



在 Kubernetes 环境中部署 ASA 容器

您可以在任何云平台上运行的开源 Kubernetes 环境中部署 ASA 容器 (ASAc)。

- [概述，第 1 页](#)
- [在 Kubernetes 环境中部署 ASA 容器的准则和限制，第 1 页](#)
- [用于在 Kubernetes 环境中部署 ASA 容器的许可证，第 2 页](#)
- [在 Kubernetes 环境中部署 ASA 容器的解决方案组件，第 2 页](#)
- [用于在 Kubernetes 环境中部署 ASA 容器的拓扑示例，第 3 页](#)
- [在 Kubernetes 环境中部署 ASA 容器的前提条件，第 3 页](#)
- [在 Kubernetes 环境中部署 ASA 容器，第 4 页](#)
- [验证 Kubernetes 环境中的 ASA 容器部署，第 7 页](#)
- [在 Kubernetes 环境中访问 ASA 容器部署日志，第 7 页](#)
- [在 Kubernetes 环境中访问 ASA 容器 Pod，第 8 页](#)

概述

容器是一种软件包，它将代码和相关要求（如系统库、系统工具、默认设置、运行时等）捆绑在一起，以确保应用程序在计算环境中成功运行。从 Cisco Secure Firewall ASA 版本 9.22 开始，您可以在开源 Kubernetes 环境中部署 ASAc。在此解决方案中，ASAc 与容器网络接口 (CNI) 集成，并作为基础设施即代码 (IaC) 解决方案进行部署。与 CNI 的集成提高了网络基础设施部署的灵活性。

在 Kubernetes 环境中部署 ASA 容器的准则和限制

- ASA 容器解决方案仅在开源 Kubernetes 和 Docker 环境中进行验证。
- 其他 Kubernetes 框架，如 EKS、GKE、AKS、OpenShift 等，尚未通过验证。
- 以下功能未经验证：
 - 升级
 - 高可用性

- 集群
- IPv6
- 透明模式

用于在 Kubernetes 环境中部署 ASA 容器的许可证

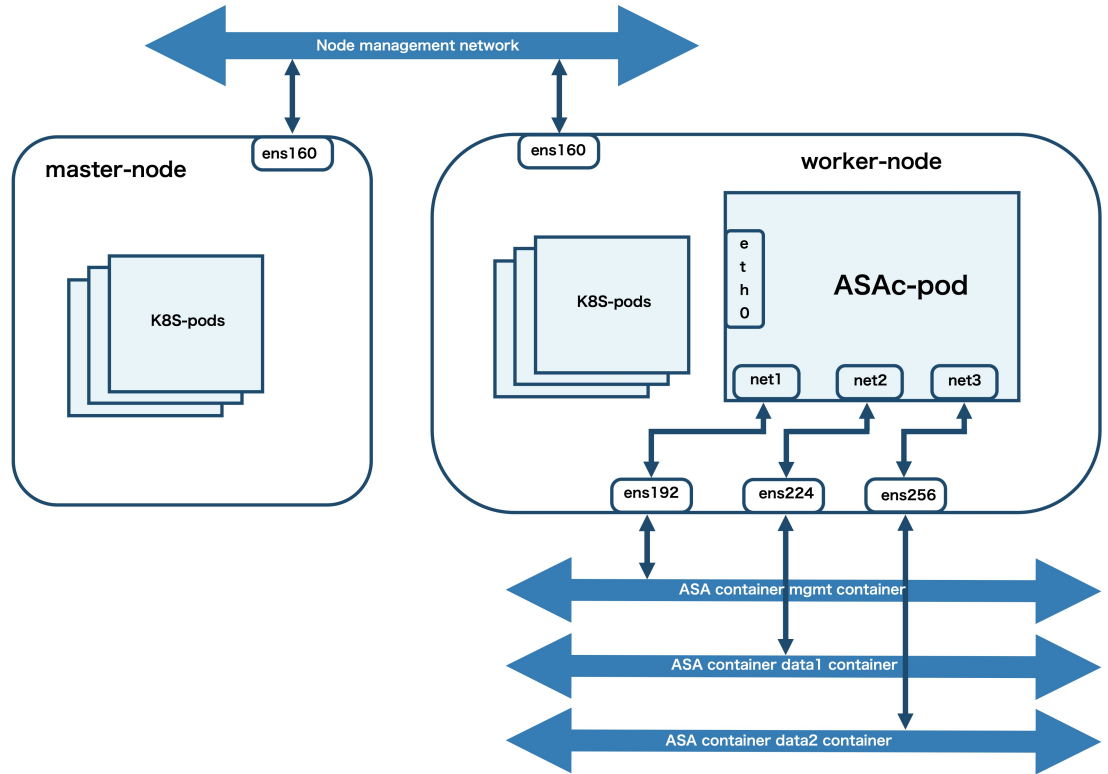
使用以下许可证之一在 Kubernetes 上部署 ASA 容器：

- ASAc5 - 1 个 vCPU、2 GB 内存和 100 Mbps 速率限制
- ASAc10 - 1 个 vCPU、2 GB 内存和 1 Gbps 速率限制

