

NSX Container Plug-in pour OpenShift - Guide d'installation et d'administration

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®

Vous trouverez la documentation technique la plus récente sur le site Web de VMware, à l'adresse :

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

Les dernières mises à jour produit se trouvent également sur le site Web de VMware.

Si vous avez des commentaires à propos de cette documentation, envoyez-les à l'adresse suivante :

docfeedback@vmware.com

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

VMware France SAS.
Tour Franklin
100-101 Terrasse Boieldieu
92042 Paris La Défense 8 Cedex
France
www.vmware.com/fr

Copyright © 2017–2019 VMware, Inc. Tous droits réservés. [Informations sur le copyright et les marques commerciales.](#)

Table des matières

NSX-T Container Plug-in pour OpenShift - Guide d'installation et d'administration 4

1 Présentation de NSX-T Container Plug-in 5

Conditions de compatibilité 6

Présentation de l'installation 6

Mise à niveau de NCP 7

2 Configuration des ressources de NSX-T 8

Configuration des ressources de NSX-T 8

Créer et configurer un routeur logique de niveau 0 11

3 Installation de NCP dans un environnement OpenShift 13

Déployer des machines virtuelles OpenShift 13

Préparer le fichier hosts Ansible 13

Installer NCP et OpenShift à l'aide d'un playbook unique 16

Installer le plug-in CNI, OVS et l'image Docker NCP 16

Installer OpenShift Container Platform 19

Exécutez NCP et l'agent de nœud NSX 19

Notes de configuration 21

4 Installation de NCP dans un environnement Bare-Metal 25

Installer le plug-in CNI NSX-T Data Center 25

Configurer la mise en réseau de centre de données NSX-T pour les nœuds OpenShift 25

Installer l'agent de nœud NSX 26

ConfigMap pour ncp.ini dans nsx-node-agent-ds.yml 27

Installer le plug-in de conteneur NSX-T 30

ConfigMap pour ncp.ini dans ncp-rc.yml 32

5 Équilibrage de charge 38

Configuration de l'équilibrage de charge 38

6 Administration de NSX-T Container Plug-in 45

Gérer les blocs d'adresses IP à partir de l'interface utilisateur de NSX Manager 45

Voir les sous-réseaux de bloc d'adresses IP à partir de l'interface utilisateur graphique de NSX Manager 46

Ports logiques attachés par CIF 46

Commandes d'interface de ligne de commande 47

Codes d'erreur 58

NSX-T Container Plug-in pour OpenShift - Guide d'installation et d'administration

Ce guide décrit comment installer et administrer NSX-T Container Plug-in (NCP) pour fournir l'intégration entre NSX-T Data Center et OpenShift.

Public visé

Ce guide est destiné aux administrateurs système et réseau. Il est impératif de connaître les procédures d'installation et d'administration de NSX-T Data Center et d'OpenShift.

Glossaire VMware Technical Publications

VMware Technical Publications fournit un glossaire de termes pouvant ne pas vous être familiers. Pour consulter la définition des termes utilisés dans la documentation technique VMware, visitez le site Web <http://www.vmware.com/support/pubs>.

Présentation de NSX-T Container Plug-in

1

NSX-T Container Plug-in (NCP) fournit l'intégration entre NSX-T Data Center et les orchestrateurs de conteneur, tels que Kubernetes, ainsi que l'intégration entre NSX-T Data Center et les logiciels PaaS (plate-forme en tant que service) basés sur un conteneur, tels qu'OpenShift. Ce guide décrit comment configurer NCP avec OpenShift.

Le composant principal de NCP s'exécute dans un conteneur et communique avec NSX Manager et avec le plan de contrôle OpenShift. NCP surveille les modifications apportées aux conteneurs et d'autres ressources et gère des ressources réseau, telles que des ports logiques, des commutateurs, des routeurs et des groupes de sécurité pour les conteneurs en appelant NSX API.

Le plug-in NSX CNI s'exécute sur chaque nœud OpenShift. Il surveille les événements de cycle de vie des conteneurs, connecte une interface de conteneur au vSwitch invité et programme le vSwitch invité pour marquer et transférer le trafic de conteneur entre les interfaces de conteneur et la carte réseau virtuelle.

NCP fournit les fonctionnalités suivantes :

- Création automatique d'une topologie logique NSX-T pour un cluster OpenShift et création d'un réseau logique distinct pour chaque espace de noms OpenShift.
- Connexion des espaces OpenShift au réseau logique et allocation des adresses IP et MAC.
- Prise en charge de la traduction d'adresse réseau (NAT) et allocation d'une adresse IP SNAT distincte pour chaque espace de noms OpenShift.

Note Lors de la configuration de NAT, le nombre total d'adresses IP traduites ne peut pas dépasser 1 000.

- Implémentation de stratégies réseau OpenShift avec pare-feu distribué NSX-T.
 - Prise en charge des stratégies réseau d'entrée et de sortie.
 - Prise en charge de sélecteur d'IPBlock dans les stratégies réseau.
 - Prise en charge de `matchLabels` et de `matchExpression` lors de la spécification de sélecteurs d'étiquette pour les stratégies réseau.
- Implémentation de route OpenShift avec un équilibrage de charge de couche 7 NSX-T.
 - Prise en charge des routes HTTP et HTTPS avec terminaison Edge TLS.

- Prise en charge des routes avec d'autres serveurs principaux et sous-domaines génériques.
- Création de balises sur le port de commutateur logique NSX-T pour l'espace de noms, le nom de l'espace et les étiquettes d'un espace, et définition par l'administrateur de groupes de sécurité et de stratégies NSX-T Data Center basées sur les balises.

Dans cette version, NCP prend en charge un seul cluster OpenShift.

Ce chapitre contient les rubriques suivantes :

- [Conditions de compatibilité](#)
- [Présentation de l'installation](#)
- [Mise à niveau de NCP](#)

Conditions de compatibilité

NSX-T Container Plug-in (NCP) présente les conditions de compatibilité suivantes.

Produit logiciel	Version
NSX-T Data Center	2.2, 2.3
Hyperviseur pour machines virtuelles d'hôte de conteneur	<ul style="list-style-type: none"> ■ Version de vSphere prise en charge ■ RHEL KVM 7.4, 7.5
Système d'exploitation d'hôte de conteneur	RHEL 7.4, 7.5
Plateforme sous forme de service	OpenShift 3.9, 3.10
Open vSwitch d'hôte de conteneur	2.9.1 (fourni avec NSX-T 2.3 et 2.2)

Présentation de l'installation

L'installation et la configuration de NCP impliquent les étapes suivantes. Pour effectuer la procédure correctement, vous devez être familiarisé avec l'installation et l'administration de NSX-T Data Center et d'OpenShift.

- 1 Installez NSX-T Data Center.
- 2 Créez une zone de transport de superposition.
- 3 Créez un commutateur logique de superposition et connectez les nœuds au commutateur.
- 4 Créez un routeur logique de niveau 0.
- 5 Créez des blocs d'adresses IP pour les espaces.
- 6 Créez des pools d'adresses IP pour SNAT (Source Network Address Translation, traduction d'adresse réseau source).
- 7 Déployez des machines virtuelles OpenShift.
- 8 Préparez le fichier hosts Ansible.

9 (Option 1) Installez NCP et OpenShift à l'aide d'un playbook unique.

(Option 2) Installez le plug-in CNI, OVS (Open vSwitch) et l'image Docker NCP. Installez ensuite OpenShift Container Platform.

10 Exécutez NCP et l'agent de nœud NSX.

Les étapes 2 à 6 ne sont pas nécessaires si vous installez NCP à l'aide des playbooks fournis. Reportez-vous aux sections [Installer NCP et OpenShift à l'aide d'un playbook unique](#) et [Installer le plug-in CNI, OVS et l'image Docker NCP](#).

Mise à niveau de NCP

Pour mettre à niveau NCP vers la version 2.3.0, procédez comme suit.

1 Mettez à niveau le module RPM CNI, le modèle DaemonSet de l'agent de nœud NSX et ReplicationController de NCP.

2 (Facultatif) Mettez à niveau NSX-T Data Center vers la version 2.3.

NCP 2.3.0 prend en charge NSX-T 2.2, mais vous pouvez également mettre à niveau vers NSX-T Data Center la version 2.3.

Configuration des ressources de NSX-T

2

Des ressources de NSX-T Data Center doivent être créées pour fournir la mise en réseau aux nœuds OpenShift. Vous pouvez configurer ces ressources manuellement en utilisant l'interface utilisateur de NSX Manager ou automatiser le processus à l'aide d'un playbook Ansible.

Cette section décrit la création manuelle des ressources de NSX-T. Pour automatiser le processus, reportez-vous à la section [Installer le plug-in CNI, OVS et l'image Docker NCP](#).

Ce chapitre contient les rubriques suivantes :

- [Configuration des ressources de NSX-T](#)
- [Créer et configurer un routeur logique de niveau 0](#)

Configuration des ressources de NSX-T

Les ressources de NSX-T Data Center que vous devez configurer incluent une zone de transport de superposition, un routeur logique de niveau 0, un commutateur logique pour connecter les machines virtuelles du nœud, des blocs d'adresses IP pour les nœuds Kubernetes et un pool d'adresses IP pour SNAT.

Vous configurez des ressources NSX-T Data Center à l'aide d'UUID ou de noms dans le fichier de configuration `ncp.ini`.

Zone de transport de superposition

Connectez-vous à NSX Manager et accédez à **Infrastructure > Zones de transport**. Recherchez la zone de transport de superposition utilisée pour la mise en réseau de conteneur ou créez-en une.

