

NSX Container Plug-In für OpenShift – Installations- und Administratorhandbuch

VMware NSX Container Plug-In 2.3, 2.3.1, 2.3.2

VMware NSX-T Data Center 2.3

VMware NSX-T Data Center 2.3.1



vmware®

Die aktuellste technische Dokumentation finden Sie auf der VMware-Website unter:

<https://docs.vmware.com/de/>

Die VMware-Website enthält auch die neuesten Produkt-Updates.

Falls Sie Anmerkungen zu dieser Dokumentation haben, senden Sie diese an:

docfeedback@vmware.com

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

VMware Global, Inc.
Zweigniederlassung Deutschland
Willy-Brandt-Platz 2
81829 München
Germany
Tel.: +49 (0) 89 3706 17 000
Fax: +49 (0) 89 3706 17 333
www.vmware.com/de

Copyright © 2017–2019 VMware, Inc. Alle Rechte vorbehalten. [Urheberrechts- und Markenhinweise](#).

Inhalt

NSX-T Container Plug-In für OpenShift – Installations- und Administratorhandbuch 4

- 1 Übersicht über das NSX-T Container Plug-In 5**
 - [Kompatibilitätsanforderungen 6](#)
 - [Überblick über die Installation 6](#)
 - [Upgrade von NCP 7](#)
- 2 Einrichten von NSX-T-Ressourcen 8**
 - [Konfigurieren von NSX-T-Ressourcen 8](#)
 - [Erstellen und Konfigurieren eines logischen Tier-0-Routers 11](#)
- 3 Installieren von NCP in einer OpenShift-Umgebung 13**
 - [Bereitstellen von OpenShift-VMs 13](#)
 - [Vorbereiten der Ansible-Hostdatei 13](#)
 - [Installieren von NCP und OpenShift mit einem einzelnen Playbook 16](#)
 - [Installieren von CNI-Plug-In, OVS und NCP-Docker-Image 16](#)
 - [Installieren von OpenShift Container Platform 19](#)
 - [Ausführen des NCP- und NSX-Knotenagenten 19](#)
 - [Hinweise zur Einrichtung 21](#)
- 4 Installieren von NCP in einer Bare-Metal-Umgebung 25**
 - [Installieren des NSX-T Data Center -CNI-Plug-Ins 25](#)
 - [Konfigurieren von NSX-T Data Center-Netzwerken für OpenShift-Knoten 25](#)
 - [Installieren des NSX-Knotenagenten 26](#)
 - [Configmap für ncp.ini in nsx-node-agent-ds.yml 27](#)
 - [Installieren des NSX-T Container Plug-Ins 30](#)
 - [Configmap für ncp.ini in ncp-rc.yml 32](#)
- 5 Lastausgleich 38**
 - [Konfigurieren des Lastausgleichs 38](#)
- 6 Verwalten von NSX-T Container Plug-in 45**
 - [Verwalten von IP-Blöcken über die NSX Manager -GUI 45](#)
 - [Anzeigen von IP-Block-Subnetzen über die GUI von NSX Manager 46](#)
 - [CIF-verknüpfte logische Ports 46](#)
 - [CLI-Befehle 47](#)
 - [Fehlercodes 58](#)

NSX-T Container Plug-In für OpenShift – Installations- und Administratorhandbuch

Dieses Handbuch beschreibt die Installation und Verwaltung von NSX-T Container Plug-in (NCP) für die Bereitstellung der Integration zwischen NSX-T Data Center und OpenShift.

Zielgruppe

Dieses Handbuch ist für die System- und Netzwerkadministratoren bestimmt. Es wird vorausgesetzt, dass Sie mit der Installation und Verwaltung von NSX-T Data Center und OpenShift vertraut sind.

VMware Technical Publications – Glossar

VMware Technical Publications enthält ein Glossar mit Begriffen, die Ihnen möglicherweise unbekannt sind. Definitionen von Begriffen, die in der technischen Dokumentation von VMware verwendet werden, finden Sie unter <http://www.vmware.com/support/pubs>.

Übersicht über das NSX-T Container Plug-In

1

NSX-T Container Plug-in (NCP) stellt eine Integration zwischen NSX-T Data Center und der Container-Orchestrierung wie z. B. Kubernetes sowie die Integration zwischen NSX-T Data Center und Container-basierten PaaS-Softwareprodukten (Plattform als Dienst), wie z. B. OpenShift, bereit. Dieses Handbuch beschreibt das Einrichten von NCP mit OpenShift.

Die Hauptkomponente von NCP wird in einem Container ausgeführt und kommuniziert mit NSX Manager und mit der OpenShift-Steuerungsebene. NCP überwacht Änderungen an den Containern und anderen Ressourcen und verwaltet Netzwerkressourcen wie logische Ports, Switches, Router und Sicherheitsgruppen für die Container per Aufruf der NSX API.

Das NSX CNI-Plugin wird auf jedem OpenShift-Knoten ausgeführt. Es überwacht Ereignisse im Container-Lebenszyklus, verbindet eine Containerschnittstelle mit dem Gast-vSwitch und programmiert den Gast-vSwitch für die Kennzeichnung und Weiterleitung des Containerdatenverkehrs zwischen den Containerschnittstellen und der VNIC.

NCP bietet die folgenden Funktionen:

- Es erstellt automatisch eine logische NSX-T-Topologie für einen OpenShift-Cluster und erstellt ein separates logisches Netzwerk für jeden OpenShift-Namespace.
- Es verbindet OpenShift-Pods mit dem logischen Netzwerk und weist IP- und MAC-Adressen zu.
- Es unterstützt Netzwerkadressübersetzung (Network Address Translation – NAT) und weist eine separate SNAT-IP für jeden OpenShift-Namespace zu.

Hinweis Bei der Konfiguration von NAT darf die Gesamtzahl der übersetzten IPs 1000 nicht überschreiten.

- Es implementiert OpenShift-Netzwerkrichtlinien mit verteilter NSX-T-Firewall.
 - Unterstützung für Ingress- und Egress-Netzwerkrichtlinien.
 - Unterstützung für IPBlock-Selektor in Netzwerkrichtlinien.
 - Unterstützung für matchLabels und matchExpression beim Angeben von Bezeichnungsselektoren für Netzwerkrichtlinien.
- Es implementiert eine OpenShift-Route mit NSX-T-Load Balancer der Schicht 7.
 - Unterstützung für HTTP-Route und HTTPS-Route mit TLS-Edge-Beendigung.

- Unterstützung für Routen mit alternativen Backends und Unterdomänen als Platzhalter.
- Es erstellt auf dem logischen NSX-T-Switch-Port Tags für den Namespace, den Pod-Namen und die Bezeichnungen eines Pods und lässt zu, dass der Administrator die NSX-T Data Center-Sicherheitsgruppen und -richtlinien basierend auf den Tags festlegt.

In dieser Version unterstützt NCP einen einzelnen OpenShift-Cluster.

Dieses Kapitel enthält die folgenden Themen:

- [Kompatibilitätsanforderungen](#)
- [Überblick über die Installation](#)
- [Upgrade von NCP](#)

Kompatibilitätsanforderungen

NSX-T Container Plug-in (NCP) weist die folgenden Kompatibilitätsanforderungen auf.

Softwareprodukt	Version
NSX-T Data Center	2.2, 2.3
Hypervisor für Container-Host-VMs	<ul style="list-style-type: none"> ■ Unterstützte vSphere-Version ■ RHEL KVM 7.4, 7.5
Container-Host-Betriebssystem	RHEL 7.4, 7.5
Platform as a Service	OpenShift 3.9, 3.10
Container-Host-Open vSwitch	2.9.1 (im Lieferumfang von NSX-T 2.3 und 2.2 enthalten)

Überblick über die Installation

Das Installieren und Konfigurieren von NCP umfasst die folgenden Schritte. Zur erfolgreichen Durchführung der Schritte müssen Sie mit NSX-T Data Center sowie der Installation und Verwaltung von OpenShift vertraut sein.

- 1 Installieren Sie NSX-T Data Center.
- 2 Erstellen Sie eine Overlay-Transportzone.
- 3 Erstellen Sie einen logischen Overlay-Switch, und verbinden Sie die Knoten mit dem Switch.
- 4 Erstellen Sie einen logischen Tier-0-Router.
- 5 Erstellen Sie IP-Blöcke für die Pods.
- 6 Erstellen Sie IP-Pools für SNAT (Source Network Address Translation).
- 7 Stellen Sie OpenShift-VMs bereit.
- 8 Bereiten Sie die Ansible-Hostdatei vor.

- 9 (Option 1) Installieren Sie NCP und OpenShift mit einem einzelnen Playbook.

(Option 2) Installieren Sie CNI-Plug-In, OVS (Open vSwitch) und NCP-Docker-Image. Installieren Sie anschließend OpenShift Container Platform.

- 10 Führen Sie NCP und den NSX-Knotenagent aus.

Die Schritte 2 bis 6 sind nicht erforderlich, wenn Sie NCP mit den bereitgestellten Playbooks installieren. Siehe [Installieren von NCP und OpenShift mit einem einzelnen Playbook](#) und [Installieren von CNI-Plug-In, OVS und NCP-Docker-Image](#).

Upgrade von NCP

Um NCP auf Version 2.3.0 zu aktualisieren, führen Sie die folgenden Schritte durch.

- 1 Führen Sie ein Upgrade für das CNI-RPM-Paket, das DaemonSet des NSX-Knotenagents und den NCP-ReplicationController durch.
- 2 (Optional) Aktualisieren Sie NSX-T Data Center auf Version 2.3.

NCP 2.3.0 unterstützt NSX-T 2.2, doch Sie können auch ein Upgrade auf NSX-T Data Center 2.3 durchführen.

Einrichten von NSX-T-Ressourcen

2

Zum Bereitstellen der Netzwerkfunktionen für OpenShift-Knoten müssen NSX-T Data Center-Ressourcen erstellt werden. Sie können diese Ressourcen manuell über die NSX Manager-GUI konfigurieren oder den Prozess mithilfe eines Ansible Playbooks automatisieren.

In diesem Abschnitt wird die manuelle Erstellung der NSX-T-Ressourcen beschrieben. Informationen zum Automatisieren des Vorgangs finden Sie unter [Installieren von CNI-Plug-In, OVS und NCP-Docker-Image](#).

Dieses Kapitel enthält die folgenden Themen:

- [Konfigurieren von NSX-T-Ressourcen](#)
- [Erstellen und Konfigurieren eines logischen Tier-0-Routers](#)

Konfigurieren von NSX-T-Ressourcen

Zu den zu konfigurierenden NSX-T Data Center-Ressourcen gehören eine Overlay-Transportzone, ein logischer Tier-0-Router, ein logischer Switch zum Verbinden der virtuellen Maschinen des Knotens, IP-Blöcke für Kubernetes-Knoten und ein IP-Pool für SNAT.

Sie konfigurieren NSX-T Data Center-Ressourcen unter Verwendung von UUIDs oder Namen in der Konfigurationsdatei `ncp.ini`.

Overlay-Transportzone

Melden Sie sich bei NSX Manager an und navigieren Sie zu **Fabric > Transportzonen**. Suchen Sie nach der für Containernetzwerke verwendeten Overlay-Transportzone oder erstellen Sie eine neue.

Geben Sie eine Overlay-Transportzone für einen Cluster an, indem Sie die Option `over_lay_tz` im Abschnitt `[nsx_v3]` der Datei `ncp.ini` festlegen. Dieser Schritt ist optional. Wenn Sie `over_lay_tz` nicht festlegen, ruft NCP die ID der Overlay-Transportzone automatisch vom Tier-0-Router ab.

Logisches Tier-0-Routing

Melden Sie sich bei NSX Manager an und navigieren Sie zu **Netzwerk > Routing > Router**. Suchen Sie nach dem für Containernetzwerke verwendeten Router oder erstellen Sie einen neuen.

Geben Sie einen logischen Tier-0-Router für einen Cluster an, indem Sie die Option `tier0_router` im Abschnitt `[nsx_v3]` der Datei `ncp.ini` festlegen.

Hinweis Der Router muss im Aktiv/Standby-Modus erstellt werden.

Logischer Switch

Die vom Knoten für den Netzwerkdatenverkehr verwendeten vNICs müssen mit einem logischen Overlay-Switch verbunden sein. Die Verwaltungsschnittstelle des Knotens muss nicht zwingend mit NSX-T Data Center verbunden sein, obwohl dies die Einrichtung erleichtert. Sie können einen logischen Switch erstellen, indem Sie sich bei NSX Manager anmelden und zu **Netzwerk > Switching > Switches** navigieren. Erstellen Sie auf dem Switch logische Ports und hängen Sie die vNICs des Knotens daran an. Die logischen Ports müssen die folgenden Tags aufweisen:

- Tag: `<cluster_name>`, Geltungsbereich: `ncp/cluster`
- Tag: `<node_name>`, Geltungsbereich: `ncp/node_name`

Der Wert für `<cluster_name>` muss mit dem Wert der Option `cluster` im Abschnitt `[coe]` von `ncp.ini` übereinstimmen.

IP-Blöcke für Kubernetes-Pods

Melden Sie sich bei NSX Manager an und navigieren Sie zu **Netzwerk > IPAM**, um einen oder mehrere IP-Blöcke zu erstellen. Geben Sie den IP-Block im CIDR-Format an.

Geben Sie IP-Blöcke für Kubernetes-Pods an, indem Sie die Option `container_ip_blocks` im Abschnitt `[nsx_v3]` der Datei `ncp.ini` festlegen.

Sie können IP-Blöcke auch speziell für Nicht-SNAT-Namespace erstellen.

Geben Sie Nicht-SNAT-IP-Blöcke ein, indem Sie die Option `no_snat_ip_blocks` im Abschnitt `[nsx_v3]` der Datei `ncp.ini` festlegen.

Wenn Sie Nicht-SNAT-IP-Blöcke erstellen, während NCP ausgeführt wird, müssen Sie NCP neu starten. Andernfalls verwendet NCP weiterhin die freigegebenen IP-Blöcke, bis sie erschöpft sind.

Hinweis Wenn Sie einen IP-Block erstellen, darf das Präfix nicht größer als der Wert des Parameters `subnet_prefix` in der NCP-Konfigurationsdatei `ncp.ini` sein.

IP-Pool für SNAT

Der IP-Pool wird für die Zuteilung von IP-Adressen verwendet, die der Übersetzung von Pod-IP-Adressen über SNAT-Regeln dienen. Zudem werden sie zum Verfügbarmachen der Ingress-Controller über SNAT/DNAT-Regeln verwendet – genau wie Openstack Floating IP-Adressen. Diese IP-Adressen werden auch als „externe IP-Adressen“ bezeichnet.

Mehrere Kubernetes-Cluster verwenden denselben externen IP-Pool. Jede NCP-Instanz verwendet einen Teil dieses Pools für den Kubernetes-Cluster, den sie verwaltet. Standardmäßig wird dasselbe Subnetzpräfix für Pod-Subnetze verwendet. Wenn Sie eine andere Subnetzgröße verwenden möchten, aktualisieren Sie die `external_subnet_prefix`-Option im `[nsx_v3]`-Abschnitt in `ncp.ini`.

Melden Sie sich bei NSX Manager an und navigieren Sie zu **Bestand > Gruppen > IP-Pools**, um einen Pool zu erstellen oder nach einem vorhandenen Pool zu suchen.

Geben Sie IP-Pools für SNAT an, indem Sie die Option `external_ip_pools` im Abschnitt `[nsx_v3]` der Datei `ncp.ini` festlegen.