在 Kubernetes 环境中部署 ASA 容器的解决方案组件

- 操作系统
 - Ubuntu 20.04.6
 - Kubernetes 版本 v1.26
 - Helm 版本 v3.13.1
- Kubernetes 集群节点 - 主设备和工作节点
- Kubernetes CNI
 - POD 管理 CNI - Calico
 - ASAc 网络 CNI - Multus macvlan
- 以 yaml 文件形式提供的 Helm 图表用于设置基础设施即代码 (IaC)

用于在 Kubernetes 环境中部署 ASA 容器的拓扑示例



在此拓扑示例中，ASA 容器 (ASAc) Pod 有三个虚拟网络接口 - **net1**、**net2** 和 **net3**，它们会连接到以下工作节点接口 - **ens192**、**ens224** 和 **ens256**。工作节点接口映射到 ASAc mgmt、data1 和 data2 网络。接口 **ens160** 是节点管理接口。接口 **eth0** 派生自 Calico CNI。接口 **net1**、**net2** 和 **net3** 派生自 multi-macvlan CNI。

在 Kubernetes 环境中部署 ASA 容器的前提条件

- 确保在主节点和工作节点上安装了 Ubuntu 20.04.6 LTS。
- 在工作节点上为 ASA 容器 (ASAc) 操作分配三个虚拟接口。
- 设置工作节点的管理接口，以用于对工作节点进行 ssh 访问。
- 在工作节点上启用 Hugepage。
- 将 Calico CNI 设置为 POD 管理。
- 使用要用于管理 ASAc 接口的 MACvlan CNI 设置 Multus。

有关这些前提条件中提到的常规 Kubernetes 操作的详细信息，请参阅 [Kubernetes 文档](#)。

在 Kubernetes 环境中部署 ASA 容器

执行以下步骤在 Kubernetes 环境中部署 ASA 容器 (ASAc)。

过程

步骤 1 设置在 Kubernetes 环境中部署 ASA 容器的前提条件中提到的要求。

步骤 2 运行 **kubectl get nodes**、**kubectl get pods**和 **kubectl get all** 命令，以便分别显示所有节点、Pod 和所有资源的状态。确保 Kubernetes Pod 和节点处于就绪状态。

注释

下面给出的输出结果只是示例输出结果。

```
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

```
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-dxpm	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-dxppm	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

步骤 3 运行 `route -n` 命令以验证网络接口配置。在本例中，ens160 是节点的管理接口。节点 ens192、ens224 和 ens256 会别映射到 ASAc 接口。

```
ubuntu@k8s-worker:~$ route -n
```

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	vxlان.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

步骤 4 运行以下给出的 `cat` 命令以验证 hugepage 配置。

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

```
HugePages_Total: 2048
HugePages_Free: 2048
```

步骤 5 将包含 ASA 容器映像的 ASA Docker tar 捆绑包从 software.cisco.com 下载到本地 Docker 注册表。

步骤 6 将下载的 ASA 容器镜像加载到本地 Docker 注册表中。

步骤 7 从 [ASAc GitHub](#) 存储库中的 **helm** 文件夹下载模板和其他文件。

步骤 8 在 values.yaml 文件中输入所需的参数值。

```
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 文件中的参数名称和参数说明如下。

变量名称	说明
repository	本地 Docker 注册表中的 ASAc 映像路径。
保持卷路径	工作节点的有效路径，其中存储来自 ASAc 的持久配置文件。
asacMgmtInterface	用作 ASAc 管理接口的工作节点接口的名称。
asacInsideInterface	用作 ASAc 内部数据接口的工作节点接口的名称。
asacOutsideInterface	用作 ASAc 外部数据接口的工作节点接口的名称。

步骤 9 验证 **day0-config** 文件中的默认参数值。您也可以根据需要更新这些值。

步骤 10 运行 **helm install** 命令以部署 helm 图表，并在 Kubernetes 框架中部署 ASAc。

```
$ 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
```

步骤 11 运行 **helm list -all** 命令以列出已部署的资源，并检查 ASAc 部署的状态。

```
$ 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
```

验证 Kubernetes 环境中的 ASA 容器部署

通过检查 hem 图表、ASAc Pod 的状态和 Pod 事件，验证 ASA 容器 (ASAc) 部署是否成功。

```
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
```

在 Kubernetes 环境中访问 ASA 容器部署日志

检查 Pod 日志和容器日志，排除可能出现的任何问题。

要显示 Pod 日志，请执行以下操作：

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

要显示容器日志，请执行以下操作：

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

在 Kubernetes 环境中访问 ASA 容器 Pod

运行 **kubecttl attach** 命令以访问 ASA 容器 (ASAc) Pod 的 CLI 并获取所需的输出。在本示例中，我们访问 ASAc Pod 的 CLI 并运行 **show version** 命令。



注释 您也可以使用 ASDM 来访问 Kubernetes 环境中的 ASAc。

```
ubuntu@k8s-master:~$ kubecttl 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
```

当地语言翻译版本说明

思科可能会在某些地方提供本内容的当地语言翻译版本。请注意，翻译版本仅供参考，如有任何不一致之处，以本内容的英文版本为准。