Spécifiez une zone de transport de superposition pour un cluster en définissant l'option `overlay_tz` dans la section `[nsx_v3]` de `ncp.ini`. Cette étape est facultative. Si vous ne définissez pas `overlay_tz`, NCP récupère automatiquement l'ID de zone de transport de superposition à partir du routeur de niveau 0.

Routage logique de niveau 0

Connectez-vous à NSX Manager et accédez à **Mise en réseau > Routage > Routeurs**. Recherchez le routeur utilisé pour la mise en réseau de conteneur ou créez-en un.

Spécifiez un routeur logique de niveau 0 pour un cluster en définissant l'option `tier0_router` dans la section `[nsx_v3]` de `ncp.ini`.

Note Le routeur doit être créé en mode actif-en veille.

Commutateur logique

Les cartes réseau virtuelles utilisées par le nœud pour le trafic de données doivent être connectées à un commutateur logique de superposition. Il n'est pas obligatoire que l'interface de gestion du nœud soit connectée à NSX-T Data Center, bien que cela facilite la configuration. Vous pouvez créer un commutateur logique en vous connectant à NSX Manager et en accédant à **Mise en réseau > Commutation > Commutateurs**. Sur le commutateur, créez des ports logiques et attachez-y les cartes réseau virtuelles du nœud. Les ports logiques doivent avoir les balises suivantes :

- balise : `<cluster_name>`, étendue : `ncp/cluster`
- balise : `<node_name>`, étendue : `ncp/node_name`

La valeur `<cluster_name>` doit correspondre à la valeur de l'option `cluster` dans la section `[coe]` de `ncp.ini`.

Blocs d'adresses IP pour des espaces Kubernetes

Connectez-vous à NSX Manager et accédez à **Mise en réseau > IPAM** pour créer un ou plusieurs blocs d'adresses IP. Spécifiez le bloc d'adresses IP au format CIDR.

Spécifiez les blocs d'adresses IP pour les espaces Kubernetes en définissant l'option `container_ip_blocks` dans la section `[nsx_v3]` de `ncp.ini`.

Vous pouvez également créer des blocs d'adresses IP spécialement pour les espaces de noms non SNAT.

Spécifiez les blocs d'adresses IP non SNAT en définissant l'option `no_snat_ip_blocks` dans la section `[nsx_v3]` de `ncp.ini`.

Si vous créez des blocs d'adresses IP non SNAT alors que NCP est en cours d'exécution, vous devez redémarrer NCP. Sinon, NCP continuera d'utiliser les blocs d'adresses IP partagés jusqu'à leur épuisement.

Note Lorsque vous créez un bloc d'adresses IP, le préfixe ne doit pas être supérieur à la valeur du paramètre `subnet_prefix` dans le fichier de configuration de NCP `ncp.ini`.

Pool d'adresses IP pour SNAT

Le pool d'adresses IP sert à allouer des adresses IP qui seront utilisées pour la traduction d'adresses IP d'espace via des règles SNAT et pour l'exposition de contrôleurs d'entrée via des règles SNAT/DNAT, comme des adresses IP flottantes OpenStack. Ces adresses IP sont également appelées adresses IP externes.

Plusieurs clusters Kubernetes utilisent le même pool d'adresses IP externes. Chaque instance NCP utilise un sous-ensemble de ce pool pour le cluster Kubernetes qu'il gère. Par défaut, le même préfixe de sous-réseau pour les sous-réseaux d'espace sera utilisé. Pour utiliser une taille de sous-réseau différente, mettez à jour l'option `external_subnet_prefix` dans la section `[nsx_v3]` dans `ncp.ini`.

Connectez-vous à NSX Manager et accédez à **Inventaire > Groupes > Pools d'adresses IP** pour créer un pool ou rechercher un pool existant.

Spécifiez les pools d'adresses IP pour SNAT en définissant l'option `external_ip_pools` dans la section `[nsx_v3]` de `ncp.ini`.

Vous pouvez également configurer SNAT pour un service spécifique en ajoutant une annotation au service. Par exemple,

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

NCP configurera la règle SNAT pour ce service. L'adresse IP source de la règle correspond à l'ensemble des espaces de serveur principal. L'adresse IP de destination est l'adresse IP SNAT allouée à partir du pool d'adresses IP externe spécifié. Notez les points suivants :

- Le pool d'adresses IP spécifié par `ncp/snat_pool` doit déjà exister dans NSX-T Data Center avant que le service ne soit configuré. À partir de NCP 2.3.1, le pool d'adresses IP doit avoir la balise `{"ncp/owner": cluster:<cluster>}`.
- Dans NSX-T Data Center, la priorité de la règle SNAT pour le service est supérieure à celle du projet.
- Si un espace est configuré avec plusieurs règles SNAT, une seule fonctionnera.

Vous pouvez spécifier l'espace de noms auquel les adresses IP sont allouées à partir du pool d'adresses IP SNAT en ajoutant la balise suivante au pool d'adresses IP.

- étendue : `ncp/owner`, balise : `ns:<namespace_UUID>`

Vous pouvez obtenir l'UUID de l'espace de noms à l'aide de l'une des commandes suivantes :

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

Notez les points suivants :

- Chaque balise doit spécifier un UUID. Vous pouvez créer plusieurs balises pour le même pool.
- Si vous modifiez les balises après l'allocation d'adresses IP à des espaces de noms selon les anciennes balises, ces adresses IP sont récupérées uniquement après la modification des configurations SNAT des services ou le redémarrage de NCP.

- La balise de propriétaire d'un espace de noms est facultative. Sans cette balise, les adresses IP du pool IP SNAT peuvent être allouées à n'importe quel espace de noms.

(Facultatif) Sections de marqueur de pare-feu

Pour permettre à l'administrateur de créer des règles de pare-feu et que celles-ci n'interfèrent pas avec les sections de pare-feu créées par NCP et basées sur des stratégies réseau, connectez-vous à NSX Manager, accédez à **Sécurité > Pare-feu distribué > Général** et créez deux sections de pare-feu.

Spécifiez les sections de pare-feu de marqueur en définissant les options `bottom_firewall_section_marker` et `top_firewall_section_marker` dans la section `[nsx_v3]` de `ncp.ini`.

La section de pare-feu inférieure doit se trouver sous la section de pare-feu supérieure. Une fois ces sections de pare-feu créées, toutes les sections de pare-feu créées par NCP pour une isolation sont créées au-dessus de la section de pare-feu inférieure, et toutes les sections de pare-feu créées par NCP pour une stratégie sont créées en dessous de la section de pare-feu supérieure. Si ces sections de marqueur ne sont pas créées, toutes les règles d'isolation sont créées en bas et toutes les sections de stratégie sont créées en haut. L'utilisation de plusieurs sections de pare-feu de marqueur possédant la même valeur par cluster n'est pas prise en charge et provoque une erreur.

Créer et configurer un routeur logique de niveau 0

Le routeur logique de niveau 0 connecte les nœuds Kubernetes à des réseaux externes.

Procédure

- 1 Dans un navigateur, connectez-vous à NSX Manager sur <https://nsx-manager-ip-address>.
- 2 Accédez à **Mise en réseau > Routage > Routeurs** et cliquez sur **Ajouter > Routeur de niveau 0**.
- 3 Entrez un nom et éventuellement une description.
- 4 Sélectionnez un cluster Edge existant dans le menu déroulant pour sauvegarder ce routeur logique de niveau 0.
- 5 Sélectionnez un mode haute disponibilité.
Sélectionnez **actif-en veille**.
- 6 Cliquez sur **Enregistrer**.
Le nouveau routeur logique s'affiche sous forme de lien.
- 7 Cliquez sur le lien du routeur logique.
- 8 Cliquez sur **Routage > Redistribution d'itinéraire**.
- 9 Cliquez sur **Ajouter** pour ajouter un nouveau critère de redistribution.
Pour les sources, dans une topologie routée (non-NAT), sélectionnez **NSX statique**. Dans une topologie NAT, sélectionnez **NAT de niveau 0**.
- 10 Cliquez sur **Enregistrer**.

- 11 Cliquez sur le routeur qui vient d'être créé.
- 12 Cliquez sur **Configuration > Ports de routeur**
- 13 Cliquez sur **Ajouter** pour ajouter un port de liaison montante.
- 14 Sélectionnez un nœud de transport.
- 15 Sélectionnez le commutateur logique qui a été créé précédemment.
- 16 Spécifiez une adresse IP de votre réseau externe.
- 17 Cliquez sur **Enregistrer**.

Le nouveau routeur logique s'affiche sous forme de lien.

Installation de NCP dans un environnement OpenShift

Ce chapitre décrit l'installation et la configuration de NSX-T Container Plug-in (NCP) et OpenShift.

Ce chapitre contient les rubriques suivantes :

- [Déployer des machines virtuelles OpenShift](#)
- [Préparer le fichier hosts Ansible](#)
- [Installer NCP et OpenShift à l'aide d'un playbook unique](#)
- [Installer le plug-in CNI, OVS et l'image Docker NCP](#)
- [Installer OpenShift Container Platform](#)
- [Exécutez NCP et l'agent de nœud NSX](#)
- [Notes de configuration](#)

Déployer des machines virtuelles OpenShift

Avant d'installer NSX-T Container Plug-in, OpenShift doit être installé. Vous devez déployer au moins un maître.

Pour plus d'informations, consultez <https://docs.openshift.com>.

Étape suivante

Préparez le fichier hosts Ansible. Reportez-vous à la section [Préparer le fichier hosts Ansible](#).

Préparer le fichier hosts Ansible

Le fichier hosts Ansible définit les nœuds du cluster OpenShift.