Sie können SNAT auch für einen bestimmten Dienst konfigurieren, indem Sie dem Dienst eine Anmerkung hinzufügen. Beispiel:

```
apiVersion: v1
kind: Service
metadata:
  name: svc-example
  annotations:
    ncp/snat_pool: <external IP pool ID or name>
  selector:
    app: example
...
```

NCP konfiguriert die SNAT-Regel für diesen Dienst. Bei der Quell-IP der Regel handelt es sich um die Gruppe der Backend-Pods. Bei der Ziel-IP handelt es sich um die SNAT-IP, die aus dem angegebenen externen IP-Pool zugeteilt wurde. Beachten Sie Folgendes:

- Der von `ncp/snat_pool` angegebene IP-Pool sollte bereits in NSX-T Data Center vorhanden sein, bevor der Dienst konfiguriert wird. Ab NCP 2.3.1 muss der IP-Pool das Tag `{"ncp/owner": cluster:<cluster>}` aufweisen.
- In NSX-T Data Center ist die Priorität der SNAT-Regel für den Dienst höher als die für das Projekt.
- Wenn ein Pod mit mehreren SNAT-Regeln konfiguriert wird, funktioniert nur eine der Regeln.

Sie können durch Hinzufügen des folgenden Tags zum IP-Pool festlegen, welchem Namespace IPs aus dem SNAT-IP-Pool zugeteilt werden können.

- Geltungsbereich: `ncp/owner`, Tag: `ns:<namespace_UUID>`

Sie können die Namespace-UUID mit einem der folgenden Befehle abrufen:

```
oc get ns -o yaml
```

Beachten Sie Folgendes:

- Jedes Tag sollte eine UUID enthalten. Sie können mehrere Tags für denselben Pool erstellen.
- Wenn Sie die Tags ändern, nachdem einigen Namespaces basierend auf den alten Tags IPs zugewiesen wurden, werden diese IPs nicht wiederhergestellt, bis sich die SNAT-Konfigurationen der Dienste ändern oder NCP neu gestartet wird.

- Das Owner-Tag für den Namespace ist optional. Ohne dieses Tag können jedem Namespace IPs aus dem SNAT-IP-Pool zugeteilt werden.

(Optional) Markierte Firewallabschnitte

Damit der Administrator Firewallregeln erstellen kann und diese die von NCP erstellten, auf Netzwerkrichtlinien basierenden Firewallabschnitte nicht beeinträchtigen, melden Sie sich bei NSX Manager an, navigieren Sie zu **Sicherheit > Verteilte Firewall > Allgemein** und erstellen Sie zwei Firewallabschnitte.

Geben Sie Firewall-Markierungsabschnitte an, indem Sie die Optionen `bottom_firewall_section_marker` und `top_firewall_section_marker` im Abschnitt `[nsx_v3]` der Datei `ncp.ini` festlegen.

Der untere Firewallabschnitt muss sich unterhalb des oberen Firewallabschnitts befinden. Wenn diese Firewallabschnitte erstellt sind, werden alle von NCP zur Isolierung erstellten Firewallabschnitte oberhalb des unteren Firewallabschnitts und alle von NCP für Richtlinien erstellten Firewallabschnitte unterhalb des oberen Firewallabschnitts erstellt. Wenn diese Markierungsabschnitte nicht erstellt werden, werden alle Isolierungsregeln unten und alle Richtlinienabschnitte oben erstellt. Mehrere markierte Firewallabschnitte mit demselben Wert pro Cluster werden nicht unterstützt und führen zu einem Fehler.

Erstellen und Konfigurieren eines logischen Tier-0-Routers

Der logische Tier-0-Router verbindet die Kubernetes-Knoten mit externen Netzwerken.

Verfahren

- 1 Melden Sie sich über einen Browser bei NSX Manager unter `https://nsx-manager-ip-address` an.
- 2 Navigieren Sie zu **Netzwerk > Routing > Router** und klicken Sie auf **Hinzufügen > Tier-0-Router**.
- 3 Geben Sie einen Namen und optional eine Beschreibung ein.
- 4 Wählen Sie im Dropdown-Menü einen vorhandenen Edge-Cluster für eine Unterstützung dieses logischen Tier-0-Routers aus.
- 5 Wählen Sie einen Modus für die Hochverfügbarkeit aus.
Wählen Sie den Aktiv/Standby-Modus aus.
- 6 Klicken Sie auf **Speichern**.
Der neue logische Router wird als Link angezeigt.
- 7 Klicken Sie auf den Link für den logischen Router.
- 8 Klicken Sie auf **Routing > Route Redistribution**.
- 9 Klicken Sie auf **Hinzufügen**, um ein neues Kriterium für die Neuverteilung hinzuzufügen.
Wählen Sie für Quellen in einer gerouteten (Nicht-NAT-) Topologie **NSX statisch** aus. Wählen Sie in einer NAT-Topologie **Tier-0 NAT** aus.
- 10 Klicken Sie auf **Speichern**.

- 11 Klicken Sie auf den neu erstellten Router.
- 12 Klicken Sie auf **Konfiguration > Router-Ports**.
- 13 Klicken Sie auf **Hinzufügen**, um einen Uplink-Port hinzuzufügen.
- 14 Wählen Sie einen Transportknoten aus.
- 15 Wählen Sie den zuvor erstellten logischen Switch aus.
- 16 Geben Sie eine IP-Adresse in Ihrem externen Netzwerk an.
- 17 Klicken Sie auf **Speichern**.

Der neue logische Router wird als Link angezeigt.

Installieren von NCP in einer OpenShift-Umgebung

3

In diesem Kapitel wird das Installieren und Konfigurieren von NSX-T Container Plug-in (NCP) und OpenShift beschrieben.

Dieses Kapitel enthält die folgenden Themen:

- [Bereitstellen von OpenShift-VMs](#)
- [Vorbereiten der Ansible-Hostdatei](#)
- [Installieren von NCP und OpenShift mit einem einzelnen Playbook](#)
- [Installieren von CNI-Plug-In, OVS und NCP-Docker-Image](#)
- [Installieren von OpenShift Container Platform](#)
- [Ausführen des NCP- und NSX-Knotenagenten](#)
- [Hinweise zur Einrichtung](#)

Bereitstellen von OpenShift-VMs

Vor der Installation von NSX-T Container Plug-in muss OpenShift installiert werden. Sie müssen mindestens einen Master bereitstellen.

Weitere Informationen finden Sie unter <https://docs.openshift.com>.

Nächste Schritte

Bereiten Sie die Ansible-Hostdatei vor. Siehe [Vorbereiten der Ansible-Hostdatei](#).

Vorbereiten der Ansible-Hostdatei

Die Ansible-Datei `hosts` definiert die Knoten im OpenShift-Cluster.

Verfahren

- 1 Klonen Sie das NCP-GitHub-Repository unter <https://github.com/vmware/nsx-integration-for-openshift>. Die Datei `hosts` befindet sich im Verzeichnis `openshift-ansible-nsx`. Sie müssen die Datei `hosts` im Verzeichnis `openshift-ansible-nsx` beibehalten. Einige Playbooks gehen davon aus, dass dies der Pfad für die `hosts`-Datei ist.

- 2 Geben Sie in den [master]- und [nodes]-Abschnitten die Hostnamen und IP-Adressen der virtuellen OpenShift-Maschinen an. Beispiel:

```
[masters]
admin.rhel.osmaster ansible_ssh_host=101.101.101.4

[single_master]
admin.rhel.osmaster ansible_ssh_host=101.101.101.4

[nodes]
admin.rhel.osmaster ansible_ssh_host=101.101.101.4 openshift_ip=101.101.101.4 openshift_schedule=true openshift_hostname=admin.rhel.osmaster
admin.rhel.osnode ansible_ssh_host=101.101.101.5 openshift_ip=101.101.101.5 openshift_hostname=admin.rhel.osnode

[etcd]

[OSEv3:children]
masters
nodes
etcd
```

Beachten Sie, dass `openshift_ip` die interne Cluster-IP-Adresse identifiziert und festgelegt werden muss, wenn die zu verwendende Schnittstelle nicht die Standardkonfiguration ist. Die Variable `single_master` wird von ncp-bezogenen Rollen aus einem Master-Knoten für bestimmte Aufgaben nur einmal verwendet, z. B. für die Ressourcenkonfiguration der NSX-T Data Center-Managementebene.

- 3 Richten Sie den SSH-Zugriff so ein, dass der Zugriff auf alle Knoten ohne Kennwort von dem Knoten aus möglich ist, auf dem die Ansible-Rolle ausgeführt wird (in der Regel ist dies der Master-Knoten):

```
ssh-keygen
ssh-copy-id -i ~/.ssh/id_rsa.pub root@admin.rhel.osnode
```

- 4 Aktualisieren Sie den Abschnitt [OSEv3:vars]. Details über sämtliche Parameter finden Sie in der Dokumentation zu OpenShift Container Platform für die erweiterte Installation (suchen Sie nach „advanced installation“ in <https://docs.openshift.com>). Beispiel:

```
# Set the default route fqdn
openshift_master_default_subdomain=apps.corp.local

os_sdn_network_plugin_name=cni
openshift_use_openshift_sdn=false
openshift_node_sdn_mtu=1500

# If ansible_ssh_user is not root, ansible_become must be set to true
ansible_become=true

openshift_master_default_subdomain
    This is the default subdomain used in the OpenShift routes for External LB

os_sdn_network_plugin_name
    Set to 'cni' for the NSX Integration
```

```

openshift_use_openshift_sdn
    Set to false to disable the built-in OpenShift SDN solution

openshift_hosted_manage_router
    Set to false to disable creation of router during installation. The router has to be manual-
    ly started after NCP and nsx-node-agent are running.

openshift_hosted_manage_registry
    Set to false to disable creation of registry during installation. The registry has to be
    manually started after NCP and nsx-node-agent are running.

deployment_type
    Set to openshift-enterprise

openshift_hosted_manage_registry
    Set to false to disable auto creation of registry

openshift_hosted_manage_router
    Set to false to disable auto creation of router

openshift_enable_service_catalog
    Set to false to disable service_catalog

(For OpenShift 3.9 only) skip_sanity_checks
    Set to true

(For OpenShift 3.9 only) openshift_web_console_install
    Set to false

```

5 Überprüfen Sie, ob eine Verbindung zu allen Hosts besteht:

```
ansible OSEv3 -i /PATH/TO/HOSTS/hosts -m ping
```

Die Ergebnisse sollte wie folgt aussehen. Falls dies nicht der Fall ist, beheben Sie das Konnektivitätsproblem.

```

openshift-node1 | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
openshift-master | SUCCESS => {
  "changed": false,
  "ping": "pong"
}

```

Nächste Schritte

Installieren Sie das CNI-Plugin und OVS. Siehe [Installieren von CNI-Plug-In, OVS und NCP-Docker-Image](#).

Installieren von NCP und OpenShift mit einem einzelnen Playbook

Sie können NCP und OpenShift mit einem einzelnen Playbook installieren oder die Installation in separaten Schritten ausführen.

Das einzelne Ansible Playbook `install.yaml` führt die folgenden Aufgaben durch:

- NCP-Vorbereitung
- OpenShift-Installation
- NCP-Installation

Alternativ dazu können Sie NCP und OpenShift anhand der Anweisungen in den folgenden zwei Abschnitten installieren: [Installieren von CNI-Plug-In, OVS und NCP-Docker-Image](#) und [Installieren von OpenShift Container Platform](#).

Legen Sie vor dem Ausführen des `install.yaml`-Playbooks die erforderlichen und optionalen Parameter für die Playbook-Rollen `ncp_prep` und `ncp` fest. Die Parameter sind in [Installieren von CNI-Plug-In, OVS und NCP-Docker-Image](#) beschrieben.

Der folgende Befehl führt das Playbook aus:

```
ansible-playbook -i /PATH/TO/HOSTS/hosts install.yaml
```

Installieren von CNI-Plug-In, OVS und NCP-Docker-Image

Das CNI(Container Network Interface)-Plug-In, Open vSwitch (OVS) und das NCP-Docker-Image müssen auf den OpenShift-Knoten installiert sein. Die Installation erfolgt durch die Ansible Playbook-Ausführung.

Hinweis Dieser Schritt ist nicht erforderlich, wenn Sie NCP und OpenShift mit einem einzelnen Playbook installieren. Siehe [Installieren von NCP und OpenShift mit einem einzelnen Playbook](#).

Das Playbook enthält Anweisungen zum Konfigurieren von NSX-T-Ressourcen für die Knoten. Sie können die NSX-T Data Center-Ressourcen auch manuell konfigurieren, wie unter [Kapitel 2Einrichten von NSX-T-Ressourcen](#) beschrieben. Der Parameter `perform_nsx_config` gibt an, ob die Ressourcen bei der Playbook-Ausführung konfiguriert werden oder nicht.

Verfahren

- 1 Aktualisieren Sie die Parameterwerte in `roles/ncp_prep/default/main.yaml` und `roles/nsx_config/default/main.yaml`, einschließlich der URLs, unter denen das CNI-Plug-In RPM, OVS und das entsprechende Kernelmodul RPM heruntergeladen werden können. Darüber hinaus ist `uplink_port` der Name der virtuellen Netzwerkkarte des Uplink-Ports auf dem VM-Knoten. Die verbleibenden Variablen beziehen sich auf die Konfiguration der NSX-T Data Center-Managementebene.

Parameter, die angegeben werden müssen:

- `perform_nsx_config`: gibt an, ob die Ressourcenkonfiguration durchgeführt wird. Legen Sie die Einstellung auf „false“ fest, wenn die Konfiguration manuell vorgenommen und das Skript `nsx_config` nicht ausgeführt wird.
- `nsx_manager_ip`: IP-Adresse des NSX Manager
- `nsx_edge_cluster_name`: Name des Edge-Clusters für die Verwendung durch den Tier-0-Router
- `nsx_transport_zone_name`: Name der Overlay-Transportzone
- `os_node_name_list`: kommagetrennte Liste der Knotennamen

Zum Beispiel `node1,node2,node3`

- `subnet_cidr`: CIDR-Adresse für IP-Administrator, die „br-int“ auf dem Knoten zugewiesen wird
- `vc_host`: IP-Adresse von vCenter Server
- `vc_user`: Benutzername des vCenter Server-Administrators
- `vc_password`: Kennwort des vCenter Server-Administrators
- `vms`: kommagetrennte Liste der VM-Namen. Die Reihenfolge muss mit `os_node_name_list` übereinstimmen.

Die folgenden Parameter weisen Standardwerte auf. Sie können diese je nach Bedarf ändern.

- `nsx_t0_router_name`: Name des logischen Tier-0-Routers für den Cluster. Standard: **t0**
- `pod_ipblock_name`: Name des IP-Blocks für Pods. Standard: **podIPBlock**
- `pod_ipblock_cidr`: CIDR-Adresse für diesen IP-Block. Standard: **172.20.0.0/16**
- `snat_ippool_name`: Name des IP-Blocks für SNAT. Standard: **externalIP**.
- `snat_ippool_cidr`: CIDR-Adresse für diesen IP-Block. Standard: **172.30.0.0/16**
- `start_range`: die für diesen IP-Pool angegebene IP-Startadresse von CIDR. Standard: **172.30.0.1**
- `end_range`: die für diesen IP-Pool angegebene IP-Endadresse von CIDR. Standard: **172.30.255.254**
- `os_cluster_name`: Name des OpenShift-Clusters. Standard: **occl-one**
- `nsx_node_ls_name`: Name des logischen Switches, der mit den Knoten verbunden ist. Standard: **node_ls**

- `nsx_node_lr_name`: Name des logischen Routers für den Switch **node_ls**. Standard: **node_lr**

Das `nsx-config`-Playbook unterstützt nur die Erstellung jeweils eines IP-Pools und eines IP-Blocks. Wenn Sie weitere Pools und Blöcke benötigen, müssen Sie diese manuell erstellen.

