OVHcloud
Documentation

On-Prem Cloud Platform

Généré le 11 août 2026

Sommaire

Premiers pas

Intégration réseau et connectivité pour OPCP

Découvrez comment OPCP s'intègre à votre réseau : réseau de données, réseau d'administration (OOB) et flux à autoriser selon votre mode de gestion

Objectif

On-Prem Cloud Platform (OPCP) est une solution de cloud privé déployée sur vos propres infrastructures, livrée sous forme d'une pile intégrée (serveurs, commutateurs, automatisation, plan de contrôle). Avant et pendant le déploiement, votre équipe doit comprendre comment OPCP s'interconnecte avec votre réseau d'entreprise : par où il communique, quels flux il émet, quels services il consomme chez vous, et – si vous avez souscrit le mode Fully managed by OVHcloud – comment OVHcloud accède à la plateforme pour l'opérer à distance.

Ce guide a pour objectif de vous présenter l'architecture réseau d'OPCP, en distinguant :

- le **réseau d'administration (Out-of-Band, OOB)** utilisé pour piloter la plateforme,
- le **réseau de données (*data plane*)** utilisé par les charges de travail hébergées sur les serveurs OPCP.

Il s'adresse aussi bien aux **architectes** préparant un déploiement qu'aux **administrateurs** prenant en main une plateforme déjà livrée.

Prérequis

- Une vision générale de votre architecture réseau d'entreprise (VLAN, plan d'adressage, équipements de bordure, firewalls).
- Connaître le **modèle de souscription OPCP** retenu (Self-managed ou Fully managed by OVHcloud).
- Avoir parcouru la liste des prérequis techniques OPCP à fournir avant le déploiement.

En pratique

1. Les niveaux de gestion d'OPCP

OPCP est proposé selon deux niveaux de délégation. Le niveau choisi détermine **qui opère** le plan de contrôle (*control plane*) de la plateforme et donc **quels flux d'interconnexion** doivent être ouverts vers OVHcloud.

Mode	**Qui opère le Core (*control plane*) ?**	**Qui opère les services applicatifs ?**	**Interconnexion vers OVHcloud requise ?**
Self-managed	Le client	Le client	Non
Fully managed by OVHcloud	OVHcloud	OVHcloud	**Oui** (IPsec)

Quel que soit le mode, **la gestion de ce que vous déployez au-dessus des services fournis par OPCP reste de votre responsabilité** : machines virtuelles, conteneurs, instances bare-metal, applications, données

métier, sauvegardes applicatives, etc. OPCP fournit la plateforme ; vous restez maître de l'usage que vous en faites.

Quel que soit le mode, l'architecture matérielle et la séparation logique entre les réseaux sont **identiques**. Les différences entre Self-managed et Fully managed by OVHcloud portent sur deux points :

1. **Le niveau d'accès aux interfaces d'administration.** En mode Self-managed, vos équipes disposent du contrôle complet des interfaces de gestion d'OPCP : API OpenStack, SSH aux contrôleurs, et deux outils en ligne de commande embarqués sur la plateforme :
    - `opcp-cli`, qui permet de **configurer OPCP** (déclaration des ressources de la plateforme, gestion des composants, opérations d'exploitation),
    - `opcp-diag`, qui produit un **état de santé de la plateforme** et **extrait les logs** nécessaires à un diagnostic à distance lorsqu'un incident survient.

    En mode Fully managed by OVHcloud, ces interfaces sont opérées par les équipes OVHcloud ; vos équipes conservent un accès en consultation et en exploitation applicative, mais l'administration plateforme est déléguée.
2. **Le besoin d'un lien IPsec entre votre réseau d'administration et OVHcloud.** Indispensable en mode Fully managed by OVHcloud pour que les équipes OVHcloud puissent opérer la plateforme à distance, ce lien n'existe pas en mode Self-managed.

2. Les deux plans réseau d'OPCP

OPCP repose sur une séparation stricte entre deux réseaux ayant des rôles, des équipements et des usages très différents. **Cette séparation est structurante** : elle conditionne la sécurité, la résilience et la qualité de service de la plateforme.

Le réseau de données (*data plane*)

Le réseau de données est le réseau **emprunté par les charges de travail** hébergées sur les serveurs OPCP (machines virtuelles, conteneurs, instances bare-metal exposées via OpenStack). C'est par lui que transitent :

- le trafic applicatif entrant et sortant des instances,
- les communications entre instances au sein de l'OPCP,
- les flux entre OPCP et le reste de votre système d'information côté production (bases de données, services internes, utilisateurs finaux, etc.).

Côté matériel, le réseau de données s'appuie sur :

- les **interfaces data** des serveurs,
- les **commutateurs Top-of-Rack (ToR)** redondés,
- pour les déploiements multi-baies, deux familles supplémentaires de commutateurs :
    - les commutateurs **edge**, qui assurent la **jonction entre le réseau interne d'OPCP et votre réseau amont** : ce sont eux qui portent la configuration des uplinks vers votre *Customer Upstream Data Network*,
    - les commutateurs **spine**, qui permettent de **croître au-delà de 3 baies** en mutualisant l'interconnexion entre les ToR de plusieurs baies,

- des **liens uplink fibre** vers votre réseau amont (*Customer Upstream Data Network*) en **40 GbE ou 100 GbE**, selon les optiques validées par OVHcloud pour votre déploiement.

Le réseau de données est **piloté de bout en bout par OPCP** via OpenStack Neutron : la création d'un réseau, d'un sous-réseau ou d'une règle de sécurité depuis Horizon ou l'API se propage automatiquement aux commutateurs sous-jacents.

Le réseau d'administration (Out-of-Band, OOB)

Le réseau OOB est le réseau **par lequel OPCP s'administre lui-même** et par lequel vous (et, en mode Fully managed by OVHcloud, les équipes OVHcloud) accédez aux interfaces de gestion. Il est **physiquement et logiquement séparé** du réseau de données. Il porte :

- l'accès aux **API OpenStack**, à l'interface **Horizon** et aux outils de gestion d'OPCP (HTTPS),
- l'accès SSH aux **OPCP Core Controllers** (commandes `opcp-cli`, `opcp-diag`),
- les **échanges inter-contrôleurs** qui assurent la **résilience du plan de contrôle** en modes STANDARD POD et FABRIC (synchronisation d'état, élection de quorum, bascule de la VIP entre les trois OPCP Core Controllers),
- la collecte de logs et de métriques émise depuis OPCP vers vos services internes,
- les flux sortants d'OPCP vers vos services d'infrastructure (NTP, DNS, S3[1] de sauvegarde, LDAP, syslog).

Côté matériel, le réseau OOB s'appuie sur :

- les **OPCP Core Controllers**, équipés d'un uplink OOB qui négocie en **10 GbE** en lien avec les capacités de votre infrastructure amont, et d'un port 1 GbE BMC,
- des **liens uplink** vers votre réseau OOB amont (*Customer Upstream OOB Network*).

> **Warning**
>
> En modes STANDARD POD et FABRIC, les trois OPCP Core Controllers sont accessibles via une adresse IP virtuelle commune dont le bon fonctionnement nécessite une communication au sein d'un réseau L2 non taggé commun entre les trois OPCP Core Controllers. La disponibilité du plan de contrôle dépend donc directement de la disponibilité de cette connexion réseau : une panne prolongée du réseau OOB client peut perturber l'accès à l'administration de la plateforme. Le mode **NANOPOD** n'est pas concerné, puisqu'il ne dispose que d'un seul contrôleur. Les données échangées entre les OPCP Core Controllers le sont au travers du réseau d'OPCP. Seul l'accès des administrateurs au plan de contrôle peut être affecté par des incidents sur le réseau OOB.

> **Warning**
>
> Aucune communication directe n'est possible entre le réseau de données et le réseau OOB depuis l'intérieur d'OPCP. Toute communication transverse passe obligatoirement par votre réseau amont, ce qui vous laisse la maîtrise des règles de sécurité entre les deux périmètres.

3. Topologies de déploiement

OPCP est livré selon trois modes de déploiement, dimensionnés en fonction du contexte d'usage. Ces modes diffèrent principalement par le nombre de baies supportées et la résilience du plan de contrôle (*control plane*).

La présence ou non de commutateurs **spine** et **edge** entre les baies et votre réseau amont en est une conséquence directe.

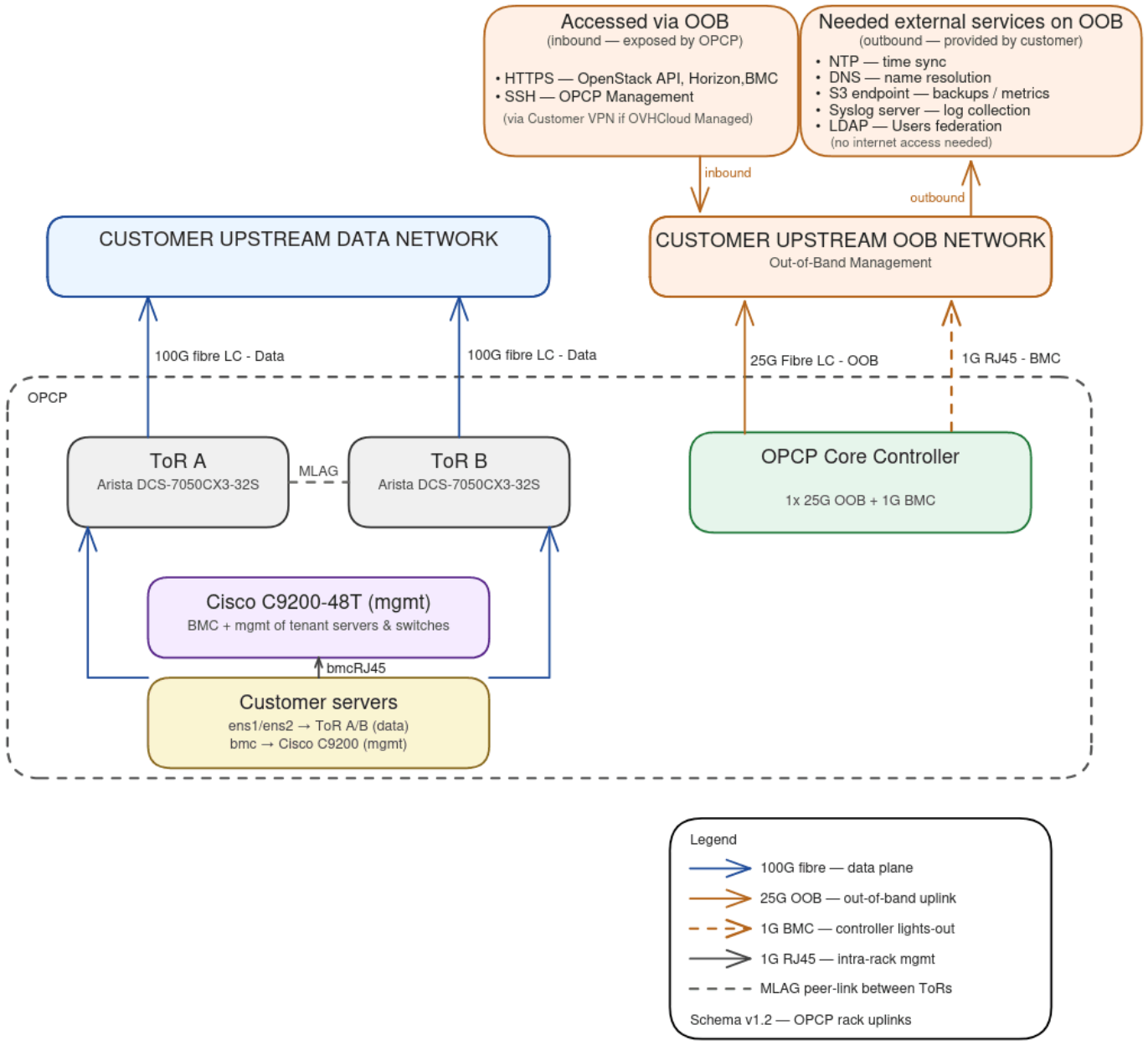
Mode	Cas d'usage	Résilience du plan de contrôle	Nombre de baies	Spine / Edge
NANOPOD	Démonstration, sites distants, infrastructures non résilientes (bureaux, edge)	Non résilient (1 contrôleur)	1	Non
STANDARD POD	Infrastructure résiliente compacte	Résilient (3 contrôleurs)	2 à 3	Edges uniquement
FABRIC	Déploiement datacenter avec infrastructure partagée	Résilient (3 contrôleurs)	3 à 10 (et au-delà avec *megaspine*)	Spine + Edges

Du point de vue de l'intégration réseau, le point clé à retenir est :

- en **NANOPOD**, le **réseau de données est configuré directement sur les commutateurs Top-of-Rack (ToR)** : ce sont eux qui remontent vers votre *Customer Upstream Data Network* ;
- en **STANDARD POD** et **FABRIC**, le **réseau de données est configuré sur les commutateurs edge**, qui s'intercalent entre les ToR et votre réseau amont. La configuration des ToR reste interne à OPCP.

Le nombre d'**OPCP Core Controllers** suit le mode de déploiement : **1 contrôleur en NANOPOD**, **3 contrôleurs en STANDARD POD et FABRIC**.

Configuration à 1 contrôleur (NANOPOD)

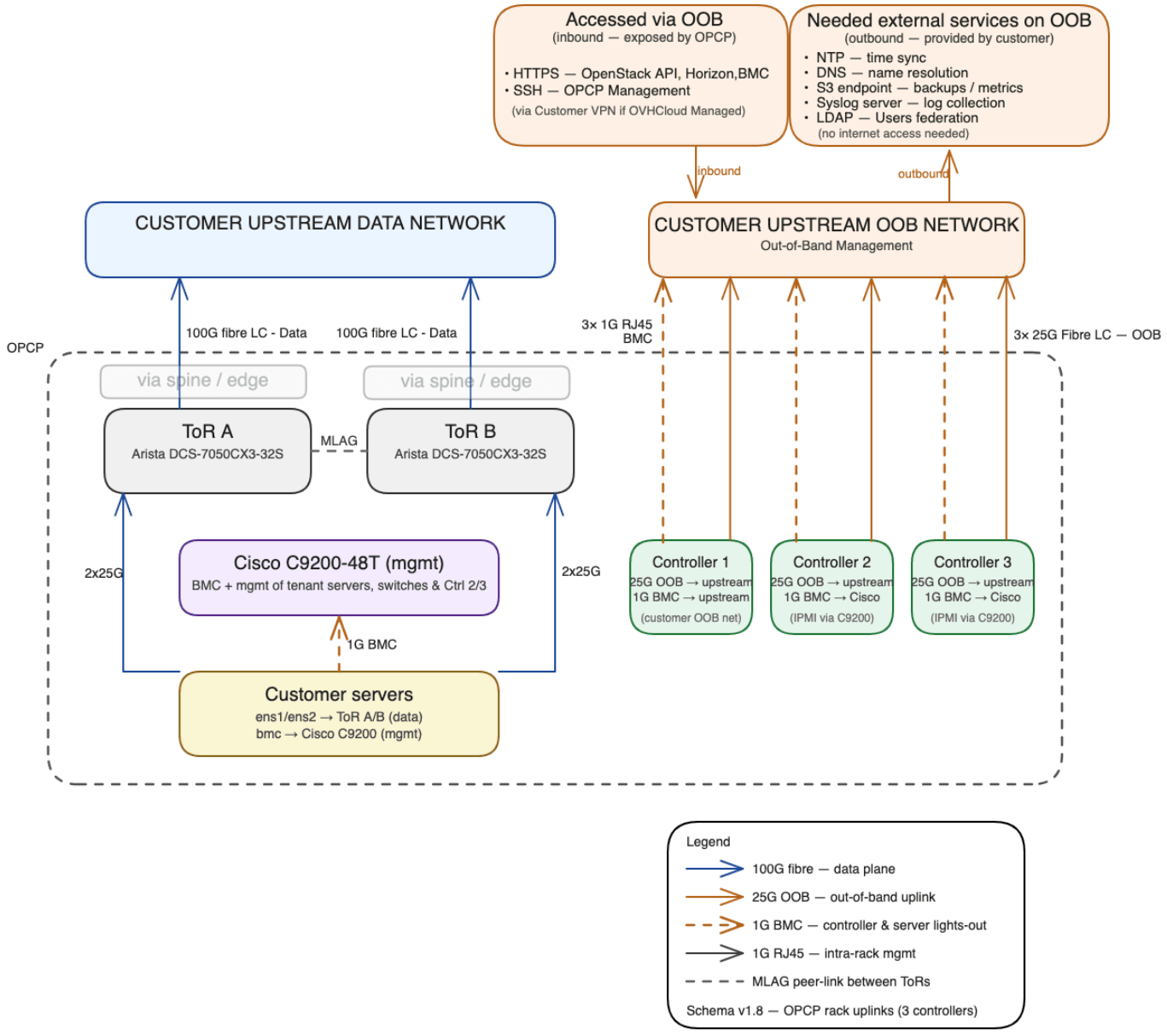


Les éléments clés sont :

- **Customer Servers** : les serveurs hébergeant les charges de travail. Leurs interfaces data sont connectées aux deux ToR ; leur BMC est rattaché au switch de management.
- **ToR A / ToR B** (Arista DCS-7050CX3-32S) : deux commutateurs Top-of-Rack redondés. En mode NANOPOD, **ils portent directement le réseau de données et remontent vers votre *Customer Upstream Data Network*** via des liens fibre en **40 GbE ou 100 GbE**, selon les optiques validées par OVHcloud.
- **Switch de management (mgmt)** : commutateur de management dédié, qui agrège les BMC des serveurs et les ports de management des ToR.
- **OPCP Core Controller** unique, raccordé via un uplink OOB en **25 GbE ou 10 GbE** vers votre *Customer Upstream OOB Network*, plus un port 1 GbE BMC pour son propre lights-out.

> **Warning**
>
> Avec un seul contrôleur, **il n'y a pas de redondance du plan de contrôle** : une panne matérielle ou une opération de maintenance sur ce contrôleur interrompt l'administration de la plateforme (les charges de travail déjà déployées continuent à s'exécuter sur les serveurs). Pour tout environnement de production exigeant une haute disponibilité du plan de contrôle, retenez les modes STANDARD POD ou FABRIC.

Configuration à 3 contrôleurs (STANDARD POD et FABRIC)



Par rapport au mode NANOPOD, les différences sont :

- **3 OPCP Core Controllers** au lieu d'un seul, chacun raccordé au réseau OOB amont avec un uplink en **25 GbE ou 10 GbE** et un lien BMC 1 GbE. Une **adresse VIP** (*Virtual IP*) est partagée entre les trois contrôleurs : c'est cette adresse qui est utilisée pour atteindre les API OpenStack et l'interface Horizon, indépendamment du contrôleur actif.
- Les BMC des trois contrôleurs, comme ceux des serveurs, sont exposés sur votre réseau OOB amont. Cela garantit un accès lights-out à l'ensemble des équipements en cas de défaillance d'un composant interne à OPCP.

- Des **commutateurs edge** s'intercalent entre les ToR et votre réseau amont. **Le réseau de données est configuré sur ces edges**, et non plus sur les ToR. En mode **FABRIC**, des équipements **spine** sont également présents pour mutualiser plusieurs baies ; au-delà de dix baies, une couche **megaspine** peut être ajoutée.

4. Comment OPCP s'intègre à votre environnement

L'intégration d'OPCP repose sur **deux périmètres réseau** que vous fournissez et que vous gardez sous votre contrôle.

Côté réseau de données (*Customer Upstream Data Network*)

C'est votre réseau de production. OPCP s'y raccorde via les liens fibre des ToR (en mode NANOPOD) ou des edges (en mode STANDARD POD et FABRIC), à des débits de **40 GbE ou 100 GbE** selon les optiques validées par OVHcloud.

La configuration du raccordement réseau (uplinks vers votre amont) est pilotée par OPCP, à travers une API exposée par la plateforme. Selon le mode de déploiement :

- en **NANOPOD**, la configuration est appliquée directement sur les **ToR** ;
- en **STANDARD POD** et **FABRIC**, elle est appliquée sur les **edges**.

Au-delà de ce raccordement, le contenu et la segmentation du réseau de données restent **entièrement pilotés par OPCP** via OpenStack Neutron : chaque réseau Neutron est représenté sur les uplinks selon l'un des trois modes ci-dessous.

Mode access

Le port uplink est rattaché à **un unique réseau Neutron** et présente le trafic en **802.1Q** classique à votre équipement amont. C'est le mode le plus simple, adapté lorsqu'un port amont est dédié à un seul réseau OPCP.

Si vos instances poussent leurs propres tags VLAN à l'intérieur du réseau Neutron (taggage au niveau de l'hyperviseur, besoins spécifiques des invités, etc.), ces tags internes sont gérés de manière transparente par OPCP et votre équipement amont ne voit toujours que des trames 802.1Q standard — il n'y a rien de plus à configurer de votre côté.

Mode trunk

Le port uplink transporte **plusieurs réseaux Neutron** sur une seule interface physique, chacun étant distingué par son propre identifiant VLAN. Comme le tag VLAN défini par OPCP est préservé de bout en bout et jamais décapsulé, l'encapsulation que voit votre équipement amont dépend de ce qui se passe **à l'intérieur** de chaque réseau Neutron :

- **802.1Q** — si vos instances n'utilisent pas de VLAN à l'intérieur du réseau Neutron. Votre équipement amont doit être configuré pour reconnaître les identifiants VLAN attribués par OPCP.
- **802.1ad (QinQ)** — si vos instances poussent leurs propres tags VLAN à l'intérieur du réseau Neutron. Le VLAN attribué par OPCP devient le tag externe et les VLAN des instances deviennent les tags internes. **Votre équipement amont doit être capable de gérer la double encapsulation.**

Les deux peuvent coexister sur un même trunk : chaque réseau Neutron est indépendant, donc certains peuvent être en 802.1Q simple et d'autres en 802.1ad, selon l'usage qui est fait de chacun.

Côté réseau d'administration (*Customer Upstream OOB Network*)

C'est le périmètre depuis lequel OPCP est administré. Vous devez y prévoir :

- un **sous-réseau dédié** à OPCP, avec une passerelle par défaut. C'est sur ce sous-réseau que sont positionnés les contrôleurs (et leur VIP en configuration HA),
- une **résolution DNS** : un nom de domaine que vous déléguez à OPCP (par exemple `*.opcp01.exemple.com`), accompagné d'un *forwarder* depuis vos résolveurs DNS internes vers la VIP OPCP,
- des **règles de filtrage** autorisant les flux décrits dans la matrice ci-dessous,
- un **accès à au moins une source de temps NTP** depuis le réseau d'administration d'OPCP (obligatoire) et, selon les options retenues, à vos autres services internes (syslog, S3, LDAP).

5. Matrice des flux à autoriser

Tous les flux suivants partent ou arrivent sur le **réseau d'administration (OOB)** d'OPCP. Aucun n'utilise le réseau de données.

Flux entrants (vous → OPCP)

Source	Destination	Service	Protocole / Port
Postes d'administration / utilisateurs OPCP	VIP des contrôleurs OPCP	API OpenStack, Horizon, interfaces de gestion	TCP/443
Administrateurs (mode Self-managed)	Contrôleurs OPCP	SSH (`opcp-cli`, `opcp-diag`)	TCP/22

Flux sortants (OPCP → vous)

Source	Destination	Service	Protocole / Port	Obligatoire
Réseau d'administration OPCP	Vos serveurs NTP	Synchronisation horaire	UDP/123	**Obligatoire**
Réseau d'administration OPCP	Vos résolveurs DNS	Résolution de noms externes	UDP/53	Optionnel
Réseau d'administration OPCP	Vos serveurs Syslog	Centralisation des logs	UDP/514 ou TCP/514	Recommandé
Réseau d'administration OPCP	Votre endpoint S3	Backup de l'infrastructure	TCP/443	Recommandé
Réseau d'administration OPCP	Votre endpoint S3	Stockage long terme des métriques	TCP/443	Recommandé
Réseau d'administration OPCP	Votre Active Directory / IdP	Fédération d'identité LDAP	TCP/636 (LDAPS)	Optionnel

> **Info**
>
> OPCP est conçu pour fonctionner en mode **air-gap** : aucun de ces flux n'a besoin d'Internet. Les seules destinations attendues sont les services de votre propre système d'information.

6. L'interconnexion OVHcloud (mode Fully managed by OVHcloud)

Si vous avez souscrit l'offre **Fully managed by OVHcloud**, OVHcloud doit pouvoir atteindre votre réseau d'administration OPCP pour assurer l'exploitation et le support à distance. Cette interconnexion repose sur un **tunnel IPsec site-à-site** entre un équipement de bordure côté OVHcloud et un équipement de bordure côté client.

Les paramètres par défaut proposés par OVHcloud sont :

- **IKEv2 uniquement** (IKEv1 n'est pas supporté),
- authentification par **PSK** (Pre-Shared Key),
- chiffrement **AES-256**,
- hash **SHA-256**,
- groupe Diffie-Hellman **14**,
- **PFS** (Perfect Forward Secrecy) activé.

> **Warning**
>
> OVHcloud déconseille fortement d'abaisser le niveau de chiffrement en deçà d'AES-256. Si une contrainte de votre côté empêche d'utiliser ces paramètres, contactez votre point de contact OVHcloud pour étudier les alternatives.