Procédure

- 1 Clonez le référentiel NCP GitHub à l'adresse <https://github.com/vmware/nsx-integration-for-openshift>. Le fichier hosts se trouve dans le répertoire `openshift-ansible-nsx`. Vous devez conserver le fichier hosts dans le répertoire `openshift-ansible-nsx`. Certains playbooks supposent qu'il s'agit du chemin d'accès au fichier hosts.

- 2 Dans les sections [masters] et [nodes], spécifiez les noms d'hôte et les adresses IP des machines virtuelles OpenShift. Par exemple,

```
[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_schedulable=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
```

Notez que `openshift_ip` identifie l'adresse IP interne du cluster et doit être défini si l'interface à utiliser n'est pas celle par défaut. La variable `single_master` est utilisée par les rôles liés à NCP à partir d'un nœud maître pour effectuer certaines tâches une seule fois, par ex. configuration des ressources du plan de gestion de NSX-T Data Center.

- 3 Configurez l'accès SSH afin que tous les nœuds soient accessibles sans mot de passe à partir du nœud sur lequel est exécuté le rôle Ansible (il s'agit en général du nœud maître) :

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

- 4 Mettez à jour la section [OSEv3:vars]. Des détails sur tous les paramètres sont disponibles dans la documentation d'OpenShift Container Platform pour l'installation avancée (recherchez « advanced installation » dans <https://docs.openshift.com>). Par exemple,

```
# 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
    manually 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 Vérifiez que vous pouvez vous connecter à tous les hôtes :

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

Les résultats devraient ressembler à ce qui suit. Si ce n'est pas le cas, résolvez les problèmes de connectivité.

```

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

```

Étape suivante

Installez le plug-in CNI et OVS. Reportez-vous à la section [Installer le plug-in CNI, OVS et l'image Docker NCP](#).

Installer NCP et OpenShift à l'aide d'un playbook unique

Vous pouvez installer NCP et OpenShift à l'aide d'un seul playbook, ou effectuer l'installation en étapes distinctes.

Le playbook Ansible unique `install.yaml` effectue les tâches suivantes :

- Préparation de NCP
- Installation d'OpenShift
- Installation de NCP

Vous pouvez également installer NCP et OpenShift en suivant les instructions des deux sections suivantes : [Installer le plug-in CNI, OVS et l'image Docker NCP](#) et [Installer OpenShift Container Platform](#).

Avant d'exécuter le playbook `install.yaml`, définissez les paramètres obligatoires et facultatifs pour les rôles de playbook `ncp_prep` et `ncp`. Les paramètres sont décrits dans [Installer le plug-in CNI, OVS et l'image Docker NCP](#).

La commande suivante exécute le playbook :

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

Installer le plug-in CNI, OVS et l'image Docker NCP

Le plug-in CNI (Container Network Interface), Open vSwitch (OVS) et l'image Docker NCP doivent être installés sur les nœuds OpenShift. L'installation est effectuée en exécutant un playbook Ansible.

Note Cette étape n'est pas nécessaire si vous installez NCP et OpenShift à l'aide d'un playbook unique. Reportez-vous à la section [Installer NCP et OpenShift à l'aide d'un playbook unique](#).

Le playbook contient des instructions pour configurer les ressources de NSX-T pour les nœuds. Vous pouvez également configurer les ressources de NSX-T Data Center manuellement comme décrit dans [Chapitre 2 Configuration des ressources de NSX-T](#). Le paramètre `perform_nsx_config` indique s'il convient ou non de configurer les ressources lorsque le playbook est exécuté.

Procédure

- 1 Mettez à jour les valeurs de paramètre dans `roles/ncp_prep/default/main.yaml` et `roles/nsx_config/default/main.yaml`, y compris les URL où le RPM du plug-in CNI, OVS et son RPM de module de noyau correspondant peuvent être téléchargés. De plus, `uplink_port` est le nom de la vNIC du port de liaison montante sur la machine virtuelle du nœud. Les autres variables se rapportent à la configuration du plan de gestion NSX-T Data Center.

Paramètres à spécifier :

- `perform_nsx_config` : si vous voulez effectuer la configuration des ressources. Définissez cette variable sur `false` si la configuration sera effectuée manuellement et le script `nsx_config` ne sera pas exécuté.

- `nsx_manager_ip` : adresse IP de NSX Manager
- `nsx_edge_cluster_name` : nom du cluster Edge que le routeur de niveau 0 doit utiliser
- `nsx_transport_zone_name` : nom de la zone de transport de superposition
- `os_node_name_list` : liste de noms de nœud séparés par des virgules
Par exemple, `noeud1, noeud2, noeud3`
- `subnet_cidr` : adresse CIDR de l'administrateur d'adresse IP qui sera attribuée à `br-int` sur le nœud
- `vc_host` : adresse IP de vCenter Server
- `vc_user` : nom d'utilisateur de l'administrateur vCenter Server
- `vc_password` : mot de passe de l'administrateur vCenter Server
- `vms` : liste des noms de machine virtuelle séparés par des virgules. L'ordre doit correspondre à `os_node_name_list`.

Les paramètres suivants ont des valeurs par défaut. Vous pouvez les modifier si nécessaire.

- `nsx_t0_router_name` : nom du routeur logique de niveau 0 du cluster. Valeur par défaut : **`t0`**
- `pod_ipblock_name` : nom du bloc d'adresses IP pour les espaces. Valeur par défaut : **`podIPBlock`**
- `pod_ipblock_cidr` : adresse CIDR pour ce bloc d'adresses IP. Valeur par défaut : **`172.20.0.0/16`**
- `snat_ippool_name` : nom du bloc d'adresses IP pour SNAT. La valeur par défaut est `externalIP`.
- `snat_ippool_cidr` : adresse CIDR pour ce bloc d'adresses IP. Valeur par défaut : **`172.30.0.0/16`**
- `start_range` : adresse IP de début de CIDR spécifiée pour ce pool d'adresses IP. Valeur par défaut : **`172.30.0.1`**
- `end_range` : adresse IP de fin de CIDR spécifiée pour ce pool d'adresses IP. Valeur par défaut : **`172.30.255.254`**
- `os_cluster_name` : nom du cluster OpenShift. Valeur par défaut : **`occl-one`**
- `nsx_node_ls_name` : nom du commutateur logique connecté aux nœuds. Valeur par défaut : **`node_ls`**
- `nsx_node_lr_name` : nom d'un routeur logique pour le commutateur **`node_ls`**. Valeur par défaut : **`node_lr`**

Le playbook `nsx-config` prend en charge la création d'un seul pool d'adresses IP et d'un seul bloc d'IP. Si vous en souhaitez davantage, vous devez les créer manuellement.

2 Accédez au répertoire openshift-ansible-nsx et exécutez le rôle ncp_prep.

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

Le manuel contient des instructions pour effectuer les actions suivantes :

- Téléchargez le fichier d'installation du plug-in CNI.

Le nom de fichier est `nsx-cni-1.0.0.0.0.xxxxxxx-1.x86_64.rpm`, où `xxxxxxx` est le numéro de build.

- Installez le fichier d'installation du plug-in CNI.

Le plug-in est installé dans `/opt/cni/bin`. Le fichier de configuration CNI `10.net.conf` est copié dans `/etc/cni/net.d`. Le RPM installe également le fichier de configuration `/etc/cni/net.d/99-loopback.conf` pour le plug-in de bouclage.

- Téléchargez et installez les fichiers d'installation OVS.

Les fichiers sont `openvswitch-2.9.1.xxxxxxx-1.x86_64.rpm` et `openvswitch-kmod-2.9.1.xxxxxxx-1.el7.x86_64.rpm`, où `xxxxxxx` est le numéro de build.

- Créez l'instance de *br-int* si elle n'est pas déjà créée.

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

- Ajoutez l'interface réseau (*node-if*) qui est attachée au commutateur logique du nœud à *br-int*.
- Vérifiez que l'état de *br-int* et *node-if link* est actif.

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

- Mettez à jour le fichier de configuration réseau pour vous assurer que l'interface réseau est active après un redémarrage.
- Téléchargez le fichier `.tar` de NCP et chargez l'image de Docker à partir de ce fichier.
- Téléchargez le fichier YAML `ncp-rbac` et définissez le paramètre `apiVersion` sur **v1**.
- Créez une topologie logique et les ressources associées dans NSX-T Data Center, puis créez des balises dessus afin que NCP puisse les reconnaître.
- Mettez à jour `ncp.ini` avec les informations sur les ressources NSX-T Data Center.

Étape suivante

Installez OpenShift Container Platform. Reportez-vous à la section [Installer OpenShift Container Platform](#).

Installer OpenShift Container Platform

OpenShift Container Platform (OCP) est une plateforme sous forme de service (PaaS) qui regroupe Docker et Kubernetes.

Note Cette étape n'est pas nécessaire si vous installez NCP et OpenShift à l'aide d'un playbook unique. Reportez-vous à la section [Installer NCP et OpenShift à l'aide d'un playbook unique](#).

Pour plus d'informations sur l'installation d'OCP, reportez-vous à la section <https://docs.openshift.com>.

Étape suivante

Exécutez NCP et l'agent de nœud NSX. Reportez-vous à la section [Exécutez NCP et l'agent de nœud NSX](#).

Exécutez NCP et l'agent de nœud NSX

Configurez et exécutez NCP et l'agent de nœud NSX.

Procédure

- 1 Modifiez `roles/ncp/defaults/main.yaml` et spécifiez l'adresse IP du serveur d'API OpenShift, l'adresse IP de NSX Manager et les URL de téléchargement des fichiers yaml NCP ReplicationController et nsx-node-agent DaemonSet.
- 2 À partir du répertoire `openshift-ansible-nsx`, exécutez le rôle `ncp` :

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

Le rôle `ncp` effectue les étapes suivantes :

- Vérifiez si le projet `nsx-system` existe et créez-en un si ce n'est pas le cas.

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

- Téléchargez le fichier YAML `ncp-rbac` et définissez le paramètre `apiVersion` sur **v1**.
- Créez le compte de service pour l'espace NCP, créez un rôle de cluster qui spécifie les ressources auxquelles NCP peut accéder et liez le rôle de cluster au compte de service NCP.
- Créez le compte de service pour l'espace `nsx-node-agent`, créez un rôle de cluster qui spécifie les ressources auxquelles l'agent de nœud peut accéder et liez le rôle de cluster au compte de service de l'agent de nœud.

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

- Obtenez le jeton associé aux comptes de service ci-dessus et enregistrez-le dans `/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
```

- Téléchargez le fichier YAML SecurityContextConstraint (SCC) `ncp-os-scc.yml` pour NCP et créez le SCC basé sur ce fichier.

