TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)                  AGE
calico-apiserver   service/calico-api                       ClusterIP   10.100.134.232   <none>        443/TCP                  94d
calico-system      service/calico-kube-controllers-metrics  ClusterIP   None             <none>        9094/TCP                 94d
calico-system      service/calico-typha                     ClusterIP   10.98.48.33      <none>        5473/TCP                 94d
default            service/kubernetes                       ClusterIP   10.96.0.1        <none>        443/TCP                  94d
kube-system        service/kube-dns                         ClusterIP   10.96.0.10       <none>        53/UDP,53/TCP,9153/TCP   94d

NAMESPACE       NAME                             DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE
calico-system   daemonset.apps/calico-node       2         2         2       2            2           kubernetes.io/os=linux   94d
calico-system   daemonset.apps/csi-node-driver   2         2         2       2            2           kubernetes.io/os=linux   94d
kube-system     daemonset.apps/kube-multus-ds    2         2         2       2            2           <none>                   94d
kube-system     daemonset.apps/kube-proxy        2         2         2       2            2           kubernetes.io/os=linux   94d

NAMESPACE          NAME                                       READY   UP-TO-DATE   AVAILABLE   AGE
calico-apiserver   deployment.apps/calico-apiserver           2/2     2            2           94d
calico-system      deployment.apps/calico-kube-controllers    1/1     1            1           94d
calico-system      deployment.apps/calico-typha               1/1     1            1           94d
kube-system        deployment.apps/coredns                    2/2     2            2           94d
tigera-operator    deployment.apps/tigera-operator            1/1     1            1           94d

NAMESPACE          NAME                                                DESIRED   CURRENT   READY   AGE
calico-apiserver   replicaset.apps/calico-apiserver-648b88b9c5         2         2         2       94d
calico-system      replicaset.apps/calico-kube-controllers-6cd4d8dd54  1         1         1       94d
calico-system      replicaset.apps/calico-typha-656cc4f7d4             1         1         1       94d
kube-system        replicaset.apps/coredns-787d4945fb                  2         2         2       94d
tigera-operator    replicaset.apps/tigera-operator-776b7d494d          1         1         1       94d
```

ステップ 3 **route -n** コマンドを実行し、ネットワーク インターフェイス構成を確認します。この例では、ens160 はノードの管理インターフェイスです。ノード ens192、ens224、および ens256 は、ASAc インターフェイスにマッピングされます。

Kubernetes 環境での ASA コンテナの展開

```
ubuntu@k8s-worker:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.10.4.1       0.0.0.0         UG    100    0        0 ens160
10.10.4.0       0.0.0.0         255.255.255.224 U     0      0        0 ens160
10.10.4.1       0.0.0.0         255.255.255.255 UH    100    0        0 ens160
10.10.4.32      0.0.0.0         255.255.255.224 U     0      0        0 ens192
10.10.4.64      0.0.0.0         255.255.255.224 U     0      0        0 ens224
10.10.4.96      0.0.0.0         255.255.255.224 U     0      0        0 ens256
10.244.235.192  10.244.235.192  255.255.255.192 UG    0      0        0 vxlan.calico
10.244.254.128  0.0.0.0         255.255.255.192 U     0      0        0 *
172.17.0.0      0.0.0.0         255.255.0.0     U     0      0        0 docker0
```

ステップ 4 次に示す **cat** コマンドを実行し、hugepage 設定を確認します。

```
ubuntu@k8s-worker:~$ cat /proc/meminfo | grep -E 'HugePages_Total|HugePages_Free'
HugePages_Total:    2048
HugePages_Free:     2048
```

ステップ 5 ASA コンテナイメージを含む ASA Docker tar バンドルを、software.cisco.com からローカルの Docker レジストリにダウンロードします。

ステップ 6 ダウンロードした ASA コンテナイメージをローカルの Docker レジストリにロードします。

ステップ 7 ASAc GitHub リポジトリの **helm** フォルダからテンプレートなどをダウンロードします。

ステップ 8 values.yaml ファイルに必要なパラメータ値を入力します。

```
Default values for helm.
This is a YAML-formatted file.
Declare variables to be passed into your templates.
replicas: 1
image:
repository: localhost:5000/asac:9.22.1.1
persistVolPath: /home/ubuntu/pod-path
asacMgmtInterface: "ens192"
asacInsideInterface: "ens224"
asacOutsideInterface: "ens256"
```

values.yaml ファイル内のパラメータの名前と説明を以下に示します。

変数名	説明
repository	ローカルの Docker レジストリからの ASAc イメージパス。
persistVolPath	ASAc からの永続的構成ファイルが保存されているワーカーノードからの有効なパス。
asacMgmtInterface	ASAc 管理インターフェイスとして使用されるワーカー ノード インターフェイスの名前。
asacInsideInterface	ASAc 内部データインターフェイスとして使用されるワーカー ノード インターフェイスの名前。

変数名	説明
asacOutsideInterface	ASAc外部データインターフェイスとして使用されるワーカー ノード インターフェイスの名前。

ステップ 9 **day0-config** ファイルに存在するデフォルトのパラメータ値を確認します。必要に応じて、これらの値を更新することもできます。

ステップ 10 **helm install** コマンドを実行して Helm チャートを展開し、Kubernetes フレームワークで ASAc を展開します。