Pour préparer cette interconnexion, vous devez fournir à OVHcloud :

- l'**IP publique** de votre endpoint IPsec,
- le **sous-réseau** côté client à exposer dans le tunnel (typiquement le sous-réseau d'administration d'OPCP),
- la **PSK** convenue (transmise via un canal sécurisé : voir avec votre point de contact OVHcloud).

Une fois le tunnel établi, les équipes OVHcloud accèdent à votre OPCP **uniquement** via cette interconnexion, sur les mêmes flux que ceux décrits dans la matrice ci-dessus. Aucun accès direct depuis Internet n'est ouvert sur votre plateforme.

7. Plan d'adressage et nommage

Lors de la préparation du déploiement, vous fournirez à OVHcloud :

- le **sous-réseau** alloué au réseau d'administration d'OPCP et sa **passerelle par défaut** (exemple : subnet `172.30.1.0/24`, gateway `172.30.1.254`),
- les **noms et adresses IP** des contrôleurs OPCP, ainsi que la **VIP** partagée en configuration HA (exemples : `opcp-controller-0` / `172.30.1.10`, `opcp-controller-1` / `172.30.1.11`, `opcp-controller-2` / `172.30.1.12`, VIP `172.30.1.5`),
- le **nom de domaine** que vous déléguez à OPCP (exemple : `opcp01.exemple.com`).

Le plan d'adressage et la convention de nommage sont **entièrement à votre main** : OVHcloud n'impose pas de format. Reportez-vous au guide « prérequis techniques OPCP » pour la liste exhaustive des éléments à transmettre.

Aller plus loin

- Prérequis techniques pour le déploiement
- Mise en route de votre OPCP

Pour une formation ou une assistance technique sur la mise en œuvre de nos solutions, contactez votre commercial ou consultez la page Professional Services pour obtenir un devis et faire analyser votre projet par nos experts.

Échangez avec notre communauté d'utilisateurs.

[1] : S3 est une marque déposée appartenant à Amazon Technologies, Inc. Les services de OVHcloud ne sont pas sponsorisés, approuvés, ou affiliés de quelque manière que ce soit.

Prérequis techniques pour le déploiement

Découvrez la liste des éléments de configuration que vous devez fournir à OVHcloud pour préparer et déployer votre plateforme OPCP

Objectif

Le déploiement d'**On-Prem Cloud Platform (OPCP)** sur votre site nécessite la collecte préalable d'un ensemble d'informations de configuration. Certaines sont indispensables au bootstrap initial et ne pourront plus être modifiées après installation ; d'autres sont optionnelles ou ajustables ultérieurement. Préparer cette liste en amont est la **condition principale d'un déploiement rapide et sans aller-retour**.

Ce guide a pour objectif de vous présenter, pour chaque prérequis, à quoi il sert, comment le formuler et quel est son caractère obligatoire ou optionnel.

Il s'adresse aussi bien aux **architectes** préparant la livraison qu'aux **administrateurs** souhaitant comprendre ce qui a été paramétré sur leur plateforme.

Prérequis

- Avoir parcouru le guide « Intégration réseau et connexion à la plateforme », qui décrit l'architecture réseau dans laquelle s'inscrivent ces prérequis.
- Connaître le **modèle de souscription OPCP** retenu (Self-managed ou Fully managed by OVHcloud).
- Disposer d'un **point de contact OVHcloud** identifié pour la livraison, à qui transmettre les éléments collectés.

En pratique

1. Comment lire ce guide

Pour chaque prérequis, vous trouverez :

- une **description** expliquant son rôle dans la plateforme,
- le **caractère obligatoire ou optionnel** de l'information,
- la **possibilité de modification après installation**,
- les **flux réseau associés** lorsque le prérequis implique une communication entre OPCP et un service de votre environnement,
- un **exemple générique** pour vous aider à formuler la valeur attendue.

> **Info**
>
> **Transmission des éléments sensibles.** Plusieurs prérequis impliquent des secrets : clé pré-partagée IPsec, clés d'accès S3[1], clés privées de certificats, etc. Ces éléments **ne doivent pas être transmis par un canal non sécurisé**. Convenez avec votre point de contact OVHcloud d'un canal de transmission approprié pour chaque secret.

2. Interconnexion OVHcloud (mode managé uniquement)

Champ	Valeur
Obligatoire	**Oui**, en mode Fully managed by OVHcloud
Modifiable après installation	Oui
Flux associé	Tunnel IPsec site-à-site entre votre endpoint et OVHcloud

Si vous avez souscrit l'offre **Fully managed by OVHcloud**, un tunnel IPsec doit être établi entre votre site et OVHcloud pour permettre l'exploitation et le support à distance. Vous devez fournir :

- l'**adresse IP publique** de votre endpoint IPsec,
- le **sous-réseau** que vous exposez côté client (typiquement le subnet d'administration d'OPCP),
- la **clé pré-partagée (PSK)** convenue pour l'authentification, à transmettre via un canal sécurisé.

Les paramètres cryptographiques par défaut sont **IKEv2 + PSK + AES-256 + SHA-256 + DH group 14 + PFS**. Si votre politique de sécurité exige des paramètres différents, abordez le sujet avec votre point de contact OVHcloud avant le déploiement.

> **Warning**
> IKEv1 n'est pas supporté. L'abaissement du niveau de chiffrement en dessous d'AES-256 est déconseillé : à n'envisager qu'en dernier recours.

En mode **Self-managed**, ce prérequis ne s'applique pas.

3. Réseau d'administration

Champ	Valeur
Obligatoire	**Oui**
Modifiable après installation	**Non**
Flux associé	Aucun (configuration locale à OPCP)

Définition du réseau de management sur lequel OPCP sera positionné. Vous devez fournir :

- le **sous-réseau** alloué à OPCP (exemple : `172.30.1.0/24`),
- l'**adresse de la passerelle par défaut** (exemple : `172.30.1.254`).

Ce réseau accueillera les contrôleurs OPCP et leur VIP. Il sert également de point de départ aux flux sortants vers vos services internes (NTP, DNS, syslog, S3, LDAP).

> **Warning**
> Ces paramètres sont **figés au déploiement** : le sous-réseau et la passerelle ne peuvent pas être modifiés après l'installation sans réinstallation complète. Validez-les soigneusement en amont.

4. Variables d'environnement OPCP

Champ	Valeur
Obligatoire	**Oui**
Modifiable après installation	**Non**
Flux associé	Aucun

OPCP utilise plusieurs variables pour identifier le déploiement de manière unique. Elles sont reprises dans Netbox, dans les logs, dans le monitoring et dans plusieurs autres composants. Vous devez fournir une valeur pour chacune :

Variable	Rôle	Exemple
`env`	Identifiant logique de l'environnement OPCP	`opcp-prod-0`
`region`	Code de la région ou de la zone géographique	`par`
`stage`	Étape de cycle de vie	`prod`, `staging`, `dev`
`org`	Organisation propriétaire	`mycompany`
`site`	Identifiant du site physique	`dc1`
`location`	Localisation précise dans le site	`R11-1`

> **Info**
>
> **Aucune convention de nommage n'est imposée par OVHcloud.** Adoptez la nomenclature qui correspond à votre référentiel interne. Choisissez-la avec soin : ces valeurs seront figées et utilisées dans de nombreux composants.

5. Noms et adresses des contrôleurs OPCP

Champ	Valeur
Obligatoire	**Oui**
Modifiable après installation	**Non**
Flux associé	Aucun (configuration locale à OPCP)

Pour chaque **OPCP Core Controller** de votre déploiement, fournissez :

- le **nom d'hôte** (FQDN ou nom court selon votre convention),
- l'**adresse IP** sur le réseau d'administration.

En configuration à 3 contrôleurs, fournissez également la **VIP (Virtual IP)** partagée entre les trois nœuds, qui constituera le point d'entrée unique vers les API OpenStack et l'interface Horizon.

Exemple en configuration 3 contrôleurs :

Élément	Nom	IP
Contrôleur 0	`opcp-controller-0`	`172.30.1.10`
Contrôleur 1	`opcp-controller-1`	`172.30.1.11`
Contrôleur 2	`opcp-controller-2`	`172.30.1.12`
VIP	`opcp.exemple.com`	`172.30.1.5`

6. Certificats

Champ	Valeur
Obligatoire	**Oui**
Modifiable après installation	Oui
Flux associé	Aucun (configuration locale à OPCP)

OPCP doit présenter des certificats TLS valides pour ses interfaces (API OpenStack, Horizon, services internes). Trois options sont supportées ; vous devez choisir l'une d'entre elles avant le déploiement.

Option A — Recommandée - Autorité de certification intermédiaire fournie par le client

Vous fournissez une CA intermédiaire dérivée de votre PKI interne. OPCP l'utilise pour signer les certificats des services. Vous devez transmettre :

- le **certificat de la CA intermédiaire**,
- la **clé privée** associée (à transmettre via un canal sécurisé convenu avec votre point de contact OVHcloud).

La rotation des certificats émis sous cette intermédiaire reste gérée par CertManager côté OPCP.

Option B — Autorité de certification auto-signée générée par OPCP

OPCP génère sa propre autorité de certification interne et signe les certificats nécessaires. **Vous n'avez rien à fournir.** La rotation est gérée automatiquement par CertManager.

C'est l'option la plus simple, adaptée si votre politique de sécurité tolère une CA interne au périmètre OPCP. Vous devrez en revanche distribuer le certificat racine d'OPCP à vos clients pour éviter les alertes de sécurité.

Option C — Let's Encrypt

OPCP demande automatiquement les certificats auprès de Let's Encrypt. Cette option nécessite :

- une **méthode de validation** compatible (HTTP-01 ou DNS-01) accessible depuis OPCP,
- les **prérequis associés** à cette méthode (résolution publique du domaine, accès sortant aux serveurs ACME, etc.).

Précisez avec votre point de contact OVHcloud la méthode retenue et les configurations à mettre en place de votre côté.

7. NTP — Synchronisation horaire

Champ	Valeur
Obligatoire	**Oui**
Modifiable après installation	Oui
Flux associé	Réseau d'administration OPCP → vos serveurs NTP — UDP/123

OPCP a besoin d'une source de temps fiable pour le bon fonctionnement de l'ensemble de ses composants (cohérence des logs, validité des certificats, élection de quorum, etc.). Fournissez :

- une ou plusieurs **adresses IP** de serveurs NTP, **ou** des **noms DNS** si la résolution DNS est configurée.

Exemple :

- `172.30.1.200` / `ntp1.exemple.com`
- `172.30.1.201` / `ntp2.exemple.com`

Prévoyez l'ouverture du flux **UDP/123** depuis le réseau d'administration d'OPCP vers vos serveurs NTP.

8. DNS — Délégation de domaine vers OPCP

Champ	Valeur
Obligatoire	**Oui**
Modifiable après installation	**Non** (le forwarder DNS pouvant être adapté si les FQDN restent stables)
Flux associé	Vos résolveurs DNS → VIP OPCP — UDP/53 et TCP/53

OPCP expose ses interfaces via des FQDN sous un domaine que vous lui déléguez. Vous devez fournir :

- le **nom de domaine** alloué à cette plateforme OPCP (exemple : `opcp01.exemple.com`).

Côté infrastructure DNS, vous devez créer un **forwarder** depuis vos résolveurs internes vers la VIP OPCP, afin que toute requête pour `*.opcp01.exemple.com` soit résolue par OPCP.

9. Résolveurs DNS — Résolution externe

Champ	Valeur
Obligatoire	Optionnel
Modifiable après installation	Oui
Flux associé	Réseau d'administration OPCP → vos résolveurs DNS — UDP/53

Si OPCP doit résoudre des noms de domaine **externes** (par exemple le FQDN de votre endpoint S3 ou de votre annuaire LDAP), fournissez l'adresse d'un ou plusieurs résolveurs DNS.

Exemple :

- `172.30.1.100`
- `172.30.1.101`

Ce prérequis n'est nécessaire que si vous activez des intégrations s'appuyant sur des FQDN externes (sauvegarde S3, métriques long terme, fédération LDAP, etc.). Pour une plateforme strictement air-gappée et configurée par IP, il peut être omis.

10. Syslog — Centralisation des logs

Champ	Valeur
Obligatoire	Optionnel mais **recommandé**
Modifiable après installation	Oui
Flux associé	Réseau d'administration OPCP → vos serveurs syslog — UDP/514 ou TCP/514

OPCP peut envoyer ses logs vers une infrastructure syslog centralisée pour rétention long terme et analyse. Fournissez :

- l'**adresse IP** ou le FQDN du serveur syslog (exemple : `172.30.1.250`),
- le **port** d'écoute,
- le **protocole** : TCP ou UDP.

> **Info**
>
> Sans syslog externe, OPCP conserve les logs **localement** avec une rétention par défaut de **7 jours** et une volumétrie maximale de **50 Go**. Au-delà, les logs les plus anciens sont supprimés. Pour toute exigence de conformité imposant une rétention plus longue, configurez un syslog externe.

11. Sauvegarde — Endpoint S3

Champ	Valeur
Obligatoire	Optionnel mais **recommandé**
Modifiable après installation	Oui
Flux associé	Réseau d'administration OPCP → votre endpoint S3 — TCP/443

OPCP peut sauvegarder l'état de l'infrastructure (configurations, état du plan de contrôle, métadonnées) vers un endpoint compatible S3 que vous fournissez. Vous devez transmettre :

- l'**endpoint S3** (exemple : `s3.exemple.com:443`),
- la **clé d'accès** (Access Key),
- la **clé secrète** (Secret Key),
- deux **noms de bucket** dédiés aux sauvegardes :
  - un pour les sauvegardes **Velero** (volumes persistants — PV), exemple : `opcp01-backup-pv-dc1`,
  - un pour les sauvegardes **CNPG** (bases de données — DB), exemple : `opcp01-backup-db-dc1`,
- le **nom de la région** S3 (exemple : `paris`).

Les clés d'accès doivent être transmises via un canal sécurisé convenu avec votre point de contact OVHcloud.

> **Warning**
> Sans sauvegarde externe, vous ne disposez d'aucun mécanisme de restauration en cas d'incident majeur sur la plateforme. Cette option est très fortement recommandée pour tout environnement de production.

12. Stockage long terme des métriques — Endpoint S3

Champ	Valeur
Obligatoire	Optionnel mais **recommandé**
Modifiable après installation	Oui
Flux associé	Réseau d'administration OPCP → votre endpoint S3 — TCP/443

OPCP collecte en permanence des métriques sur la plateforme. Pour les conserver au-delà de la fenêtre de rétention locale, vous pouvez les externaliser vers un bucket S3. Les éléments à fournir sont les mêmes que pour la sauvegarde, mais le bucket doit être **distinct** :

- **endpoint S3**,
- **clé d'accès**,
- **clé secrète**,
- **nom du bucket** dédié aux métriques (exemple : `opcp01-metrics-dc1`),
- **nom de la région**.

Vous pouvez réutiliser le même endpoint et les mêmes identifiants S3 que pour la sauvegarde, à condition d'utiliser un bucket différent.

13. LDAP — Fédération d'identité

Champ	Valeur
Obligatoire	Optionnel
Modifiable après installation	Oui
Flux associé	Réseau d'administration OPCP → votre annuaire — TCP/636 (LDAPS)

OPCP intègre Keycloak comme fournisseur d'identité. Si vous souhaitez fédérer les accès avec votre annuaire d'entreprise (Active Directory ou autre serveur LDAP), fournissez :

- les **adresses IP** ou FQDN de vos serveurs LDAP (exemple : `ldap.exemple.com`, `10.3.0.5`, `10.3.0.6`),
- le **port** d'écoute (typiquement `636` pour LDAPS).

Sans fédération, les utilisateurs sont gérés directement dans le Keycloak embarqué d'OPCP.

14. Clé SSH publique

Champ	Valeur
Obligatoire	Optionnel mais **recommandé**
Modifiable après installation	Oui
Flux associé	Aucun (configuration locale à OPCP)

Pour accéder aux contrôleurs OPCP après le bootstrap initial (notamment pour utiliser `opcp-cli` et `opcp-diag`), fournissez une ou plusieurs **clés SSH publiques** des administrateurs côté client.

En mode **Fully managed by OVHcloud**, cette clé donne au client un accès aux outils d'administration en complément de l'accès des équipes OVHcloud.

> **Info**
> Si aucune clé n'est fournie, une nouvelle paire de clés SSH sera générée lors du déploiement. Pour maîtriser vos accès dès la livraison, il est préférable de fournir vous-même la clé publique.

Récapitulatif

Prérequis	Caractère	Modifiable après installation
Interconnexion OVHcloud (IPsec)	Obligatoire en mode managé	Oui
Réseau d'administration	Obligatoire	**Non**
Variables d'environnement OPCP	Obligatoire	**Non**
Noms et adresses des contrôleurs	Obligatoire	**Non**
Certificats	Obligatoire	Oui
NTP	Obligatoire	Oui
DNS (délégation de domaine)	Obligatoire	**Non**
Résolveurs DNS	Optionnel	Oui
Syslog	Optionnel mais recommandé	Oui
Sauvegarde S3	Optionnel mais recommandé	Oui
Stockage long terme des métriques S3	Optionnel mais recommandé	Oui
LDAP	Optionnel	Oui
Clé SSH publique	Optionnel mais recommandé	Oui

Portez une attention particulière aux prérequis **non modifiables après installation** : ils conditionneront durablement votre plateforme.

Aller plus loin

- Intégration réseau et connexion à la plateforme
- Mise en route de votre OPCP

Pour une formation ou une assistance technique sur la mise en œuvre de nos solutions, contactez votre commercial ou consultez la page Professional Services pour obtenir un devis et faire analyser votre projet par nos experts.

Échangez avec notre communauté d'utilisateurs.

[1] : S3 est une marque déposée appartenant à Amazon Technologies, Inc. Les services de OVHcloud ne sont pas sponsorisés, approuvés, ou affiliés de quelque manière que ce soit.

Mise en route de votre OPCP

Découvrez comment prendre en main et configurer votre On-Prem Cloud Platform

Objectif

Ce guide a été conçu pour vous présenter comment vous connecter aux interfaces graphiques de votre **OPCP** en tant que qu'administrateur de ce service.

Prérequis

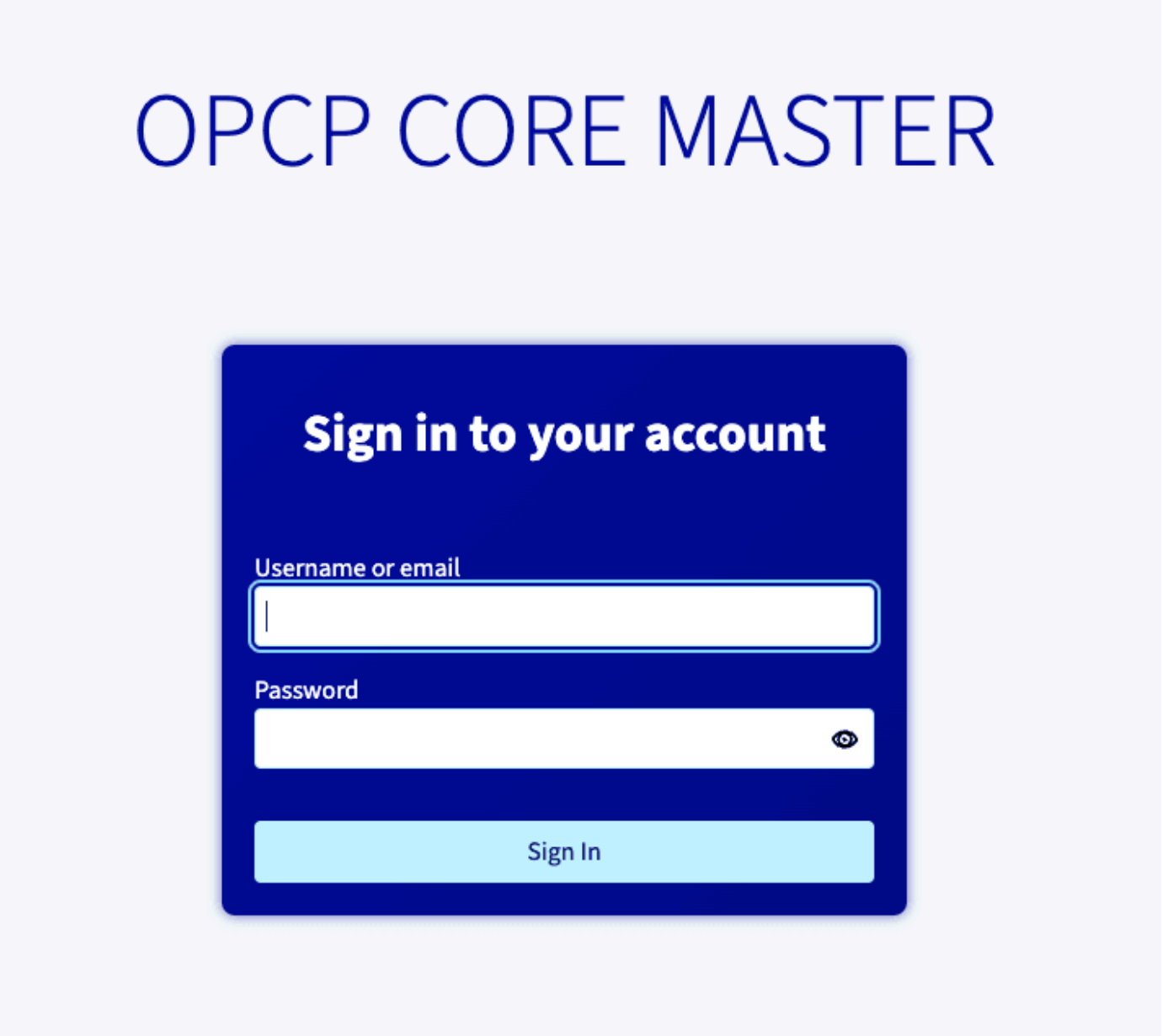
Afin de suivre ce guide, vous aurez besoin des informations suivantes :

- L'adresse **url** de l'interface de gestion définie pour la livraison du service.
- Les identifiants (login et mot de passe) fournies lors de la livraison du service.

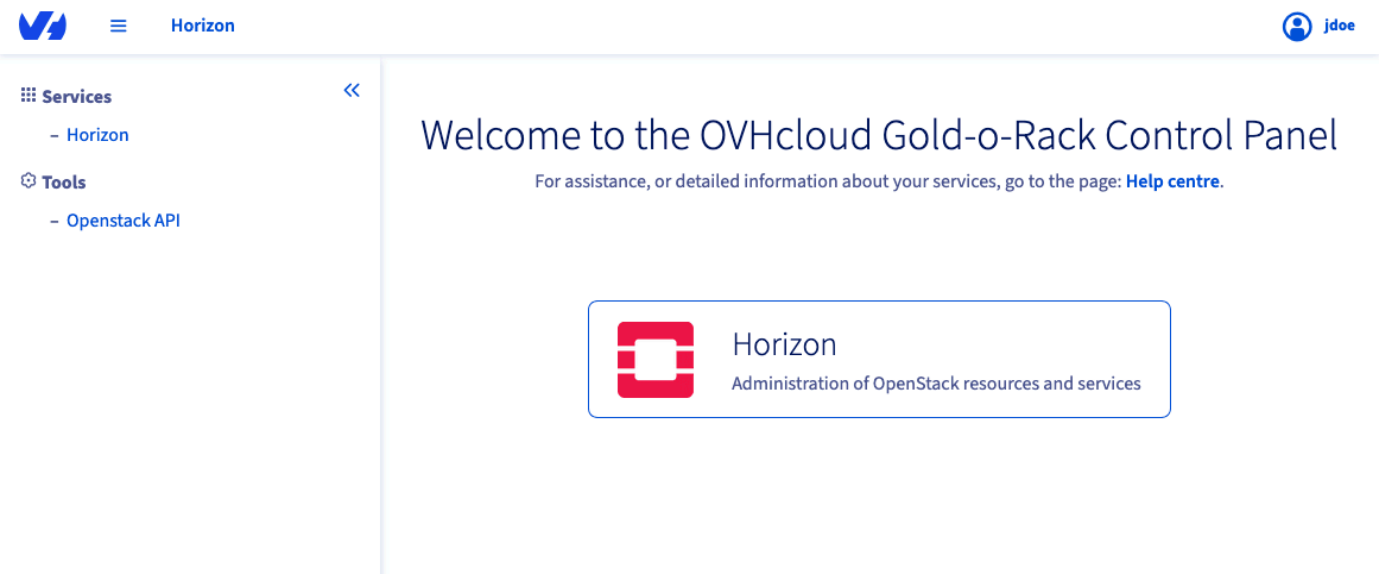
En pratique

Composition de l'interface utilisateur

L'adresse **url** fournie vous permet d'accéder à l'interface utilisateur de **OPCP**.



Une fois vos identifiants utilisés pour vous enregistrer, vous aurez accès au tableau de bord du produit.



Votre interface vous permet d'accéder à :

- la configuration des utilisateurs au sein de *Keycloak* via le lien IAM sous votre identifiant.
- l'interface de gestion de OpenStack, *Horizon*. C'est une interface web graphique pour gérer l'ensemble de l'infrastructure OpenStack. Elle permet à l'utilisateur d'exploiter les ressources machines mises à disposition par les administrateurs. Ceci passe par la création, le lancement et l'arrêt des instances, la configuration des réseaux, la gestion de l'accessibilité des instances.