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

Le fichier YAML SCC spécifie le type SELinux comme `spc_t` pour s'assurer que NCP/`nsx-node-agent` dispose d'autorisations d'accès sous SELinux. C'est,

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

Dans le fichier YAML SCC, sous `seLinuxContextseLinuxOptions`, le niveau de contrôle d'accès basé sur l'étiquette SELinux est défini sur `s0:c0:c1023` pour autoriser l'accès de `ncp/node-agent` aux cibles de catégories de fichiers différents.

- Ajoutez le fichier SCC à l'utilisateur qui crée le NCP et les espaces de l'agent de nœud NSX. Par exemple, pour ajouter le fichier SCC à l'utilisateur par défaut dans le projet actuel :

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

- Ajoutez le fichier SCC aux comptes de service de NCP et de l'agent de nœud NSX :

```
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
```

- Téléchargez les fichiers YAML pour NCP ReplicationController (RC) et `nsx-node-agent` DaemonSet (DS), et mettez à jour le ConfigMap.
- Téléchargez et chargez l'image NCP (`nsx-node-agent` utilise la même image).
- Configurez le compte de service et configurez les fichiers SecurityContextConstraint pour NCP et `nsx_node_agent` requis.

- Créez les fichiers NCP ReplicationController et nsx-node-agent DaemonSet.

Note NCP ouvre des connexions HTTP persistantes au serveur Kubernetes API pour surveiller les événements de cycle de vie des ressources Kubernetes. Si une défaillance du serveur API ou une panne de réseau provoque l'obsolescence des connexions TCP de NCP, vous devez redémarrer NCP pour lui permettre de rétablir les connexions au serveur API. Sinon, NCP ratera les nouveaux événements.

Notes de configuration

Avant de configurer OpenShift et NCP, notez les informations suivantes.

- Un espace ne doit pas comporter plus de 11 étiquettes et un espace de noms ne doit pas comporter plus de 12 étiquettes.
- Les étiquettes ajoutées pour une utilisation interne d'OpenShift, par exemple, une étiquette avec le préfixe openshift.io dans sa clé, seront ignorées par NCP et l'utilisateur ne verra pas les balises correspondantes créées sur les ressources NSX associées. Voici une liste de préfixes d'étiquette utilisés par OpenShift et vous devez éviter d'utiliser une clé d'étiquette commençant par ce qui suit :

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

- Les nœuds devront accéder aux espaces, par exemple, pour les contrôles de santé Kubelet. Vérifiez que l'interface de gestion d'hôte peut accéder au réseau de l'espace.
- Les capacités NET_ADMIN et NET_RAW de Linux peuvent être exploitées par des personnes mal intentionnées pour compromettre le réseau de l'espace. Vous devez désactiver ces deux capacités de conteneurs non approuvés. Par défaut, avec SCC restreint et anyuid, NET_ADMIN n'est pas accordé. Faites attention aux SCC qui activent NET_ADMIN explicitement ou qui activent l'espace pour s'exécuter en mode privilégié. De plus, pour les conteneurs non approuvés, créez un SCC distinct en fonction, par exemple, de SCC anyuid, avec la capacité NET_RAW supprimée. Pour cela, vous pouvez ajouter NET_RAW à la liste requiredDropCapabilities dans la définition de SCC.
- Autorisez l'accès racine dans les espaces/conteneurs (pour le test uniquement). Les commandes ci-dessous nécessiteront un accès racine dans tous les espaces du projet oc auquel vous êtes actuellement connecté.

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

- Configurez (ajoutez) le Registre OpenShift.

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

- Supprimer le Registre OpenShift

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

- Il existe une règle de pare-feu IPTables manquante pour autoriser les demandes DNS dans les conteneurs de pont Docker par défaut au processus dnsmasq sur le nœud. Il doit être ouvert manuellement. Modifiez `/etc/sysconfig/iptables` et ajoutez les règles suivantes en bas du fichier avant COMMIT :

```
-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
```

- Redémarrez iptables, docker et origin-node (redémarre kube-proxy et kubelet).

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

- Le Registre de Docker interne d'OpenShift doit être autorisé à utiliser non-TLS pour qu'OpenShift fonctionne. Normalement, cela doit être ajouté automatiquement par le programme d'installation d'OpenShift Ansible, mais il semble que cela ne fonctionne pas. Modifiez `/etc/sysconfig/docker` et ajoutez :

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

- Redémarrez Docker.

```
systemctl restart docker
```

- La prise en charge par NCP des stratégies réseau est la même que celle fournie par Kubernetes et dépend de la version de Kubernetes utilisée par OpenShift.
 - OpenShift 3.9 : les clauses de règle dans la stratégie réseau peuvent contenir au maximum un sélecteur parmi namespaceSelector, podSelector et ipBlock.
- Le serveur API Kubernetes n'effectue pas de validation d'une spécification de la stratégie réseau. Il est possible de créer une stratégie réseau qui n'est pas valide. NCP rejettera cette stratégie réseau. Si vous mettez à jour la stratégie réseau pour la rendre valide, NCP ne sera toujours pas en mesure de traiter la stratégie réseau. Vous devez supprimer la stratégie réseau et en recréer une avec une spécification valide.

- Certaines versions de Kubernetes présentent un problème lié à `subPath` (reportez-vous à la section <https://github.com/kubernetes/kubernetes/issues/61076>). Si la version d'OpenShift ne contient pas de correctif pour ce problème, la création de l'espace NCP échoue avec l'erreur `CreateContainerConfigError : impossible de préparer subPath pour volumeMount`. Vous pouvez contourner ce problème en supprimant l'utilisation de `subPath` dans le fichier YAML de NCP. En particulier, supprimez la ligne contenant `subPath: ncp.ini` et remplacez la configuration de volumes par les éléments suivants :

```
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
```

Cette modification a pour effet secondaire de définir en lecture seule tout le répertoire `/etc/nsx-ujo`. Par conséquent, la connexion avec NSX-T en utilisant un certificat et une clé privée ne fonctionnera pas, car NCP ne pourra pas créer de fichier temporaire sous `/etc/nsx-ujo` pour déplacer le certificat et la clé privée vers un seul fichier.

- Si vous exécutez le cluster OpenShift 3.10 ou que vous le mettez à niveau, notez les points suivants :
 - Vous devez spécifier la configuration de groupes de nœuds spécifiques au cluster OpenShift 3.10. La configuration ConfigMap de nœud doit être indiquée dans le fichier d'hôtes d'inventaire.
 - Tous les hôtes définis dans le groupe `[nodes]` du fichier d'hôtes d'inventaire doivent être attribués à un nom de groupe de nœuds.
 - La mise à niveau du cluster OpenShift à partir d'un manuel Ansible peut entraîner la perte de réseau. Veillez à ajouter le correctif (<https://github.com/openshift/openshift-ansible/pull/8016/files#diff-2386e21861da3f95091dbb27d72ca366>) sur le référentiel `openshift-ansible` à supprimer de l'arrêt/la désinstallation de modules Open vSwitch.
- À partir d'OpenShift 3.10, `kube-proxy` a été déplacé du service `openshift-node` vers un `DaemonSet`. Il n'est plus démarré par défaut. Pour démarrer manuellement `kube-proxy` (en supposant que le référentiel `openshift-ansible` a été cloné), procédez comme suit :
 - Accédez au répertoire `openshift-ansible` et sous `[defaults]`, définissez les éléments suivants :

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

- Créez un fichier `create_proxy.yaml` dans le répertoire des playbooks avec les entrées suivantes :

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

- Exécutez le playbook :

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

Vous verrez des messages d'erreur indiquant l'échec de certaines opérations. Ces messages peuvent être ignorés. Vous pouvez vérifier le résultat en exécutant la commande `oc get po --all-namespaces`.

Installation de NCP dans un environnement Bare-Metal

4

Les étapes d'installation NSX-T Container Plug-in (NCP) dans un environnement bare-metal sont semblables pour les étapes d'installation NCP dans un environnement autre que bare-metal. Les étapes qui sont différentes sont décrites dans cette section.

Ce chapitre contient les rubriques suivantes :

- [Installer le plug-in CNI NSX-T Data Center](#)
- [Configurer la mise en réseau de centre de données NSX-T pour les nœuds OpenShift](#)
- [Installer l'agent de nœud NSX](#)
- [ConfigMap pour ncp.ini dans nsx-node-agent-ds.yml](#)
- [Installer le plug-in de conteneur NSX-T](#)
- [ConfigMap pour ncp.ini dans ncp-rc.yml](#)

Installer le plug-in CNI NSX-T Data Center

Le plug-in CNI NSX-T Data Center doit être installé sur les nœuds OpenShift.