- 2 Wechseln Sie zum Verzeichnis `openshift-ansible-nsx`, und führen Sie die Rolle `ncp_prep` aus.

```
ansible-playbook -i /PATH/TO/HOSTS/hosts ncp_prep.yaml
```

Das Playbook enthält Anweisungen zum Ausführen der folgenden Aktionen:

- Laden Sie die Installationsdatei für das CNI-Plug-In herunter.

Der Dateiname lautet `nsx-cni-1.0.0.0.xxxxxxx-1.x86_64.rpm`, wobei `xxxxxxx` die Build-Nummer ist.

- Installieren Sie die Installationsdatei für das CNI-Plug-In.

Das Plugin wird unter `/opt/cni/bin` installiert. Die CNI-Konfigurationsdatei „`10.net.conf`“ wird nach `/etc/cni/net.d` kopiert. RPM installiert auch die Konfigurationsdatei `/etc/cni/net.d/99-loopback.conf` für das Loopback-Plugin.

- Laden Sie die OVS-Installationsdateien herunter, und installieren Sie sie.

Die Dateien lauten `openvswitch-2.9.1.xxxxxxx-1.x86_64.rpm` und `openvswitch-kmod-2.9.1.xxxxxxx-1.el7.x86_64.rpm`, wobei `xxxxxxx` die Build-Nummer ist.

- Erstellen Sie die *br-int*-Instanz, sofern dies noch nicht erfolgt ist.

```
# ovs-vsctl add-br br-int
```

- Fügen Sie die Netzwerkschnittstelle (*node-if*) hinzu, die mit dem logischen Switch des Knotens für *br-int* verbunden ist.

- Stellen Sie sicher, dass *br-int* und *node-if* einen aktiven Status aufweisen.

```
# ip link set br-int up
# ip link set <node-if> up
```

- Aktualisieren Sie die Netzwerkkonfigurationsdatei, um sicherzustellen, dass die Netzwerkschnittstelle nach einem Neustart aktiv ist.
- Laden Sie die TAR-Datei für NCP herunter und laden Sie das Docker-Image aus der TAR-Datei.
- Laden Sie die YAML-Datei `ncp-rbac` herunter und ändern Sie `apiVersion` in **v1**.
- Erstellen Sie eine logische Topologie und zugehörige Ressourcen in NSX-T Data Center und erstellen Sie Tags auf diesen, damit sie von NCP erkannt werden.
- Aktualisieren Sie `ncp.ini` mit Informationen zur NSX-T Data Center-Ressource.

Nächste Schritte

Installieren Sie OpenShift Container Platform. Siehe [Installieren von OpenShift Container Platform](#).

Installieren von OpenShift Container Platform

OpenShift Container Platform (OCP) ist ein PaaS-Produkt (Platform as a Service), das Docker und Kubernetes verbindet.

Hinweis Dieser Schritt ist nicht erforderlich, wenn Sie NCP und OpenShift mit einem einzelnen Playbook installieren. Siehe [Installieren von NCP und OpenShift mit einem einzelnen Playbook](#).

Informationen zum Installieren von OCP finden Sie unter <https://docs.openshift.com>.

Nächste Schritte

Führen Sie NCP und den NSX-Knotenagent aus. Siehe [Ausführen des NCP- und NSX-Knotenagenten](#).

Ausführen des NCP- und NSX-Knotenagenten

Richten Sie den NCP- und NSX-Knotenagent ein, und führen Sie ihn aus.

Verfahren

- 1 Bearbeiten Sie `roles/ncp/defaults/main.yaml`, und geben Sie die IP-Adresse des OpenShift-API-Servers, die IP-Adresse von NSX Manager und die URLs zum Herunterladen der YAML-Datei für NCP ReplicationController und für DaemonSet für „nsx-node-agent“ an.
- 2 Führen Sie im Verzeichnis „openshift-ansible-nsx“ ncp-Rolle aus:

```
ansible-playbook -i /PATH/TO/HOSTS/hosts ncp.yaml
```

Die ncp-Rolle führt die folgenden Schritte aus:

- Überprüfen Sie, ob das Projekt „nsx-system“ vorhanden ist, und erstellen Sie es, wenn dies nicht der Fall ist.

```
oc new-project nsx-system
```

- Laden Sie die YAML-Datei `ncp-rbac` herunter und ändern Sie `apiVersion` in **v1**.
- Erstellen Sie das Dienstkonto für den NCP-Pod, erstellen Sie eine Clusterrolle, die die Ressourcen festlegt, auf die NCP zugreifen kann, und binden Sie die Clusterrolle an das NCP-Dienstkonto.
- Erstellen Sie das Dienstkonto für den nsx-node-agent-Pod, erstellen Sie eine Clusterrolle, die die Ressourcen festlegt, auf die der Knotenagent zugreifen kann, und binden Sie die Clusterrolle an das Knotenagent-Dienstkonto.

```
oc apply -f /tmp/ncp-rbac.yaml
```

- Rufen Sie das Token ab, das mit dem oben genannten Dienstkonto verknüpft ist, und speichern Sie es unter `/etc/nsx-ujo/<service_account>_token`:

```
secret=`kubectl get serviceaccount ncp-svc-account -o yaml | grep -A1 secrets | tail -n1 | awk {'print $3'}`
kubectl get secret $secret -o yaml | grep 'token:' | awk {'print $2'} | base64 -d > /etc/nsx-ujo/ncp_token
secret=`kubectl get serviceaccount nsx-node-agent-svc-account -o yaml | grep -A1 secrets | tail -n1 | awk {'print $3'}`
kubectl get secret $secret -o yaml | grep 'token:' | awk {'print $2'} | base64 -d > /etc/nsx-ujo/node_agent_token
```

- Laden Sie die SecurityContextConstraint (SCC)-YAML-Datei `ncp-os-scc.yml` für NCP herunter und erstellen Sie das SCC-Objekt basierend auf der YAML-Datei.

```
oc apply -f ncp-os-scc.yml
```

In der SCC-YAML-Datei ist „`spc_t`“ als SELinux-Typ angegeben, um sicherzustellen, dass NCP/`nsx-node-agent` unter SELinux über Zugriffsberechtigungen verfügt. Dies sieht folgendermaßen aus:

```
seLinuxContext:
  seLinuxOptions:
    type: spc_t
```

Die auf Bezeichnungen basierende SELinux-Zugriffssteuerungsebene unter `seLinuxOptions` von `seLinuxContext` ist in der SCC-YAML-Datei auf `s0:c0:c1023` festgelegt, damit `ncp/node-agent` auf Ziele aus anderen Dateikategorien zugreifen kann.

- Fügen Sie das SCC-Objekt dem Benutzer hinzu, der die Pods der NCP- und NSX-Knotenagents erstellt. So fügen Sie beispielsweise das SCC-Objekt dem Standardbenutzer im aktuellen Projekt hinzu:

```
oc adm policy add-scc-to-user ncp-scc -z default
```

- Fügen Sie das SCC-Objekt den NCP- und NSX-Knotenagent-Dienstkonten hinzu:

```
oc adm policy add-scc-to-user ncp-scc -z ncp-svc-account
oc adm policy add-scc-to-user ncp-scc -z nsx-node-agent-svc-account
```

- Laden Sie die YAML-Dateien für NCP ReplicationController (RC) und `nsx-node-agent` DaemonSet (DS) herunter und aktualisieren Sie die ConfigMap.
- Laden Sie das NCP-Image herunter (`nsx-node-agent` verwendet dasselbe Image), und laden Sie es.
- Konfigurieren Sie das Dienstkonto, und richten Sie das erforderliche SecurityContextConstraint-Objekt für NCP und `nsx_node_agent` ein.

- Erstellen Sie den NCP ReplicationController und das nsx-node-agent DaemonSet.

Hinweis NCP öffnet dauerhafte HTTP-Verbindungen zum Kubernetes-API-Server, um Lebenszyklusereignisse von Kubernetes-Ressourcen zu überwachen. Wenn ein API-Serverausfall oder ein Netzwerkausfall dazu führt, dass TCP-Verbindungen von NCP verfallen, müssen Sie NCP neu starten, sodass es die Verbindungen zum API-Server wiederherstellen kann. Andernfalls verpasst NCP die neuen Ereignisse.

Hinweise zur Einrichtung

Beachten Sie vor dem Einrichten von OpenShift und NCP die folgenden Informationen.

- Ein Pod darf nicht mehr als 11 Bezeichnungen haben, und ein Namespace darf nicht mehr als 12 Bezeichnungen aufweisen.
- Hinzugefügte Bezeichnungen für die interne Verwendung von OpenShift, z. B. eine Bezeichnung mit Präfix „openshift.io“ im zugehörigen Schlüssel, werden von NCP ignoriert. Demzufolge werden dem Benutzer die entsprechenden Tags, die auf den entsprechenden NSX-Ressourcen erstellt wurden, nicht angezeigt. Hier finden Sie eine Liste mit von OpenShift verwendeten Bezeichnungspräfixen. Vermeiden Sie die Verwendung eines Bezeichnungsschlüssels, der wie folgt beginnt:

```
openshift.io
pod-template
```

- Die Knoten benötigen Zugriff auf die Pods, beispielsweise für Kubelet-Integritätsprüfungen. Stellen Sie sicher, dass die Host-Verwaltungsschnittstelle auf das Pod-Netzwerk zugreifen kann.
- Die Linux-Funktionen NET_ADMIN und NET_RAW können von Angreifern ausgenutzt werden, um das Pod-Netzwerk zu gefährden. Sie sollten diese beiden Funktionen der nicht vertrauenswürdigen Container deaktivieren. Standardmäßig erhält NET_ADMIN bei SCC mit „restricted“ und „anyuid“ keinen Zugriff. Seien Sie vorsichtig bei einem SCC-Objekt, das NET_ADMIN explizit aktiviert oder die Pod-Ausführung im privilegierten Modus ermöglicht. Erstellen Sie darüber hinaus für nicht vertrauenswürdige Container ein separates SCC-Objekt, z. B. basierend auf SCC mit „anyuid“, und entfernen Sie dabei die Funktion NET_RAW. Dies kann durch Hinzufügen von NET_RAW zu der Liste „RequiredDropCapabilities“ in der SCC-Definition erfolgen.
- Erlauben Sie den Root-Zugriff in PODs/Containern (nur zu Testzwecken). Die unten aufgeführten Befehle benötigen Root-Zugriff in allen PODs des oc-Projekts, bei dem Sie aktuell angemeldet sind.

```
oc new-project test-project
oc project test-project
oc adm policy add-scc-to-user anyuid -z default
```

- Konfigurieren Sie die OpenShift-Registrierung (fügen Sie sie hinzu).

```
oc login -u system:admin -n default
oc adm registry --service-account=registry --config=/etc/origin/master/admin.kubeconfig
```

- Löschen der OpenShift-Registrierung

```
oc login -u system:admin -n default
oc delete svc/docker-registry dc/docker-registry
```

- Es fehlt eine IPtables-Firewallregel zum Zulassen von DNS-Anforderungen von Docker-Standard-Bridge-Containern für den dnsmasq-Prozess auf dem Knoten. Sie muss manuell geöffnet werden. Bearbeiten Sie `/etc/sysconfig/iptables`, und fügen Sie am Ende der Datei vor COMMIT die folgenden Regeln hinzu:

```
-A OS_FIREWALL_ALLOW -p tcp -m state --state NEW -m tcp --dport 53 -j ACCEPT
-A OS_FIREWALL_ALLOW -p udp -m state --state NEW -m udp --dport 53 -j ACCEPT
COMMIT
```

- Starten Sie iptables, docker und origin-node neu (Kube-Proxy und Kubelet werden neu gestartet).

```
systemctl restart iptables
systemctl restart docker
systemctl restart origin-node
```

- Die interne Docker-Registrierung von OpenShift muss Nicht-TLS verwenden können, damit OpenShift funktioniert. In der Regel sollte dies automatisch vom OpenShift Ansible-Installationsprogramm hinzugefügt werden. Momentan scheint dies aber nicht zu funktionieren. Bearbeiten Sie `„/etc/sysconfig/docker“` und fügen Sie Folgendes hinzu:

```
INSECURE_REGISTRY='--insecure-registry 172.30.0.0/16'
```

- Starten Sie Docker neu.

```
systemctl restart docker
```

- Die NCP-Unterstützung für Netzwerkrichtlinien entspricht der von Kubernetes bereitgestellten Unterstützung und richtet sich nach der von OpenShift verwendeten Kubernetes-Version.
 - OpenShift 3.9 – Die Regelklauseln in der Netzwerkrichtlinie können höchstens einen der Selektoren `namespaceSelector`, `podSelector` und `ipBlock` enthalten.
- Der Kubernetes-API-Server unterzieht eine Netzwerkrichtlinienspezifikation keiner Validierung. Es ist daher möglich, eine Netzwerkrichtlinie zu erstellen, die ungültig ist. NCP lehnt ungültige Netzwerkrichtlinien ab. Falls Sie die Netzwerkrichtlinie aktualisieren, damit sie gültig ist, wird sie von NCP trotzdem nicht verarbeitet. Sie müssen die Netzwerkrichtlinie löschen und mit einer gültigen Spezifikation neu erstellen.

- Bestimmte Kubernetes-Versionen weisen ein auf subPath bezogenes Problem auf (siehe <https://github.com/kubernetes/kubernetes/issues/61076>). Wenn die OpenShift-Version keinen Fix für dieses Problem enthält, schlägt die Erstellung des NCP-Pods mit folgendem Fehler fehl: CreateContainerConfigError: subPath für volumeMount konnte nicht vorbereitet werden. Sie können dieses Problem umgehen, indem Sie die Verwendung von subPath aus der NCP-YAML-Datei entfernen. Entfernen Sie insbesondere die Zeile mit subPath: ncp.ini und ersetzen Sie die Konfiguration für volumes durch Folgendes:

```
volumes:
  - name: config-volume
    # ConfigMap nsx-ncp-config is expected to supply ncp.ini
    configMap:
      name: nsx-ncp-config
      items:
        - key: ncp.ini
          path: ncp.ini
```

Ein Nebeneffekt dieser Änderung besteht darin, dass das gesamte /etc/nsx-ujo-Verzeichnis mit einem Schreibschutz versehen wird. Dies führt dazu, dass eine Verbindung mit NSX-T über ein Zertifikat und einen privaten Schlüssel nicht funktioniert, da NCP kein temporäres Verzeichnis unter /etc/nsx-ujo erstellen kann, um Zertifikat und privaten Schlüssel in eine einzige Datei zu verschieben.