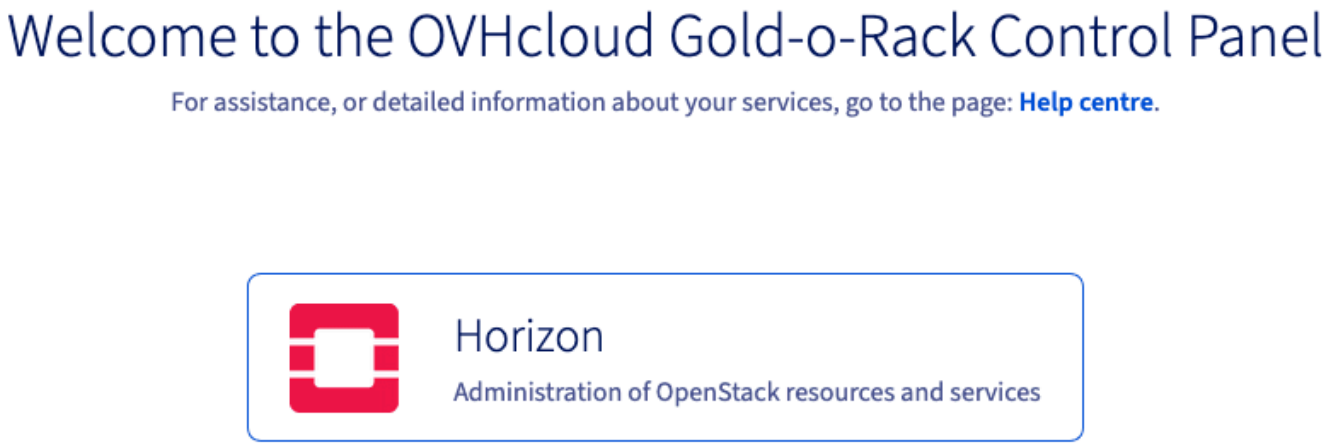
L'interface d'administration de **OPCP** regroupe également les accès aux différentes APIs telles que : Keystone (authentification et gestion des identités), Glance (gestion des images), Nova (service de calcul), Neutron (gestion du réseau), Ironic (gestion du matériel "Bare Metal") qui peuvent être utilisées au sein de vos automatisations.

Présentation de l'Interface OpenStack Horizon

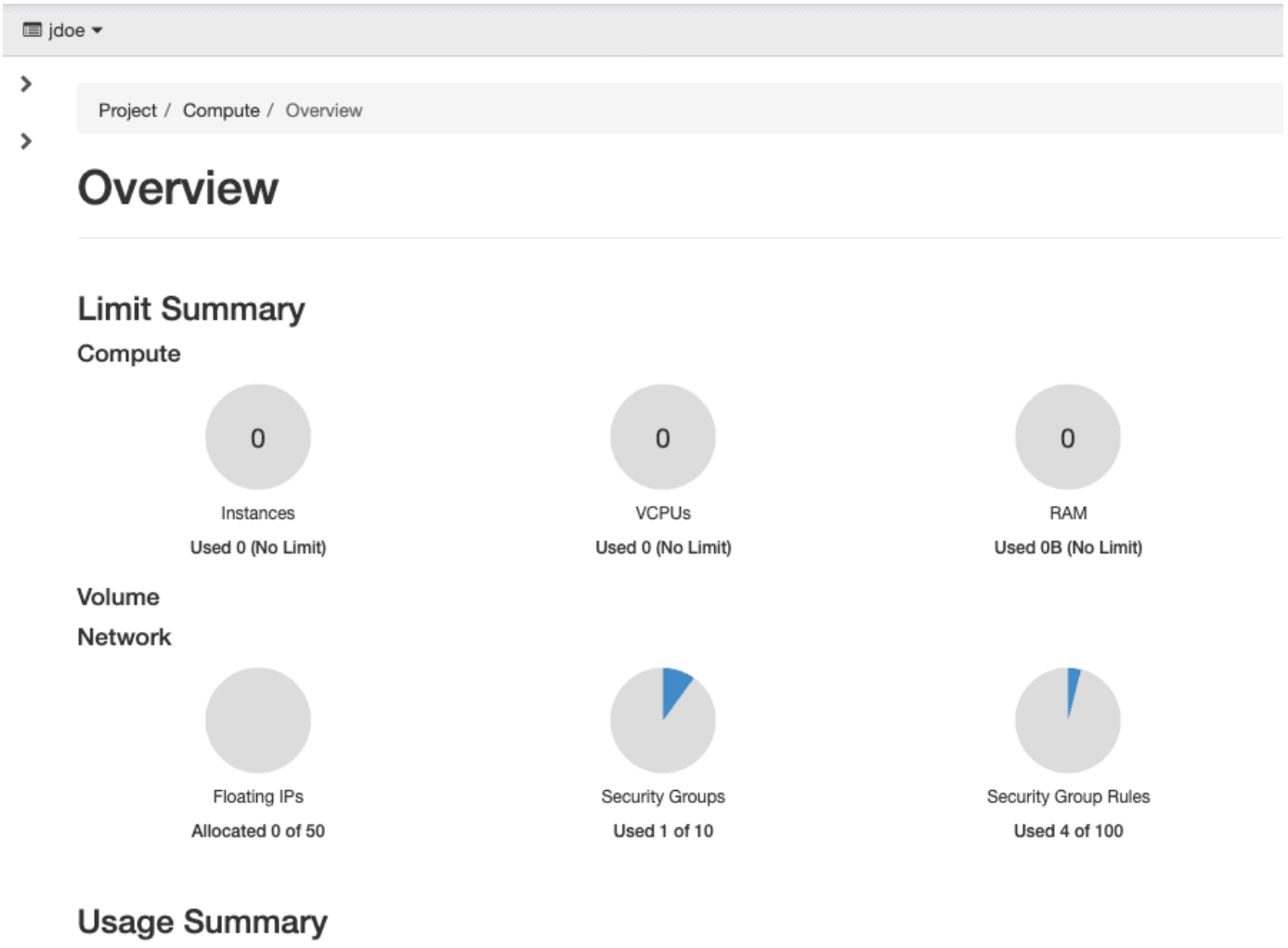
L'interface graphique d'OpenStack Horizon offre la possibilité de réaliser différentes actions en fonction de leurs autorisations et du projet auquel ells appartiennent. Parmi les principales fonctionnalités disponibles pour un utilisateur final, on peut citer : la gestion des instances, la gestion des réseaux, le suivi des ressources.

Accès à l'interface d'administration OpenStack Horizon

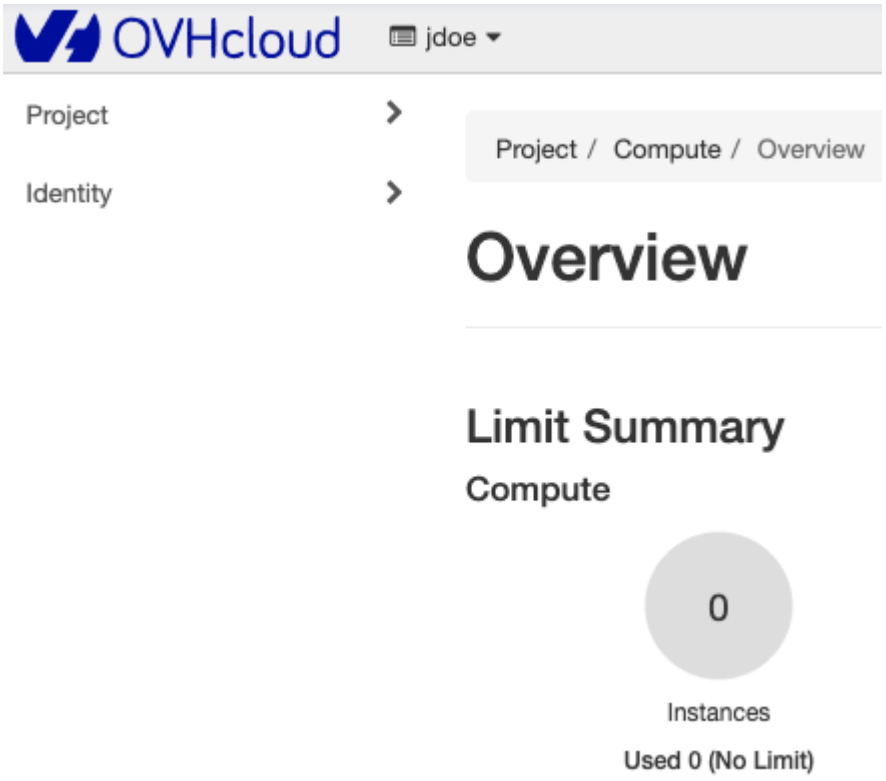
Depuis l'espace utilisateur de **OPCP**, l'interface OpenStack Horizon est accessible via le lien dans le tableau de bord.



Après s'être connecté, l'interface Horizon OpenStack se présente ainsi :



Le menu latéral situé à gauche de l'interface offre un accès aux différents éléments de l'interface. Il y a deux entrées parentes dans ce menu :

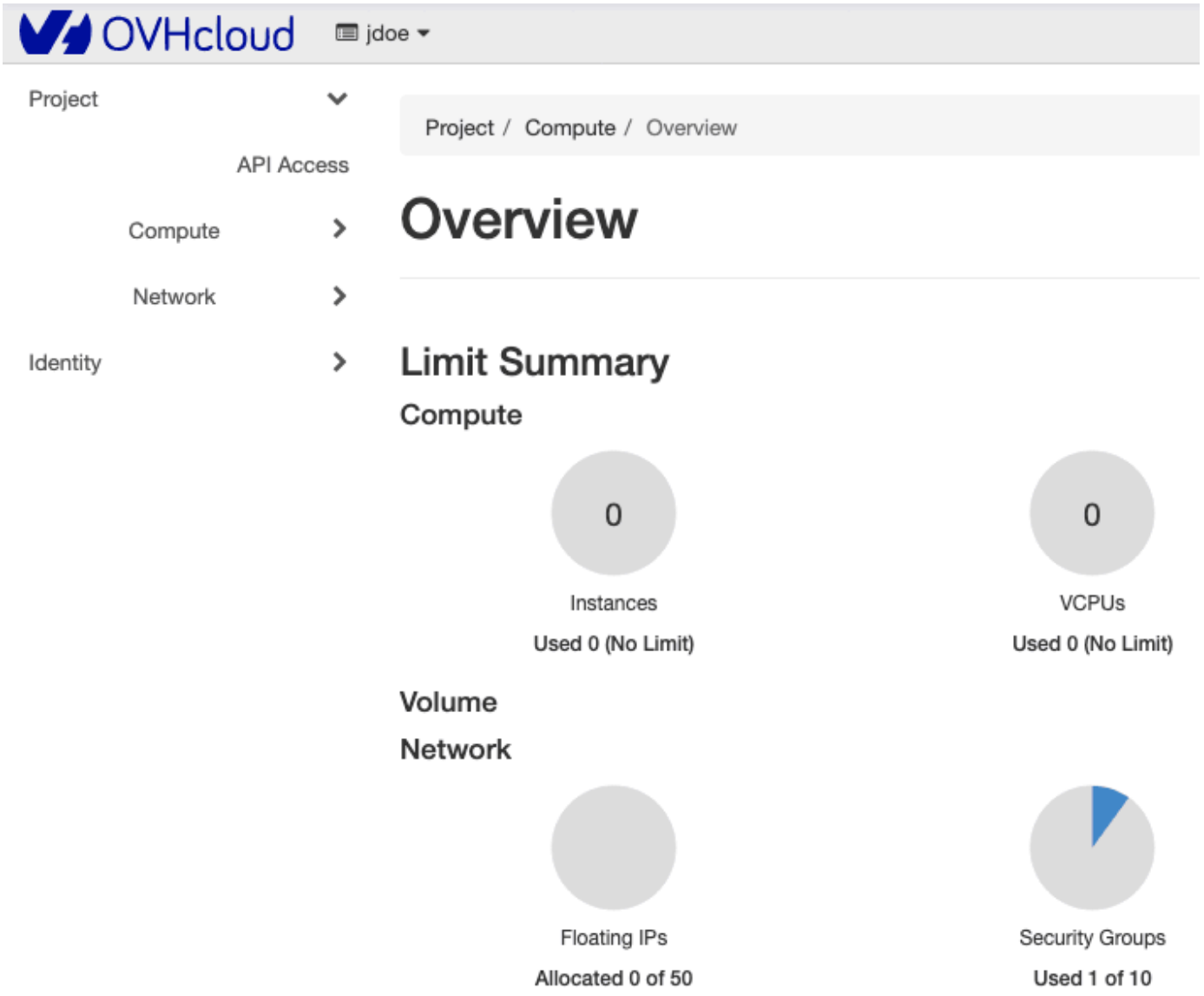


- **Projet** qui comprend quatre éléments : Aperçu (Overview), Accès à l'API (API Access), Compute et Réseau (Network). Ces éléments rassemblent l'ensemble des fonctionnalités de gestion des serveurs dédiés, de leurs réseaux, dans les limites de quotas définis.
- **Identité (Identity)** qui inclut les éléments Projets, Utilisateurs et identifiants d'application qui contiennent les fonctionnalités de gestion des utilisateurs.

Vue projet

L'élément principal *Projet* est composé de différents sous-éléments qui permettent d'accéder à toutes les fonctionnalités de gestion des ressources. Le premier sous-élément, appelé *Aperçu* (Overview), offre une vision globale des quotas de ressources attribuées au projet, ainsi qu'un suivi visuel de la consommation globale des ressources.

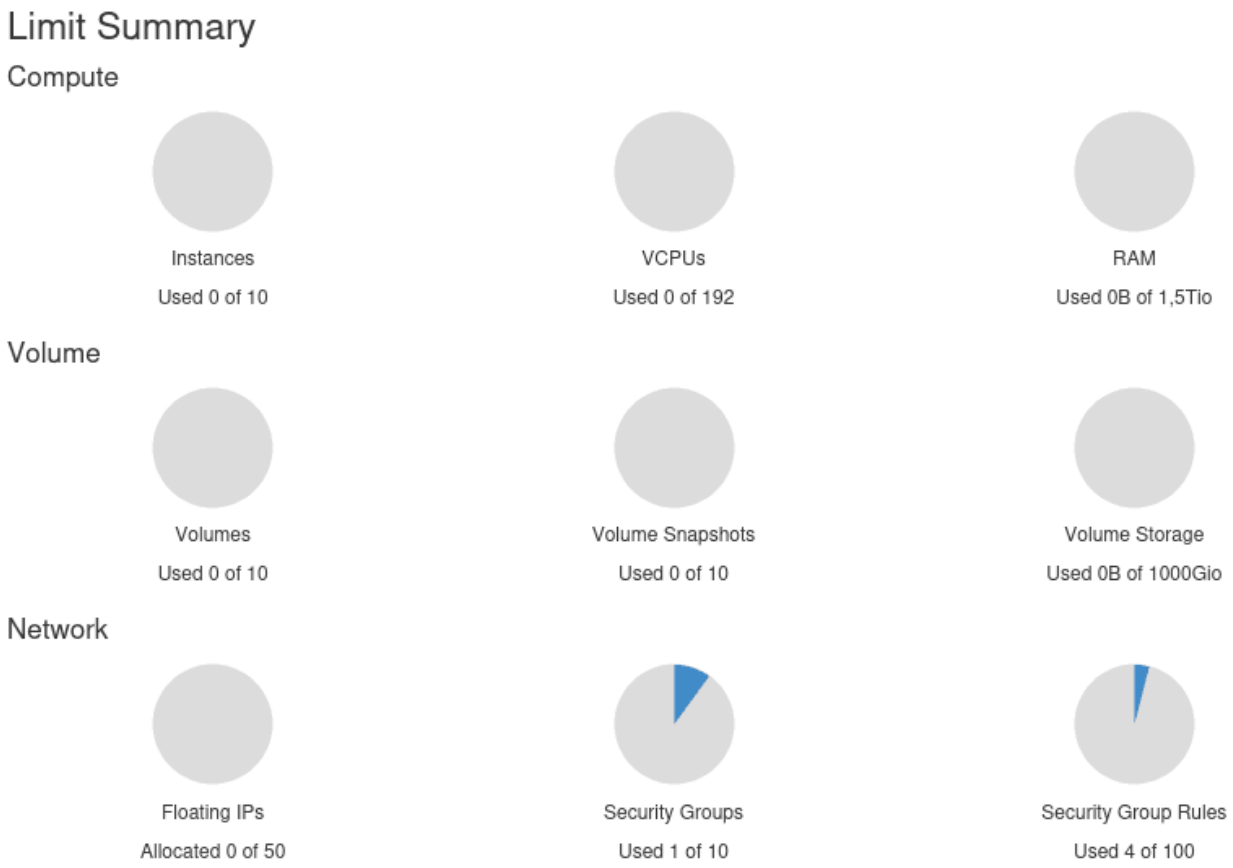
Section Overview



La section *Overview* (Aperçu) est composée de deux parties principales :

- **Limit Summary** : Les limites de quotas attribuées au projet pour chaque type de ressource. Ceci permet également de visualiser le niveau de consommation des ressources par rapport aux capacités disponibles.

  Les quotas sont regroupés en deux catégories, telles que représentées dans l'image ci-dessous :

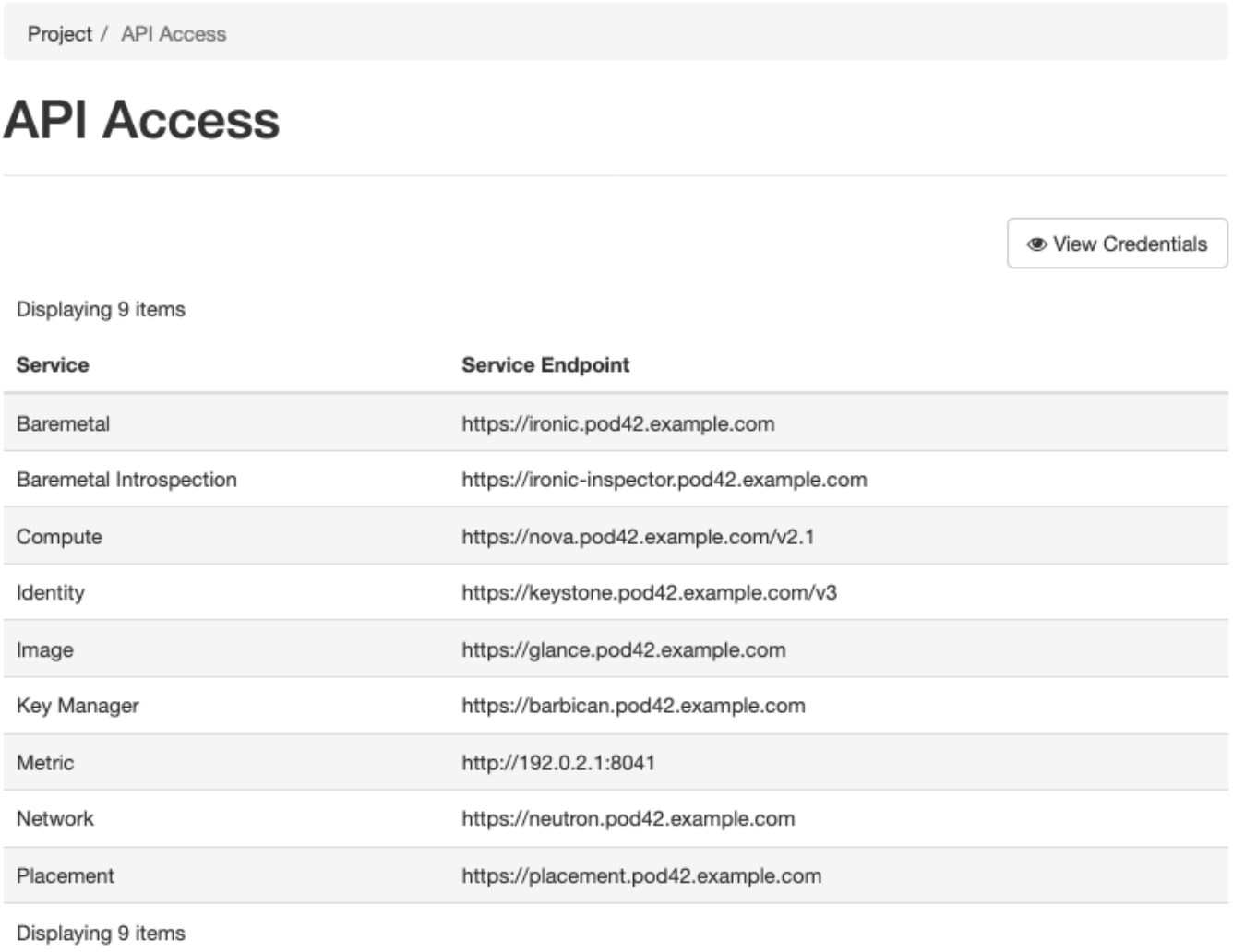


  - **Compute** qui comprend les limites des instances, les vCPUs et la RAM.
  - **Network** qui surveille les quotas des ressources réseau : les adresses Floating IP, les groupes de sécurité, les règles de sécurité des groupes, les réseaux et les ports.

- **Usage Summary** (Résumé de l'utilisation) : historique d'utilisation des ressources sur une période qui permet d'observer l'évolution de l'usage des ressources dans le temps.

Section API Access

L'onglet **API-Access** regroupe les 10 services disponibles via API tels que Bare-Metal, Compute, identité, image et réseau, ainsi que leurs URL de point d'accès.



Grâce à ces *Endpoints*, il est possible de communiquer directement avec les composants OpenStack en utilisant des requêtes API. Ces informations vous sont nécessaires si vous êtes amené à implémenter vous-même des requêtes HTTP avec les API de OpenStack.

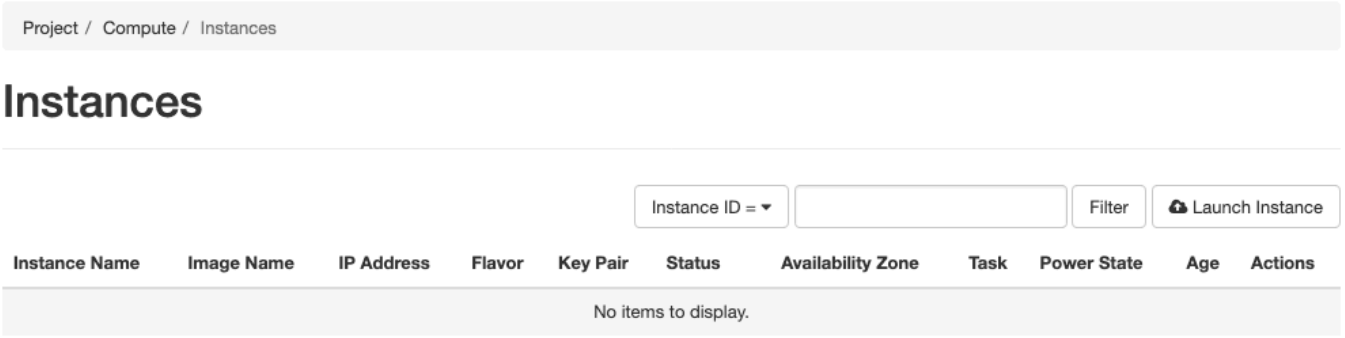
Si vous utilisez des intégrations OpenStack existantes, ces dernières récupéreront ces informations lors de la première connexion au composant OpenStack Keystone. Ce dernier se charge de fournir de manière programmatique ces informations.

Vue Compute

La vue **Compute** regroupe les fonctionnalités permettant de configurer les serveurs dédiés de votre produit. Cette vue est divisée en différentes sections :

Section Instances

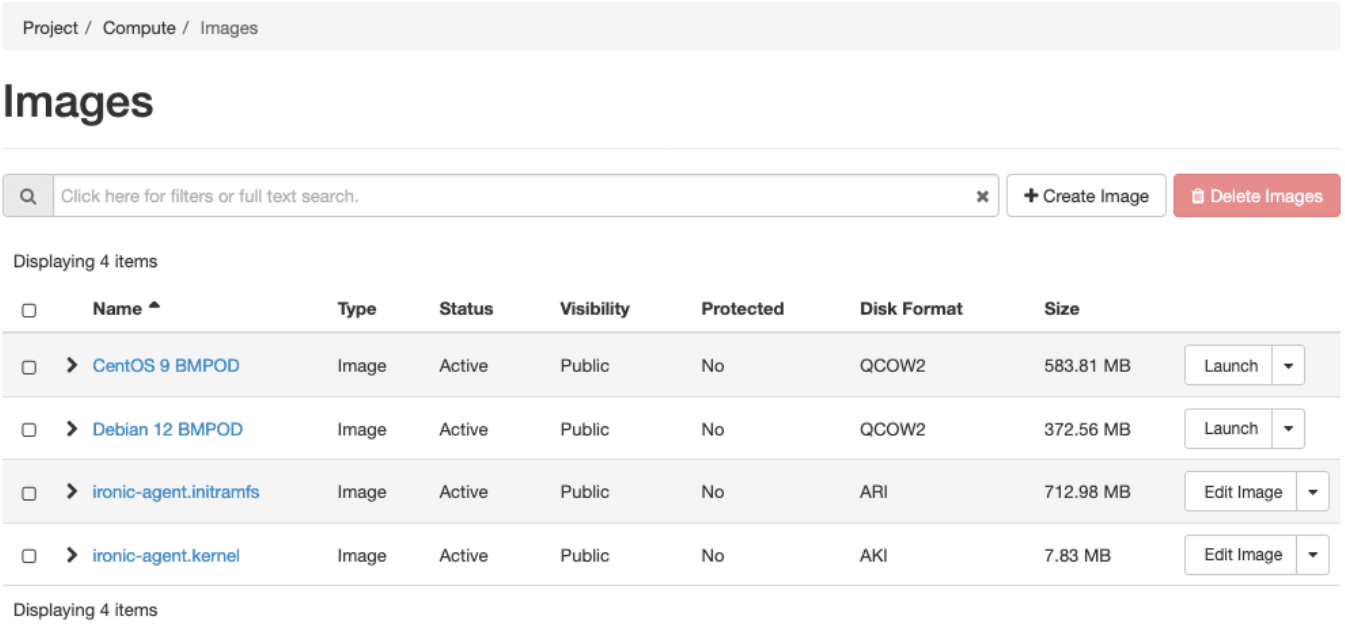
Interface permettant de lister et de gérer les serveurs dédiés déjà configurés. Une *Instance* correspond à un serveur dédié.