Procédure

- 1 Téléchargez le fichier d'installation approprié pour votre distribution Linux.

Le nom de fichier est `nsx-cni-1.0.0.0.xxxxxxx-1.x86_64.rpm`, où `xxxxxxx` est le numéro de build.

- 2 Installez le fichier `.rpm` téléchargé à l'étape 1.

Le plug-in est installé dans `/opt/cni/bin`. Le fichier de configuration CNI `10.net.conf` est copié dans `/etc/cni/net.d`. Le RPM installe également le fichier de configuration `/etc/cni/net.d/99-loopback.conf` pour le plug-in de bouclage.

Configurer la mise en réseau de centre de données NSX-T pour les nœuds OpenShift

Cette section explique comment configurer la mise en réseau NSX-T Data Center pour des nœuds maître et de calcul OpenShift.

Chaque nœud doit être enregistré avec le NSX Manager en tant que type de système d'exploitation RHEL Container. L'interface de gestion du nœud peut être utilisée pour se connecter au cluster OpenShift et peut être installée sur l'infrastructure NSX-T Data Center. Les autres interfaces fournissent une mise en réseau pour les espaces et doivent être installées sur l'infrastructure NSX-T Data Center.

Le nœud de transport correspondant doit avoir les balises suivantes :

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

Vous pouvez identifier le nœud de transport pour un nœud OpenShift en accédant à **Infrastructure > Nœuds** à partir de l'interface utilisateur graphique de NSX Manager.

Si le nom du nœud OpenShift change, vous devez mettre à jour la balise `ncp/node_name` et redémarrer NCP. Vous pouvez utiliser la commande suivante pour obtenir les noms de nœud :

```
oc get nodes
```

Si vous ajoutez un nœud à un cluster alors que NCP est en cours d'exécution, vous devez ajouter les balises au nœud de transport avant d'exécuter la commande `oc cluster add`. Sinon, le nouveau nœud ne disposera pas de connectivité réseau. Si les balises sont incorrectes ou manquantes, vous pouvez procéder comme suit pour résoudre ce problème :

- Appliquez les balises appropriées au nœud de transport.
- Redémarrez NCP.

Installer l'agent de nœud NSX

L'agent de nœud NSX est un DaemonSet où chaque espace exécute deux conteneurs. Un conteneur exécute l'agent de nœud NSX, dont la responsabilité principale consiste à gérer les interfaces réseau de conteneur. Il interagit avec le plug-in CNI et Kubernetes API Server. L'autre conteneur exécute le proxy kube NSX, dont la seule responsabilité consiste à mettre en œuvre l'abstraction du service Kubernetes en convertissant les adresses IP du cluster en adresses IP d'espace. Il met en œuvre la même fonctionnalité que le proxy kube en amont.

Procédure

- 1 Téléchargez l'image Docker de NCP.

Le nom de fichier est `nsx-ncp-xxxxxxx.tar`, où `xxxxxxx` est le numéro de build.

- 2 Téléchargez le modèle yaml DaemonSet de l'agent de nœud NSX.

Le nom de fichier est `nsx-node-agent-ds.yml`. Vous pouvez modifier ce fichier ou l'utiliser comme exemple pour votre propre fichier de modèle.

- 3 Chargez l'image Docker de NCP dans votre registre d'images.

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

4 Modifiez le fichier `nsx-node-agent-ds.yml`.

Changez le nom de l'image pour celui qui a été chargé.

Effectuez les modifications suivantes :

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

Supprimez les lignes suivantes :

```
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
```

Note Dans le fichier yaml, vous devez spécifier que le ConfigMap généré pour `ncp.ini` doit être monté en tant que volume en lecture seule. Le fichier yaml téléchargé a déjà cette spécification, qui ne doit pas être modifiée.

5 Créez le DaemonSet de l'agent de nœud NSX avec la commande suivante.

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

ConfigMap pour `ncp.ini` dans `nsx-node-agent-ds.yml`

L'exemple de fichier yaml `nsx-node-agent-ds.yml` contient un ConfigMap pour le fichier de configuration `ncp.ini` pour l'agent du nœud NSX. Cette section ConfigMap contient des paramètres que vous pouvez spécifier pour personnaliser votre installation de l'agent du nœud.

L'exemple de fichier `nsx-node-agent-ds.yml` que vous téléchargez contient les informations `ncp.ini` suivantes :

```
# 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>

```

Installer le plug-in de conteneur NSX-T

NSX-T Container Plug-in (NCP) est fourni sous forme d'image Docker. NCP doit s'exécuter sur un nœud des services d'infrastructure. Il n'est pas recommandé d'exécuter NCP sur le nœud maître.

Procédure

1 Téléchargez l'image Docker de NCP.

Le nom de fichier est `nsx-ncp-xxxxxxx.tar`, où `xxxxxxx` est le numéro de build.

2 Téléchargez le modèle yaml ReplicationController de NCP.

Le nom de fichier est `ncp-rc.yaml`. Vous pouvez modifier ce fichier ou l'utiliser comme exemple pour votre propre fichier de modèle.

3 Chargez l'image Docker de NCP dans votre registre d'images.

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

4 Modifiez `ncp-rc.yaml`.

Définissez le type de nœud sur bare metal.

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

Changez le nom de l'image pour celui qui a été chargé.

Spécifiez le paramètre `nsx_api_managers`. Cette version prend en charge un seul cluster de nœuds Kubernetes et une seule instance de NSX Manager. Par exemple :

```
nsx_api_managers = 192.168.1.180
```

(Facultatif) Spécifiez le paramètre `ca_file` dans la section `[nsx_v3]`. La valeur doit être un fichier de bundle d'autorité de certification à utiliser pour la vérification du certificat de serveur NSX Manager. Si aucune valeur n'est définie, les autorités de certification racines du système seront utilisées.

Spécifiez les paramètres `nsx_api_cert_file` et `nsx_api_private_key_file` pour l'authentification avec NSX-T Data Center.

`nsx_api_cert_file` est le chemin d'accès complet vers un fichier de certificat client au format PEM. Le contenu de ce fichier devrait ressembler à ce qui suit :

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

`nsx_api_private_key_file` est le chemin d'accès complet vers un fichier de clé privée client au format PEM. Le contenu de ce fichier devrait ressembler à ce qui suit :

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

Spécifiez le paramètre `ingress_mode = nat` si le contrôleur d'entrée est configuré pour s'exécuter en mode NAT.

Par défaut, le préfixe de sous-réseau 24 est utilisé pour tous les sous-réseaux alloués à partir des blocs d'adresses IP pour les commutateurs logiques d'espace. Pour utiliser une taille de sous-réseau différente, mettez à jour l'option `subnet_prefix` dans la section `[nsx_v3]`.

Note Dans le fichier `yaml`, vous devez spécifier que le ConfigMap généré pour `ncp.ini` doit être monté en tant que volume en lecture seule. Le fichier `yaml` téléchargé a déjà cette spécification, qui ne doit pas être modifiée.

5 Créez ReplicationController de NCP.

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

Note NCP ouvre des connexions HTTP persistantes au serveur Kubernetes API pour surveiller les événements de cycle de vie des ressources Kubernetes. Si une défaillance du serveur API ou une panne de réseau provoque l'obsolescence des connexions TCP de NCP, vous devez redémarrer NCP pour lui permettre de rétablir les connexions au serveur API. Sinon, NCP ratera les nouveaux événements.

Pendant une mise à jour tournante de NCP ReplicationController, ne redémarrez pas l'hôte de conteneur. Si l'hôte est redémarré pour une raison quelconque, vous pouvez voir deux espaces NCP en cours d'exécution après le redémarrage. Dans ce cas, procédez comme suit :

- Supprimez l'un des espaces NCP. Vous pouvez supprimer l'un ou l'autre indifféremment. Par exemple,

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

- Supprimez l'espace de noms `nsx-system`. Par exemple,

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

ConfigMap pour `ncp.ini` dans `ncp-rc.yml`

L'exemple de fichier YAML `ncp-rc.yml` contient un ConfigMap pour le fichier de configuration `ncp.ini`. Cette section ConfigMap contient les paramètres que vous devez spécifier avant d'installer NCP, comme décrit dans la section précédente.

L'exemple de fichier `ncp-rc.yml` que vous téléchargez contient les informations `ncp.ini` suivantes :

```
# 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>

```

Équilibrage de charge

L'équilibrage de charge NSX-T Data Center est intégré à OpenShift et est utilisé comme routeur OpenShift.

NCP surveille les événements de routage et de point de terminaison OpenShift, et configure les règles d'équilibrage de charge en fonction de la spécification de la route. En conséquence, l'équilibrage de charge NSX-T Data Center transférera le trafic de couche 7 entrant vers les espaces principaux en fonction des règles.

Configuration de l'équilibrage de charge

La configuration de l'équilibrage de charge implique la configuration d'un service Kubernetes LoadBalancer ou d'une route OpenShift. Vous devez également configurer le contrôleur de réplication NCP. Le service LoadBalancer est pour le trafic de couche 4 et la route OpenShift est pour le trafic de couche 7.