- Beachten Sie Folgendes, wenn Sie einen OpenShift 3.10-Cluster ausführen oder ein Upgrade darauf durchführen:
 - Sie müssen die Konfiguration der Knotengruppen angeben, die für den OpenShift 3.10-Cluster spezifisch sind. Die Konfiguration des ConfigMap-Objekts des Knotens muss in der Bestandslisten-Hostdatei angegeben werden.
 - Allen Hosts, die in der [nodes]-Gruppe in der Bestandslisten-Hostdatei definiert sind, muss ein Knotengruppenname zugewiesen werden.
 - Wenn Sie über ein Ansible-Playbook ein Upgrade des OpenShift-Clusters durchführen, kommt es möglicherweise zu einem Netzerkausfall. Fügen Sie den Patch (<https://github.com/openshift/openshift-ansible/pull/8016/files#diff-2386e21861da3f95091dbb27d72ca366>) im „openshift-ansible“-Repository hinzu, um die Beendigung/Deinstallation von Open vSwitch-Paketen zu unterbinden.
- Seit OpenShift 3.10 wurde Kube-Proxy vom „openshift-node“-Dienst zu einem DaemonSet verschoben. Es wird nicht mehr standardmäßig gestartet. Führen Sie die folgenden Schritte durch, um Kube-Proxy manuell zu starten (davon ausgehend, dass das „openshift-ansible“-Repository geklont wurde):
 - Wechseln Sie in das Verzeichnis openshift-ansible und legen Sie unter [defaults] Folgendes fest:

```
library = roles/lib_utils/library/
```

- Erstellen Sie eine „create_proxy.yaml“-Datei im Playbooks-Verzeichnis mit den folgenden Einträgen:

```
- import_playbook: byo/openshift_facts.yml
- hosts: masters
  run_once: True
  roles:
    - kube_proxy_and_dns
```

- Führen Sie das Playbook aus:

```
ansible-playbook -i hosts playbooks/create_proxy.yaml
```

Sie sehen Fehlermeldungen, die auf das Fehlschlagen einiger Vorgänge hinweisen. Diese Meldungen können ignoriert werden. Sie können das Ergebnis überprüfen, indem Sie den Befehl `oc get po --all-namespaces` ausführen.

Installieren von NCP in einer Bare-Metal-Umgebung

4

Zur Installation von NSX-T Container Plug-in (NCP) in einer Bare-Metal-Umgebung werden ähnliche Schritte ausgeführt wie bei der Installation von NCP in einer Nicht-Bare-Metal-Umgebung. In diesem Abschnitt werden die Schritte beschrieben, die sich unterscheiden.

Dieses Kapitel enthält die folgenden Themen:

- [Installieren des NSX-T Data Center-CNI-Plug-Ins](#)
- [Konfigurieren von NSX-T Data Center-Netzwerken für OpenShift-Knoten](#)
- [Installieren des NSX-Knotenagenten](#)
- [Configmap für ncp.ini in nsx-node-agent-ds.yml](#)
- [Installieren des NSX-T Container Plug-Ins](#)
- [Configmap für ncp.ini in ncp-rc.yml](#)

Installieren des NSX-T Data Center -CNI-Plug-Ins

Das NSX-T Data Center-CNI-Plug-In muss auf dem OpenShift-Knoten installiert sein.

Verfahren

- 1 Laden Sie die entsprechende Installationsdatei für Ihre Linux-Bereitstellung herunter.
Der Dateiname lautet `nsx-cni-1.0.0.0.0.xxxxxxx-1.x86_64.rpm`, wobei `xxxxxxx` die Build-Nummer ist.
- 2 Installieren Sie die in Schritt 1 heruntergeladene RPM-Datei.
Das Plugin wird unter `/opt/cni/bin` installiert. Die CNI-Konfigurationsdatei „`10.net.conf`“ wird nach `/etc/cni/net.d` kopiert. RPM installiert auch die Konfigurationsdatei `/etc/cni/net.d/99-loopback.conf` für das Loopback-Plugin.

Konfigurieren von NSX-T Data Center-Netzwerken für OpenShift-Knoten

In diesem Abschnitt wird die Konfiguration von NSX-T Data Center-Netzwerken für Master- und Berechnungsknoten für OpenShift beschrieben.

Jeder Knoten muss als Betriebssystemtyp RHEL Container bei NSX Manager registriert werden. Die Verwaltungsschnittstelle des Knotens kann zur Verbindung des OpenShift-Clusters verwendet werden. Sie kann sich auf dem NSX-T Data Center-Fabric befinden oder nicht. Die anderen Schnittstellen stellen das Netzwerk für die Pods bereit und müssen sich auf dem NSX-T Data Center-Fabric befinden.

Der entsprechende Transportknoten muss die folgenden Tags aufweisen:

```
{'ncp/node_name': '<node_name>'}
{'ncp/cluster': '<cluster_name>'}
```

Sie können den Transportknoten für einen OpenShift-Knoten identifizieren, indem Sie zu **Fabric > Knoten** aus dem NSX Manager-GUI navigieren.

Wenn der OpenShift-Knotenname geändert wird, müssen Sie das Tag `ncp/node_name` aktualisieren und NCP neu starten. Mithilfe des folgenden Befehls können Sie die Knotennamen abrufen:

```
oc get nodes
```

Wenn Sie einem Cluster einen Knoten hinzufügen, während NCP ausgeführt wird, müssen Sie die Tags dem Transportknoten hinzufügen, bevor Sie den `oc cluster add`-Befehl ausführen. Andernfalls wird der neue Knoten nicht über Netzwerkkonnektivität verfügen. Wenn die Tags falsch oder nicht vorhanden sind, können Sie die folgenden Schritte ausführen, um das Problem zu beheben:

- Wenden Sie die richtigen Tags auf dem Transportknoten an.
- Starten Sie NCP neu.

Installieren des NSX-Knotenagenten

Der NSX-Knotenagent ist ein DaemonSet, wobei von jedem Pod zwei Container ausgeführt werden. Ein Container führt den NSX-Knotenagent aus, der hauptsächlich für die Verwaltung der Container-Netzwerk-schnittstellen zuständig ist. Er interagiert mit dem CNI-Plugin und dem Kubernetes-API-Server. Der andere Container führt den NSX-Kube-Proxy aus, der lediglich der Implementierung der Kubernetes-Dienst-abstraktion dient, indem Cluster-IP-Adressen in Pod-IP-Adressen übersetzt werden. Er implementiert dieselbe Funktionalität wie der vorgeschaltete Kube-Proxy.

Verfahren

- 1 Laden Sie das NCP-Docker-Image herunter.

Der Dateiname lautet `nsx-ncp-xxxxxxx.tar`, wobei `xxxxxxx` der Build-Nummer entspricht.

- 2 Laden Sie die YAML-Datei für das DaemonSet des NSX-Knotenagents herunter.

Der Dateiname ist `nsx-node-agent-ds.yml`. Sie können diese Datei bearbeiten oder als Beispiel für Ihre eigene Vorlagendatei verwenden.

- 3 Laden Sie das NCP-Docker-Image in Ihre Image-Registrierung.

```
docker load -i <tar file>
```

4 Bearbeiten Sie `nsx-node-agent-ds.yml`.

Ändern Sie den Image-Namen in den Namen der geladenen Datei.

Nehmen Sie die folgenden Änderungen vor:

```
[coe]
node_type = 'BAREMETAL'
...
[nsx_node_agent]
ovs_bridge = 'nsx-managed'
```

Heben Sie die Auskommentierung für folgende Zeilen auf:

```
securityContext:
  capabilities:
    add:
      - NET_ADMIN
      - SYS_ADMIN
      - SYS_PTRACE
      - DAC_READ_SEARCH
      # For BMC usecase
      - DAC_OVERRIDE
  volumeMounts:
    ...
    # mount nestdb-sock for baremetal node
    - name: nestdb-sock
      mountPath: /var/run/vmware/nestdb/nestdb-server.sock
  volumes:
    ...
    # volume for baremetal node
    - name: nestdb-sock
      hostPath:
        path: /var/run/vmware/nestdb/nestdb-server.sock
        type: Socket
```

Hinweis In der YAML-Datei müssen Sie angeben, dass das für `ncp.ini` generierte ConfigMap-Objekt als ReadOnly-Volume bereitgestellt werden muss. Die heruntergeladene YAML-Datei weist bereits diese Spezifikation auf, die nicht geändert werden sollte.

5 Erstellen Sie das NSX-Knotenagent-DaemonSet mit dem folgenden Befehl.

```
oc apply -f nsx-node-agent-ds.yml
```

Configmap für `ncp.ini` in `nsx-node-agent-ds.yml`

Die yaml-Beispieldatei `nsx-node-agent-ds.yml` enthält ein ConfigMap-Objekt für die Konfigurationsdatei `ncp.ini` für den NSX-Knoten-Agent. Dieser ConfigMap-Abschnitt enthält Parameter, die Sie angeben können, um die Knoten-Agent-Installation anzupassen.

Die von Ihnen heruntergeladene Beispieldatei `nsx-node-agent-ds.yml` enthält die folgenden `ncp.ini`-Informationen:

```
# ConfigMap for ncp.ini
apiVersion: v1
kind: ConfigMap
metadata:
  name: nsx-node-agent-config
  labels:
    version: v1
data:
  ncp.ini: |
    [DEFAULT]

    # Set to True to enable logging to stderr
    #use_stderr = True
    # Set to True to send logs to the syslog daemon
    #use_syslog = False
    # Enabler debug-level logging for the root logger. If set to True, the
    # root logger debug level will be DEBUG, otherwise it will be INFO.
    #debug = True

    # The log file path must be set to something like '/var/log/nsx-ujo/'. By
    # default, logging to file is disabled.
    #log_dir = None

    # Name of log file to send logging output to. If log_dir is set but log_file is
    # not, the binary name will be used, i.e., ncp.log, nsx_node_agent.log and
    # nsx_kube_proxy.log.
    #log_file = None

    # max MB for each compressed file. Defaults to 100 MB
    #log_rotation_file_max_mb = 100

    # Total number of compressed backup files to store. Defaults to 5.
    #log_rotation_backup_count = 5
    [coe]
    #
    # Common options for Container Orchestrators
    #

    # Container orchestrator adaptor to plug in
    # Options: kubernetes (default), openshift, pcf.
    #adaptor = kubernetes

    # Specify cluster for adaptor. It is a prefix of NSX resources name to
    # distinguish multiple clusters who are using the same NSX.
    # This is also used as the tag of IP blocks for cluster to allocate
    # IP addresses. Different clusters should have different IP blocks.
    #cluster = k8scluster

    # Log level for the NCP operations. If set, overrides the level specified
    # for the root logger. Possible values are NOTSET, DEBUG, INFO, WARNING,
    # ERROR, CRITICAL
```

```

#loglevel=None

# Log level for the NSX API client operations. If set, overrides the level
# specified for the root logger. Possible values are NOTSET, DEBUG, INFO,
# WARNING, ERROR, CRITICAL
nsxlib_loglevel=INFO

# Once enabled, all projects in this cluster will be mapped to a NAT
# topology in NSX backend
#enable_snat = True

# The type of container node. Possible values are HOSTVM, BAREMETAL.
node_type = BAREMETAL

[ha]
#
# NCP High Availability configuration options
#

# Time duration in seconds of mastership timeout. NCP instance will
# remain master for this duration after elected. Note that the heartbeat
# period plus the update timeout must be less than this period. This
# is done to ensure that the master instance will either confirm
# liveness or fail before the timeout.
#master_timeout = 9

# Time in seconds between heartbeats for elected leader. Once an NCP
# instance is elected master, it will periodically confirm liveness based
# on this value.
#heartbeat_period = 3

# Timeout duration in seconds for update to election resource. If the
# update request does not complete before the timeout it will be
# aborted. Used for master heartbeats to ensure that the update finishes
# or is aborted before the master timeout occurs.
#update_timeout = 3

[k8s]
#
# From kubernetes
#

# IP address of the Kubernetes API Server. If not set, will try to
# read and use the Kubernetes Service IP from environment variable
# KUBERNETES_SERVICE_HOST.
#apiserver_host_ip = <ip_address>

# Port of the Kubernetes API Server.
# Set to 6443 for https. If not set, will try to
# read and use the Kubernetes Service port from environment
# variable KUBERNETES_SERVICE_PORT.
#apiserver_host_port = <port>

# Specify a CA bundle file to use in verifying the Kubernetes API server
# certificate. (string value)

```

```

#ca_file = <None>
ca_file = /var/run/secrets/kubernetes.io/serviceaccount/ca.crt

# Full path of the Token file to use for authenticating with the k8s API server.
#client_token_file = <None>
client_token_file = /var/run/secrets/kubernetes.io/serviceaccount/token

# Full path of the client certificate file to use for authenticating
# with the k8s API server. It must be specified together with
# "client_private_key_file"
#client_cert_file = <None>

# Full path of the client certificate file to use for authenticating
# with the k8s API server. It must be specified together with
# "client_cert_file"
#client_private_key_file = <None>

# Log level for the kubernetes adaptor. If set, overrides the level specified
# for the root logger. Possible values are NOTSET, DEBUG, INFO, WARNING,
# ERROR, CRITICAL
#loglevel=None

[nsx_node_agent]
#
# Configuration for nsx_node_agent
#

# Needs to mount node /proc to container if nsx_node_agent runs in a container.
# By default node /proc will be mounted to /host/proc, the prefix is /host.
# It should be the same setting with mounted path in the daemonset yaml file.
# Set the path to '' if nsx_node_agent is running as a process in minion node.
#proc_mount_path_prefix = /host

# The OVS bridge to configure container interface.
#ovs_bridge = br-int

[nsx_kube_proxy]
#
# Configuration for nsx_kube_proxy
#

# The OVS uplink OpenFlow port where to apply the NAT rules to.
# If not specified, the port that gets assigned ofport=1 is used.
#ovs_uplink_port = <None>

```

Installieren des NSX-T Container Plug-Ins

NSX-T Container Plug-in (NCP) wird als Docker-Image bereitgestellt. NCP (NSX Container-Plug-In) sollte auf einem Knoten für Infrastrukturdienste ausgeführt werden. Die NCP-Ausführung auf dem Master-Knoten wird nicht empfohlen.

Verfahren

- 1 Laden Sie das NCP-Docker-Image herunter.

Der Dateiname lautet `nsx-ncp-xxxxxxx.tar`, wobei `xxxxxxx` der Build-Nummer entspricht.

- 2 Laden Sie die YAML-Vorlage für NCP ReplicationController herunter.

Der Dateiname lautet `ncp-rc.yml`. Sie können diese Datei bearbeiten oder als Beispiel für Ihre eigene Vorlagendatei verwenden.

- 3 Laden Sie das NCP-Docker-Image in Ihre Image-Registrierung.

```
docker load -i <tar file>
```

- 4 Bearbeiten Sie `ncp-rc.yml`.

Legen Sie den Knotentyp auf „Bare Metal“ fest.

```
[coe]
node_type = 'BAREMETAL'
```

Ändern Sie den Image-Namen in den Namen der geladenen Datei.

Geben Sie den `nsx_api_managers`-Parameter an. Diese Version unterstützt ein einzelnes Kubernetes-Knotencluster und eine einzelne NSX Manager-Instanz. Beispiel:

```
nsx_api_managers = 192.168.1.180
```

(Optional) Geben Sie den Parameter `ca_file` im Abschnitt `[nsx_v3]` an. Der Wert sollte einer CA-Paketdatei entsprechen, die beim Überprüfen des NSX Manager-Serverzertifikats verwendet wird. Wenn Sie keine Festlegung treffen, werden die System-Root-CAs verwendet.

Geben Sie die Parameter `nsx_api_cert_file` und `nsx_api_private_key_file` für die Authentifizierung mit NSX-T Data Center an.

`nsx_api_cert_file` ist der vollständige Pfad zu einer Client-Zertifikatdatei im PEM-Format. Der Inhalt dieser Datei sollte wie folgt aussehen:

```
-----BEGIN CERTIFICATE-----
<certificate_data_base64_encoded>
-----END CERTIFICATE-----
```

„`nsx_api_private_key_file`“ ist der vollständige Pfad zu einer privaten Client-Schlüsseldatei im PEM-Format. Der Inhalt dieser Datei sollte wie folgt aussehen:

```
-----BEGIN PRIVATE KEY-----
<private_key_data_base64_encoded>
-----END PRIVATE KEY-----
```

Geben Sie den Parameter `ingress_mode = nat` an, wenn der Ingress-Controller für die Ausführung im NAT-Modus konfiguriert ist.