Section Images

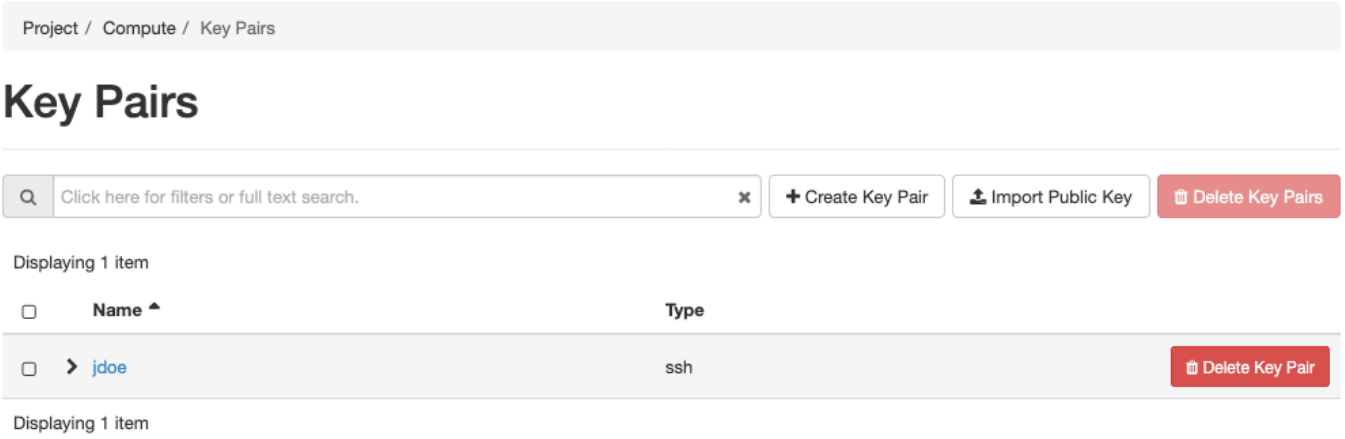
Vous avez la possibilité de gérer les images des OS disponibles pour créer des instances. Il est aussi possible de télécharger de nouvelles images ou de sélectionner parmi les images déjà disponibles afin de mettre en place des instances. Vous pouvez ainsi générer vos propres images pour gérer des systèmes d'exploitation supplémentaires.

> **Warning**
> Les images doivent prendre en compte les drivers du hardware délivré dans votre produit. De nombreuses images disponible pour Glance ne sont disponibles que pour des environnements virtuels basés sur les drivers de qemu ou de kvm.



Section Key Pairs

Afin de vous authentifier en SSH sur vos machines après l'installation, il faut utiliser des clés de chiffrement asymétriques. Cette interface permet d'importer les clés publiques ou de créer une paire de clés qui seront déployées durant l'installation des serveurs dédiés afin de vous assurer une connexion SSH.



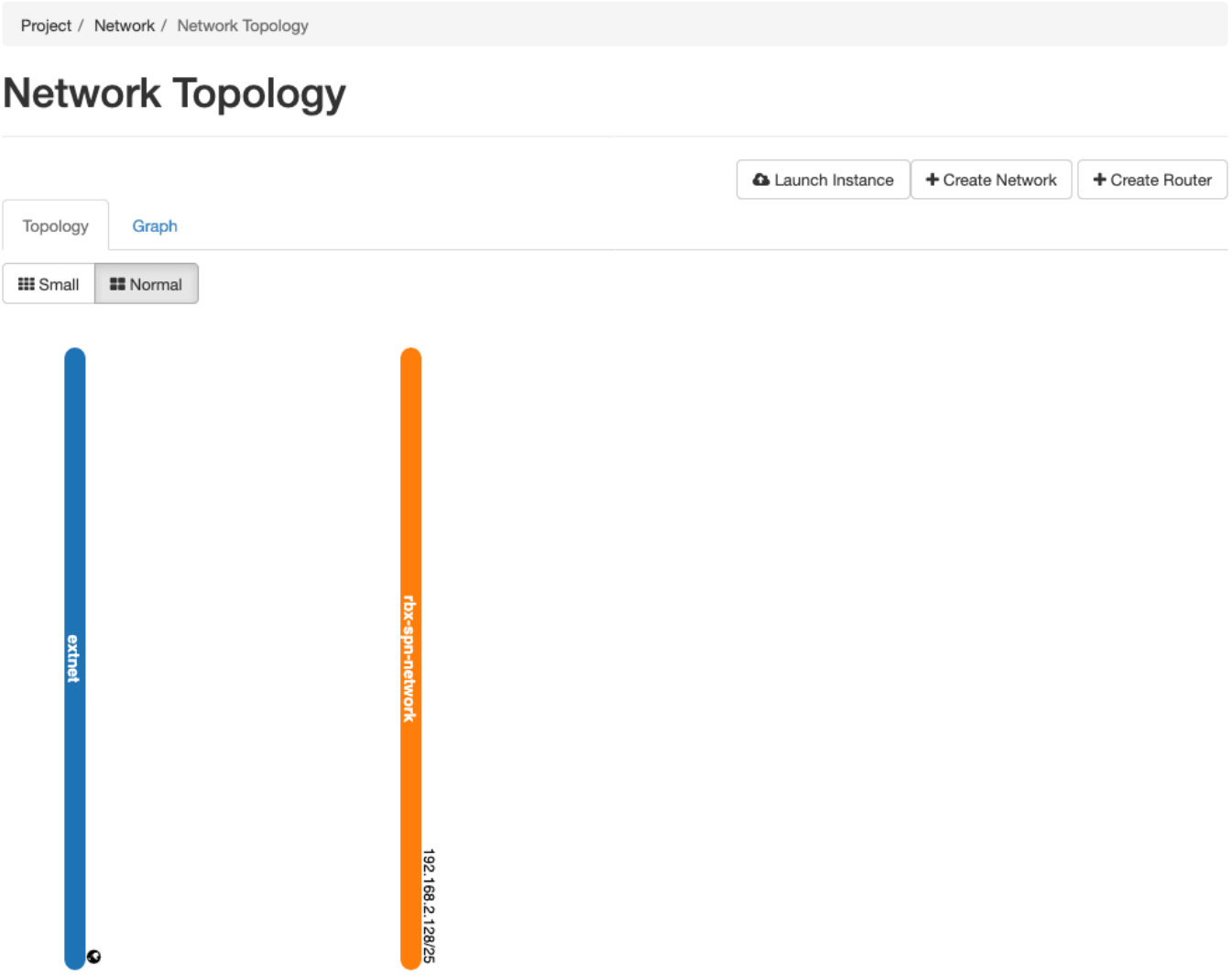
Vue Network

La Vue Network (réseau) vous permet de visualiser et gérer les réseaux de votre service **OPCP**. Cette interface permet de créer des réseaux mutualisés ou distincts entre vos serveurs dédiés.

> **Info**
>
> L'ensemble de votre configuration réseau est pilotée par cette interface graphique ou via les API de **OpenStack** avec le composant réseau nommé **Neutron**. Les switchs de votre infrastructure seront automatiquement configurés à partir des informations de OpenStack.

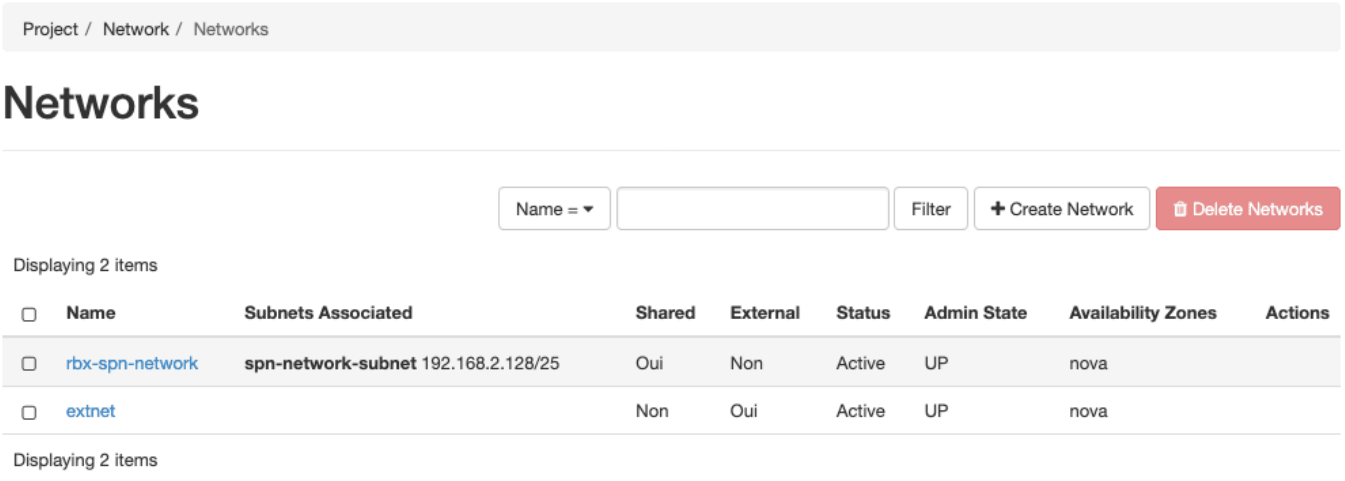
Section Network Topology

Cette section vous représente l'ensemble des réseaux créés sur ce OPCP via une ligne verticale de couleur. Les carrés correspondent à des services ou des serveurs dédiés connectés à un ou plusieurs de ces réseaux.



Section Networks

Cette section contient la liste des réseaux disponibles pour les serveurs dédiés sur votre **OPCP**.



Pour en savoir plus sur le fonctionnement des réseaux avec OpenStack, nous vous conseillons de consulter la documentation [OpenStack Networking].

Aller plus loin

Si vous avez besoin d'une formation ou d'une assistance technique pour la mise en oeuvre de nos solutions, contactez votre commercial ou cliquez sur ce lien pour obtenir un devis et demander une analyse personnalisée de votre projet à nos experts de l’équipe Professional Services.

Échangez avec notre communauté d’utilisateurs.

Installer un contrôleur OPCP

Découvrez comment installer un contrôleur OPCP à partir de l'image Debian OVHcloud, de la préparation du support à la configuration du réseau d'accès.

Objectif

Ce guide explique comment installer un contrôleur **OPCP** à partir de l'image d'installation Debian fournie par OVHcloud. Il couvre la préparation du support d'installation, le câblage du serveur, la sélection des disques pour le RAID 1, la configuration du réseau d'accès du contrôleur et les ajustements à effectuer après le premier démarrage.

> **Info**
>
> Dans ce guide, le réseau **OOB** correspond au réseau d'accès du contrôleur. Il s'agit de la première interface réseau physique du serveur, connectée à votre réseau.
>
> Ce réseau n'est pas le port d'administration dédié du serveur (IPMI, iDRAC, iLO, etc.), qui peut uniquement servir à monter l'ISO ou à ouvrir une console distante.

> **Warning**
>
> L'installation efface toutes les données présentes sur les disques sélectionnés.
>
> OVHcloud met à votre disposition des services dont vous êtes responsable. Vous devez donc vous assurer de leur bon fonctionnement après l'installation.

Prérequis

- Un serveur physique destiné à héberger le contrôleur OPCP
- Un accès en lecture seule au bucket S3[1] de livraison OPCP, avec l'endpoint S3, le nom du bucket, la région et la version OPCP à récupérer
- Une access key et une secret key S3 dédiées, limitées en lecture à ce bucket et à son préfixe de livraison
- Une clé USB, un média virtuel ou tout autre support de démarrage
- Un accès à la console locale ou à la console distante du serveur (IPMI, iDRAC, iLO, etc.)
- Au moins 2 disques physiques de taille comparable pour l'installation en RAID 1
- Un câble réseau branché sur la première interface réseau du serveur et raccordé à votre réseau
- L'adresse IP, le masque, la passerelle et les serveurs DNS du réseau d'accès du contrôleur

En pratique

1. Télécharger et vérifier l'image d'installation

Avant d'écrire l'image sur votre support de démarrage, téléchargez l'ISO et son checksum SHA-256 depuis le bucket S3 en lecture seule qui vous a été communiqué.

> **Info**
>
> Les artefacts sont publiés dans le bucket de livraison sous le préfixe `opcp-controller-debian-image/<version>/`.
>
> Utilisez une paire de credentials dédiée, limitée en lecture à ce bucket et à ce préfixe. Chargez-la uniquement dans votre shell courant ou depuis votre gestionnaire de secrets, puis supprimez-la après le téléchargement. Les endpoints de base peuvent par exemple être `s3.sbg.io.cloud.ovh.net` ou `s3.gra.io.cloud.ovh.net`.

Renseignez d'abord les variables de votre environnement :

```
export S3_ENDPOINT="<s3-endpoint>"
export S3_BUCKET="<s3-bucket>"
export S3_REGION="<s3-region>"
export OPCP_VERSION="<opcp-release-version>"
export S3_PREFIX="opcp-controller-debian-image/${OPCP_VERSION}"
export S3_ACCESS_KEY="<read-only-access-key>"
export S3_SECRET_KEY="<read-only-secret-key>"
```

Téléchargez ensuite les fichiers avec `s3cmd` (méthode recommandée pour un accès S3 authentifié) :

```
sudo apt-get update
sudo apt-get install -y s3cmd

s3cfg=$(mktemp)
chmod 600 "$s3cfg"

cat >"$s3cfg" <<EOF
[default]
access_key = ${S3_ACCESS_KEY}
secret_key = ${S3_SECRET_KEY}
host_base = ${S3_ENDPOINT}
host_bucket = %(bucket).${S3_ENDPOINT}
bucket_location = ${S3_REGION}
EOF

for artifact in \
        live-image-amd64.hybrid.iso \
        live-image-amd64.hybrid.iso.sha256; do
        s3cmd -c "$s3cfg" get \
                "s3://${S3_BUCKET}/${S3_PREFIX}/${artifact}" \
                "${artifact}"
done

rm -f "$s3cfg"
```

Si votre environnement standardise déjà les téléchargements avec `curl`, vous pouvez utiliser une requête HTTPS signée en SigV4 :

```
download_with_curl() {
        local artifact="$1"
        local curl_config
        curl_config=$(mktemp)
        chmod 600 "$curl_config"

        cat >"$curl_config" <<EOF
url = "https://${S3_BUCKET}.${S3_ENDPOINT}/${S3_PREFIX}/${artifact}"
user = "${S3_ACCESS_KEY}:${S3_SECRET_KEY}"
aws-sigv4 = "aws:amz:${S3_REGION}:s3"
output = "${artifact}"
fail
silent
show-error
EOF

        curl --config "$curl_config"
        rm -f "$curl_config"
}

for artifact in \
        live-image-amd64.hybrid.iso \
        live-image-amd64.hybrid.iso.sha256; do
        download_with_curl "$artifact"
done
```

Si votre version de `curl` ne prend pas en charge SigV4, utilisez la méthode `s3cmd`.

Vérifiez ensuite le checksum avant d'écrire l'ISO :

```
sha256sum -c live-image-amd64.hybrid.iso.sha256

unset S3_ACCESS_KEY S3_SECRET_KEY

sudo dd if=live-image-amd64.hybrid.iso of=/dev/sdX bs=4M status=progress oflag=sync
```

Remplacez `/dev/sdX` par le périphérique correspondant à votre support d'installation.

2. Câbler le serveur

Avant de démarrer l'installation :

- connectez la première interface réseau du serveur à votre réseau
- si le serveur dispose de plusieurs interfaces réseau, utilisez la première pour faciliter son identification pendant l'installation, et n'en connectez aucune autre

3. Démarrer l'installeur

Démarrez le serveur sur le support d'installation puis suivez l'installeur interactif.

Dans le menu GRUB, sélectionnez l'entrée `Auto Install` pour lancer l'installation.

Pendant l'installation, l'image :

- détecte les disques physiques disponibles
- vous demande de sélectionner les disques à utiliser pour le RAID 1

- configure automatiquement le système sur les disques sélectionnés
- vous demande de choisir l'interface réseau utilisée pour l'accès au contrôleur
- active l'accès SSH `root` sur le contrôleur au premier démarrage

4. Sélectionner les disques d'installation

Lorsque l'installeur vous le demande, sélectionnez au moins 2 disques physiques pour le contrôleur.

Utilisez de préférence des disques identiques, ou aussi proches que possible en taille et en performances, pour permettre la création correcte du RAID 1.

L'installeur crée automatiquement la configuration RAID 1 et la structure LVM du système sur les disques retenus.

5. Configurer le réseau d'accès du contrôleur

Lorsque l'installeur affiche la liste des interfaces réseau physiques :

1. Sélectionnez l'interface connectée à votre réseau.
2. Choisissez `Static` pour saisir la configuration manuellement.
3. En mode statique, renseignez l'adresse IP, le masque, la passerelle et les serveurs DNS demandés.
4. Validez le récapitulatif avant l'écriture de la configuration réseau.

À la fin de l'installation de base, cette configuration est appliquée au système installé pour permettre un premier accès SSH au contrôleur.

6. Vérifier l'accès après le premier démarrage

Après le redémarrage du serveur :

1. Connectez-vous depuis le réseau relié à la première interface réseau du serveur.
2. Vérifiez que l'adresse IP configurée pour le contrôleur répond bien en SSH.
3. Vérifiez que le contrôleur peut joindre les ressources réseau nécessaires dans votre environnement.

Si l'interface ou les paramètres réseau ne sont pas corrects, relancez l'installation et sélectionnez les valeurs attendues.

7. Ajuster la taille des volumes logiques

Après le premier démarrage, connectez-vous au contrôleur puis adaptez la taille des volumes logiques en fonction de la capacité réelle des disques et de vos besoins.

> **Warning**
> Les tailles ci-dessous sont des exemples de destination sur des disques de 900 Go. Vérifiez l'espace réellement utilisé et disponible avant toute réduction de volume, puis ajustez les valeurs selon la capacité de vos disques.

Commencez par réduire le volume `spare`, puis réallouez l'espace libéré aux volumes utiles :

```
lvreduce -r -L 100G -v /dev/mapper/vg-spare
lvresize -r -L100G -v /dev/mapper/vg-home
lvresize -r -L100G -v /dev/mapper/vg-root
lvresize -r -L30G -v /dev/mapper/vg-tmp
lvresize -r -L200G -v /dev/mapper/vg-var
```

Aller plus loin

Si vous avez besoin d'une formation ou d'une assistance technique pour la mise en oeuvre de nos solutions, contactez votre commercial ou consultez notre page Professional Services pour obtenir un devis et demander une analyse personnalisée de votre projet à nos experts de l'équipe Professional Services.

Échangez avec notre communauté d'utilisateurs.

[1] : S3 est une marque déposée appartenant à Amazon Technologies, Inc. Les services de OVHcloud ne sont pas sponsorisés, approuvés, ou affiliés de quelque manière que ce soit.

Cycle de vie d'un nœud OPCP

Découvrez le cycle de vie d'un nœud OPCP et ses différents status

Objectif

Un nœud dans openstack représente la configuration d'un serveur physique du rack OPCP. Il faut les différencer des instances qui représente le système d'exploitation sur un nœud.

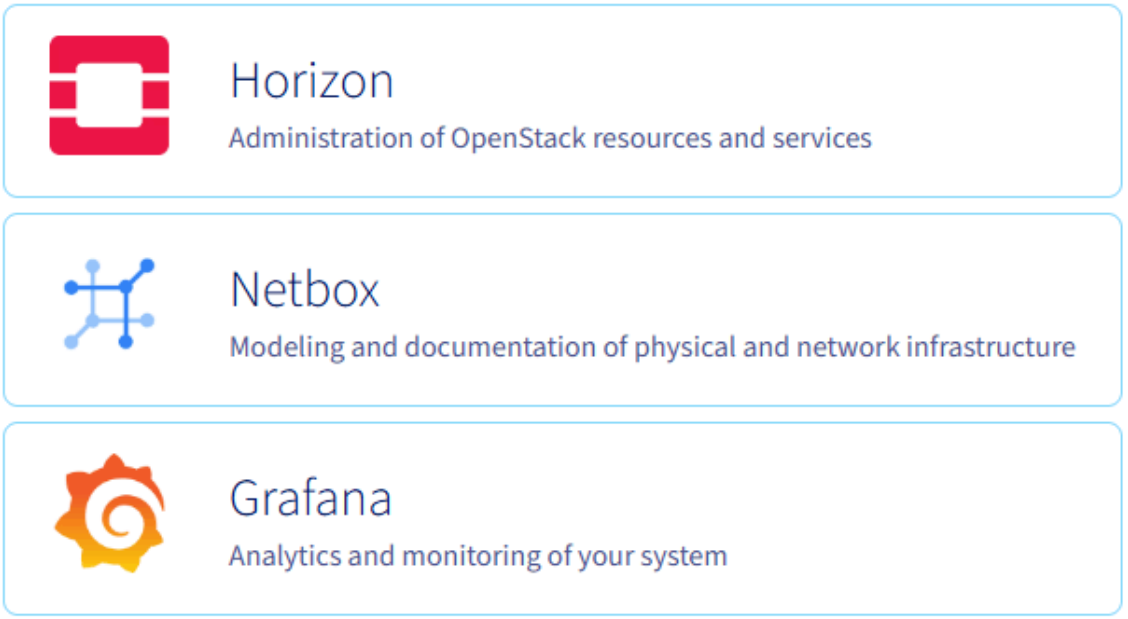
Ce guide vous détaille les différents status d'un nœud dans une baie OPCP et comment les modifier.

Prérequis

- Disposer d'un service OPCP actif.
- Posséder un compte utilisateur avec les droits admin pour se connecter à Horizon sur l'offre OPCP.
- (Optionnel) Avoir un accès aux API Openstack de votre projet.
- (Optionnel) Avoir installé le client ironic.

En pratique

Connectez vous à l'interface Horizon de votre OPCP sur le projet admin.

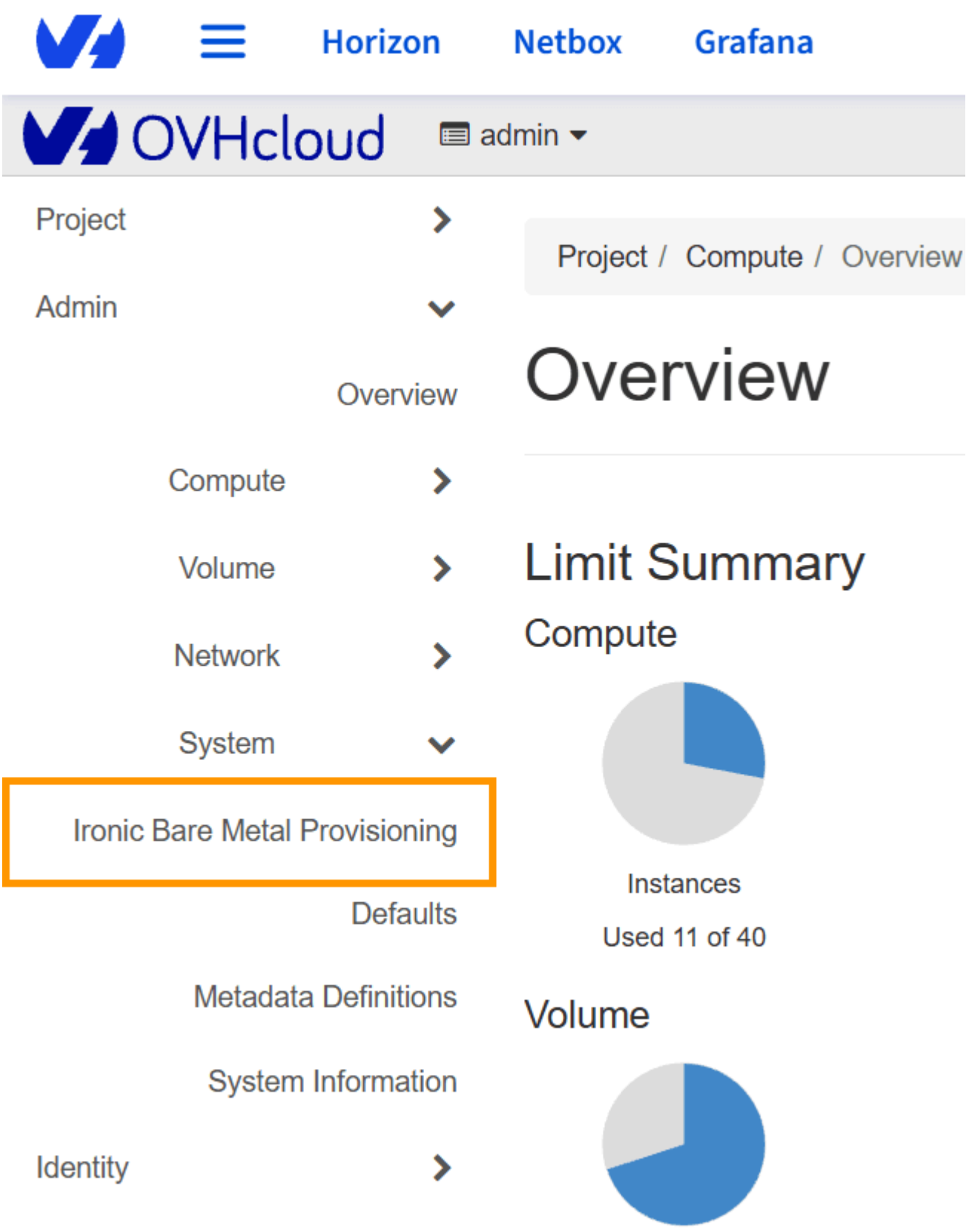


Si vous souhaitez suivre la partie API Openstack, il sera nécessaire d'installer les paquets ironic sur votre environnement :

```
pip install python-ironicclient
```

Vérifier le status d'un nœud

Vous pouvez vérifier le statut d'un nœud directement depuis Horizon via l'onglet `Admin` > `System` > `Ironic Bare Metal Provisioning` :



Vous retrouverez la liste de vos nœuds ainsi que leurs différents status.