Lorsque vous configurez un service Kubernetes LoadBalancer, une adresse IP est allouée au service à partir du bloc d'adresses IP externe que vous configurez. L'équilibrage de charge est exposé sur cette adresse IP et sur le port de service. Vous pouvez spécifier le nom ou l'ID d'un pool d'adresses IP à l'aide de la spécification `loadBalancerIP` dans la définition de l'équilibrage de charge. L'adresse IP du service d'équilibrage de charge sera allouée à partir de ce pool d'adresses IP. Si la spécification `loadBalancerIP` est vide, l'adresse IP sera allouée à partir du bloc d'adresses IP externe que vous configurez.

À partir de NCP 2.3.1, le pool d'adresses IP spécifié par `loadBalancerIP` doit avoir la balise `{"ncp/owner": cluster:<cluster>}`.

Pour utiliser l'équilibrage de charge NSX-T Data Center, vous devez configurer l'équilibrage de charge dans NCP. Dans le fichier `ncp_rc.yml`, procédez comme suit :

- 1 Définissez `use_native_loadbalancer = True`.
- 2 Définissez `pool_algorithm` sur `WEIGHTED_ROUND_ROBIN`.
- 3 Définissez `lb_default_cert_path` et `lb_priv_key_path` comme noms de chemin d'accès complet du fichier de certificat signé par une autorité de certification et du fichier de clé privée, respectivement. Reportez-vous à la section ci-dessous pour un exemple de script permettant de générer un certificat signé par une autorité de certification. En outre, placez le certificat et la clé par défaut dans l'espace NCP. Reportez-vous à la section ci-dessous pour obtenir des instructions.

- 4 (Facultatif) Spécifiez un paramètre de persistance avec les paramètres `l4_persistence` et `l7_persistence`. Les options disponibles pour la persistance de la couche 4 est l'adresse IP source. Les options disponibles pour la persistance de la couche 7 sont le cookie et l'adresse IP source. La valeur par défaut est `<None>`. Par exemple,

```
# 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 (Facultatif) Définissez `service_size` = SMALL, MEDIUM ou LARGE. La valeur par défaut est SMALL.
- 6 Si vous utilisez OpenShift 3.11, vous devez effectuer la configuration suivante pour qu'OpenShift n'attribue aucune adresse IP au service LoadBalancer.
- Définissez `ingressIPNetworkCIDR` sur `0.0.0.0/32` sous `networkConfig` dans le fichier `/etc/origin/master/master-config.yaml`.
 - Redémarrez le serveur d'API et les contrôleurs à l'aide des commandes suivantes :

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

Note Si vous configurez un équilibrage de charge de couche 4 et de couche 7, vous pouvez définir `l4_persistence` ou `l7_persistence`, ou les deux sur `source_ip`, mais vous ne pouvez pas définir `l4_persistence` sur `source_ip` et `l7_persistence` sur `cookie`. Si vous définissez par erreur `l4_persistence` sur `source_ip` et `l7_persistence` sur `cookie`, le service LoadBalancer ne fonctionnera pas. Pour résoudre le problème, vous devez supprimer la ressource d'entrée et le service LoadBalancer, modifier les paramètres de persistance, redémarrer NCP et recréer la ressource d'entrée et le service LoadBalancer.

Exemple d'équilibrage de charge de couche 7

Le fichier YAML suivant configure deux contrôleurs de réplication (`tea-rc` et `coffee-rc`), deux services (`tea-svc` et `coffee-svc`) et deux routes (`cafe-route-multi` et `cafe-route`) pour assurer l'équilibrage de charge de couche 7.

```
# 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

```

Remarques supplémentaires

- Seule la terminaison Edge est prise en charge pour le trafic HTTPS.
- Un sous-domaine de caractères génériques est pris en charge. Par exemple, si wildcardPolicy est défini sur **Sous-domaine** et que le nom d'hôte est défini sur **wildcard.example.com**, toute demande à ***.example.com** est traitée.
- Si NCP génère une erreur lors du traitement d'un événement Route en raison d'une mauvaise configuration, vous devez corriger le fichier Route YAML, puis supprimer et recréer la ressource Route.
- NCP n'applique pas la propriété de nom d'hôte par espaces de noms.
- Un service LoadBalancer est pris en charge par cluster Kubernetes.
- NSX-T Data Center crée un serveur virtuel et un pool d'équilibrage de charge de couche 4 pour chaque port de service LoadBalancer. TCP et UDP sont pris en charge.

- L'équilibrage de charge NSX-T Data Center est fourni dans différentes tailles. Pour plus d'informations sur la configuration d'un équilibrage de charge NSX-T Data Center, consultez le *Guide d'administration NSX-T*.

Le petit équilibrage de charge NSX-T Data Center prend en charge les éléments suivants :

- 10 serveurs virtuels NSX-T.
- 10 pools NSX-T.
- 30 membres de pool NSX-T.
- 8 ports pour les services LoadBalancer.
- Un total de 10 ports définis par les services LoadBalancer et les ressources Route.
- Un total de 30 points de terminaison référencés par les services LoadBalancer et les ressources Route.

L'équilibrage de charge NSX-T Data Center moyen prend en charge les éléments suivants :

- 100 serveurs virtuels NSX-T.
- 100 pools NSX-T.
- 300 membres de pool NSX-T.
- 98 ports pour les services LoadBalancer.
- Un total de 100 ports définis par les services LoadBalancer et les ressources Route.
- Un total de 300 points de terminaison référencés par les services LoadBalancer et les ressources Route.

Le grand équilibrage de charge NSX-T Data Center prend en charge les éléments suivants :

- 1 000 serveurs virtuels NSX-T.
- 1 000 pools NSX-T.
- 3 000 membres de pool NSX-T.
- 998 ports pour les services LoadBalancer.
- Un total de 1 000 ports définis par les services LoadBalancer et les ressources Route.
- Un total de 3 000 points de terminaison référencés par les services LoadBalancer et les ressources Route.

Après la création de l'équilibrage de charge, la taille d'équilibrage de charge ne peut pas être modifiée en mettant à jour le fichier de configuration. Elle peut être modifiée au moyen de l'interface utilisateur ou de l'API.

- À partir de NCP 2.3.1, la mise à l'échelle automatique de l'équilibrage de charge de couche 4 est prise en charge. Si un service LoadBalancer Kubernetes est créé ou modifié pour qu'il requiert des serveurs virtuels supplémentaires et que l'équilibrage de charge de couche 4 existant n'a pas la capacité, un nouvel équilibrage de charge de couche 4 sera créé. NCP supprimera également les équilibres de charge de couche 4 qui n'ont plus aucun serveur virtuel associé. Cette fonctionnalité est activée par défaut. Vous pouvez la désactiver en définissant `l4_lb_auto_scaling` sur **false** dans le ConfigMap NCP. Cette fonctionnalité requiert NSX-T Data Center 2.3 ou version ultérieure.

Exemple de script permettant de générer un certificat signé par une autorité de certification

Le script ci-dessous génère un certificat signé par une autorité de certification et une clé privée stockés dans les fichiers `<filename>.crt` et `<filename>.key`, respectivement. La commande `genrsa` génère une clé d'autorité de certification. La clé d'autorité de certification doit être chiffrée. Vous pouvez spécifier une méthode de chiffrement avec la commande `aes256`, par exemple.

```
#!/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
```

Placer le certificat et la clé par défaut dans l'espace NCP

Une fois le certificat et la clé privée générés, placez-les dans le répertoire `/etc/nsx-ujo` sur la machine virtuelle hôte. En supposant que les fichiers de certificat et de clé sont nommés `lb-default.crt` et `lb-default.key`, respectivement, modifiez `ncp-rc.yaml` afin que ces fichiers, stockés sur l'hôte, soient placés dans l'espace. Par exemple,