Standardmäßig wird das Subnetzpräfix 24 für alle Subnetze verwendet, die von den IP-Blöcken für die logischen Pod-Switches zugeordnet sind. Wenn Sie eine andere Subnetzgröße verwenden möchten, aktualisieren Sie die `subnet_prefix`-Option im Abschnitt `[nsx_v3]`.

Hinweis In der YAML-Datei müssen Sie angeben, dass das für `ncp.ini` generierte ConfigMap-Objekt als ReadOnly-Volume bereitgestellt wird. Die heruntergeladene YAML-Datei weist bereits diese Spezifikation auf, die nicht geändert werden sollte.

5 Erstellen Sie den NCP ReplicationController.

```
kubectl create -f ncp-rc.yml
```

Hinweis NCP öffnet dauerhafte HTTP-Verbindungen zum Kubernetes-API-Server, um Lebenszyklusereignisse von Kubernetes-Ressourcen zu überwachen. Wenn ein API-Serverausfall oder ein Netzwerkausfall dazu führt, dass TCP-Verbindungen von NCP verfallen, müssen Sie NCP neu starten, sodass es die Verbindungen zum API-Server wiederherstellen kann. Andernfalls verpasst NCP die neuen Ereignisse.

Starten Sie während eines parallelen Updates des NCP ReplicationController den Container-Host nicht neu. Wenn der Host aus irgendeinem Grund neu gestartet wird, werden nach dem Neustart möglicherweise zwei NCP-Pods ausgeführt. In diesem Fall gehen Sie wie folgt vor:

- Löschen Sie einen der NCP-Pods. Dabei spielt es keine Rolle, welchen der beiden Pods Sie löschen. Beispiel:

```
oc delete pods <NCP pod name> -n nsx-system
```

- Löschen Sie den Namespace `nsx-system`. Beispiel:

```
oc delete -f ncp-rc.yml -n nsx-system
```

Configmap für `ncp.ini` in `ncp-rc.yml`

Die YAML-Beispieldatei `ncp-rc.yml` enthält ein ConfigMap-Objekt für die Konfigurationsdatei `ncp.ini`. Dieser ConfigMap-Abschnitt enthält Parameter, die Sie angeben müssen, bevor Sie NCP wie im vorherigen Abschnitt beschrieben installieren.

Die von Ihnen heruntergeladene Beispieldatei `ncp-rc.yml` enthält die folgenden `ncp.ini`-Informationen:

```
# ConfigMap for ncp.ini
apiVersion: v1
kind: ConfigMap
metadata:
  name: nsx-ncp-config
  labels:
    version: v1
data:
```



```

ncp.ini: |
[DEFAULT]

# Set to True to enable logging to stderr
#use_stderr = True
# Set to True to send logs to the syslog daemon
#use_syslog = False
# Enabler debug-level logging for the root logger. If set to True, the
# root logger debug level will be DEBUG, otherwise it will be INFO.
#debug = True

# The log file path must be set to something like '/var/log/nsx-ujo/'. By
# default, logging to file is disabled.
#log_dir = None

# Name of log file to send logging output to. If log_dir is set but log_file is
# not, the binary name will be used, i.e., ncp.log, nsx_node_agent.log and
# nsx_kube_proxy.log.
#log_file = None

# max MB for each compressed file. Defaults to 100 MB
#log_rotation_file_max_mb = 100

# Total number of compressed backup files to store. Defaults to 5.
#log_rotation_backup_count = 5
[coe]
#
# Common options for Container Orchestrators
#

# Container orchestrator adaptor to plug in
# Options: kubernetes (default), openshift, pcf.
#adaptor = kubernetes

# Specify cluster for adaptor. It is a prefix of NSX resources name to
# distinguish multiple clusters who are using the same NSX.
# This is also used as the tag of IP blocks for cluster to allocate
# IP addresses. Different clusters should have different IP blocks.
#cluster = k8scluster

# Log level for the NCP operations. If set, overrides the level specified
# for the root logger. Possible values are NOTSET, DEBUG, INFO, WARNING,
# ERROR, CRITICAL
#loglevel=None

# Log level for the NSX API client operations. If set, overrides the level
# specified for the root logger. Possible values are NOTSET, DEBUG, INFO,
# WARNING, ERROR, CRITICAL
nsxlib_loglevel=INFO

# Once enabled, all projects in this cluster will be mapped to a NAT
# topology in NSX backend
#enable_snat = True

# The type of container node. Possible values are HOSTVM, BAREMETAL.

```

```

node_type = BAREMETAL

[ha]
#
# NCP High Availability configuration options
#

# Time duration in seconds of mastership timeout. NCP instance will
# remain master for this duration after elected. Note that the heartbeat
# period plus the update timeout must be less than this period. This
# is done to ensure that the master instance will either confirm
# liveness or fail before the timeout.
#master_timeout = 9

# Time in seconds between heartbeats for elected leader. Once an NCP
# instance is elected master, it will periodically confirm liveness based
# on this value.
#heartbeat_period = 3

# Timeout duration in seconds for update to election resource. If the
# update request does not complete before the timeout it will be
# aborted. Used for master heartbeats to ensure that the update finishes
# or is aborted before the master timeout occurs.
#update_timeout = 3

[k8s]
#
# From kubernetes
#

# IP address of the Kubernetes API Server. If not set, will try to
# read and use the Kubernetes Service IP from environment variable
# KUBERNETES_SERVICE_HOST.
#apiserver_host_ip = <ip_address>

# Port of the Kubernetes API Server.
# Set to 6443 for https. If not set, will try to
# read and use the Kubernetes Service port from environment
# variable KUBERNETES_SERVICE_PORT.
#apiserver_host_port = <port>

# Specify a CA bundle file to use in verifying the Kubernetes API server
# certificate. (string value)
#ca_file = <None>
ca_file = /var/run/secrets/kubernetes.io/serviceaccount/ca.crt

# Full path of the Token file to use for authenticating with the k8s API server.
#client_token_file = <None>
client_token_file = /var/run/secrets/kubernetes.io/serviceaccount/token

# Full path of the client certificate file to use for authenticating
# with the k8s API server. It must be specified together with
# "client_private_key_file"
#client_cert_file = <None>

```

```

# Full path of the client certificate file to use for authenticating
# with the k8s API server. It must be specified together with
# "client_cert_file"
#client_private_key_file = <None>

# Log level for the kubernetes adaptor. If set, overrides the level specified
# for the root logger. Possible values are NOTSET, DEBUG, INFO, WARNING,
# ERROR, CRITICAL
#loglevel=None

# Specify how ingress controllers are expected to be deployed. Possible values:
# hostnetwork or nat. NSX will create NAT rules only in the second case.
#ingress_mode = hostnetwork

[nsx_v3]
#
# From nsx
#

# IP address of one or more NSX managers separated by commas. The IP address
# should be of the form (list value):
# <ip_address1>[:<port1>],<ip_address2>[:<port2>],...
# HTTPS will be used for communication with NSX. If port is not provided,
# port 443 will be used.
#nsx_api_managers = <ip_address>

# If true, the NSX Manager server certificate is not verified. If false the CA
# bundle specified via "ca_file" will be used or if unset the default system
# root CAs will be used. (boolean value)
#insecure = False

# Specify one or a list of CA bundle files to use in verifying the NSX Manager
# server certificate. This option is ignored if "insecure" is set to True. If
# "insecure" is set to False and ca_file is unset, the system root CAs will be
# used to verify the server certificate. (list value)
#ca_file = <None>

# Path to NSX client certificate file. If specified, the nsx_api_user and
# nsx_api_password options will be ignored. This option must be specified
# along with "nsx_api_private_key_file" option.
#nsx_api_cert_file = <None>

# Path to NSX client private key file. If specified, the nsx_api_user and
# nsx_api_password options will be ignored. This option must be specified
# along with "nsx_api_cert_file" option.
#nsx_api_private_key_file = <None>

# The time in seconds before aborting a HTTP connection to a NSX manager.
# (integer value)
#http_timeout = 10

# The time in seconds before aborting a HTTP read response from a NSX manager.
# (integer value)
#http_read_timeout = 180

```

```

# Maximum number of times to retry a HTTP connection. (integer value)
#http_retries = 3

# Maximum concurrent connections to each NSX manager. (integer value)
#concurrent_connections = 10

# The amount of time in seconds to wait before ensuring connectivity to the NSX
# manager if no manager connection has been used. (integer value)
#conn_idle_timeout = 10

# Number of times a HTTP redirect should be followed. (integer value)
#redirects = 2

# Maximum number of times to retry API requests upon stale revision errors.
# (integer value)
#retries = 10

# Subnet prefix of IP block. IP block will be retrieved from NSX API and
# recognised by tag 'cluster'.
# Prefix should be less than 31, as two addresses(the first and last addresses)
# need to be network address and broadcast address.
# The prefix is fixed after the first subnet is created. It can be changed only
# if there is no subnets in IP block.
#subnet_prefix = 24

# Indicates whether distributed firewall DENY rules are logged.
#log_dropped_traffic = False

# Option to use native loadbalancer support.
#use_native_loadbalancer = False

# Used when ingress class annotation is missing
# if set to true, the ingress will be handled by nsx lbs
# otherwise will be handled by 3rd party ingress controller (e.g. nginx)
#default_ingress_class_nsx = True

# Path to the default certificate file for HTTPS load balancing
#lb_default_cert_path = <None>

# Path to the private key file for default certificate for HTTPS load balancing
#lb_priv_key_path = <None>

# Option to set load balancing algorithm in load balancer pool object.
# Available choices are
# ROUND_ROBIN/LEAST_CONNECTION/IP_HASH/WEIGHTED_ROUND_ROBIN
#pool_algorithm = 'ROUND_ROBIN'

# Option to set load balancer service size. Available choices are
# SMALL/MEDIUM/LARGE.
# MEDIUM Edge VM (4 vCPU, 8GB) only supports SMALL LB.
# LARGE Edge VM (8 vCPU, 16GB) only supports MEDIUM and SMALL LB.
# Bare Metal Edge (IvyBridge, 2 socket, 128GB) supports LARGE, MEDIUM and
# SMALL LB
#service_size = 'SMALL'

```

```

# Choice of persistence type for ingress traffic through L7 Loadbalancer.
# Accepted values:
# 'cookie'
# 'source_ip'
#l7_persistence = <None>

# Choice of persistence type for ingress traffic through L4 Loadbalancer.
# Accepted values:
# 'source_ip'
#l4_persistence = <None>

# Name or UUID of the tier0 router that project tier1 routers connect to
#tier0_router = <None>

# Name or UUID of the NSX overlay transport zone that will be used for creating
# logical switches for container networking. It must refer to an existing
# transport zone on NSX and every hypervisor that hosts the Kubernetes
# node VMs must join this transport zone
#overlay_tz = <None>

# Name or UUID of the NSX lb service that can be attached by virtual servers
#lb_service = <None>

# Name or UUID of the container ip blocks that will be used for creating
# subnets. If name, it must be unique
#container_ip_blocks = <None>

# Name or UUID of the container ip blocks that will be used for creating
# subnets for no-SNAT projects. If specified, no-SNAT projects will use these
# ip blocks ONLY. Otherwise they will use container_ip_blocks
#no_snat_ip_blocks = <None>

# Name or UUID of the external ip pools that will be used for allocating IP
# addresses which will be used for translating container IPs via SNAT rules
#external_ip_pools = <None>

# Firewall sections for this cluster will be created below this mark section
#top_firewall_section_marker = <None>

# Firewall sections for this cluster will be created above this mark section
#bottom_firewall_section_marker = <None>

```

Lastausgleich

Der NSX-T Data Center-Load Balancer ist in OpenShift integriert und fungiert als OpenShift-Router.

NCP überwacht OpenShift-Routen- und -Endpoint-Ereignisse und konfiguriert Lastausgleichsregeln auf dem Load Balancer basierend auf der Routenspezifikation. Dies führt dazu, dass der NSX-T Data Center-Load Balancer eingehenden Schicht 7-Datenverkehr basierend auf den Regeln zu den geeigneten Backend-Pods weiterleitet.

Konfigurieren des Lastausgleichs

Das Konfigurieren des Lastausgleichs umfasst das Konfigurieren eines Kubernetes-LoadBalancer-Diensts oder einer OpenShift-Route. Sie müssen auch den NCP ReplicationController konfigurieren. Der LoadBalancer-Dienst wird für den Schicht 4-Datenverkehr und die OpenShift-Route für den Schicht 7-Datenverkehr verwendet.

Wenn Sie einen Kubernetes-LoadBalancer-Dienst konfigurieren, wird diesem eine IP-Adresse aus dem von Ihnen konfigurierten externen IP-Block zugeteilt. Der Load Balancer wird auf dieser IP-Adresse und dem Dienst-Port bereitgestellt. Sie können den Namen oder die ID eines IP-Pools mithilfe der `loadBalancerIP`-Spezifikation in der Definition des LoadBalancer-Diensts angeben. Die IP-Adresse des LoadBalancer-Diensts wird aus diesem IP-Pool zugeteilt. Wenn die `loadBalancerIP`-Spezifikation leer ist, wird die IP-Adresse aus dem externen IP-Block zugeteilt, den Sie konfigurieren.

Ab NCP 2.3.1 muss der von `loadBalancerIP` angegebene IP-Pool das Tag `{"ncp/owner": "cluster:<cluster>"}` aufweisen.

Um den Load Balancer von NSX-T Data Center zu verwenden, müssen Sie den Lastausgleich in NCP konfigurieren. Führen Sie in der Datei `ncp_rc.yml` die folgenden Schritte aus:

- 1 Legen Sie `use_native_loadbalancer = True` fest.
- 2 Legen Sie `pool_algorithm` auf `WEIGHTED_ROUND_ROBIN` fest.
- 3 Legen Sie „`lb_default_cert_path`“ und „`lb_priv_key_path`“ als vollständige Pfadnamen des von der Zertifizierungsstelle signierten Zertifikats und der Datei mit dem privaten Schlüssel fest. Ein Beispielskript zum Generieren eines von einer Zertifizierungsstelle signierten Zertifikats finden Sie unten. Mounten Sie darüber hinaus das Standardzertifikat und den Standardschlüssel in den NCP-Pod. Anweisungen hierzu finden Sie unten.

- 4 (Optional) Legen Sie mit den Parametern `l4_persistence` und `l7_persistence` eine Persistenzeinstellung fest. Die verfügbare Option für die Schicht-4-Persistenz ist „Quell-IP“. Die verfügbaren Optionen für die Schicht-7-Persistenz sind „Cookie“ und „Quell-IP“. Die Standardeinstellung ist `<None>`.
Beispiel:

```
# Choice of persistence type for ingress traffic through L7 Loadbalancer.
# Accepted values:
# 'cookie'
# 'source_ip'
l7_persistence = cookie

# Choice of persistence type for ingress traffic through L4 Loadbalancer.
# Accepted values:
# 'source_ip'
l4_persistence = source_ip
```

- 5 (Optional) Legen Sie `service_size = SMALL, MEDIUM` oder `LARGE` fest. Die Standardeinstellung ist `SMALL`.
- 6 Wenn Sie OpenShift 3.11 ausführen, müssen Sie die folgende Konfiguration ausführen, damit OpenShift dem LoadBalancer-Dienst keine IP zuweist.
- Legen Sie `ingressIPNetworkCIDR` unter `networkConfig` in der Datei `/etc/origin/master/master-config.yaml` auf „0.0.0.0/32“ fest.
 - Starten Sie den API-Server und die API-Controller mit den folgenden Befehlen neu:

```
master-restart api
master-restart controllers
```

Hinweis Wenn Sie einen Load Balancer auf Schicht 4 und einen auf Schicht 7 erstellen, können Sie entweder `l4_persistence` oder `l7_persistence` oder beide auf `source_ip` festlegen, Sie können jedoch nicht `l4_persistence` auf `source_ip` und `l7_persistence` auf `cookie` festlegen. Wenn Sie irrtümlicherweise `l4_persistence` auf `source_ip` und `l7_persistence` auf `cookie` festlegen, funktioniert der LoadBalancer-Dienst nicht. Um dieses Problem zu beheben, müssen Sie die Ingress-Ressource und den LoadBalancer-Dienst löschen, die Persistenzeinstellungen ändern, NCP neu starten und die Ingress-Ressource und den LoadBalancer-Dienst neu erstellen.