Depuis les API Openstack, vous pouvez retrouver la même liste via la commande suivante :

```
baremetal node list
```

Vous pouvez également vérifier le statut d'un nœud spécifique :

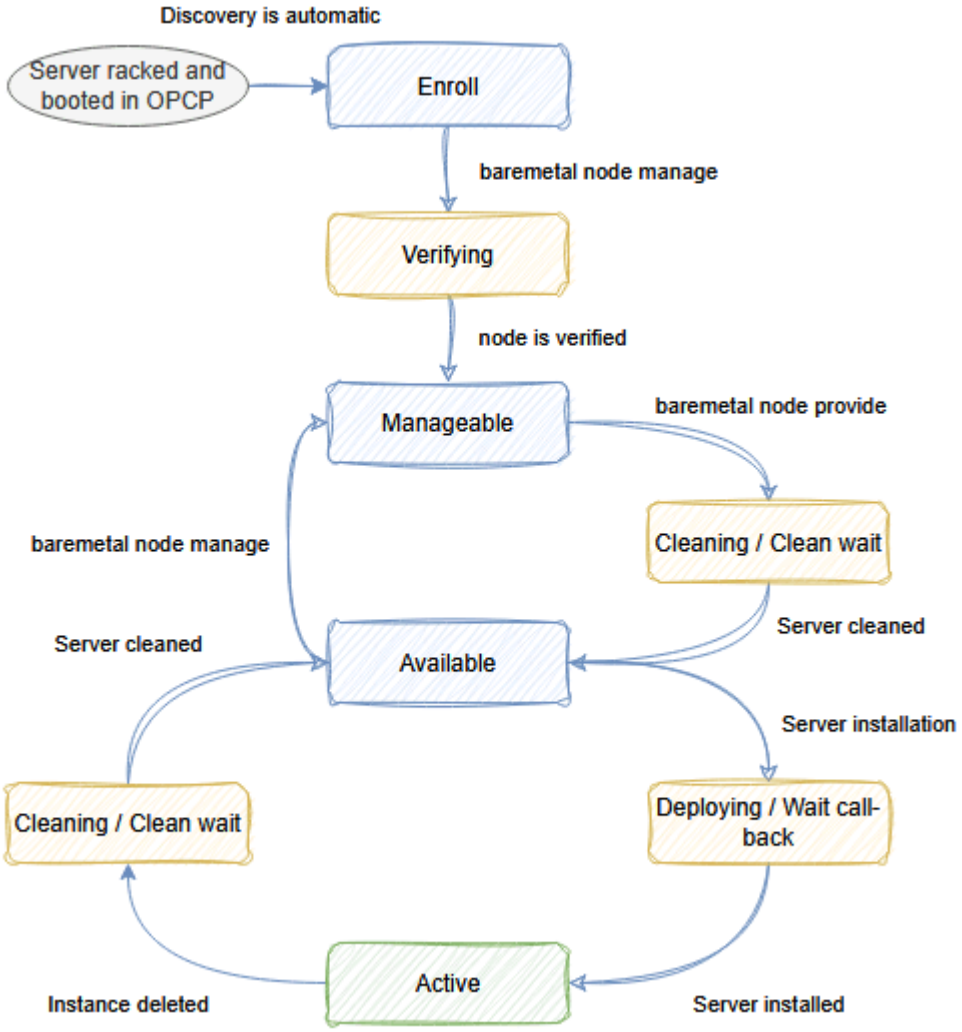
```
baremetal node show $BAREMETAL_NODE_ID
```

Différents status possibles

Statuts	Description
Enroll	Premier état du nœud lorsqu'il a été découvert automatiquement par OPCP. Le serveur n'a pas encore été vérifié et doit être rendu `Manageable` manuellement.
Manageable	Le nœud a été vérifié et est géré par Ironic, il n'est cependant pas encore installable. Le nœud doit être passé en état `Available` avant de pouvoir être installé.
Available	Le nœud est disponible et peut être installé.
Active	Le nœud est installé et a une instance active sur celui-ci.
Verifying	Etat transitoire lorsqu'un nœud passe de l'état `Enroll` à `Manageable`. Ironic vérifie qu'il peut gérer le nœud via les drivers et propriétés hardware configurés lors de la découverte faite par le control plane.
Cleaning / Clean-wait	Etat transitoire lorsqu'une instance est supprimée ou sort de l'état `Manageable` avant de redevenir `Available`. Les disques sont formatés durant cette étape.
Deploying / Wait call-back	Etat transitoire lorsque le nœud est en cours d'installation.

Vous pouvez retrouver le détail des différents status dans la documentation OpenStack officielle.

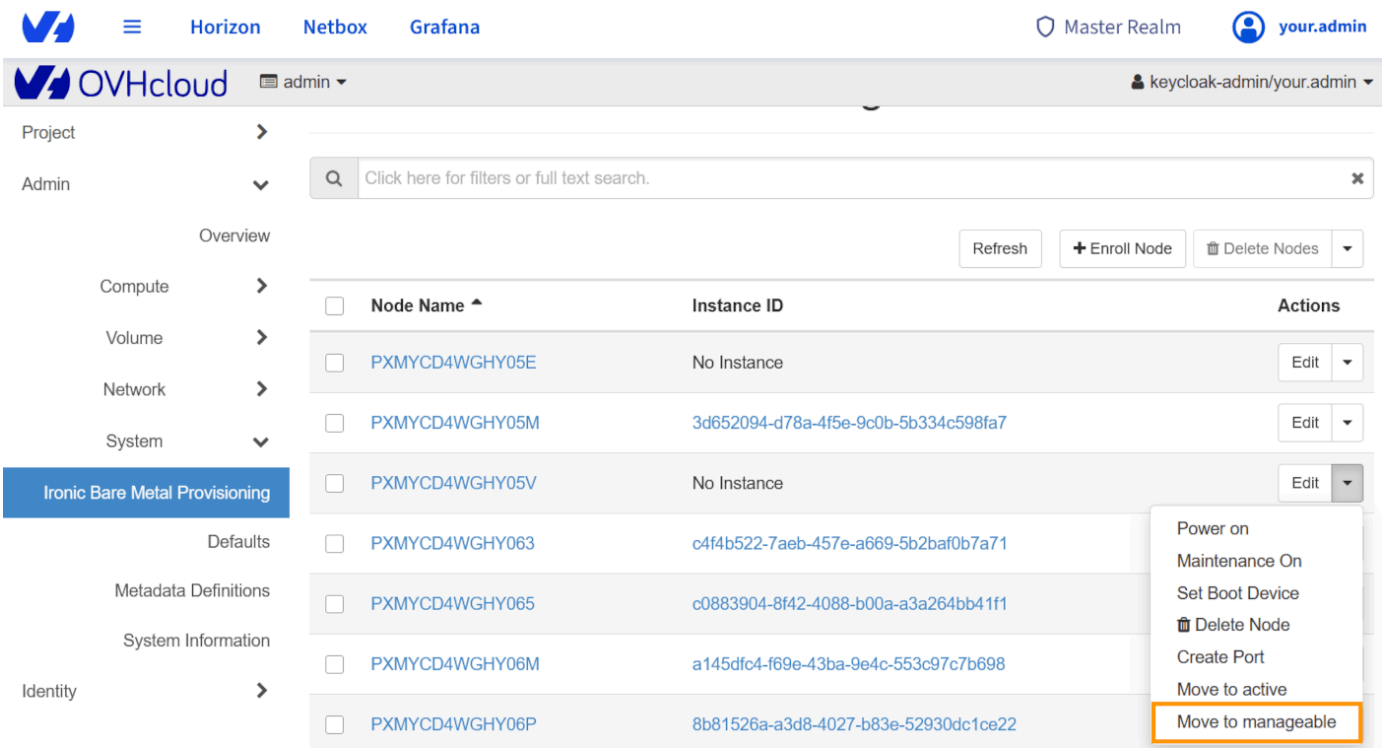
Cycle de vie d'un nœud



Lorsqu'un nœud est installé et démarré dans une baie OPCP, la découverte du nœud est automatiquement effectuée par le control plane. C'est à ce moment que le nœud récupère ses propriétés et **traits** en fonction du profil hardware de celui-ci.

Une fois que le nœud est dans l'état `Enroll`, vous pouvez modifier son état pour qu'il soit géré par Ironic.

Depuis l'interface Horizon :

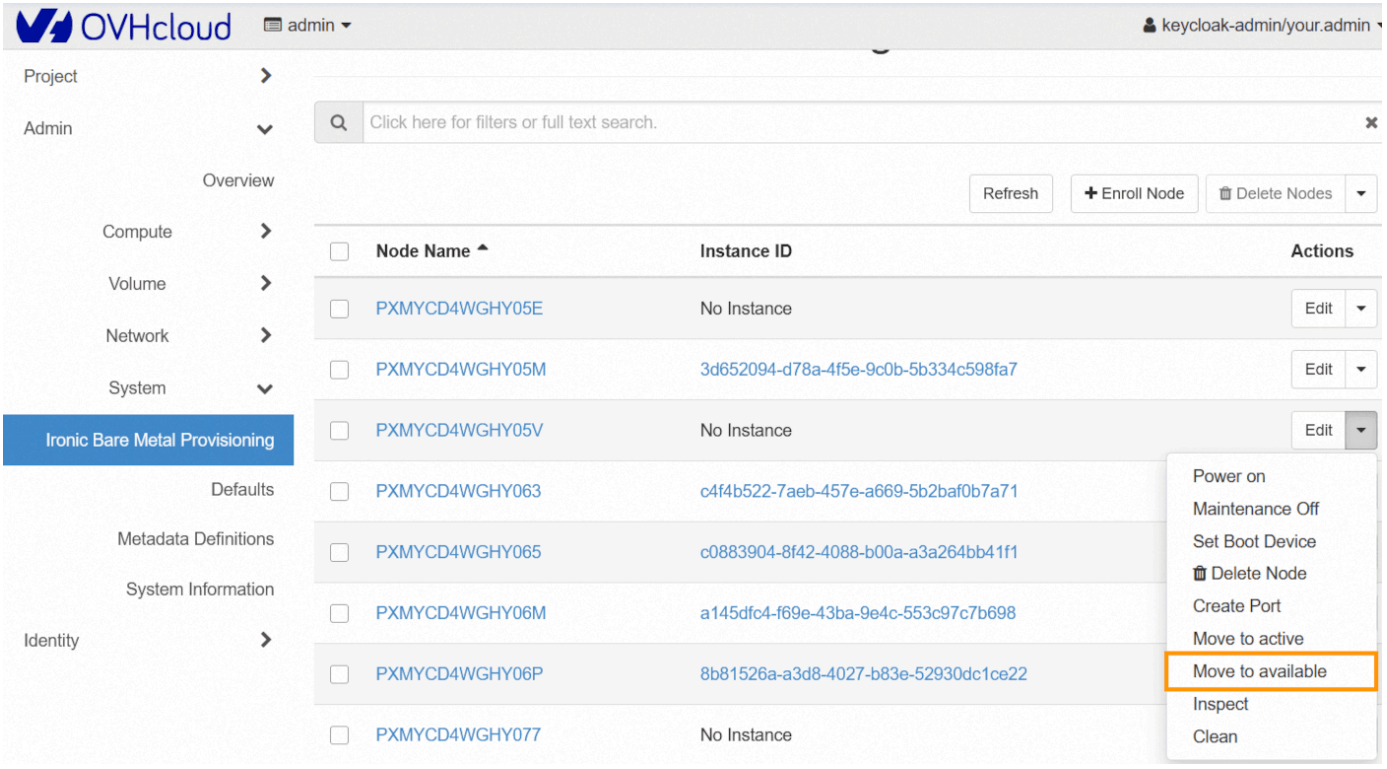


Depuis les API Openstack :

```
baremetal node manage $BAREMETAL_NODE_ID
```

Pour rendre le nœud disponible à l'installation, il faut ensuite le passer en statut `Available` :

Depuis l'interface Horizon :



Depuis les API Openstack :

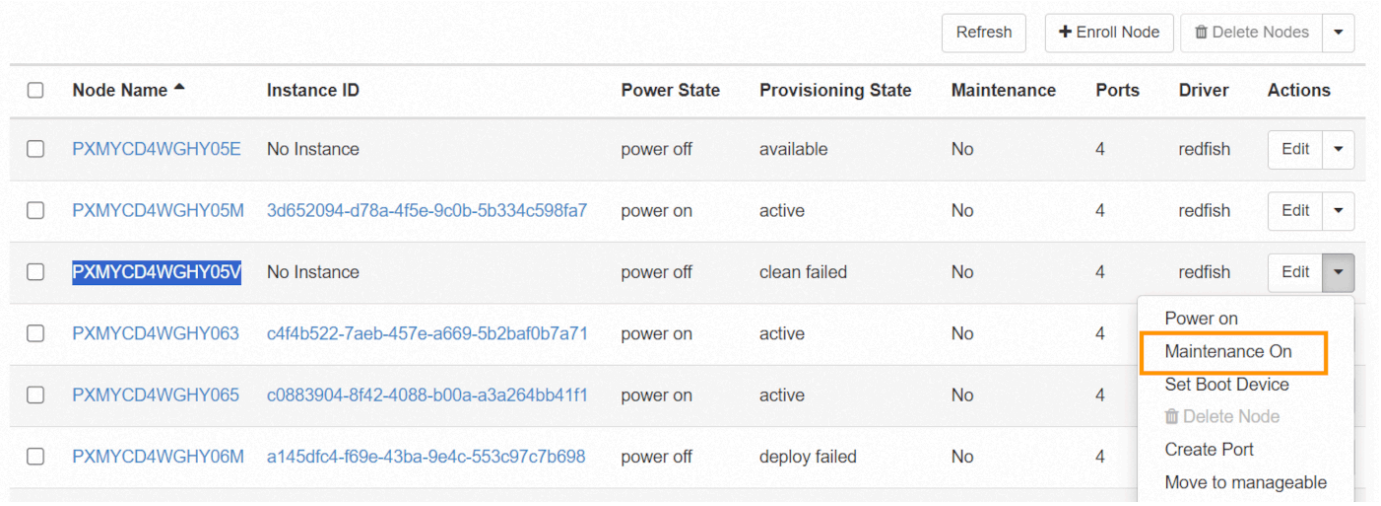
```
baremetal node provide $BAREMETAL_NODE_ID
```

Le nœud passe alors en status `Cleaning` puis `Available`, ce qui le rend installable par les différents projets de votre environnement OPCP.

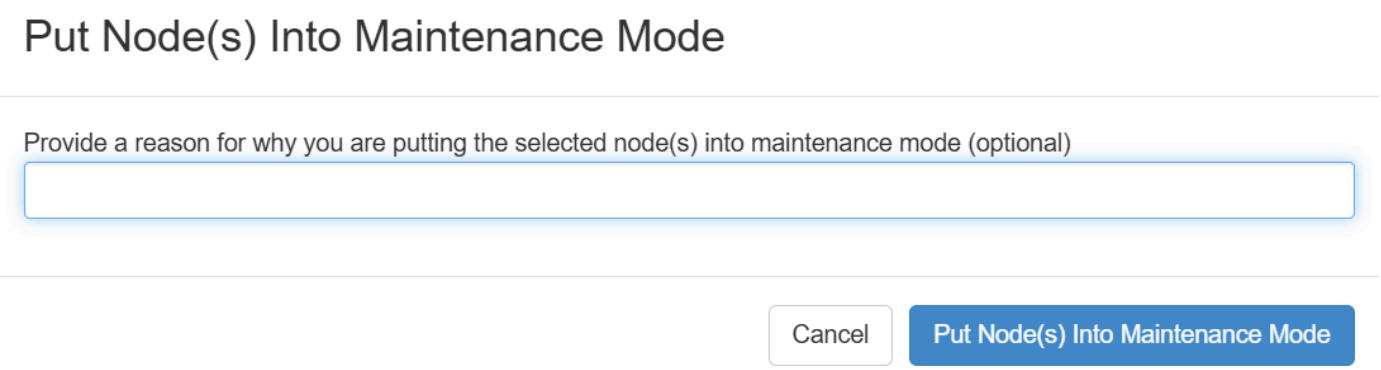
Mode maintenance

Ce mode peut être activé afin de rendre un nœud non disponible pour un installation, même si celui-ci est dans le statut `Available`.

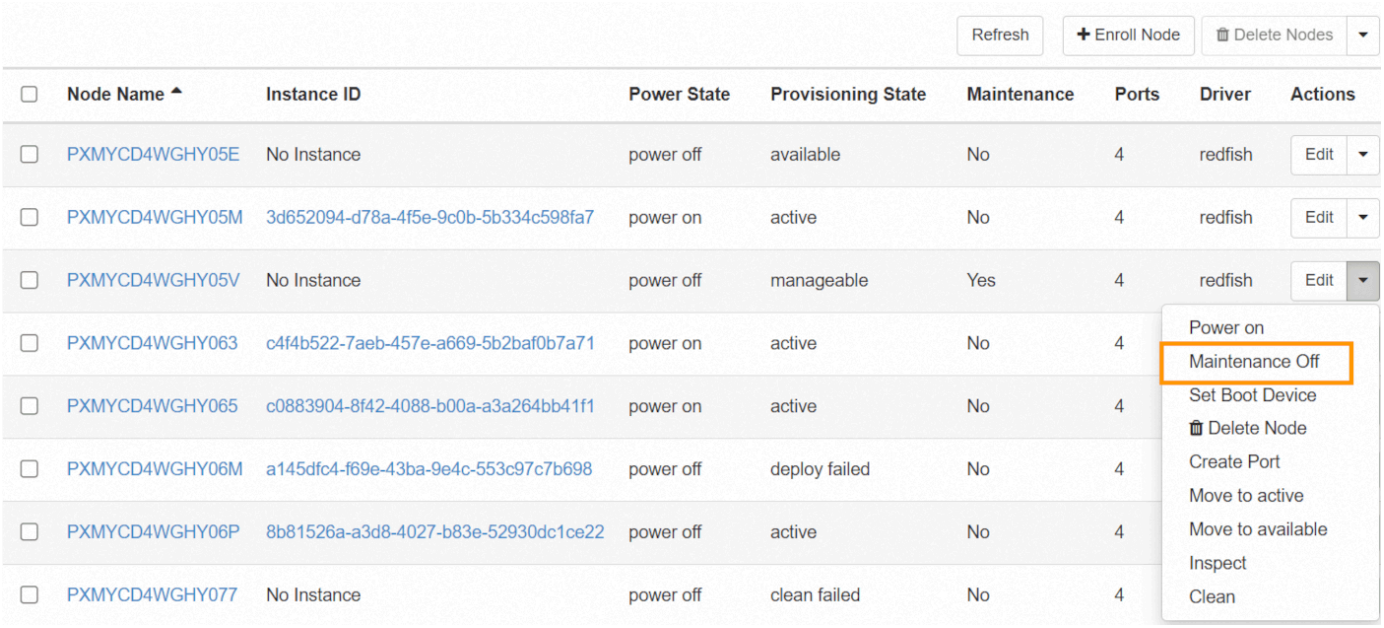
Depuis l'interface Horizon :



Lors de la mise en maintenance, vous pouvez indiquer un motif afin que l'équipe responsable des nœuds dispose de cette information. Cette raison reste optionnelle.



Une fois votre maintenance terminée, vous pouvez retirer la maintenance :



Depuis les API openstack :

```
baremetal node maintenance set $BAREMETAL_NODE_ID --reason "Maintenance reason"
```

Vous pouvez ensuite retrouver la maintenance et la raison via la commande suivante :

```
baremetal node show $BAREMETAL_NODE_ID
```

Vous trouverez les lignes :

```
| maintenance             | True
| maintenance_reason      | "Maintenance reason"
```

Pour sortir le nœud de la maintenance, vous pouvez utiliser la commande :

```
baremetal node maintenance unset $BAREMETAL_NODE_ID
```

Références

- Openstack Official Documentation - Horizon
- Openstack Ironic States
- Openstack Ironic Troubleshooting - Maintenance

Go further

If you need training or technical assistance for the implementation of our solutions, contact your sales representative or click this link to request a quote and have your project analyzed by our Professional Services team experts.

Join our community of users.

Comment utiliser les API et obtenir les informations d'identification

Découvrez les étapes nécessaires pour configurer Keycloak et le CLI OpenStack afin de permettre l’authentification via Keycloak

Objectif

OPCP intègre une authentification centralisée avec **Keycloak**. Il est donc nécessaire de configurer la **CLI OpenStack** afin qu’il utilise Keycloak comme fournisseur d’identité (Identity Provider).

Ce guide décrit les étapes nécessaires pour configurer Keycloak et la CLI OpenStack afin de permettre l’authentification via Keycloak.

Prérequis

- Être administrateur de l'infrastructure OPCP et avoir accès à l'interface d'administration (admin.dashboard).
- Avoir accès à l'interface d'administration Keyloack admin.
- Avoir un utilisateur avec les droits suffisants pour se connecter à Horizon sur l'offre OPCP.

En pratique

Création d’un client Keycloak pour la CLI OpenStack

Un client **Keycloak dédié** est nécessaire pour permettre à la CLI OpenStack de communiquer de manière sécurisée avec le serveur Keycloak.

Étapes

1. **Connexion à l’interface d’administration Keycloak**

   - Connectez-vous à votre instance Keycloak et sélectionnez le *realm* dans lequel les utilisateurs OpenStack sont définis.

2. **Création d’un nouveau client**

   - Allez dans la section `Clients` et cliquez sur **`Créer un client`**.
   - Renseignez un **Client ID**, par exemple :

     ```
     openstack-cli
     ```

   - Cliquez sur **`Suivant`**.

3. **Activation de l’authentification du client**

   - Activez le **Client Authentication** (mettre sur **ON**).
   - Cliquez sur **`Suivant`**, puis sur **`Enregistrer`**.

4. **Configuration des portées (Client Scopes)**

- Ouvrez l'onglet `Client Scopes`.
- Sélectionnez la portée nommée :

```
[votre-client-id]-dedicated
```

- Cliquez sur **Configurer un nouveau mapper**.

5. **Ajout d'un mapper d'attributs de groupe utilisateur**

- Choisissez le type de mapper **aggregated-user-group-attribute-mapper**.
- Configurez les champs suivants :

Champ	Valeur
Name	`projects`
User Attribute	`project`
Token Claim Name	`projects`

- Cliquez sur **Enregistrer**.

6. **Ajout d'un mapper d'audience** (à partir de la version 3.0.5 d'OPCP)

> **Info**
> Depuis la version **3.0.5** d'OPCP, vous devez ajouter un mapper d'audience au client afin que les tokens émis incluent l'audience `keystone`. Sans ce mapper, l'authentification auprès de Keystone échoue.

- Depuis l'onglet `Client Scopes`, sélectionnez de nouveau la portée `[votre-client-id]-dedicated`.
- Cliquez sur **Configurer un nouveau mapper** et choisissez le type **Audience**.
- Configurez les champs suivants :

Champ	Valeur
Name	`keystone-audience`
Included Client Audience	`keystone`
Included Custom Audience	*(laisser vide)*

- Cliquez sur **Enregistrer**.

7. **Récupération des identifiants du client**

- Allez dans l'onglet `Credentials` du client que vous venez de créer.
- Copiez et conservez de manière sécurisée la **Client Secret** — il sera nécessaire lors de la configuration du CLI OpenStack.

Configuration de la CLI OpenStack

Une fois le client Keycloak créé, la CLI OpenStack doit être configurée pour utiliser ce client comme fournisseur d'identité OIDC (OpenID Connect).