```
$ helm install test-asac helm
NAME: test-asac
LAST DEPLOYED: Sun Jan 21 07:41:03 2024
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
```

ステップ 11 **helm list -all** コマンドを実行して展開されたリソースを一覧表示し、ASAc 展開のステータスを確認します。

```
$ helm list –all
NAME        NAMESPACE    REVISION   UPDATED                                  STATUS     CHART
          APP VERSION
test-asac   default      1          2024-01-21 07:41:03.175728953 +0000 UTC   deployed   helm-0.1.0
        1.16.0
```

Kubernetes 環境での ASA コンテナの展開の検証

Helm チャート、ASAc ポッドのステータスを確認し、ポッドイベントを進めることで、ASA コンテナ（ASAc）の展開が成功したかどうかを検証します。

```
ubuntu@k8s-master:~$ helm status test-asac
NAME: test-asac
LAST DEPLOYED: Sun Jan 21 07:41:03 2024
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
```

```
ubuntu@k8s-master:~$ kubectl get pod
NAME                    READY   STATUS    RESTARTS   AGE
asac-5d8c4d547f-6k479   1/1     Running   0          43m
```

```
ubuntu@k8s-master:~$ kubectl events asac-5d8c4d547f-6k479
LAST SEEN   TYPE     REASON                OBJECT                              MESSAGE
52m         Normal   SuccessfulCreate      ReplicaSet/asac-5d8c4d547f          Created
pod: asac-5d8c4d547f-6k479
52m         Normal   ScalingReplicaSet     Deployment/asac                     Scaled up
```

```
 replica set asac-5d8c4d547f to 1
52m         Normal   WaitForFirstConsumer   PersistentVolumeClaim/local-pvc   waiting
for first consumer to be created before binding
51m         Normal   Scheduled              Pod/asac-5d8c4d547f-6k479         Successfully
 assigned default/asac-5d8c4d547f-6k479 to k8s-worker
51m         Normal   AddedInterface         Pod/asac-5d8c4d547f-6k479         Add eth0
[10.244.254.160/32] from k8s-pod-network
51m         Normal   AddedInterface         Pod/asac-5d8c4d547f-6k479         Add net1
[] from default/macvlan-mgmt-bridge
51m         Normal   AddedInterface         Pod/asac-5d8c4d547f-6k479         Add net2
[] from default/macvlan-in-bridge
51m         Normal   AddedInterface         Pod/asac-5d8c4d547f-6k479         Add net3
[] from default/macvlan-out-bridge
51m         Normal   Pulling                Pod/asac-5d8c4d547f-6k479         Pulling
image "dockerhub.cisco.com/asac-dev-docker/asac:9.22.x.x"
50m         Normal   Pulled                 Pod/asac-5d8c4d547f-6k479         Successfully
 pulled image "dockerhub.cisco.com/asac-dev-docker/asac:9.22.x.x" in 1m10.641397525s
(1m10.641428591s including waiting)
50m         Normal   Created                Pod/asac-5d8c4d547f-6k479         Created
container asac
50m         Normal   Started                Pod/asac-5d8c4d547f-6k479         Started
container asac
```

Kubernetes 環境での ASA コンテナ展開ログへのアクセス

何らかの問題が発生した場合は、トラブルシュートのためにポッドログとコンテナログを確認してください。

ポッドログを表示するには、次の手順を実行します。

```
ubuntu@k8s-master:~$ kubectl describe pod asac-5d8c4d547f-6k479
```

コンテナログを表示するには、次の手順を実行します。

```
ubuntu@k8s-master:~$ kubectl logs asac-5d8c4d547f-6k479
```

Kubernetes 環境での ASA コンテナポッドへのアクセス

kubectl attach コマンドを実行して ASA コンテナ（ASAc）ポッドの CLI にアクセスし、必要な出力を取得します。この例では、ASAc ポッドの CLI にアクセスして **show version** コマンドを実行しています。

（注） ASDM を使用して Kubernetes 環境で ASAc にアクセスすることもできます。

```
ubuntu@k8s-master:~$ kubectl attach -it asac-5d8c4d547f-6k479
If you don't see a command prompt, try pressing enter.
ciscoasa> show version
Cisco Adaptive Security Appliance Software Version 9.22
SSP Operating System Version 82.16(0.179i)
Device Manager Version 7.20
Compiled on Thu 02-Nov-23 13:30 GMT by builders
System image file is "Unknown, monitor mode tftp booted image"
Config file at boot was "startup-config"
```

```
ciscoasa up 55 mins 53 secs
Start-up time 12 secs
Hardware: ASAc, 2048 MB RAM, CPU Xeon E5 series 2100 MHz, 1 CPU (1 core)
BIOS Flash Firmware Hub @ 0x0, 0KB
0: Ext: Management0/0 : address is ae15.c291.86b1, irq 0
1: Ext: GigabitEthernet0/0 : address is faff.65b8.73a9, irq 0
2: Ext: GigabitEthernet0/1 : address is be89.078a.a560, irq 0
3: Int: Internal-Data0/0   : address is 0000.0100.0001, irq 0
```

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。