Beispiel für einen Load Balancer auf Schicht 7:

Zur Bereitstellung des Lastausgleichs auf Schicht 7 konfiguriert die folgende YAML-Datei zwei Replikations-Controller (`tea-rc` und `coffee-rc`), zwei Dienste (`tea-svc` und `coffee-svc`) sowie zwei Routen (`cafe-route-multi` und `cafe-route`).

```
# RC
apiVersion: v1
kind: ReplicationController
metadata:
  name: tea-rc
spec:
```

```

replicas: 2
template:
  metadata:
    labels:
      app: tea
  spec:
    containers:
      - name: tea
        image: nginxdemos/hello
        imagePullPolicy: IfNotPresent
        ports:
          - containerPort: 80
---
apiVersion: v1
kind: ReplicationController
metadata:
  name: coffee-rc
spec:
  replicas: 2
  template:
    metadata:
      labels:
        app: coffee
    spec:
      containers:
        - name: coffee
          image: nginxdemos/hello
          imagePullPolicy: IfNotPresent
          ports:
            - containerPort: 80
---
# Services
apiVersion: v1
kind: Service
metadata:
  name: tea-svc
  labels:
    app: tea
spec:
  ports:
    - port: 80
      targetPort: 80
      protocol: TCP
      name: http
  selector:
    app: tea
---
apiVersion: v1
kind: Service
metadata:
  name: coffee-svc
  labels:
    app: coffee
spec:
  ports:

```



```

- port: 80
  targetPort: 80
  protocol: TCP
  name: http
  selector:
    app: coffee
---
# Routes
apiVersion: v1
kind: Route
metadata:
  name: cafe-route-multi
spec:
  host: www.cafe.com
  path: /drinks
  to:
    kind: Service
    name: tea-svc
    weight: 1
  alternateBackends:
  - kind: Service
    name: coffee-svc
    weight: 2
---
apiVersion: v1
kind: Route
metadata:
  name: cafe-route
spec:
  host: www.cafe.com
  path: /tea-svc
  to:
    kind: Service
    name: tea-svc
    weight: 1

```

Zusätzliche Hinweise

- Nur die Edge-Beendigung wird für HTTPS-Datenverkehr unterstützt.
- Unterdomänen als Platzhalter werden unterstützt. Wenn zum Beispiel wildcardPolicy auf **Unterdomäne** und der Hostname auf **wildcard.example.com** festgelegt ist, werden alle Anforderungen an ***.example.com** verarbeitet.
- Wenn NCP aufgrund einer fehlerhaften Konfiguration einen Fehler während der Verarbeitung eines Routen-Ereignisses auslöst, müssen Sie die YAML-Datei der Route korrigieren und die Routen-Resource löschen und neu erstellen.
- NCP erzwingt den Hostnamen-Besitz nicht nach Namespaces.
- Ein LoadBalancer-Dienst wird pro Kubernetes-Cluster unterstützt.
- NSX-T Data Center erstellt für jeden LoadBalancer-Dienst-Port einen virtuellen Server und einen Pool des Load Balancers auf Schicht 4. TCP und UDP werden unterstützt.

- Der Load Balancer von NSX-T Data Center ist in verschiedenen Größen erhältlich. Weitere Informationen zum Konfigurieren eines Load Balancers für NSX-T Data Center finden Sie im *NSX-T-Administratorhandbuch*.

Der kleine Load Balancer von NSX-T Data Center unterstützt Folgendes:

- 10 virtuelle NSX-T-Server
- 10 NSX-T-Pools
- 30 NSX-T-Poolmitglieder
- 8 Ports für LoadBalancer-Dienste.
- Insgesamt 10 von den LoadBalancer-Diensten und Routenressourcen definierte Ports.
- Insgesamt 30 von den LoadBalancer-Diensten und Routenressourcen referenzierte Endpoints.

Der mittlere Load Balancer von NSX-T Data Center unterstützt Folgendes:

- 100 virtuelle NSX-T-Server
- 100 NSX-T-Pools
- 300 NSX-T-Poolmitglieder
- 98 Ports für LoadBalancer-Dienste.
- Insgesamt 100 von den LoadBalancer-Diensten und Routenressourcen definierte Ports.
- Insgesamt 300 von den LoadBalancer-Diensten und Routenressourcen referenzierte Endpoints.

Der große Load Balancer von NSX-T Data Center unterstützt Folgendes:

- 1000 virtuelle NSX-T-Server
- 1000 NSX-T-Pools
- 3000 NSX-T-Poolmitglieder
- 998 Ports für LoadBalancer-Dienste.
- Insgesamt 1000 von den LoadBalancer-Diensten und Routenressourcen definierte Ports.
- Insgesamt 3000 von den LoadBalancer-Diensten und Routenressourcen referenzierte Endpoints.

Nachdem der Load Balancer erstellt wurde, kann seine Größe nicht durch Aktualisierung der Konfigurationsdatei geändert werden. Die Größe kann über die Benutzeroberfläche oder API geändert werden.

- Ab NCP 2.3.1 wird die automatische Skalierung des Load Balancers auf Schicht 4 unterstützt. Wenn ein Kubernetes-LoadBalancer-Dienst erstellt oder geändert wird, sodass er zusätzliche virtuelle Server erfordert, und der vorhandene Load Balancer auf Schicht 4 nicht über ausreichend Kapazität verfügt, wird ein neuer Load Balancer auf Schicht 4 erstellt. NCP löscht einen Load Balancer auf Schicht 4 auch, wenn keine virtuellen Server mehr an ihn angehängt sind. Diese Funktion ist standardmäßig aktiviert. Sie können sie deaktivieren, indem Sie in der NCP-ConfigMap `l4_lb_auto_scaling` auf **false** festlegen. Diese Funktion erfordert NSX-T Data Center 2.3 oder höhere Versionen.

Beispielskript zum Generieren eines von einer Zertifizierungsstelle signierten Zertifikats

Das nachfolgende Skript generiert ein von einer Zertifizierungsstelle signiertes Zertifikat und einen privaten in den Dateien <Dateiname>.crt und <Dateiname>.key gespeicherten privaten Schlüssel. Der Befehl `genrsa` generiert einen Zertifizierungsstellen-Schlüssel. Der Zertifizierungsstellen-Schlüssel muss verschlüsselt werden. Sie können eine Verschlüsselungsmethode mit einem Befehl wie zum Beispiel `aes256` angeben.

```
#!/bin/bash
host="www.example.com"
filename=server

openssl genrsa -out ca.key 4096
openssl req -key ca.key -new -x509 -days 365 -sha256 -extensions v3_ca -out ca.crt -subj
"/C=US/ST=CA/L=Palo Alto/O=OS3/OU=Eng/CN=${host}"
openssl req -out ${filename}.csr -new -newkey rsa:2048 -nodes -keyout ${filename}.key -subj
"/C=US/ST=CA/L=Palo Alto/O=OS3/OU=Eng/CN=${host}"
openssl x509 -req -days 360 -in ${filename}.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out ${filename}.crt -sha256
```

Mounten Sie das Standardzertifikat und den Standardschlüssel in den NCP-Pod

Wenn Sie das Zertifikat und den privaten Schlüssel erstellt haben, platzieren Sie sie im Verzeichnis `/etc/nsx-ujo` auf dem VM-Host. Vorausgesetzt, dass die Zertifikats- und Schlüsseldateien die Namen `lb-default.crt` und `lb-default.key` aufweisen, bearbeiten Sie `ncp-rc.yaml`, sodass diese Dateien auf dem Host in den Pod gemountet werden. Beispiel:

```
spec:
  ...
  containers:
  - name: nsx-ncp
    ...
    volumeMounts:
    ...
    - name: lb-default-cert
      # Mount path must match nsx_v3 option "lb_default_cert_path"
      mountPath: /etc/nsx-ujo/lb-default.crt
    - name: lb-priv-key
      # Mount path must match nsx_v3 option "lb_priv_key_path"
      mountPath: /etc/nsx-ujo/lb-default.key
  volumes:
  ...
  - name: lb-default-cert
    hostPath:
```

```
    path: /etc/nsx-ujo/lb-default.crt  
- name: lb-priv-key  
  hostPath:  
    path: /etc/nsx-ujo/lb-default.key
```

Verwalten von NSX-T Container Plug-in

6

Sie können NSX-T Container Plug-in über die NSX Manager-GUI oder über die Befehlszeilenschnittstelle (Command Line Interface, CLI) verwalten.

Hinweis Wenn eine Container-Host-VM auf ESXi 6.5 ausgeführt wird und die VM über vMotion auf einen anderen ESXi 6.5-Host migriert wird, geht die Verbindung von Containern, die auf dem Container-Host ausgeführt werden, mit Containern, die auf anderen Container-Hosts ausgeführt werden, verloren. Sie können das Problem lösen, indem Sie die vNIC des Container-Hosts trennen und erneut verbinden. Dieses Problem tritt bei ESXi 6.5 Update 1 oder höher nicht auf.

Hyperbus reserviert die VLAN-ID 4094 auf dem Hypervisor für die PVLAN-Konfiguration, und die ID kann nicht geändert werden. Um VLAN-Konflikte zu vermeiden, konfigurieren Sie logische VLAN-Switches oder VTEP-vmknics mit derselben VLAN-ID.

Dieses Kapitel enthält die folgenden Themen:

- [Verwalten von IP-Blöcken über die NSX Manager-GUI](#)
- [Anzeigen von IP-Block-Subnetzen über die GUI von NSX Manager](#)
- [CIF-verknüpfte logische Ports](#)
- [CLI-Befehle](#)
- [Fehlercodes](#)

Verwalten von IP-Blöcken über die NSX Manager -GUI

Sie können die Tags für einen IP-Block über die NSX Manager-GUI hinzufügen, löschen, bearbeiten, verwalten und die zugehörigen Details anzeigen.

Verfahren

- 1 Melden Sie sich über einen Browser bei NSX Manager unter `https://<nsx-manager-IP-address-or-domain-name>` an.
- 2 Navigieren Sie zu **Netzwerk > IPAM**.
Es wird eine Liste der vorhandenen IP-Blöcke angezeigt.

3 Führen Sie eine der folgenden Aktionen durch.

Option	Aktion
Hinzufügen eines IP-Blocks	Klicken Sie auf HINZUFÜGEN .
Löschen eines oder mehrerer IP-Blöcke	Wählen Sie einen oder mehrere IP-Blöcke aus, und klicken Sie auf LÖSCHEN .
Bearbeiten eines IP-Blocks	Wählen Sie einen IP-Block aus, und klicken Sie auf BEARBEITEN .
Anzeigen von Details zu einem IP-Block	Klicken Sie auf den Namen des IP-Blocks. Klicken Sie auf die Registerkarte Übersicht , um allgemeine Informationen anzuzeigen. Klicken Sie auf die Registerkarte Subnetze , um die Subnetze dieses IP-Blocks anzuzeigen.
Verwalten von Tags für einen IP-Block	Wählen Sie einen IP-Block aus, und klicken Sie auf AKTIONEN > Tags verwalten .

IP-Blöcke mit zugeteilten Subnetzen können nicht gelöscht werden.

Anzeigen von IP-Block-Subnetzen über die GUI von NSX Manager

Sie können Subnetze für einen IP-Block über die NSX Manager-GUI anzeigen. Es wird nicht empfohlen, IP-Block-Subnetze nach der Installation und Ausführung von NCP hinzuzufügen oder zu löschen.

Verfahren

- 1 Melden Sie sich über einen Browser bei NSX Manager unter `https://<nsx-manager-IP-address-or-domain-name>` an.
- 2 Navigieren Sie zu **Netzwerk > IPAM**.
Es wird eine Liste der vorhandenen IP-Blöcke angezeigt.
- 3 Klicken Sie auf den Namen eines IP-Blocks.
- 4 Klicken Sie auf die Registerkarte **Subnetze**.

CIF-verknüpfte logische Ports

CIFs (Container Interfaces) sind Netzwerkschnittstellen für Container, die mit logischen Ports auf einem Switch verbunden sind. Diese Ports werden als CIF-verknüpfte logische Ports bezeichnet.

Sie können CIF-verknüpfte logische Ports über die NSX Manager-GUI verwalten.

Verwalten von CIF-verknüpften logischen Ports

Navigieren Sie zu **Netzwerk > Switching > Ports**, um alle logischen Ports anzuzeigen, einschließlich der CIF-verknüpften logischen Ports. Klicken Sie auf die Anhangsverknüpfung eines CIF-verknüpften logischen Ports, um die Anlageninformationen anzuzeigen. Klicken Sie auf den Link des logischen Ports, um einen Fensterbereich mit vier Registerkarten zu öffnen: „Übersicht“, „Überwachen“, „Verwalten“ und „Zugehörig“. Wenn Sie auf **Zugehörig > Logische Ports** klicken, wird der zugehörige logische Port auf einem Uplink-Switch angezeigt. Weitere Informationen zu Switch-Ports finden Sie im *NSX-T-Administratorhandbuch*.

Netzwerküberwachungstools

Die folgenden Tools unterstützen CIF-verknüpfte logische Ports. Weitere Informationen zu diesen Tools finden Sie im *NSX-T-Administratorhandbuch*.

- Traceflow
- Portverbindung
- IPFIX
- Die Remote-Portspiegelung mit GRE-Kapselung eines logischen Switch-Ports, der mit einem Container verbunden ist, wird unterstützt. Weitere Informationen finden Sie unter „Grundlegendes zum Switching-Profil für die Portspiegelung“ im *NSX-T-Administratorhandbuch*. Die Portspiegelung des CIF-Ports zum VIF-Port wird jedoch über die Manager-Benutzeroberfläche nicht unterstützt.

CLI-Befehle

Um CLI-Befehle auszuführen, melden Sie sich beim NSX-T Container Plug-in-Container an, öffnen Sie ein Terminal, und führen Sie den Befehl `nsxcli` aus.