Étapes

1. **Installer les outils CLI OpenStack**

Si ce n’est pas déjà fait :

```
sudo pip install python-openstackclient
```

2. **Définir les variables d’environnement pour l’authentification Keycloak**

Exemple :

```
export OS_INTERFACE=public
export OS_IDENTITY_API_VERSION=3
export OS_AUTH_URL="https://keystone.domain.ovh"
export OS_AUTH_TYPE="v3oidcpassword"
export OS_PROTOCOL="openid"
export OS_IDENTITY_PROVIDER="keycloak-admin"
export OS_CLIENT_ID="keycloak-client-id"
export OS_CLIENT_SECRET="keycloak-client-credentials"
export OS_DISCOVERY_ENDPOINT="https://admin.keycloak.domain.ovh/realms/master/.well-known/openid-configuration"
export OS_USERNAME="keycloak-user-username"
export OS_PASSWORD="keycloak-user-password"
export OS_PROJECT_ID="project-id"
```

> **Tip**
>
> Vous pouvez utiliser le script suivant afin de générer le fichier de configuration openrc.sh facilement :

```
 #!/usr/bin/env bash

read -p "Your environment's base FQDN (e.g. example.bmp.ovhgoldorack.ovh): " FQDN_ENV

read -p 'master or pod realm ? (master/pod): ' REALM
if [ "$REALM" != "master" ] && [ "$REALM" != "pod" ]; then
    echo "Invalid input. Please enter either 'master' or 'pod'."
    exit 1
fi

read -p 'Keycloak client ID: ' KC_CLIENT_ID
read -srp 'Keycloak client secret: ' KC_CLIENT_SECRET && echo

read -p 'Keycloak username: ' KC_USERNAME_INPUT
read -srp 'Keycloak password: ' KC_PASSWORD_INPUT && echo

read -p 'Openstack Project ID (not the name): ' PROJECT_ID

printf "\n\nHere is your configuration, paste it to your shell or use the generate openrc.sh file\n\n"
cat << EOM
export OS_INTERFACE=public
export OS_IDENTITY_API_VERSION=3
export OS_AUTH_URL="https://keystone.${FQDN_ENV}"
export OS_AUTH_TYPE="v3oidcpassword"
export OS_PROTOCOL="openid"
export OS_IDENTITY_PROVIDER=$([ "$REALM" = "master" ] && echo "keycloak-admin" || echo "keycloak")
export OS_CLIENT_ID="$KC_CLIENT_ID"
export OS_CLIENT_SECRET="$KC_CLIENT_SECRET"
export OS_DISCOVERY_ENDPOINT="https://$([ "$REALM" = "master" ] && echo "admin.keycloak" || echo
"keycloak").${FQDN_ENV}\`/realms\`/$REALM/.well-known/openid-configuration"
export OS_USERNAME="$KC_USERNAME_INPUT"
export OS_PASSWORD="$KC_PASSWORD_INPUT"
export OS_PROJECT_ID="$PROJECT_ID"
EOM

echo "#!/usr/bin/env bash

export OS_INTERFACE=public
export OS_IDENTITY_API_VERSION=3
export OS_AUTH_URL="https://keystone.${FQDN_ENV}"
export OS_AUTH_TYPE="v3oidcpassword"
export OS_PROTOCOL="openid"
export OS_IDENTITY_PROVIDER=$([ "$REALM" = "master" ] && echo "keycloak-admin" || echo "keycloak")
export OS_CLIENT_ID="$KC_CLIENT_ID"
export OS_CLIENT_SECRET="$KC_CLIENT_SECRET"
export OS_DISCOVERY_ENDPOINT="https://$([ "$REALM" = "master" ] && echo "admin.keycloak" || echo
"keycloak").${FQDN_ENV}\`/realms\`/$REALM/.well-known/openid-configuration"
export OS_USERNAME="$KC_USERNAME_INPUT"
export OS_PASSWORD="$KC_PASSWORD_INPUT"
export OS_PROJECT_ID="$PROJECT_ID > $PROJECT_ID."-openrc.sh"
```

Tip

Configuration d’un proxy :

Si vous utilisez un proxy pour accéder à votre service, vous devez configurer vos variables d’environnement pour prendre en compte ce proxy.

Pour ce faire, ajoutez les lignes de commande suivantes :

```
export https_proxy=http://your-adress-ip:port/
export http_proxy=http://your-adress-ip:port/
```

Vérification de la configuration

Vous pouvez tester votre configuration à l'aide de quelques commandes simples :

```
openstack token issue
openstack project list
openstack server list
```

Si ces commandes retournent des résultats, l'intégration **Keycloak ↔ OpenStack** est correctement configurée.

Dépannage (Troubleshooting)

Problème	Cause possible	Solution
`Invalid client credentials`	Mauvais ou manquant `Client Secret`	Vérifiez le secret dans l'onglet **Credentials** du client Keycloak
`Unauthorized`	L'utilisateur n'est pas associé au bon groupe ou projet	Vérifiez les attributs `project` de l'utilisateur dans Keycloak
`OIDC discovery failed`	Mauvaise URL dans `DISCOVERY_ENDPOINT`	Assurez-vous qu'elle pointe bien vers le *realm* correct de Keycloak

Références

- Documentation Keycloak – OpenID Connect
- Documentation OpenStack Keystone

Aller plus loin

Si vous avez besoin d'une formation ou d'une assistance technique pour la mise en oeuvre de nos solutions, contactez votre commercial ou cliquez sur ce lien pour obtenir un devis et demander une analyse personnalisée de votre projet à nos experts de l'équipe Professional Services.

Échangez avec notre communauté d'utilisateurs.

Comment installer une instance depuis l'interface Horizon

Découvrez comment installer une instance OPCP via Horizon, en configurant réseaux, sous-réseaux, instances et clés SSH.

Objectif

Avant de pouvoir déployer des services sur vos baies **OPCP**, il est nécessaire de disposer d'une instance installée et active.

Ce guide détaille les étapes à suivre pour installer une instance OPCP à partir de l'interface Horizon.

Prérequis

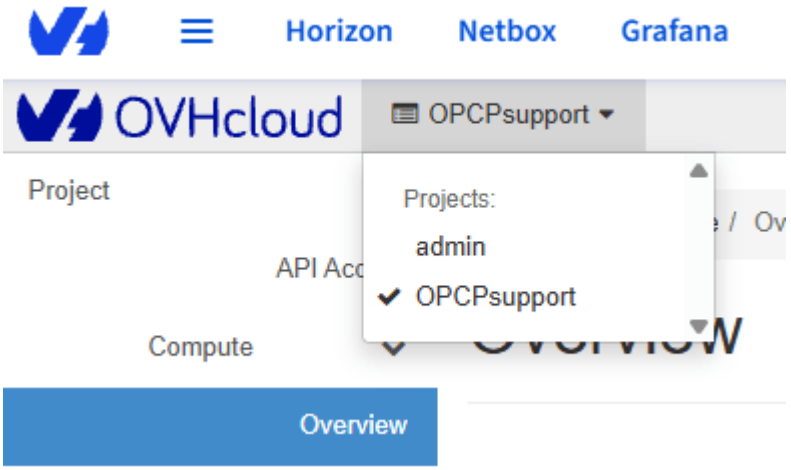
- Disposer d'un service OPCP actif.
- Posséder un **compte utilisateur** avec les droits suffisants pour se connecter à Horizon sur l'offre OPCP.

En pratique

1. Connexion à Horizon

Connectez-vous à l'interface **Horizon** de votre environnement OPCP.

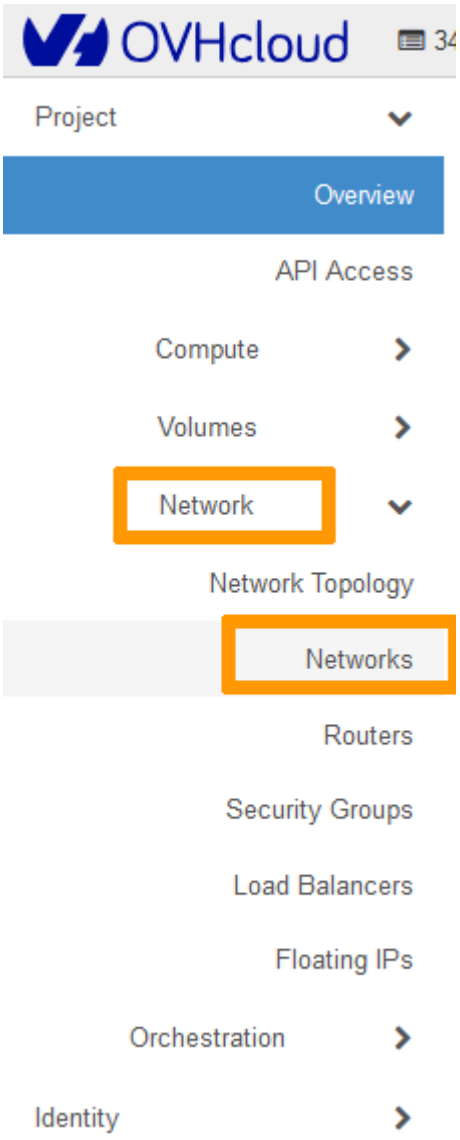
Une fois connecté, sélectionnez le **projet** dans lequel vous souhaitez installer votre instance.



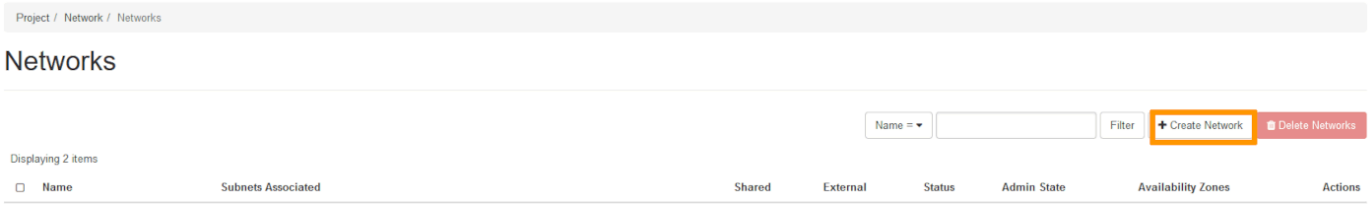
2. Création d’un réseau privé

Avant de déployer votre instance, il est généralement nécessaire de créer un **réseau privé** afin qu’il soit accessible au sein de votre infrastructure locale.

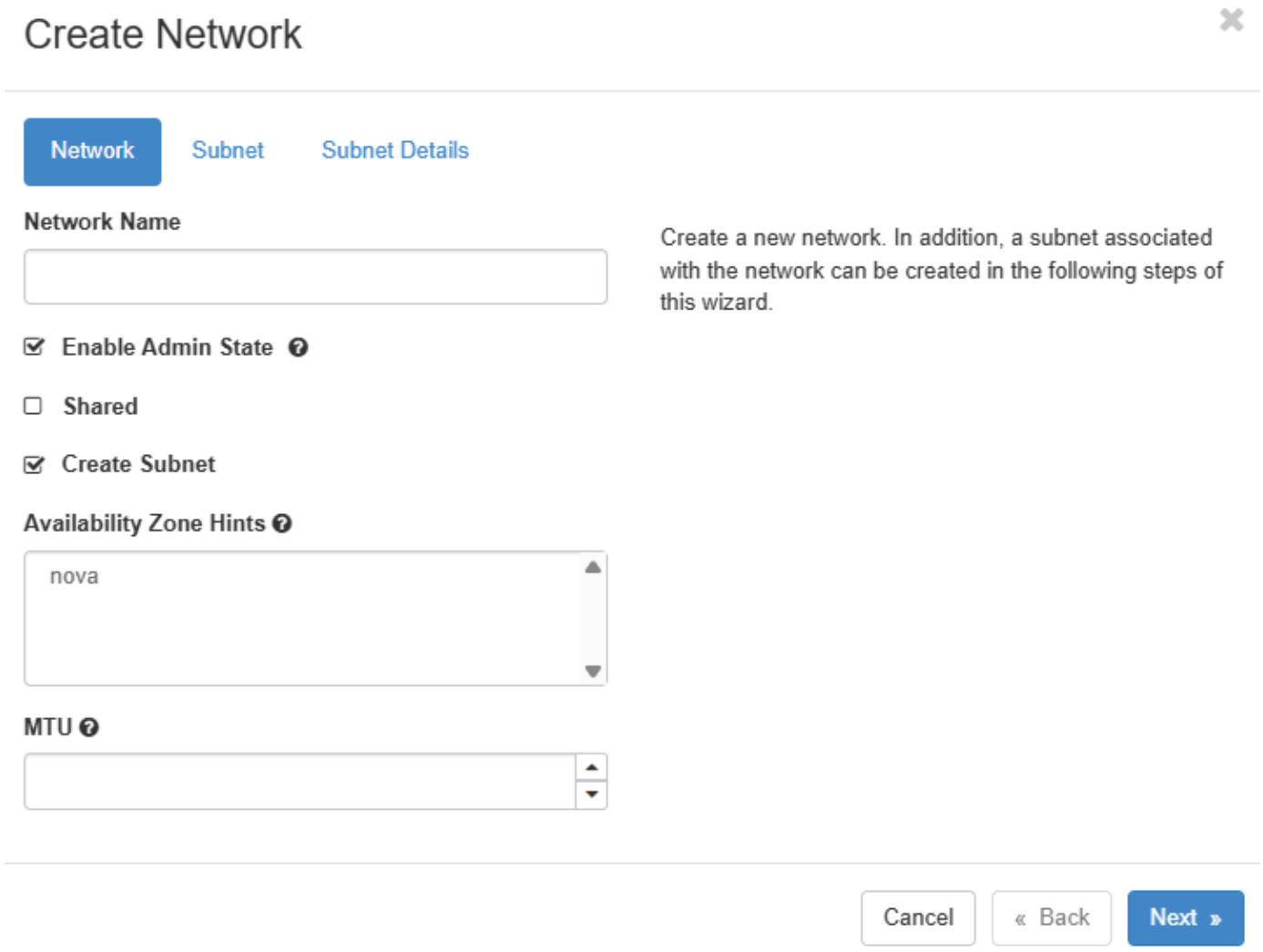
Dans le menu de gauche, cliquez sur **Network**, puis sur **Networks**.



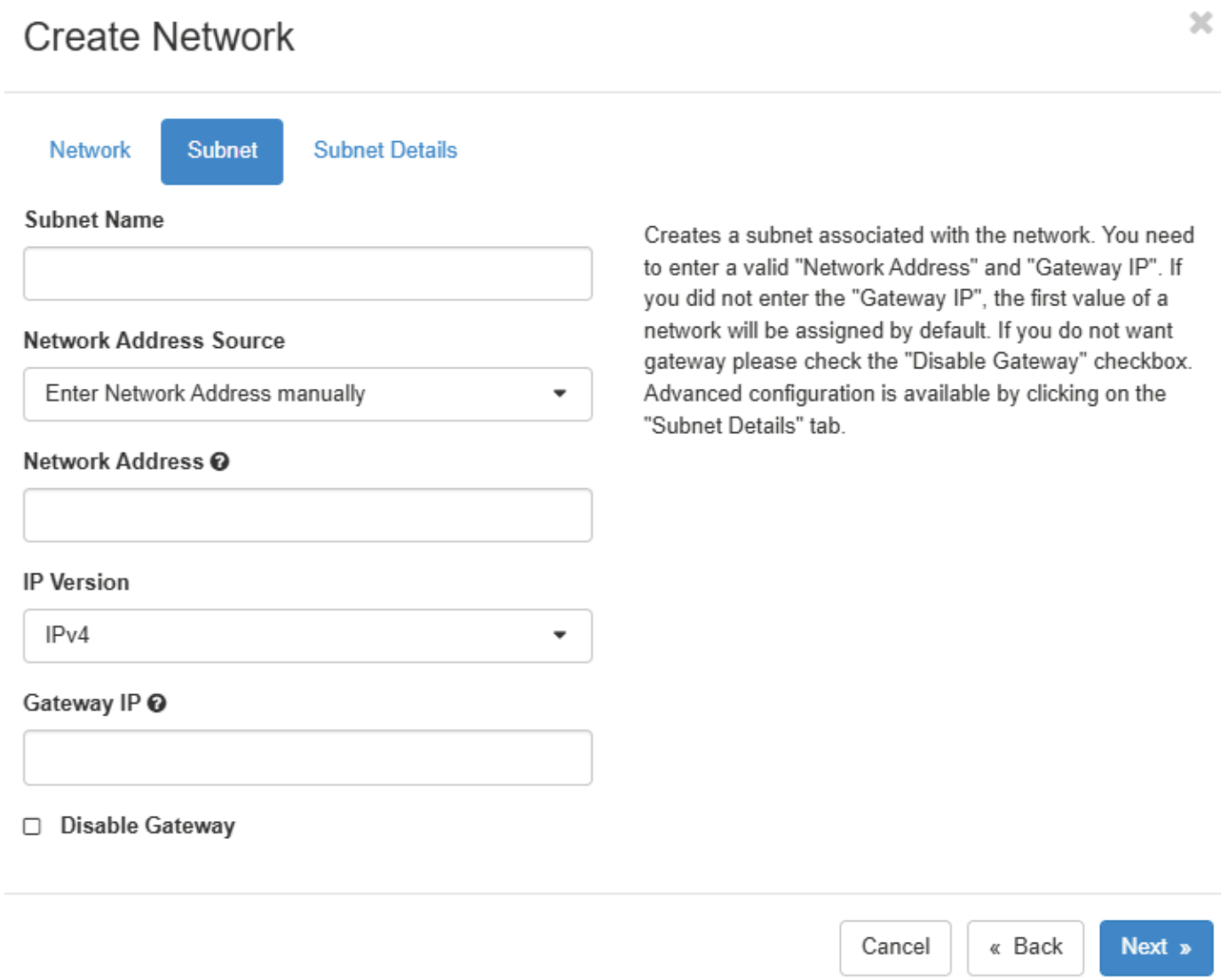
Cliquez sur **Create Network**.



Network
Subnet
Subnet Details



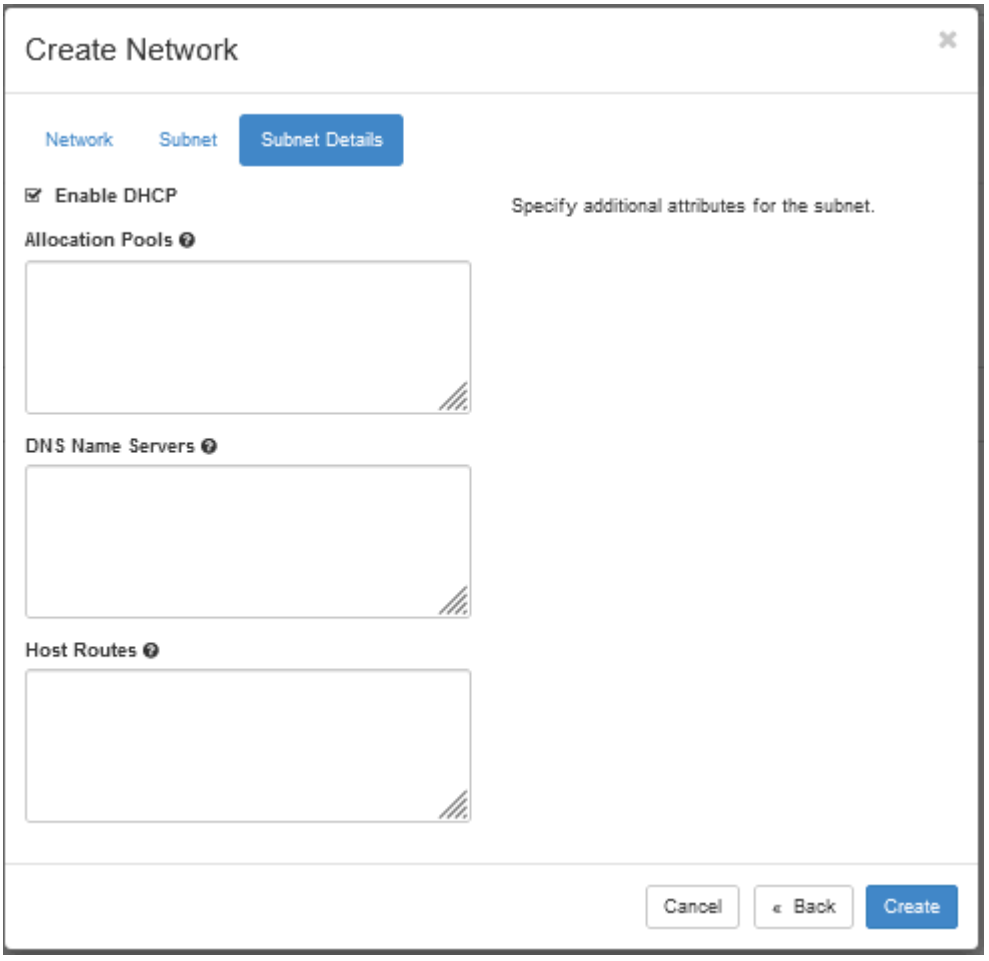
Champ	Description
Network Name	Saisissez un nom pour votre réseau.
Enable Admin State	Laissez cette option cochée pour activer le réseau.
Shared	Cochez cette case si vous souhaitez rendre le réseau disponible pour plusieurs projets.
Create Subnet	Laissez cette option cochée pour créer un sous-réseau.
Availability Zone Hints	Laissez la valeur par défaut.



Info

Bien qu'il soit possible de créer un réseau sans sous-réseau, celui-ci ne pourra pas être attaché à une instance s'il n'a pas de sous-réseau.

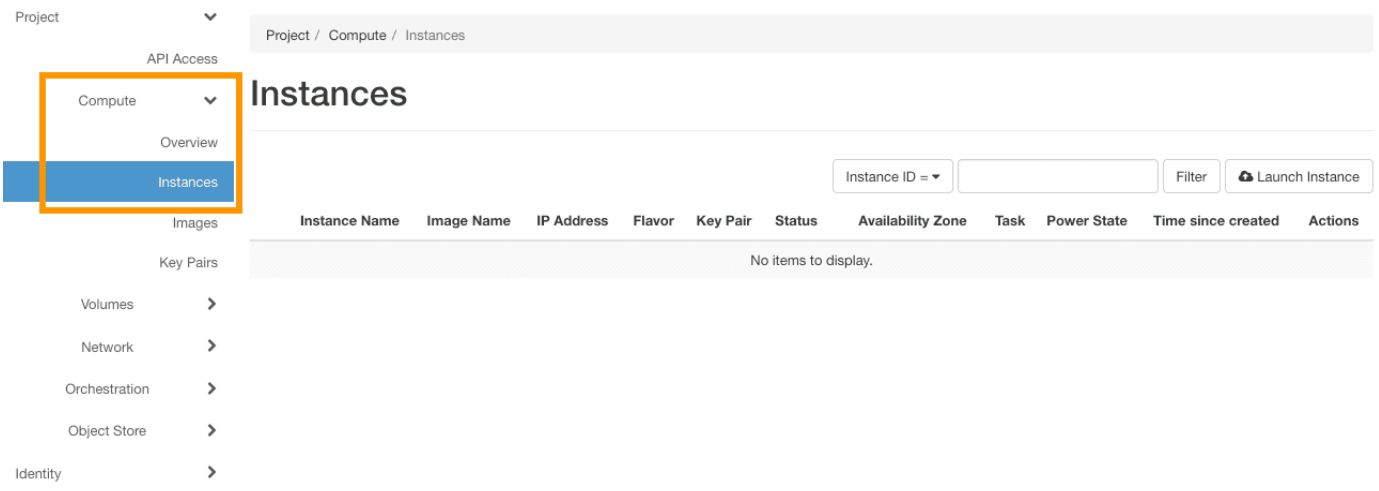
Champ	Description
Subnet Name	Entrez un nom pour votre sous-réseau.
Network Address	Définissez une plage d'adresses privées, par exemple `192.168.100.0/24`.
IP Version	Laissez la valeur par défaut **IPv4**.
Gateway IP	Optionnel. Si non renseignée, une adresse sera sélectionnée automatiquement.
Disable Gateway	Cochez cette case pour ne pas attribuer d'adresse passerelle.



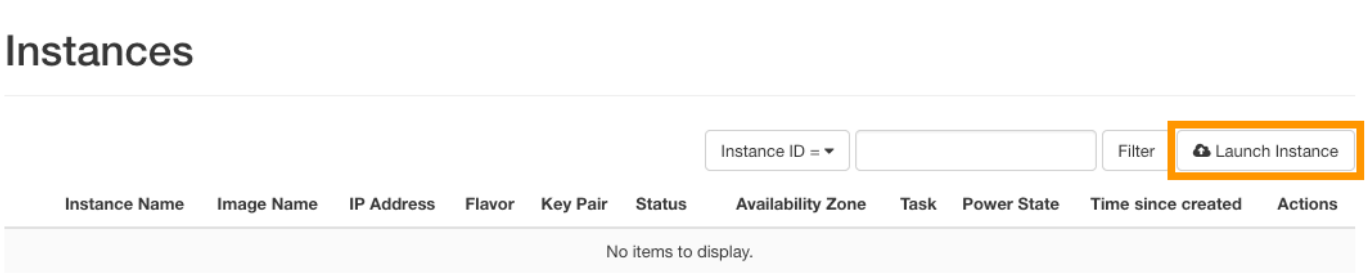
Champ	Description
Enable DHCP	Laissez activé si vous souhaitez que les adresses IP soient attribuées automatiquement.
Allocation Pools	Optionnel. Permet de définir une plage d'adresses IP spécifique.
DNS Name Servers	Optionnel. Permet de spécifier un ou plusieurs serveurs DNS.
Host Routes	Optionnel. Permet d'ajouter des routes statiques.