```
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-uj0/lb-default.crt  
- name: lb-priv-key  
  hostPath:  
    path: /etc/nsx-uj0/lb-default.key
```

Administration de NSX-T Container Plug-in

6

Vous pouvez administrer NSX-T Container Plug-in à partir de l'interface utilisateur NSX Manager ou à partir de l'interface de ligne de commande.

Note Si une machine virtuelle d'hôte de conteneur s'exécute sur ESXi 6.5 et si la machine virtuelle est migrée via vMotion vers un autre hôte ESXi 6.5, les conteneurs en cours d'exécution sur l'hôte de conteneur perdent la connectivité aux conteneurs en cours d'exécution sur d'autres hôtes de conteneur. Vous pouvez résoudre ce problème en déconnectant et reconnectant la carte réseau virtuelle de l'hôte de conteneur. Ce problème ne se produit pas avec ESXi 6.5 Update 1 ou version ultérieure.

Hyperbus réserve l'ID de VLAN 4094 sur l'hyperviseur pour la configuration PVLAN et cet ID ne peut pas être modifié. Pour éviter tout conflit de VLAN, ne configurez pas les commutateurs logiques de VLAN ou les vmknics VTEP avec les mêmes ID de VLAN.

Ce chapitre contient les rubriques suivantes :

- [Gérer les blocs d'adresses IP à partir de l'interface utilisateur de NSX Manager](#)
- [Voir les sous-réseaux de bloc d'adresses IP à partir de l'interface utilisateur graphique de NSX Manager](#)
- [Ports logiques attachés par CIF](#)
- [Commandes d'interface de ligne de commande](#)
- [Codes d'erreur](#)

Gérer les blocs d'adresses IP à partir de l'interface utilisateur de NSX Manager

Vous pouvez ajouter, supprimer, modifier, afficher les détails et gérer les balises d'un bloc d'adresses IP à partir de l'interface utilisateur de NSX Manager.

Procédure

- 1 Dans un navigateur, connectez-vous au dispositif NSX Manager sur `https://<nsx-manager-IP-address-or-domain-name>`.
- 2 Accédez à **Mise en réseau > IPAM**.

La liste des blocs d'adresses IP existants s'affiche.

3 Utilisez l'une des actions suivantes.

Option	Action
Ajouter un bloc d'adresses IP	Cliquez sur Ajouter .
Supprimer un ou plusieurs blocs d'adresses IP	Sélectionnez un ou plusieurs blocs d'adresses IP et cliquez sur SUPPRIMER .
Modifier un bloc d'adresses IP	Sélectionnez un bloc d'adresses IP et cliquez sur MODIFIER .
Afficher des détails sur un bloc d'adresses IP	Cliquez sur le nom du bloc d'adresses IP. Cliquez sur l'onglet Présentation pour voir des informations générales. Cliquez sur l'onglet Sous-réseaux pour voir les sous-réseaux du bloc d'adresses IP.
Gérer les balises d'un bloc d'adresses IP	Sélectionnez un bloc d'adresses IP et cliquez sur ACTIONS > Gérer les balises .

Vous ne pouvez pas supprimer un bloc d'adresses IP dont des sous-réseaux sont alloués.

Voir les sous-réseaux de bloc d'adresses IP à partir de l'interface utilisateur graphique de NSX Manager

Vous pouvez voir les sous-réseaux pour un bloc d'adresses IP à partir de l'interface utilisateur graphique de NSX Manager. Il n'est pas recommandé d'ajouter ou de supprimer des sous-réseaux de bloc d'adresses IP après avoir installé et exécuté NCP.

Procédure

- 1 Dans un navigateur, connectez-vous au dispositif NSX Manager sur `https://<nsx-manager-IP-address-or-domain-name>`.
- 2 Accédez à **Mise en réseau > IPAM**.
La liste des blocs d'adresses IP existants s'affiche.
- 3 Cliquez sur un nom de bloc d'adresses IP.
- 4 Cliquez sur l'onglet **Sous-réseaux**.

Ports logiques attachés par CIF

Les CIF (interfaces de conteneur) sont des interfaces réseau sur des conteneurs qui sont connectés à des ports logiques sur un commutateur. Ces ports sont appelés ports logiques attachés par CIF.

Vous pouvez gérer des ports logiques attachés par CIF depuis l'interface utilisateur de NSX Manager.

Gestion des ports logiques attachés par CIF

Accédez à **Mise en réseau > Commutation > Ports** pour voir tous les ports logiques, y compris les ports logiques attachés par CIF. Cliquez sur le lien d'attachement d'un port logique attaché par CIF pour afficher les informations de l'attachement. Cliquez sur le lien du port logique pour ouvrir un volet de fenêtre avec quatre onglets : Présentation, Surveiller, Gérer et Éléments associés. Cliquez sur **Éléments associés > Ports logiques** pour voir le port logique associé sur un commutateur de liaison montante. Pour plus d'informations sur les ports de commutateur, consultez le *Guide d'administration de NSX-T*.

Outils de surveillance du réseau

Les outils suivants prennent en charge les ports logiques attachés par CIF. Pour plus d'informations sur ces outils, consultez le *Guide d'administration de NSX-T*.

- Traceflow
- Connexion de port
- IPFIX
- La mise en miroir de port distant à l'aide de l'encapsulation GRE d'un port de commutateur logique qui se connecte à un conteneur est prise en charge. Pour plus d'informations, reportez-vous à la section « Comprendre le profil de commutation de mise en miroir de ports » dans le *Guide d'administration de NSX-T*. Toutefois, la mise en miroir des ports CIF-VIF n'est pas prise en charge via l'interface utilisateur du gestionnaire.

Commandes d'interface de ligne de commande

Pour exécuter des commandes d'interface de ligne de commande, connectez-vous au conteneur NSX-T Container Plug-in, ouvrez un terminal et exécutez la commande `nsxcli`.

Vous pouvez également obtenir l'invite d'interface de ligne de commande en exécutant la commande suivante sur un nœud :

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

Tableau 6-1. Commandes d'interface de ligne de commande pour le conteneur NCP

Type	vdmadmin
Statut	get ncp-master status
Statut	get ncp-nsx status
Statut	get ncp-watcher <watcher-name>
Statut	get ncp-watchers
Statut	get ncp-k8s-api-server status
Statut	check projects
Statut	check project <nom_projet>
Cache	get project-cache <project-name>
Cache	get project-caches
Cache	get namespace-cache <namespace-name>
Cache	get namespace-caches
Cache	get pod-cache <pod-name>
Cache	get pod-caches
Cache	get ingress-caches

Tableau 6-1. Commandes d'interface de ligne de commande pour le conteneur NCP (Suite)

Type	vdmadmin
Cache	get ingress-cache <nom_entrée>
Cache	get ingress-controllers
Cache	get ingress-controller <nom_contrôleur_entrée>
Cache	get network-policy-caches
Cache	get network-policy-cache <nom_espace>
Support	get ncp-log file <filename>
Support	get ncp-log-level
Support	set ncp-log-level <niveau_journal>
Support	get support-bundle file <filename>
Support	get node-agent-log file <filename>
Support	get node-agent-log file <filename> <node-name>

Tableau 6-2. Commandes d'interface de ligne de commande pour le conteneur de l'agent du nœud NSX

Type	vdmadmin
Statut	get node-agent-hyperbus status
Cache	get container-cache <nom-conteneur>
Cache	get container-caches

Tableau 6-3. Commandes d'interface de ligne de commande pour le conteneur du proxy Kube NSX

Type	vdmadmin
Statut	get ncp-k8s-api-server status
Statut	get kube-proxy-watcher <watcher-name>
Statut	get kube-proxy-watchers
Statut	dump ovs-flows

Commandes d'état pour le conteneur NCP

- Afficher l'état du nœud maître NCP

```
get ncp-master status
```


Exemple :

```
kubecode> 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
```

- Afficher l'état de la connexion entre NCP et NSX Manager

```
get ncp-nsx status
```

Exemple :

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

- Afficher l'état de l'observateur de l'entrée, l'espace de noms, l'espace et le service

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

Exemple 1 :

```
kubecode> 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
```

Exemple 2 :

```
kubecode> 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

```

- Afficher l'état de la connexion entre NCP et Kubernetes API Server

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

Exemple :

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

- Vérifier tous les projets ou un projet spécifique

```
check projects
check project <project-name>
```

Exemple :

```
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
```

Commandes de cache pour le conteneur NCP

- Obtenir le cache interne pour des projets ou des espaces de noms

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

Exemple :

```
kubecall> 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

kubecall> 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

kubecall> 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

```

■ Obtenir le cache interne pour des espaces

```

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

```

Exemple :

```

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

kubens> get pod-cache nsx.default.nginx-rc-uc2lv
  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

```

■ Obtenir les caches de stratégie réseau ou un cache spécifique

```

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

```

Exemple :

```

kubens> 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
```

Commandes de support pour le conteneur NCP

- Enregistrer le bundle de support NCP dans le magasin de fichiers

Le bundle de support est constitué des fichiers journaux de tous les conteneurs d'espaces avec l'étiquette **tier:nsx-networking**. Le fichier de bundle est au format tgz et enregistré dans le répertoire du magasin de fichiers par défaut d'interface de ligne de commande `/var/vmware/nsx/file-store`. Vous pouvez utiliser la commande de magasin de fichiers d'interface de ligne de commande pour copier le fichier de bundle sur un site distant.

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

Exemple :

```
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
```

- Enregistrer les journaux NCP dans le magasin de fichiers

Le fichier journal est enregistré au format tgz dans le répertoire du magasin de fichiers par défaut d'interface de ligne de commande `/var/vmware/nsx/file-store`. Vous pouvez utiliser la commande de magasin de fichiers d'interface de ligne de commande pour copier le fichier de bundle sur un site distant.

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

Exemple :

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

- Enregistrer les journaux de l'agent de nœud dans le magasin de fichiers

Enregistrez les journaux de l'agent de nœud d'un seul nœud ou de tous les nœuds. Les journaux sont enregistrés au format tgz dans le répertoire du magasin de fichiers par défaut d'interface de ligne de commande `/var/vmware/nsx/file-store`. Vous pouvez utiliser la commande de magasin de fichiers d'interface de ligne de commande pour copier le fichier de bundle sur un site distant.

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

Exemple :

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

- Obtenir et définir le niveau de journalisation

Les niveaux de journalisation disponibles sont NOTSET, DEBUG, INFO, WARNING, ERROR et CRITICAL.

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

Exemple :

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

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

Commandes d'état pour le conteneur de l'agent du nœud NSX

- Affichez l'état de la connexion entre l'agent de nœud et HyperBus sur ce nœud.

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

Exemple :

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

Commandes de cache pour le conteneur de l'agent du nœud NSX

- Obtenir le cache interne pour les conteneurs d'agents du nœud NSX.

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

Exemple 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
```


Exemple 2 :

```
kubecfg> 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
```

Commandes d'état pour le conteneur du proxy Kube NSX

- Afficher l'état de la connexion entre Kube Proxy et Kubernetes API Server

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

Exemple :

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

- Afficher l'état de l'observateur Kube Proxy

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

Exemple 1 :

```
kubecfg> 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
```

Exemple 2 :

```
kubecfg> 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

```

■ Vider les flux OVS sur un nœud

```
dump ovs-flows
```

Exemple :

```

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
  actions=ct(table=1)
    cookie=0x0, duration=8.898s, table=0, n_packets=0, n_bytes=0, idle_age=8, priority=0
    actions=NORMAL
      cookie=0x0, duration=8.759s, table=1, n_packets=0, n_bytes=0, idle_age=8,
      priority=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,
        priority=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,
          priority=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
            actions=NORMAL
              cookie=0x0, duration=8.856s, table=2, n_packets=0, n_bytes=0, idle_age=8, actions=NORMAL

```

Codes d'erreur

Cette section répertorie les codes d'erreur renvoyés par les différents composants.