Sie können die CLI-Eingabeaufforderung auch aufrufen, indem Sie den folgenden Befehl für einen Knoten ausführen:

```
kubectl exec -it <pod name> nsxcli
```

Tabelle 6-1. CLI-Befehle für den NCP-Container

Typ	Befehl
Status	<code>get ncp-master status</code>
Status	<code>get ncp-nsx status</code>
Status	<code>get ncp-watcher <Watcher-Name></code>
Status	<code>get ncp-watchers</code>
Status	<code>get ncp-k8s-api-server status</code>
Status	<code>check projects</code>
Status	<code>check project <project-name></code>
Cache	<code>get project-cache <Projektname></code>
Cache	<code>get project-caches</code>
Cache	<code>get namespace-cache <Namensraum-Name></code>
Cache	<code>get namespace-caches</code>
Cache	<code>get pod-cache <Pod-Name></code>
Cache	<code>get pod-caches</code>
Cache	<code>get ingress-caches</code>
Cache	<code>get ingress-cache <ingress-name></code>

Tabelle 6-1. CLI-Befehle für den NCP-Container (Fortsetzung)

Typ	Befehl
Cache	get ingress-controllers
Cache	get ingress-controller <ingress-controller-name>
Cache	get network-policy-caches
Cache	get network-policy-cache <pod-name>
Support	get ncp-log file <Dateiname>
Support	get ncp-log-level
Support	set ncp-log-level <log-level>
Support	get support-bundle file <Dateiname>
Support	get node-agent-log file <Dateiname>
Support	get node-agent-log file <Dateiname> <Knotenname>

Tabelle 6-2. CLI-Befehle für den NSX-Knoten-Agent-Container

Typ	Befehl
Status	get node-agent-hyperbus status
Cache	get container-cache <Containername>
Cache	get container-caches

Tabelle 6-3. CLI-Befehle für den NSX-Kube-Proxy-Container

Typ	Befehl
Status	get ncp-k8s-api-server status
Status	get kube-proxy-watcher <Watcher-Name>
Status	get kube-proxy-watchers
Status	dump ovs-flows

Statusbefehle für den NCP-Container

- Status des NCP-Masters anzeigen

```
get ncp-master status
```

Beispiel:

```
kubenode> get ncp-master status
This instance is not the NCP master
Current NCP Master id is a4h83eh1-b8dd-4e74-c71c-cbb7cc9c4c1c
Last master update at Wed Oct 25 22:46:40 2017
```


- Verbindungsstatus zwischen NCP und NSX Manager anzeigen

```
get ncp-nsx status
```

Beispiel:

```
kubenode> get ncp-nsx status
NSX Manager status: Healthy
```

- Watcher-Status für eingehenden Datenverkehr, Namensraum, Pod und Dienst anzeigen

```
get ncp-watcher <watcher-name>
get ncp-watchers
```

Beispiel 1:

```
kubenode> get ncp-watcher pod
Average event processing time: 1174 msec (in past 3600-sec window)
Current watcher started time: Mar 02 2017 10:47:35 PST
Number of events processed: 1 (in past 3600-sec window)
Total events processed by current watcher: 1
Total events processed since watcher thread created: 1
Total watcher recycle count: 0
Watcher thread created time: Mar 02 2017 10:47:35 PST
Watcher thread status: Up
```

Beispiel 2:

```
kubenode> get ncp-watchers
pod:
Average event processing time: 1145 msec (in past 3600-sec window)
Current watcher started time: Mar 02 2017 10:51:37 PST
Number of events processed: 1 (in past 3600-sec window)
Total events processed by current watcher: 1
Total events processed since watcher thread created: 1
Total watcher recycle count: 0
Watcher thread created time: Mar 02 2017 10:51:37 PST
Watcher thread status: Up

namespace:
Average event processing time: 68 msec (in past 3600-sec window)
Current watcher started time: Mar 02 2017 10:51:37 PST
Number of events processed: 2 (in past 3600-sec window)
Total events processed by current watcher: 2
Total events processed since watcher thread created: 2
Total watcher recycle count: 0
Watcher thread created time: Mar 02 2017 10:51:37 PST
Watcher thread status: Up

ingress:
Average event processing time: 0 msec (in past 3600-sec window)
Current watcher started time: Mar 02 2017 10:51:37 PST
```

```

Number of events processed: 0 (in past 3600-sec window)
Total events processed by current watcher: 0
Total events processed since watcher thread created: 0
Total watcher recycle count: 0
Watcher thread created time: Mar 02 2017 10:51:37 PST
Watcher thread status: Up

```

service:

```

Average event processing time: 3 msec (in past 3600-sec window)
Current watcher started time: Mar 02 2017 10:51:37 PST
Number of events processed: 1 (in past 3600-sec window)
Total events processed by current watcher: 1
Total events processed since watcher thread created: 1
Total watcher recycle count: 0
Watcher thread created time: Mar 02 2017 10:51:37 PST
Watcher thread status: Up

```

- Verbindungsstatus zwischen NCP und Kubernetes-API-Server anzeigen

```
get ncp-k8s-api-server status
```

Beispiel:

```

kubecall> get ncp-k8s-api-server status
Kubernetes ApiServer status: Healthy

```

- Alle Projekte oder ein bestimmtes Projekt überprüfen

```

check projects
check project <project-name>

```

Beispiel:

```

kubecall> check projects
default:
  Tier-1 link port for router 1b90a61f-0f2c-4768-9eb6-ea8954b4f327 is missing
  Switch 40a6829d-c3aa-4e17-ae8a-7f7910fdf2c6 is missing

ns1:
  Router 8accc9cd-9883-45f6-81b3-0d1fb2583180 is missing

kubecall> check project default
Tier-1 link port for router 1b90a61f-0f2c-4768-9eb6-ea8954b4f327 is missing
Switch 40a6829d-c3aa-4e17-ae8a-7f7910fdf2c6 is missing

```

Cachebefehle für den NCP-Container

- Internen Cache für Projekte oder Namensräume abrufen

```
get project-cache <project-name>
get project-caches
get namespace-cache <namespace-name>
get namespace-caches
```

Beispiel:

```
kubenode> get project-caches
default:
  logical-router: 8accc9cd-9883-45f6-81b3-0d1fb2583180
  logical-switch:
    id: 9d7da647-27b6-47cf-9cdb-6e4f4d5a356d
    ip_pool_id: 519ff57f-061f-4009-8d92-3e6526e7c17e
    subnet: 10.0.0.0/24
    subnet_id: f75fd64c-c7b0-4b42-9681-fc656ae5e435

kube-system:
  logical-router: 5032b299-acad-448e-a521-19d272a08c46
  logical-switch:
    id: 85233651-602d-445d-ab10-1c84096cc22a
    ip_pool_id: ab1c5b09-7004-4206-ac56-85d9d94bffa2
    subnet: 10.0.1.0/24
    subnet_id: 73e450af-b4b8-4a61-a6e3-c7ddd15ce751

testns:
  ext_pool_id: 346a0f36-7b5a-4ecc-ad32-338dcb92316f
  labels:
    ns: myns
    project: myproject
  logical-router: 4dc8f8a9-69b4-4ff7-8fb7-d2625dc77efa
  logical-switch:
    id: 6111a99a-6e06-4faa-a131-649f10f7c815
    ip_pool_id: 51ca058d-c3dc-41fd-8f2d-e69006ab1b3d
    subnet: 50.0.2.0/24
    subnet_id: 34f79811-bd29-4048-a67d-67ceac97eb98
  project_nsgroup: 9606afee-6348-4780-9dbe-91abfd23e475
  snat_ip: 4.4.0.3

kubenode> get project-cache default
logical-router: 8accc9cd-9883-45f6-81b3-0d1fb2583180
logical-switch:
  id: 9d7da647-27b6-47cf-9cdb-6e4f4d5a356d
  ip_pool_id: 519ff57f-061f-4009-8d92-3e6526e7c17e
  subnet: 10.0.0.0/24
  subnet_id: f75fd64c-c7b0-4b42-9681-fc656ae5e435

kubenode> get namespace-caches
default:
  logical-router: 8accc9cd-9883-45f6-81b3-0d1fb2583180
```

```

logical-switch:
  id: 9d7da647-27b6-47cf-9cdb-6e4f4d5a356d
  ip_pool_id: 519ff57f-061f-4009-8d92-3e6526e7c17e
  subnet: 10.0.0.0/24
  subnet_id: f75fd64c-c7b0-4b42-9681-fc656ae5e435

kube-system:
  logical-router: 5032b299-acad-448e-a521-19d272a08c46
  logical-switch:
    id: 85233651-602d-445d-ab10-1c84096cc22a
    ip_pool_id: ab1c5b09-7004-4206-ac56-85d9d94bffa2
    subnet: 10.0.1.0/24
    subnet_id: 73e450af-b4b8-4a61-a6e3-c7ddd15ce751

testns:
  ext_pool_id: 346a0f36-7b5a-4ecc-ad32-338dcb92316f
  labels:
    ns: myns
    project: myproject
  logical-router: 4dc8f8a9-69b4-4ff7-8fb7-d2625dc77efa
  logical-switch:
    id: 6111a99a-6e06-4faa-a131-649f10f7c815
    ip_pool_id: 51ca058d-c3dc-41fd-8f2d-e69006ab1b3d
    subnet: 50.0.2.0/24
    subnet_id: 34f79811-bd29-4048-a67d-67ceac97eb98
  project_nsgroup: 9606afee-6348-4780-9dbe-91abfd23e475
  snat_ip: 4.4.0.3

kubenode> get namespace-cache default
logical-router: 8accc9cd-9883-45f6-81b3-0d1fb2583180
logical-switch:
  id: 9d7da647-27b6-47cf-9cdb-6e4f4d5a356d
  ip_pool_id: 519ff57f-061f-4009-8d92-3e6526e7c17e
  subnet: 10.0.0.0/24
  subnet_id: f75fd64c-c7b0-4b42-9681-fc656ae5e435

```

■ Internen Cache für Pods abrufen

```

get pod-cache <pod-name>
get pod-caches

```

Beispiel:

```

kubenode> get pod-caches
nsx.default.nginx-rc-uq2lv:
  cif_id: 2af9f734-37b1-4072-ba88-abbf935bf3d4
  gateway_ip: 10.0.0.1
  host_vif: d6210773-5c07-4817-98db-451bd1f01937
  id: 1c8b5c52-3795-11e8-ab42-005056b198fb
  ingress_controller: False
  ip: 10.0.0.2/24
  labels:
    app: nginx
  mac: 02:50:56:00:08:00

```

```

    port_id: d52c833a-f531-4bdf-bfa2-e8a084a8d41b
    vlan: 1

nsx.testns.web-pod-1:
  cif_id: ce134f21-6be5-43fe-afbf-aaca8c06b5cf
  gateway_ip: 50.0.2.1
  host_vif: d6210773-5c07-4817-98db-451bd1f01937
  id: 3180b521-270e-11e8-ab42-005056b198fb
  ingress_controller: False
  ip: 50.0.2.3/24
  labels:
    app: nginx-new
    role: db
    tier: cache
  mac: 02:50:56:00:20:02
  port_id: 81bc2b8e-d902-4cad-9fc1-aabdc32ecaf8
  vlan: 3

kubenode> get pod-cache nsx.default.nginx-rc-uj2lv
  cif_id: 2af9f734-37b1-4072-ba88-abbf935bf3d4
  gateway_ip: 10.0.0.1
  host_vif: d6210773-5c07-4817-98db-451bd1f01937
  id: 1c8b5c52-3795-11e8-ab42-005056b198fb
  ingress_controller: False
  ip: 10.0.0.2/24
  labels:
    app: nginx
  mac: 02:50:56:00:08:00
  port_id: d52c833a-f531-4bdf-bfa2-e8a084a8d41b
  vlan: 1

```

- Netzwerkrichtlinien-Caches oder einen bestimmten Netzwerkrichtlinien-Cache abrufen

```

get network-policy caches
get network-policy-cache <network-policy-name>

```

Beispiel:

```

kubenode> get network-policy-caches
nsx.testns.allow-tcp-80:
  dest_labels: None
  dest_pods:
    50.0.2.3
  match_expressions:
    key: tier
    operator: In
    values:
      cache
  name: allow-tcp-80
  np_dest_ip_set_ids:
    22f82d76-004f-4d12-9504-ce1cb9c8aa00
  np_except_ip_set_ids:
  np_ip_set_ids:
    14f7f825-f1a0-408f-bbd9-bb2f75d44666

```

```

np_isol_section_id: c8d93597-9066-42e3-991c-c550c46b2270
np_section_id: 04693136-7925-44f2-8616-d809d02cd2a9
ns_name: testns
src_egress_rules: None
src_egress_rules_hash: 97d170e1550eee4afc0af065b78cda302a97674c
src_pods:
  50.0.2.0/24
src_rules:
  from:
    namespaceSelector:
      matchExpressions:
        key: tier
        operator: DoesNotExist
      matchLabels:
        ns: myns
    ports:
      port: 80
      protocol: TCP
src_rules_hash: e4ea7b8d91c1e722670a59f971f8fcc1a5ac51f1

```

```

kubenset> get network-policy-cache nsx.testns.allow-tcp-80
dest_labels: None
dest_pods:
  50.0.2.3
match_expressions:
  key: tier
  operator: In
  values:
    cache
name: allow-tcp-80
np_dest_ip_set_ids:
  22f82d76-004f-4d12-9504-ce1cb9c8aa00
np_except_ip_set_ids:
np_ip_set_ids:
  14f7f825-f1a0-408f-bbd9-bb2f75d44666
np_isol_section_id: c8d93597-9066-42e3-991c-c550c46b2270
np_section_id: 04693136-7925-44f2-8616-d809d02cd2a9
ns_name: testns
src_egress_rules: None
src_egress_rules_hash: 97d170e1550eee4afc0af065b78cda302a97674c
src_pods:
  50.0.2.0/24
src_rules:
  from:
    namespaceSelector:
      matchExpressions:
        key: tier
        operator: DoesNotExist
      matchLabels:
        ns: myns

```

```
ports:
  port: 80
  protocol: TCP
src_rules_hash: e4ea7b8d91c1e722670a59f971f8fcc1a5ac51f1
```

Supportbefehle für den NCP-Container

■ NCP-Support-Paket im Dateispeicher speichern

Das Support-Paket umfasst die Protokolldateien für alle Container in Pods mit der Bezeichnung **tier:nsx-networking**. Die Paketdatei liegt im TGZ-Format vor und befindet sich im CLI-Standard-Dateispeicherverzeichnis `/var/vmware/nsx/file-store`. Mithilfe des CLI-Befehls „file-store“ können Sie die Paketdatei in eine Remote-Site kopieren.

```
get support-bundle file <filename>
```

Beispiel:

```
kubenode>get support-bundle file foo
Bundle file foo created in tgz format
kubenode>copy file foo url scp://nicira@10.0.0.1:/tmp
```

■ NCP-Protokolle im Dateispeicher speichern

Die Protokolldatei wird im TGZ-Format im CLI-Standard-Dateispeicherverzeichnis `/var/vmware/nsx/file-store` gespeichert. Mithilfe des CLI-Befehls „file-store“ können Sie die Paketdatei in eine Remote-Site kopieren.

```
get ncp-log file <filename>
```

Beispiel:

```
kubenode>get ncp-log file foo
Log file foo created in tgz format
```

■ Knoten-Agent-Protokolle im Dateispeicher speichern

Speichern Sie die Knoten-Agent-Protokolle von einem oder allen Knoten. Die Protokolle werden im TGZ-Format im CLI-Standard-Dateispeicherverzeichnis `/var/vmware/nsx/file-store` gespeichert. Mithilfe des CLI-Befehls „file-store“ können Sie die Paketdatei in eine Remote-Site kopieren.

```
get node-agent-log file <filename>
get node-agent-log file <filename> <node-name>
```

Beispiel:

```
kubenode>get node-agent-log file foo
Log file foo created in tgz format
```

- Protokollebene abrufen und einrichten

Zu den verfügbaren Protokollebenen gehören: NOTSET, DEBUG, INFO, WARNING, ERROR und CRITICAL.

```
get ncp-log-level
set ncp-log-level <log level>
```

Beispiel:

```
kubenode>get ncp-log-level
NCP log level is INFO

kubenode>set ncp-log-level DEBUG
NCP log level is changed to DEBUG
```

Status-Befehle für den NSX-Knoten-Agent-Container

- Zeigen Sie den Verbindungsstatus zwischen dem Knoten-Agent und HyperBus auf diesem Knoten an.

```
get node-agent-hyperbus status
```