3. Création d'une instance

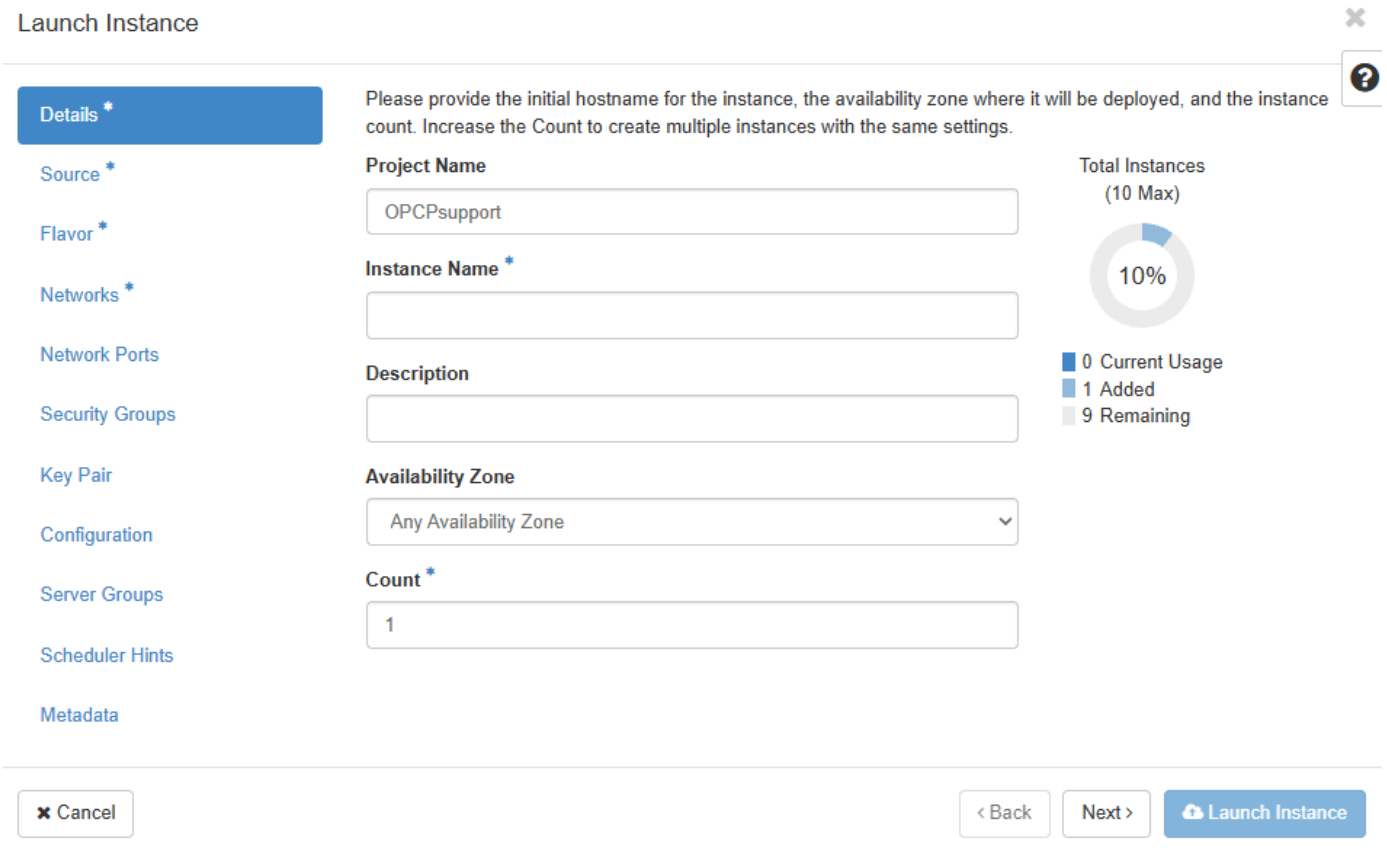
Dans le menu de gauche, cliquez sur **Compute**, puis sur **Instances**.



Cliquez sur **Launch Instance** pour lancer la création d'une nouvelle instance.

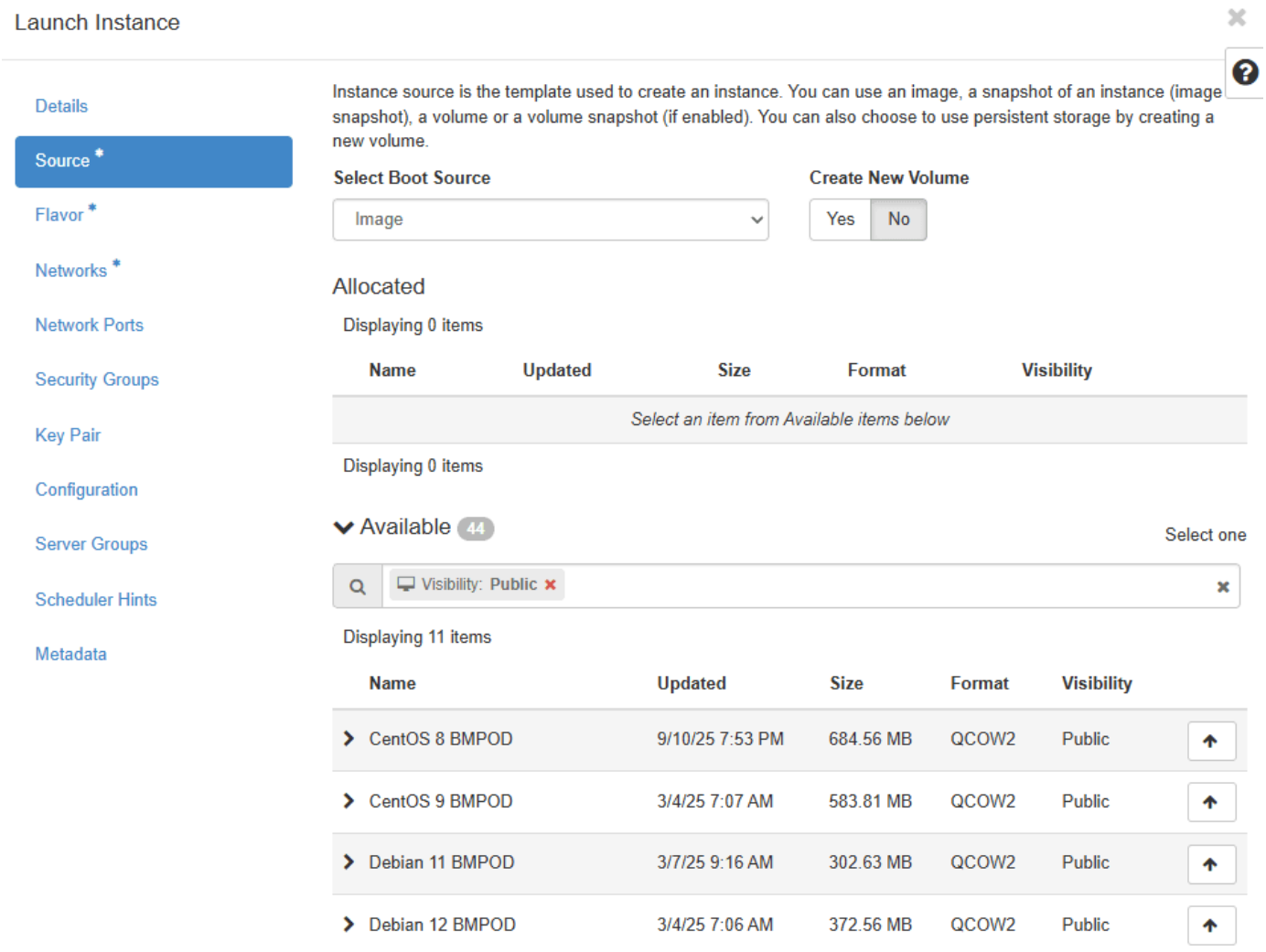


Onglet : Details



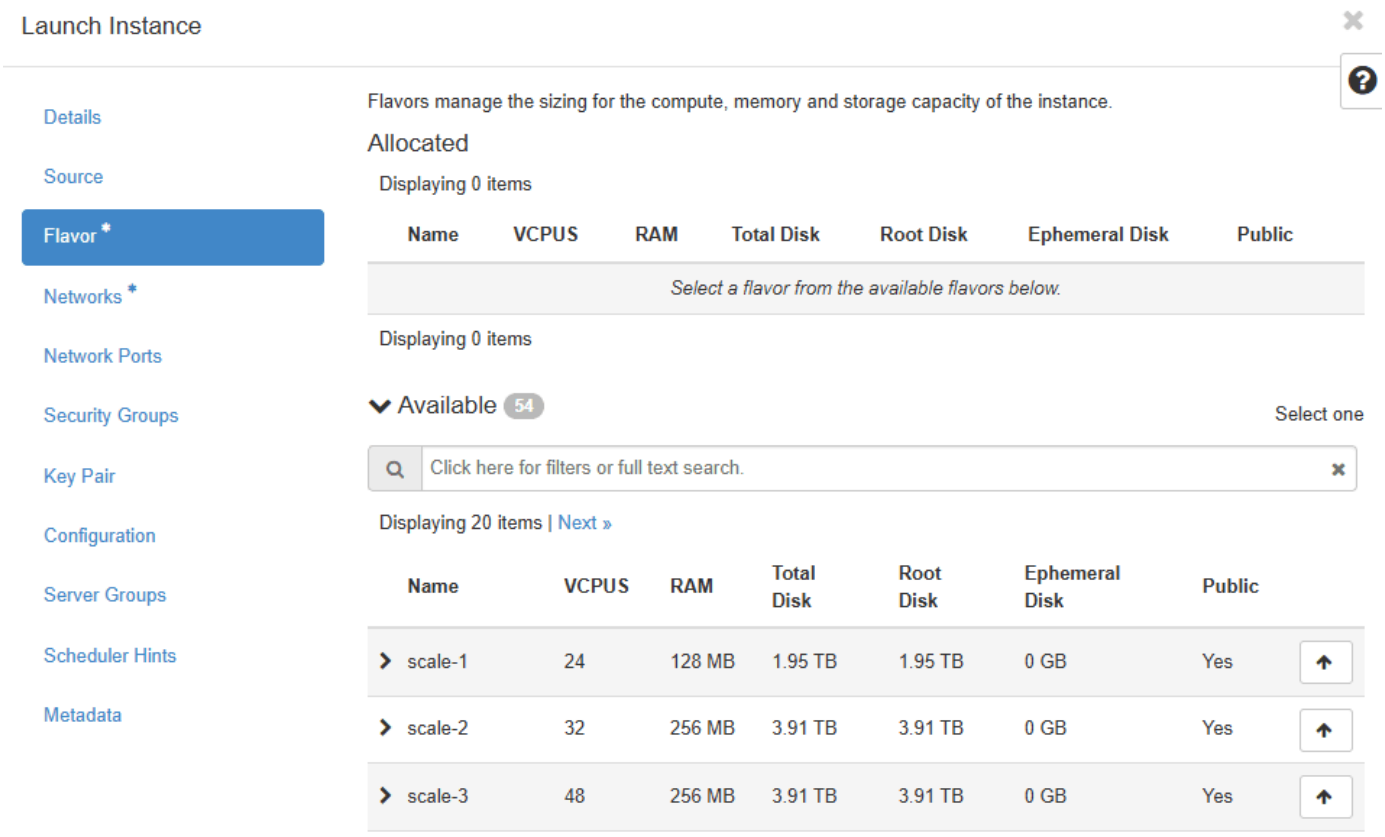
Champ	Description
Instance Name	Saisissez le nom de l'instance à créer.
Description	Optionnel. Ajoutez une description si nécessaire.
Availability Zone	Laissez la valeur par défaut **nova**.
Count	Indiquez le nombre d'instances à déployer.

Onglet : Source



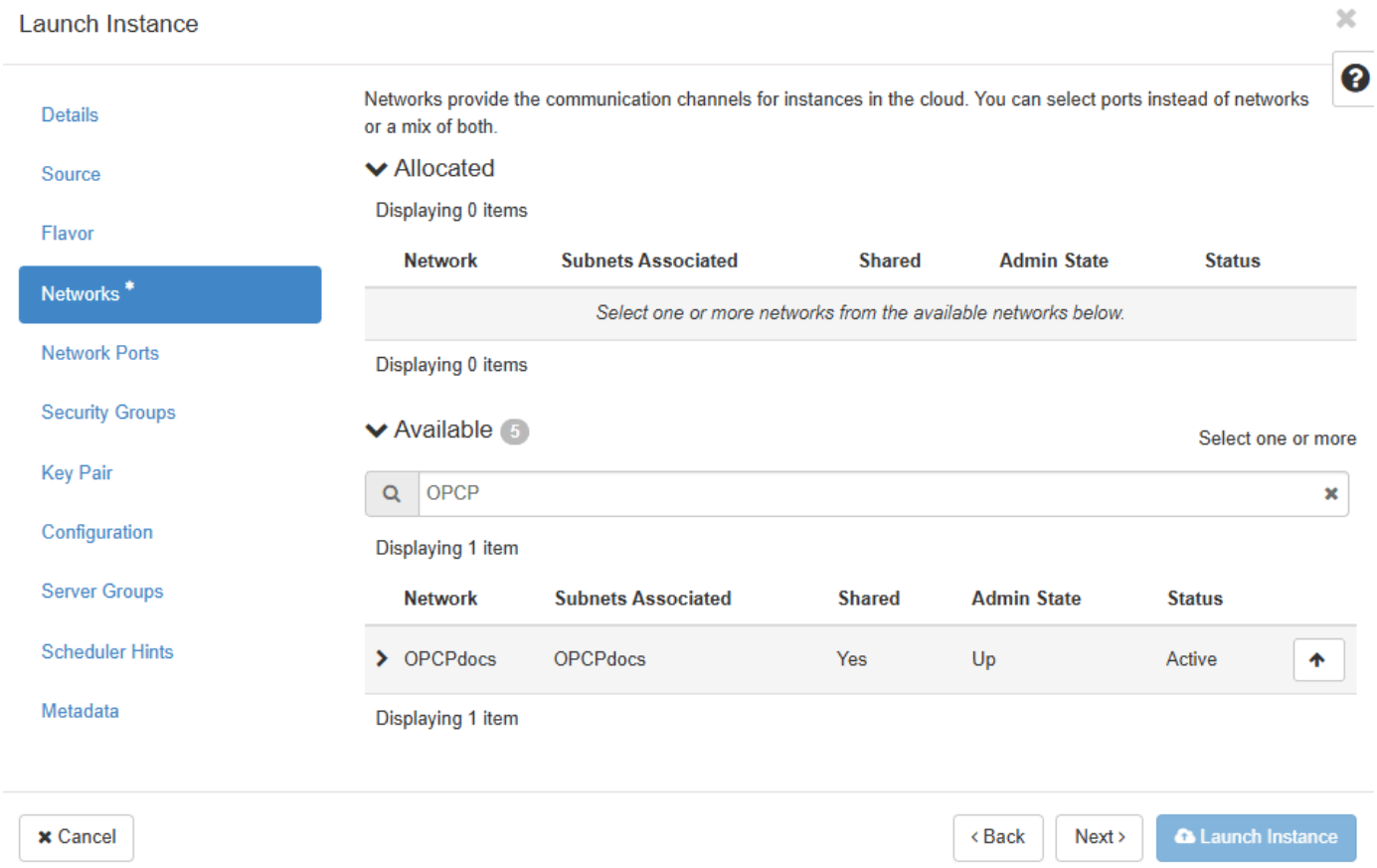
Champ	Description
Boot Source	Sélectionnez la source de démarrage : *Image* ou *Instance Snapshot*.
Image Name	Choisissez l'image à utiliser (ex. : *Debian 12 BMPOD*).

Onglet : Flavor



Sélectionnez la **configuration matérielle** adaptée (vCPU, mémoire, stockage).

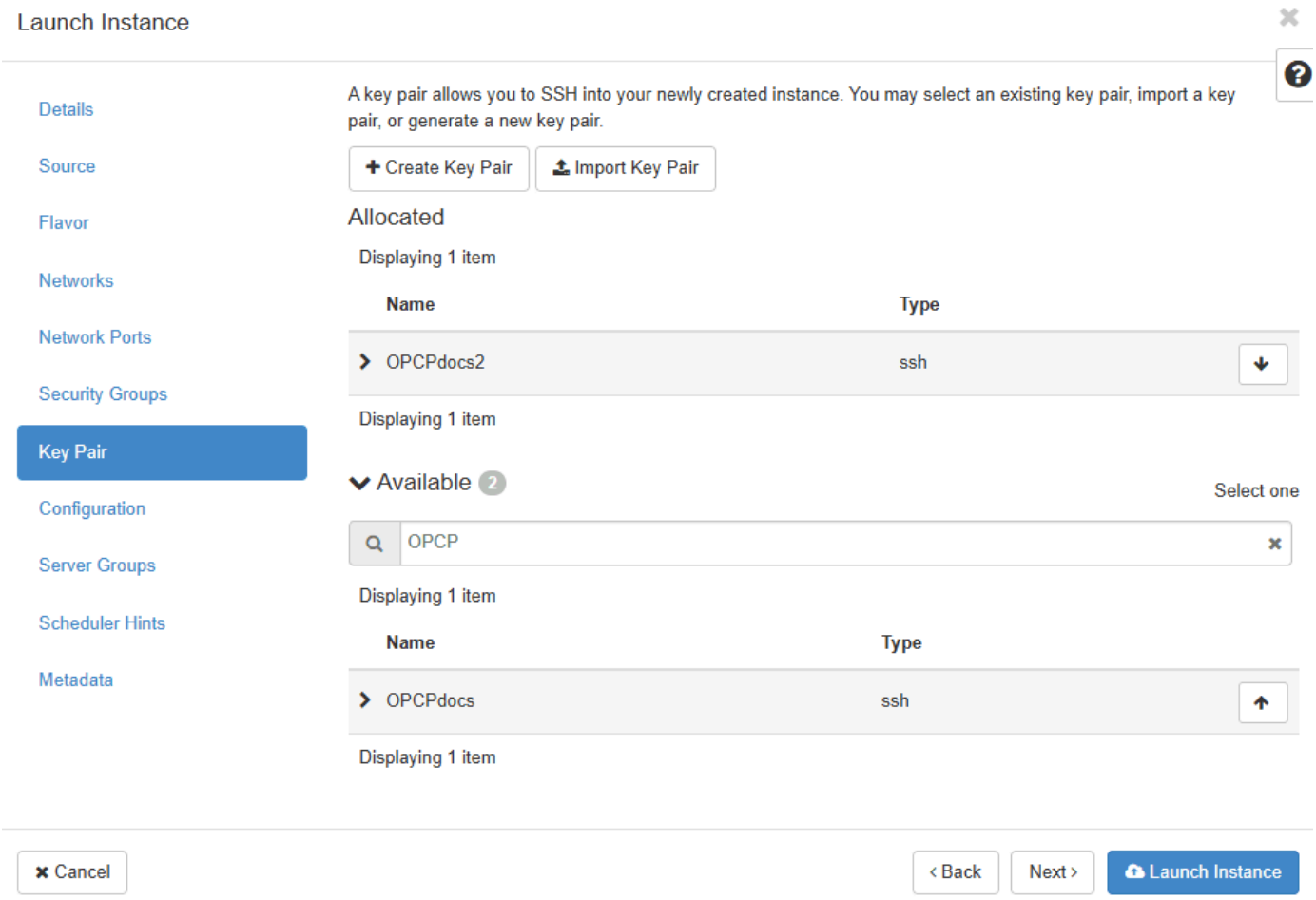
Onglet : Networks



Sélectionnez le **réseau privé** précédemment créé. Vous pouvez également attacher un **port réseau** existant depuis l'onglet *Network Ports*.

4. Gestion des paires de clés SSH

> **Warning**
> Bien que la sélection d'une clé SSH ne soit pas obligatoire dans Horizon, elle est **indispensable pour se connecter à l'instance** une fois celui-ci créé.

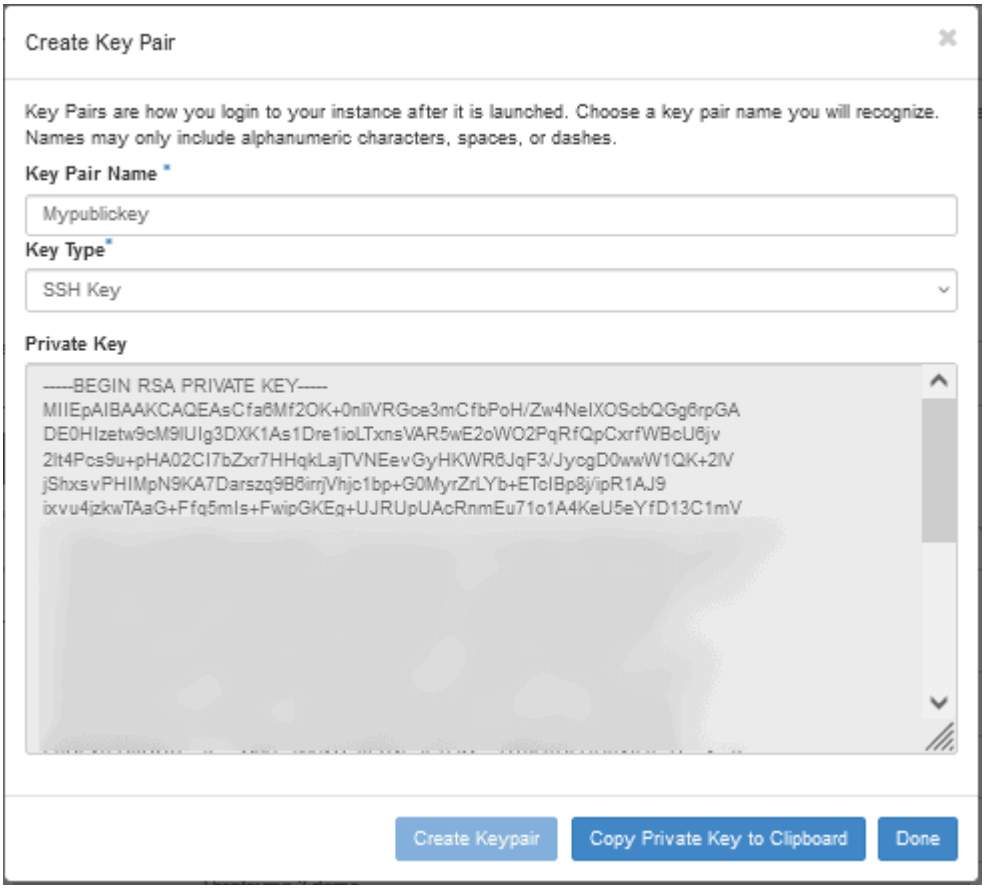


Créer une nouvelle paire de clés
Importer une clé existante

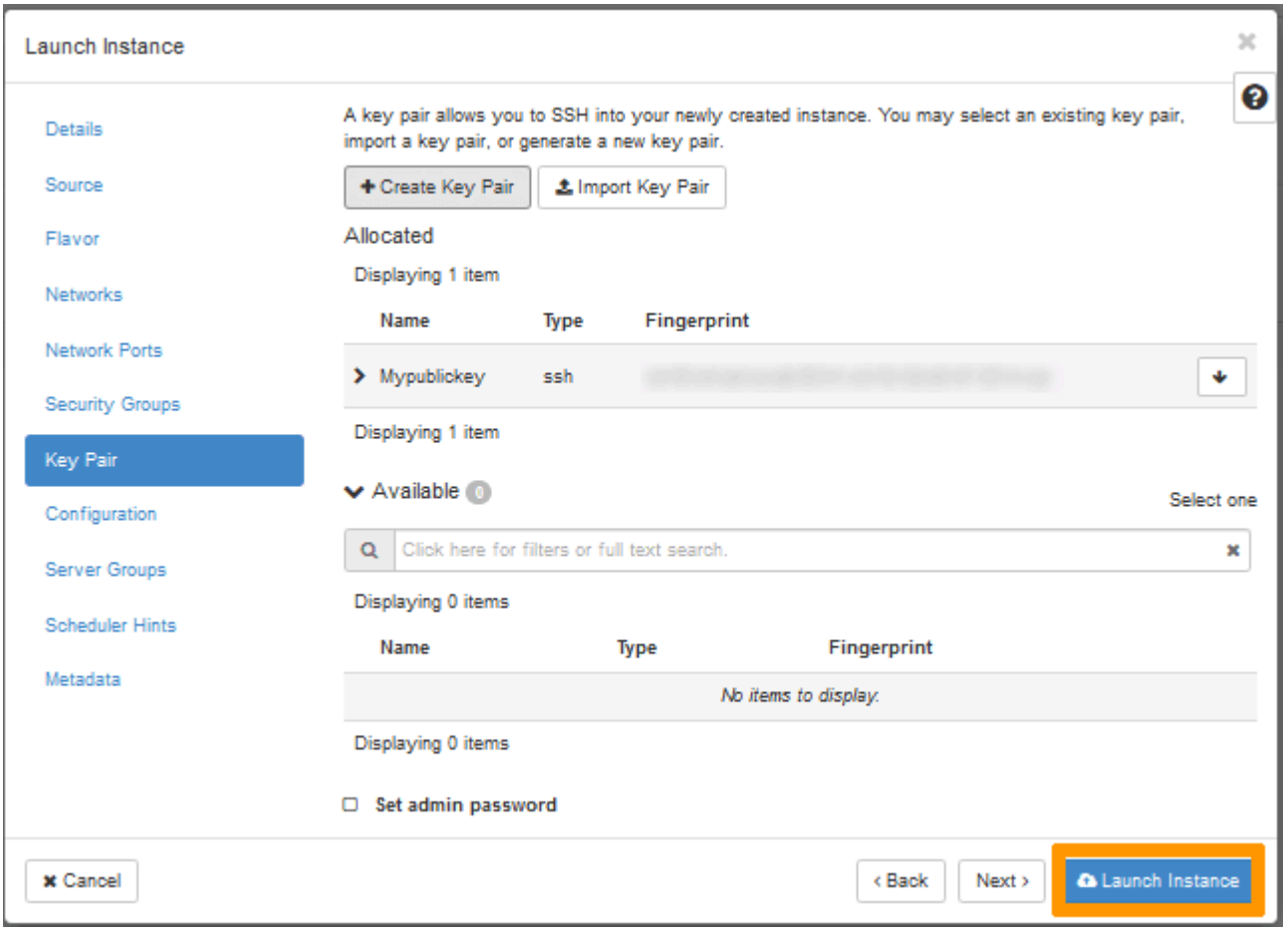
Cliquez sur **+ Create Key Pair** et renseignez les champs suivants :

Champ	Description
Key Pair Name	Saisissez un nom pour la clé.
Key Type	Sélectionnez **SSH Key**.