Codes d'erreur NCP

Code d'erreur	Description
NCP00001	Configuration non valide
NCP00002	Échec de l'initialisation
NCP00003	État non valide
NCP00004	Adaptateur non valide
NCP00005	Certificat introuvable
NCP00006	Jeton introuvable
NCP00007	Configuration de NSX non valide
NCP00008	Balise NSX non valide
NCP00009	Échec de la connexion à NSX

Code d'erreur	Description
NCP00010	Balise de nœud introuvable
NCP00011	Port de commutateur logique du nœud non valide
NCP00012	Échec de la mise à jour de la VIF parent
NCP00013	VLAN épuisé
NCP00014	Échec de la libération du VLAN
NCP00015	Pool d'adresses IP épuisé
NCP00016	Échec de la libération de l'adresse IP
NCP00017	Bloc d'adresses IP épuisé
NCP00018	Échec de la création du sous-réseau IP
NCP00019	Échec de la suppression du sous-réseau IP
NCP00020	Échec de la création du pool d'adresses IP
NCP00021	Échec de la suppression du pool d'adresses IP
NCP00022	Échec de la création du routeur logique
NCP00023	Échec de la mise à jour du routeur logique
NCP00024	Échec de la suppression du routeur logique
NCP00025	Échec de la création du commutateur logique

Code d'erreur	Description
NCP00026	Échec de la mise à jour du commutateur logique
NCP00027	Échec de la suppression du commutateur logique
NCP00028	Échec de la création du port de routeur logique
NCP00029	Échec de la suppression du port de routeur logique
NCP00030	Échec de la création du port de commutateur logique
NCP00031	Échec de la mise à jour du port de commutateur logique
NCP00032	Échec de la suppression du port de commutateur logique
NCP00033	Stratégie réseau introuvable
NCP00034	Échec de la création du pare-feu
NCP00035	Échec de la lecture du pare-feu
NCP00036	Échec de la mise à jour du pare-feu
NCP00037	Échec de la suppression du pare-feu
NCP00038	Plusieurs pare-feu trouvés
NCP00039	Échec de la création du NSGroup
NCP00040	Échec de la suppression du NSGroup
NCP00041	Échec de la création de l'ensemble d'adresses IP
NCP00042	Échec de la mise à jour de l'ensemble d'adresses IP

Code d'erreur	Description
NCP00043	Échec de la suppression de l'ensemble d'adresses IP
NCP00044	Échec de la création de la règle SNAT
NCP00045	Échec de la suppression de la règle SNAT
NCP00046	Échec de la connexion à l'API d'adaptateur
NCP00047	Exception de l'observateur d'adaptateur
NCP00048	Échec de la suppression du service d'équilibrage de charge
NCP00049	Échec de la création du serveur virtuel d'équilibrage de charge
NCP00050	Échec de la mise à jour du serveur virtuel d'équilibrage de charge

Code d'erreur	Description
NCP00051	Échec de la suppression du serveur virtuel d'équilibrage de charge
NCP00052	Échec de la création du pool d'équilibrage de charge
NCP00053	Échec de la mise à jour du pool d'équilibrage de charge
NCP00054	Échec de la suppression du pool d'équilibrage de charge
NCP00055	Échec de la création de la règle d'équilibrage de charge
NCP00056	Échec de la mise à jour de la règle d'équilibrage de charge
NCP00057	Échec de la suppression de la règle d'équilibrage de charge
NCP00058	Échec de la libération de l'adresse IP de pool d'équilibrage de charge
NCP00059	Serveur virtuel d'équilibrage de charge et association de service introuvables
NCP00060	Échec de la mise à jour du NSGroup
NCP00061	Échec de l'obtention des règles de pare-feu
NCP00062	Aucun critère de NSGroup
NCP00063	VM de nœud introuvable
NCP00064	VIF de nœud introuvable
NCP00065	Échec de l'importation de certificat
NCP00066	Échec de l'annulation de l'importation de certificat
NCP00067	Échec de la mise à jour de la liaison SSL
NCP00068	Profil SSL introuvable
NCP00069	Pool d'adresses IP introuvable
NCP00070	Cluster Edge T0 introuvable
NCP00071	Échec de la mise à jour du pool d'adresses IP
NCP00072	Échec du répartiteur
NCP00073	Échec de la suppression de la règle NAT
NCP00074	Échec de l'obtention du port de routeur logique
NCP00075	Échec de la validation de la configuration de NSX

Code d'erreur	Description
NCP00076	Échec de la mise à jour de la règle SNAT
NCP00077	Chevauchement de la règle SNAT
NCP00078	Échec de l'ajout des points de terminaison d'équilibrage de charge
NCP00079	Échec de la mise à jour des points de terminaison d'équilibrage de charge
NCP00080	Échec de la création du pool de règles d'équilibrage de charge
NCP00081	Serveur virtuel d'équilibrage de charge introuvable
NCP00082	Échec de la lecture de l'ensemble d'adresses IP
NCP00083	Échec de l'obtention du pool SNAT
NCP00084	Échec de la création du service d'équilibrage de charge
NCP00085	Échec de la mise à jour du service d'équilibrage de charge
NCP00086	Échec de la mise à jour du port de routeur logique
NCP00087	Échec de l'initialisation de l'équilibrage de charge
NCP00088	Pool d'adresses IP non unique
NCP00089	Erreur de synchronisation du cache d'équilibrage de charge de couche 7
NCP00090	Erreur : le pool d'équilibrage de charge n'existe pas
NCP00091	Erreur d'initialisation du cache de la règle d'équilibrage de charge
NCP00092	Échec du processus SNAT
NCP00093	Erreur de certificat par défaut d'équilibrage de charge
NCP00094	Échec de la suppression du point de terminaison d'équilibrage de charge
NCP00095	Projet introuvable
NCP00096	Accès au pool refusé
NCP00097	Échec de l'obtention d'un service d'équilibrage de charge
NCP00098	Échec de la création d'un service d'équilibrage de charge
NCP00099	Erreur de synchronisation du cache de pool d'équilibrage de charge

Codes d'erreur de l'agent de nœud NSX

Code d'erreur	Description
NCP01001	Liaison montante OVS introuvable
NCP01002	Adresse MAC de l'hôte introuvable
NCP01003	Échec de la création du port OVS
NCP01004	Aucune configuration de l'espace
NCP01005	Échec de la configuration de l'espace
NCP01006	Échec de l'annulation de la configuration de l'espace
NCP01007	Socket CNI introuvable

Code d'erreur	Description
NCP01008	Échec de la connexion à CNI
NCP01009	Incompatibilité de la version de CNI
NCP01010	Échec de la réception du message CNI
NCP01011	Échec de la transmission du message CNI
NCP01012	Échec de la connexion à Hyperbus
NCP01013	Incompatibilité de la version d'Hyperbus
NCP01014	Échec de la réception du message Hyperbus
NCP01015	Échec de la transmission du message Hyperbus
NCP01016	Échec de l'envoi du paquet GARP
NCP01017	Échec de la configuration de l'interface

Codes d'erreur de nsx-kube-proxy

Code d'erreur	Description
NCP02001	Port de la passerelle de proxy non valide
NCP02002	Échec de la commande de proxy
NCP02003	Échec de la validation du proxy

Codes d'erreur de la CLI

Code d'erreur	Description
NCP03001	Échec du démarrage de la CLI
NCP03002	Échec de la création du socket de la CLI
NCP03003	Exception de socket de la CLI
NCP03004	Demande du client de CLI non valide
NCP03005	Échec de la transmission du serveur de CLI
NCP03006	Échec de la réception du serveur de CLI
NCP03007	Échec de l'exécution de la commande de CLI

Codes d'erreur Kubernetes

Code d'erreur	Description
NCP05001	Échec de la connexion à Kubernetes
NCP05002	Configuration de Kubernetes non valide
NCP05003	Échec de la demande Kubernetes
NCP05004	Clé Kubernetes introuvable

Code d'erreur	Description
NCP05005	Type Kubernetes introuvable
NCP05006	Exception de l'observateur Kubernetes
NCP05007	Longueur de la ressource Kubernetes non valide
NCP05008	Type de ressource Kubernetes non valide
NCP05009	Échec du handle de ressource Kubernetes
NCP05010	Échec du handle de service Kubernetes
NCP05011	Échec du handle de point de terminaison Kubernetes
NCP05012	Échec du handle d'entrée Kubernetes
NCP05013	Échec du handle de stratégie réseau Kubernetes
NCP05014	Échec du handle de nœud Kubernetes
NCP05015	Échec du handle d'espace de noms Kubernetes
NCP05016	Échec du handle d'espace Kubernetes
NCP05017	Échec du handle de secret Kubernetes
NCP05018	Échec du serveur principal Kubernetes par défaut
NCP05019	Expression de correspondance Kubernetes non prise en charge
NCP05020	Échec de la mise à jour de l'état Kubernetes
NCP05021	Échec de la mise à jour de l'annotation Kubernetes
NCP05022	Cache d'espace de noms Kubernetes introuvable
NCP05023	Secret Kubernetes introuvable
NCP05024	Serveur principal Kubernetes par défaut en cours d'utilisation
NCP05025	Échec du handle de service Kubernetes LoadBalancer

Codes d'erreur OpenShift

Code d'erreur	Description
NCP07001	Échec du handle de la route OC
NCP07002	Échec de la mise à jour de l'état de la route OC