Beispiel:

```
kubenode> get node-agent-hyperbus status
HyperBus status: Healthy
```

Cache-Befehle für den NSX-Knoten-Agent-Container

- Internen Cache für NSX-Knoten-Agent-Container abrufen

```
get container-cache <container-name>
get container-caches
```

Beispiel 1:

```
kubenode> get container-cache cif104
ip: 192.168.0.14/32
mac: 50:01:01:01:01:14
gateway_ip: 169.254.1.254/16
vlan_id: 104
```


Beispiel 2:

```
kubenode> get container-caches
cif104:
  ip: 192.168.0.14/32
  mac: 50:01:01:01:01:14
  gateway_ip: 169.254.1.254/16
  vlan_id: 104
```

Status-Befehle für den NSX-Kube-Proxy-Container

- Verbindungsstatus zwischen Kube-Proxy und Kubernetes-API-Server anzeigen

```
get ncp-k8s-api-server status
```

Beispiel:

```
kubenode> get kube-proxy-k8s-api-server status
Kubernetes ApiServer status: Healthy
```

- Kube-Proxy-Watcher-Status anzeigen

```
get kube-proxy-watcher <watcher-name>
get kube-proxy-watchers
```

Beispiel 1:

```
kubenode> get kube-proxy-watcher endpoint
Average event processing time: 15 msec (in past 3600-sec window)
Current watcher started time: May 01 2017 15:06:24 PDT
Number of events processed: 90 (in past 3600-sec window)
Total events processed by current watcher: 90
Total events processed since watcher thread created: 90
Total watcher recycle count: 0
Watcher thread created time: May 01 2017 15:06:24 PDT
Watcher thread status: Up
```

Beispiel 2:

```
kubenode> get kube-proxy-watchers
endpoint:
  Average event processing time: 15 msec (in past 3600-sec window)
  Current watcher started time: May 01 2017 15:06:24 PDT
  Number of events processed: 90 (in past 3600-sec window)
  Total events processed by current watcher: 90
  Total events processed since watcher thread created: 90
  Total watcher recycle count: 0
  Watcher thread created time: May 01 2017 15:06:24 PDT
  Watcher thread status: Up

service:
```

```

Average event processing time: 8 msec (in past 3600-sec window)
Current watcher started time: May 01 2017 15:06:24 PDT
Number of events processed: 2 (in past 3600-sec window)
Total events processed by current watcher: 2
Total events processed since watcher thread created: 2
Total watcher recycle count: 0
Watcher thread created time: May 01 2017 15:06:24 PDT
Watcher thread status: Up

```

■ OVS-Flows für Speicherabbild an einem Knoten

```
dump ovs-flows
```

Beispiel:

```

kubenode> dump ovs-flows
NXST_FLOW reply (xid=0x4):
  cookie=0x0, duration=8.876s, table=0, n_packets=0, n_bytes=0, idle_age=8, priority=100,ip ac-
  tions=ct(table=1)
    cookie=0x0, duration=8.898s, table=0, n_packets=0, n_bytes=0, idle_age=8, priority=0 ac-
    tions=NORMAL
      cookie=0x0, duration=8.759s, table=1, n_packets=0, n_bytes=0, idle_age=8, priori-
      ty=100,tcp,nw_dst=10.96.0.1,tp_dst=443 actions=mod_tp_dst:443
        cookie=0x0, duration=8.719s, table=1, n_packets=0, n_bytes=0, idle_age=8, priori-
        ty=100,ip,nw_dst=10.96.0.10 actions=drop
          cookie=0x0, duration=8.819s, table=1, n_packets=0, n_bytes=0, idle_age=8, priori-
          ty=90,ip,in_port=1 actions=ct(table=2,nat)
            cookie=0x0, duration=8.799s, table=1, n_packets=0, n_bytes=0, idle_age=8, priority=80,ip ac-
            tions=NORMAL
              cookie=0x0, duration=8.856s, table=2, n_packets=0, n_bytes=0, idle_age=8, actions=NORMAL

```

Fehlercodes

In diesem Abschnitt werden die Fehlercodes der verschiedenen Komponenten aufgelistet.

NCP-Fehlercodes

Fehlercode	Beschreibung
NCP00001	Ungültige Konfiguration
NCP00002	Initialisierung fehlgeschlagen
NCP00003	Ungültiger Zustand
NCP00004	Ungültiger Adapter
NCP00005	Zertifikat wurde nicht gefunden
NCP00006	Token wurde nicht gefunden
NCP00007	Ungültige Konfiguration für NSX
NCP00008	Ungültiges NSX-Tag
NCP00009	NSX-Verbindung fehlgeschlagen

Fehlercode	Beschreibung
NCP00010	Knoten-Tag nicht gefunden
NCP00011	Ungültiger logischer Switch-Port des Knotens
NCP00012	Aktualisieren der übergeordneten VIF fehlgeschlagen
NCP00013	VLAN ausgeschöpft
NCP00014	VLAN-Version ist fehlgeschlagen
NCP00015	IP-Pool ist ausgeschöpft
NCP00016	IP-Version ist fehlgeschlagen
NCP00017	IP-Block ausgeschöpft
NCP00018	Erstellen des IP-Subnetzes ist fehlgeschlagen
NCP00019	Löschen des IP-Subnetzes ist fehlgeschlagen
NCP00020	Erstellen des IP-Pools ist fehlgeschlagen
NCP00021	Löschen des IP-Pools ist fehlgeschlagen
NCP00022	Erstellen des logischen Routers ist fehlgeschlagen
NCP00023	Aktualisieren des logischen Routers ist fehlgeschlagen
NCP00024	Löschen des logischen Routers ist fehlgeschlagen
NCP00025	Erstellen des logischen Switches ist fehlgeschlagen

Fehlercode	Beschreibung
NCP00026	Aktualisieren des logischen Switches ist fehlgeschlagen
NCP00027	Löschen des logischen Switches ist fehlgeschlagen
NCP00028	Erstellen des Ports für den logischen Router ist fehlgeschlagen
NCP00029	Löschen des Ports für den logischen Router ist fehlgeschlagen
NCP00030	Erstellen des Ports für den logischen Switch ist fehlgeschlagen
NCP00031	Aktualisieren des Ports für den logischen Switch ist fehlgeschlagen
NCP00032	Löschen des Ports für den logischen Switch ist fehlgeschlagen
NCP00033	Netzwerkrichtlinie wurde nicht gefunden
NCP00034	Erstellen der Firewall ist fehlgeschlagen
NCP00035	Lesen der Firewall ist fehlgeschlagen
NCP00036	Aktualisieren der Firewall ist fehlgeschlagen
NCP00037	Löschen der Firewall ist fehlgeschlagen
NCP00038	Mehrere Firewalls gefunden
NCP00039	Erstellen der NS-Gruppe ist fehlgeschlagen
NCP00040	Löschen der NS-Gruppe ist fehlgeschlagen
NCP00041	Erstellen von IP Set ist fehlgeschlagen
NCP00042	Aktualisieren von IP Set ist fehlgeschlagen

Fehlercode	Beschreibung
NCP00043	Löschen von IP Set ist fehlgeschlagen
NCP00044	Erstellen der SNAT-Regel ist fehlgeschlagen
NCP00045	Löschen der SNAT-Regel ist fehlgeschlagen
NCP00046	Adapter-API-Verbindung ist fehlgeschlagen
NCP00047	Adapter-Watcher-Ausnahme
NCP00048	Löschen des Load Balancer-Diensts ist fehlgeschlagen
NCP00049	Erstellen des virtuellen Servers für den Load Balancer ist fehlgeschlagen
NCP00050	Aktualisieren des virtuellen Servers für den Load Balancer ist fehlgeschlagen

Fehlercode	Beschreibung
NCP00051	Löschen des virtuellen Servers für den Load Balancer ist fehlgeschlagen
NCP00052	Erstellen des Load Balancer-Pools ist fehlgeschlagen
NCP00053	Aktualisieren des Load Balancer-Pools ist fehlgeschlagen
NCP00054	Löschen des Load Balancer-Pools ist fehlgeschlagen
NCP00055	Erstellen der Load Balancer-Regel ist fehlgeschlagen
NCP00056	Aktualisieren des Load Balancer-Pools ist fehlgeschlagen
NCP00057	Löschen der Load Balancer-Regel ist fehlgeschlagen
NCP00058	IP-Freigabe für Load Balancer-Pool ist fehlgeschlagen
NCP00059	Zuordnung von virtuellem Server und Dienstzuordnung für Load Balancer nicht gefunden
NCP00060	Aktualisieren der NS-Gruppe ist fehlgeschlagen
NCP00061	Abrufen der Firewallregeln ist fehlgeschlagen
NCP00062	NS-Gruppe – keine Kriterien
NCP00063	Knoten-VM nicht gefunden
NCP00064	Knoten-VIF nicht gefunden
NCP00065	Zertifikatimport ist fehlgeschlagen
NCP00066	Rückgängigmachen des Zertifikats ist fehlgeschlagen
NCP00067	Aktualisieren der SSL-Bindung ist fehlgeschlagen
NCP00068	SSL-Profil nicht gefunden
NCP00069	IP-Pool nicht gefunden
NCP00070	T0-Edge-Cluster nicht gefunden
NCP00071	Aktualisieren des IP-Pools ist fehlgeschlagen
NCP00072	Dispatcher ist fehlgeschlagen
NCP00073	Löschen der NAT-Regel ist fehlgeschlagen
NCP00074	Abrufen des Ports für den logischen Router ist fehlgeschlagen
NCP00075	NSX-Konfigurationsvalidierung ist fehlgeschlagen

Fehlercode	Beschreibung
NCP00076	Aktualisieren der SNAT-Regel ist fehlgeschlagen
NCP00077	SNAT-Regel überlagert
NCP00078	Hinzufügen der Load Balancer-Endpoints ist fehlgeschlagen
NCP00079	Aktualisieren der Load Balancer-Endpoints ist fehlgeschlagen
NCP00080	Erstellen des Load Balancer-Regelpools ist fehlgeschlagen
NCP00081	Virtueller Server für Load Balancer nicht gefunden
NCP00082	Lesen von IP Set ist fehlgeschlagen
NCP00083	Abrufen von SNAT-Pool ist fehlgeschlagen
NCP00084	Erstellen des Load Balancer-Diensts ist fehlgeschlagen
NCP00085	Aktualisieren des Load Balancer-Diensts ist fehlgeschlagen
NCP00086	Aktualisieren des Ports für den logischen Router ist fehlgeschlagen
NCP00087	Load Balancer-Initialisierung ist fehlgeschlagen
NCP00088	IP-Pool nicht eindeutig
NCP00089	Layer 7 des Load Balancers – Cache-Synchronisierungsfehler
NCP00090	Fehler, da Load Balancer-Pool nicht vorhanden
NCP00091	Fehler beim Initialisieren des Load Balancer-Regelcaches
NCP00092	SNAT-Prozess ist fehlgeschlagen
NCP00093	Fehler bei Load Balancer-Standardzertifikat
NCP00094	Löschen des Load Balancer-Endpoints ist fehlgeschlagen
NCP00095	Projekt nicht gefunden
NCP00096	Pool-Zugriff verweigert
NCP00097	Fehler beim Abrufen eines Load Balancer-Diensts
NCP00098	Fehler beim Erstellen eines Load Balancer-Diensts
NCP00099	Fehler bei Synchronisierung des Load Balancer-Pool-Caches

Fehlercodes für NSX-Knoten-Agent

Fehlercode	Beschreibung
NCP01001	OVS-Uplink nicht gefunden
NCP01002	Host-MAC nicht gefunden
NCP01003	OVS-Porterstellung ist fehlgeschlagen
NCP01004	Keine Pod-Konfiguration
NCP01005	Pod-Konfiguration ist fehlgeschlagen
NCP01006	Aufheben der Pod-Konfiguration ist fehlgeschlagen
NCP01007	CNI-Socket nicht gefunden

Fehlercode	Beschreibung
NCP01008	CNI-Verbindung ist fehlgeschlagen
NCP01009	CNI-Version stimmt nicht überein
NCP01010	CNI-Nachrichtenempfang ist fehlgeschlagen
NCP01011	CNI-Nachrichtenübertragung ist fehlgeschlagen
NCP01012	Hyperbus-Verbindung ist fehlgeschlagen
NCP01013	Hyperbus-Version stimmt nicht überein
NCP01014	Fehler beim Empfang der Hyperbus-Nachricht
NCP01015	Hyperbus-Nachrichtenübertragung ist fehlgeschlagen
NCP01016	GARP-Senden ist fehlgeschlagen
NCP01017	Schnittstellenkonfiguration ist fehlgeschlagen

Fehlercodes für nsx-kube-proxy

Fehlercode	Beschreibung
NCP02001	Ungültiger Gateway-Port des Proxys
NCP02002	Proxy-Befehl ist fehlgeschlagen
NCP02003	Proxy-Validierung ist fehlgeschlagen

CLI-Fehlercodes

Fehlercode	Beschreibung
NCP03001	CLI-Start ist fehlgeschlagen
NCP03002	Erstellen des CLI-Sockets ist fehlgeschlagen
NCP03003	CLI-Socket-Ausnahme
NCP03004	Ungültige Anforderung von CLI-Client
NCP03005	CLI-Server-Übertragung ist fehlgeschlagen
NCP03006	CLI-Server-Empfang ist fehlgeschlagen
NCP03007	CLI-Befehlsausführung ist fehlgeschlagen

Fehlercodes für Kubernetes

Fehlercode	Beschreibung
NCP05001	Kubernetes-Verbindung ist fehlgeschlagen
NCP05002	Ungültige Konfiguration für Kubernetes
NCP05003	Kubernetes-Anforderung ist fehlgeschlagen
NCP05004	Kubernetes-Schlüssel nicht gefunden

Fehlercode	Beschreibung
NCP05005	Kubernetes-Typ nicht gefunden
NCP05006	Ausnahme bei Kubernetes-Wächter
NCP05007	Kubernetes-Ressource weist ungültige Länge auf
NCP05008	Kubernetes-Ressource weist ungültigen Typ auf
NCP05009	Kubernetes-Ressourcen-Handle ist fehlgeschlagen
NCP05010	Kubernetes-Dienst-Handle ist fehlgeschlagen
NCP05011	Kubernetes-Endpoint-Handle ist fehlgeschlagen
NCP05012	Kubernetes-Ingress-Handle ist fehlgeschlagen
NCP05013	Kubernetes-Netzwerkrichtlinien-Handle ist fehlgeschlagen
NCP05014	Kubernetes-Knoten-Handle ist fehlgeschlagen
NCP05015	Kubernetes-Namespace-Handle ist fehlgeschlagen
NCP05016	Kubernetes-Pod-Handle ist fehlgeschlagen
NCP05017	Kubernetes-Secret-Handle ist fehlgeschlagen
NCP05018	Kubernetes-Standard-Backend ist fehlgeschlagen
NCP05019	Nicht unterstützter Übereinstimmungsausdruck für Kubernetes
NCP05020	Aktualisieren des Kubernetes-Status ist fehlgeschlagen
NCP05021	Aktualisieren des Kubernetes-Kommentars ist fehlgeschlagen
NCP05022	Kubernetes-Namespace-Cache nicht gefunden
NCP05023	Kubernetes-Secret nicht gefunden
NCP05024	Kubernetes-Standard-Backend wird verwendet
NCP05025	Kubernetes-LoadBalancer-Dienst-Handle ist fehlgeschlagen

Fehlercodes für OpenShift

Fehlercode	Beschreibung
NCP07001	OC-Routen-Handle ist fehlgeschlagen
NCP07002	Aktualisieren des OC-Routenstatus ist fehlgeschlagen