Cliquez sur **Create Keypair**.
Copiez la clé privée avec **Copy Private Key to Clipboard**, puis cliquez sur **Done**.



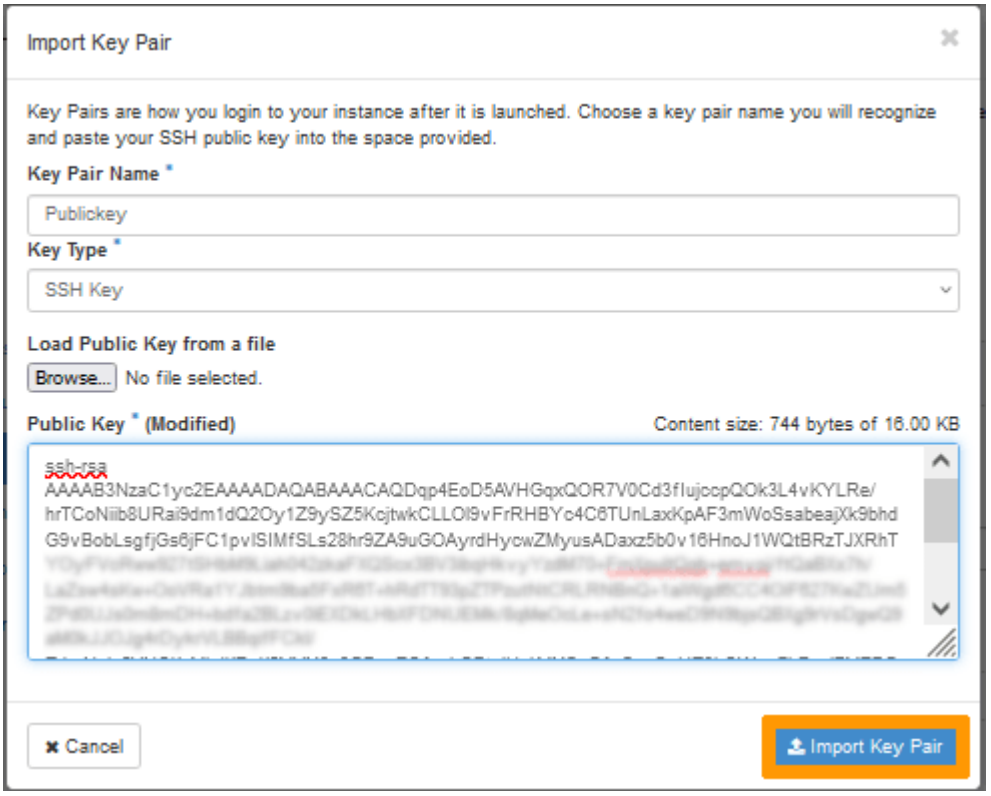
La clé est désormais sélectionnée par défaut. Cliquez sur **Launch Instance** pour démarrer la création de l'instance.



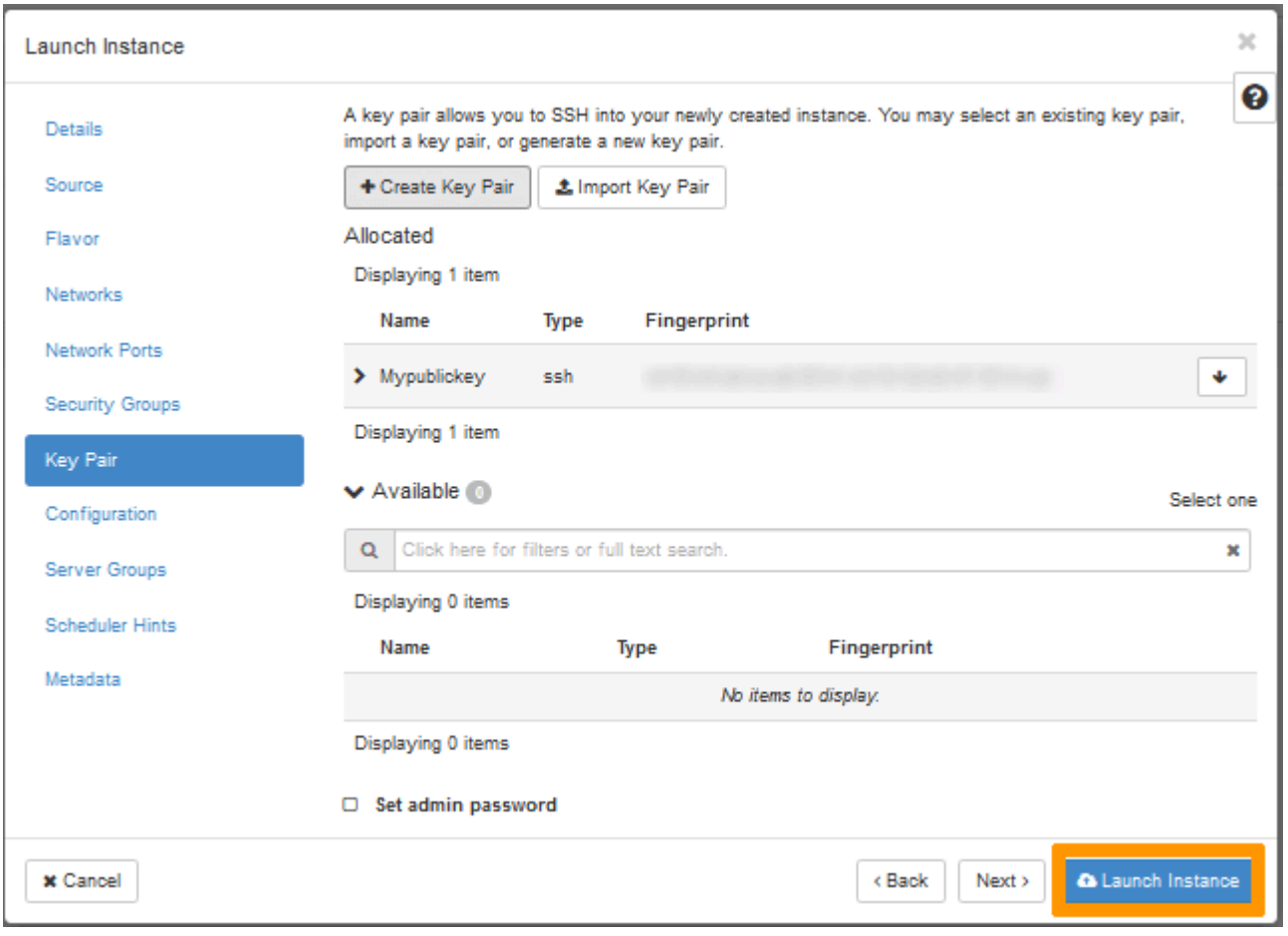
Cliquez sur **Import Key Pair** et renseignez les champs suivants :

Champ	Description
Key Pair Name	Nom de la clé.
Key Type	Sélectionnez **SSH Key**.
Public Key	Collez votre clé publique ou importez le fichier correspondant.

Cliquez sur **Import Key Pair**.



La clé est désormais sélectionnée par défaut. Cliquez sur **Launch Instance** pour démarrer la création de l'instance.



5. Autres options

Les autres onglets de configuration (Security Groups, Configuration, Metadata, etc.) ne sont pas nécessaires pour une installation standard.
Pour aller plus loin, consultez la documentation officielle OpenStack.

6. Références

- OpenStack Official Documentation – Horizon
- OpenStack Networking Guide (Neutron)
- OpenStack Compute Guide (Nova)
- OpenStack Key Pairs
- Debian 12 Official Site

Aller plus loin

Échangez avec notre communauté d’utilisateurs.

Comment installer une instance depuis la CLI OpenStack

Découvrez comment déployer une instance OPCP via l'outil en ligne de commande OpenStack en configurant les réseaux, sous-réseaux, instances et clés SSH

Objectif

Ce guide détaille les étapes à suivre pour installer un nœud OPCP via la création d'une instance à partir de la CLI OpenStack. Avant de pouvoir déployer des services sur vos baies **OPCP**, il est nécessaire de disposer au moins d'un nœud installé et actif.

Prérequis

- Disposer d'un service OPCP actif.
- Posséder un compte utilisateur avec les droits suffisants pour se connecter aux API OpenStack.
- Préparer l'environnement pour utiliser l'API OpenStack.
- Charger les variables d'environnement pour le projet.

En pratique

Vous pouvez obtenir la liste des commandes OpenStack disponibles à l'aide de la commande suivante :

```
openstack command list
```

Vous pouvez filtrer les commandes affichées en indiquant le groupe :

```
openstack command list --group compute
```

Il est aussi possible d'avoir des informations concernant une commande en ajoutant `help` devant celle ci :

```
openstack help flavor list

usage: openstack flavor list [-h] [-f {csv,json,table,value,yaml}] [-c COLUMN]
                             [--quote {all,minimal,none,nonnumeric}] [--noindent]
                             [--max-width `<integer>`] [--fit-width] [--print-empty]
                             [--sort-column SORT_COLUMN]
                             [--sort-ascending | --sort-descending] [--public | --private | --all]
                             [--min-disk `<min-disk>`] [--min-ram `<min-ram>`] [--long]
                             [--marker `<flavor-id>`] [--limit `<num-flavors>`]

List flavors ...
```

> **Tip**
> Consultez la documentation du client OpenStack directement sur le site OpenStack.

Récupérer les paramètres nécessaires à la création d'une instance

Créer un réseau privé (*private network*) et un sous réseau (*subnet*)

Etape 1 : Créer le réseau privé

Avant de déployer votre instance, il est généralement nécessaire de créer un **réseau privé** afin qu'il soit accessible au sein de votre infrastructure locale. Si vous avez déjà un réseau privé avec un subnet sur votre projet que vous souhaitez utiliser, vous pouvez ignorer l'étape de création et directement lister vos réseaux pour récupérer le nom ou l'ID du réseau concerné.

```
openstack network create $NETWORK_NAME
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | UP                                   |
| availability_zone_hints   |                                      |
| availability_zones        |                                      |
| created_at                | 2025-11-05T15:18:06Z                 |
| description               |                                      |
| dns_domain                | None                                 |
| id                        | da5ed9ae-65c2-441b-99f4-75d8eb87c214 |
| ipv4_address_scope        | None                                 |
| ipv6_address_scope        | None                                 |
| is_default                | False                                |
| is_vlan_transparent       | None                                 |
| mtu                       | 1500                                 |
| name                      | opcp-docs                            |
| port_security_enabled     | True                                 |
| project_id                | 057ac6e82ade49078a5c66f4371eaf22     |
| provider:network_type     | vlan                                 |
| provider:physical_network | physnet1                             |
| provider:segmentation_id  | 2140                                 |
| qos_policy_id             | None                                 |
| revision_number           | 1                                    |
| router:external           | Internal                             |
| segments                  | None                                 |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tags                      |                                      |
| updated_at                | 2025-11-05T15:18:06Z                 |
+---------------------------+--------------------------------------+
```

Par défaut, un réseau n'est visible que par le projet qui l'a créé (ainsi que par les utilisateurs administrateurs).

Si vous souhaitez créer un réseau **partagé entre tous vos projets**, vous pouvez utiliser le paramètre `--share`.

Pour partager un réseau uniquement avec certains projets spécifiques, il est nécessaire d'utiliser le mécanisme **Role-Based Access Control (RBAC)** d'OpenStack : Documentation RBAC Neutron.

Par ailleurs, si vous souhaitez créer le réseau privé dans un **VLAN particulier**, vous pouvez le préciser à l'aide des paramètres suivants :

- `--provider-network-type vlan`
- `--provider-physical-network physnet1`
- `--provider-segment $VLAN_ID`

Par exemple, si vous souhaitez créer un réseau privé partagé dans le VLAN 2025 qui se nomme opcpdocs:

```
openstack network create --share --provider-network-type vlan --provider-physical-network physnet1 --provider-segment
2025 opcpdocs
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | UP                                   |
| availability_zone_hints   |                                      |
| availability_zones        |                                      |
| created_at                | 2025-11-05T15:34:30Z                 |
| description               |                                      |
| dns_domain                | None                                 |
| id                        | 2a92f464-e1c0-4daa-8ecd-b3ba6b3b96da |
| ipv4_address_scope        | None                                 |
| ipv6_address_scope        | None                                 |
| is_default                | False                                |
| is_vlan_transparent       | None                                 |
| mtu                       | 1500                                 |
| name                      | opcpdocs                             |
| port_security_enabled     | True                                 |
| project_id                | 07f0c728fe844d778cda63ca28547937     |
| provider:network_type     | vlan                                 |
| provider:physical_network | physnet1                             |
| provider:segmentation_id  | 2025                                 |
| qos_policy_id             | None                                 |
| revision_number           | 1                                    |
| router:external           | Internal                             |
| segments                  | None                                 |
| shared                    | True                                 |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tags                      |                                      |
| updated_at                | 2025-11-05T15:34:30Z                 |
+---------------------------+--------------------------------------+
```

Une fois le réseau créé, vous pouvez le lister via la commande :

```
openstack network list --name $NETWORK_NAME
+--------------------------------------+-----------+---------+
| ID                                   | Name      | Subnets |
+--------------------------------------+-----------+---------+
| 2a92f464-e1c0-4daa-8ecd-b3ba6b3b96da | opcp-docs |         |
+--------------------------------------+-----------+---------+
```

Si nécessaire, vous pouvez lister l'ensemble des réseaux en retirant l'argument `--name`.

Etape 2 : Créer le subnet

Par défaut, le seul élément nécessaire pour créer un subnet sur votre réseau est le CIDR que vous souhaitez configurer et le réseau que vous venez de créer :

```
openstack subnet create --network $NETWORK_NAME --subnet-range 192.168.120.0/24 $SUBNET_NAME
```

Si vous souhaitez cependant préciser un *allocation pool*, vous pouvez le spécifier via différents paramètres. Par exemple, si vous souhaitez créer un sous réseau avec le CIDR 192.168.120.0/24 en allouant uniquement 50 adresses IP du CIDR et avec une gateway spécifique, vous pouvez utiliser la commande suivante :

```
openstack subnet create --network opcpdocs --subnet-range 192.168.120.0/24 --allocation-pool
start=192.168.120.11,end=192.168.120.60 --gateway 192.168.120.8 opcpdocs-subnet
+----------------------+--------------------------------------+
| Field                | Value                                |
+----------------------+--------------------------------------+
| allocation_pools     | 192.168.120.11-192.168.120.60        |
| cidr                 | 192.168.120.0/24                     |
| created_at           | 2025-11-05T16:17:58Z                 |
| description          |                                      |
| dns_nameservers      |                                      |
| dns_publish_fixed_ip | None                                 |
| enable_dhcp          | True                                 |
| gateway_ip           | 192.168.120.8                        |
| host_routes          |                                      |
| id                   | b532e25f-3aae-4396-99e9-ad6811a03a6e |
| ip_version           | 4                                    |
| ipv6_address_mode    | None                                 |
| ipv6_ra_mode         | None                                 |
| name                 | opcpdocs-subnet                      |
| network_id           | 2a92f464-e1c0-4daa-8ecd-b3ba6b3b96da |
| project_id           | 07f0c728fe844d778cda63ca28547937     |
| revision_number      | 0                                    |
| segment_id           | None                                 |
| service_types        |                                      |
| subnetpool_id        | None                                 |
| tags                 |                                      |
| updated_at           | 2025-11-05T16:17:58Z                 |
+----------------------+--------------------------------------+
```

Ce subnet pourra être utilisé pour déployer une instance, permettant ainsi à OpenStack d'attribuer une adresse IP à celle-ci lors de l'installation.

Ajout d'une clé SSH publique

Dans un premier temps, il est nécessaire d'ajouter une clé SSH publique qui permettra de se connecter sur les instances.

- Lister les commandes liées aux clés SSH :

```
openstack help | grep keypair
  keypair create  Create new public or private key for server ssh access
  keypair delete  Delete public or private key(s)
  keypair list    List key fingerprints
  keypair show    Display key details
```

- Ajouter la clé SSH publique :

```
openstack keypair create --public-key ~/.ssh/id_rsa.pub $SSHKEY
```

- Lister les clés SSH disponibles :

```
openstack keypair list
+--------------+-------------------------------------------------+------+
| Name         | Fingerprint                                     | Type |
+--------------+-------------------------------------------------+------+
| SSHKEY       | 5c:fd:9d:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:3a | ssh  |
+--------------+-------------------------------------------------+------+
```

Lister les modèles d'instance

Il faudra ensuite récupérer l'ID ou le nom du modèle que l'on souhaite utiliser :

```
openstack flavor list
+--------------------------------------+---------------------+---------+------+-----------+-------+-----------+
| ID                                   | Name                |     RAM | Disk | Ephemeral | VCPUs | Is Public |
+--------------------------------------+---------------------+---------+------+-----------+-------+-----------+
| 4e731d33-4c1e-4286-bc63-68c2cba3bc59 | gpu                 |     400 |  900 |         0 |   128 | True      |
| 8bc6b1e0-dfdd-44d8-83fb-cef5e8fb6ec2 | scale-2             |     256 | 4000 |         0 |    32 | True      |
| f975e49a-fd75-4a7e-868e-2842972e82e2 | hsm                 |     512 | 4000 |         0 |   128 | True      |
| fcd2c556-9a16-4532-a5c8-e9d9275ba78f | scale-1             |     128 | 2000 |         0 |    24 | True      |
| ff5b4d7d-46c9-4bf5-b300-f1f465661e64 | scale-3             |     256 | 4000 |         0 |    48 | True      |
+--------------------------------------+---------------------+---------+------+-----------+-------+-----------+
```

Lister les images disponibles

Pour finir, il suffit de récupérer l'ID ou le nom de l'image qui sera utilisée pour l'instance :

```
openstack image list
+--------------------------------------+-----------------------------------------------+--------+
| ID                                   | Name                                          | Status |
+--------------------------------------+-----------------------------------------------+--------+
| 6540686f-0150-496b-a894-03d95ef8bc7e | ironic-agent.initramfs                        | active |
| 5f6d4237-023b-4a25-8ceb-7051ffc6d632 | ironic-agent.kernel                           | active |
| 7600f9fc-fc1f-4fb8-aaa3-878bb9399d26 | CentOS 9 OPCP                                 | active |
| ebabdaaa-a52d-4246-9790-e77ba5c49519 | Debian 12 LVM OPCP                            | active |
+--------------------------------------+-----------------------------------------------+--------+
```

Installation d'une instance

Avec les éléments récupérés précédemment, vous pouvez créer une instance pour déployer un système d'exploitation, ainsi que votre clé publique SSH sur la flavor souhaitée :

```
openstack server create --key-name OPCPdocs2 --flavor scale-1 --image "Debian 12 LVM OPCP" --network opcpdocs
OPCPdocs-server
+-------------------------------------+-------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------
---------------------------------------------+
| Field                               | Value
|
+-------------------------------------+-------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------
---------------------------------------------+
| OS-DCF:diskConfig                   | MANUAL
|
| OS-EXT-AZ:availability_zone         | None
|
| OS-EXT-SRV-ATTR:host                | None
|
| OS-EXT-SRV-ATTR:hostname            | opcpdocs-server
|
| OS-EXT-SRV-ATTR:hypervisor_hostname | None
|
| OS-EXT-SRV-ATTR:instance_name       | None
|
| OS-EXT-SRV-ATTR:kernel_id           | None
|
| OS-EXT-SRV-ATTR:launch_index        | None
|
| OS-EXT-SRV-ATTR:ramdisk_id          | None
|
| OS-EXT-SRV-ATTR:reservation_id      | r-900xkw3k
|
| OS-EXT-SRV-ATTR:root_device_name    | None
|
| OS-EXT-SRV-ATTR:user_data           | None
|
| OS-EXT-STS:power_state              | N/A
|
| OS-EXT-STS:task_state               | scheduling
|
| OS-EXT-STS:vm_state                 | building
|
| OS-SRV-USG:launched_at              | None
|
| OS-SRV-USG:terminated_at            | None
|
| accessIPv4                          | None
|
| accessIPv6                          | None
|
| addresses                           | N/A
|
| adminPass                           | VbRFzvGnk9rP
|
| config_drive                        | None
|
| created                             | 2025-11-05T16:41:32Z
|
| description                         | None
|
| flavor                              | description=, disk='2000', ephemeral='0',
extra_specs.:architecture='bare_metal', extra_specs.resources:CUSTOM_DISCOVERED='1',
extra_specs.resources:DISK_GB='0', extra_specs.resources:MEMORY_MB='0', extra_specs.resources:VCPU='0',         |
|                                     | extra_specs.trait:CUSTOM_SCALE1='required', id='scale-1', is_disabled=,
is_public='True', location=, name='scale-1', original_name='scale-1', ram='128', rxtx_factor=, swap='0', vcpus='24'
|
```

```
| hostId                              | None
|
| host_status                         | None
|
| id                                  | 2f9c2e41-b4dc-4787-a3ef-9b2b2d686dbb
|
| image                               | Debian 12 LVM OPCP (ebabdaaa-a52d-4246-9790-e77ba5c49519)
|
| key_name                            | OPCPdocs2
|
| locked                              | None
|
| locked_reason                       | None
|
| name                                | OPCPdocs-server
|
| pinned_availability_zone            | None
|
| progress                            | None
|
| project_id                          | 07f0c728fe844d778cda63ca28547937
|
| properties                          | None
|
| security_groups                     | name='default'
|
| server_groups                       | None
|
| status                              | BUILD
|
| tags                                |
|
| trusted_image_certificates          | None
|
| updated                             | 2025-11-05T16:41:32Z
|
| user_id                             | b1481b3f72e8aecda9a623b215085227f9d85170f4fc4524b3a4ab1a0b97dfde
|
| volumes_attached                    |
|
+-------------------------------------+--------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------
-------------------------------------------+
```

Par défaut, l'instance qui va être installée est sélectionnée automatiquement parmi le pool de nœuds ayant le statut `Available` et pour lesquels les **traits** requis par la flavor correspondent avec l'installation demandée. Cela signifie qu'un serveur physique correpondant aux contraintes décrites par les **traits** sera sélectionné.

Après plusieurs minutes, l'instance est déployée et vous pouvez retrouver vos instances installées via la commande suivante :

```
openstack server list
+--------------------------------------+-----------------+--------+-------------------------+---------------------+--
-------+
| ID                                   | Name            | Status | Networks                | Image               |
Flavor  |
+--------------------------------------+-----------------+--------+-------------------------+---------------------+--
-------+
| 2f9c2e41-b4dc-4787-a3ef-9b2b2d686dbb | OPCPdocs-server | ACTIVE | opcpdocs=192.168.120.59 | Debian 12 LVM OPCP  |
scale-1 |
+--------------------------------------+-----------------+--------+-------------------------+---------------------+--
-------+
```

Si vous souhaitez installer l'instance sur un nœud spécifique, vous pouvez spécifier l'identifiant de votre nœud dans votre commande :

```
openstack server create --flavor $flavor_ID --image $image_ID --network $network_ID --key-name $your_keyname --
availability-zone nova::$baremetal_nœud_ID $server_name
```

Il faudra cependant vous assurer que le nœud a le statut `Available` et possède bien les **traits** nécessaires pour installer la flavor souhaitée.

Pour vérifier l'état actuel du nœud et récupérer son identifiant, consultez le guide « Cycle de vie d'un nœud OPCP ».

Redémarrer ou arrêter une instance

Vous pouvez gérer l'état d'alimentation de votre instance directement depuis l'API OpenStack :

```
# Redémarrage logiciel (soft reboot)
openstack server reboot $INSTANCE_ID

# Redémarrage matériel (hard reboot)
openstack server reboot --hard $INSTANCE_ID

# Arrêter l'instance
openstack server stop $INSTANCE_ID

# Démarrer l'instance
openstack server start $INSTANCE_ID
```

> **Warning**
> L'arrêt d'une instance doit toujours être effectué depuis l'API OpenStack ou l'interface Horizon, et non depuis le système d'exploitation via des commandes telles que `systemctl poweroff`. Une action réalisée directement depuis l'OS n'est pas propagée à OpenStack, qui continuerait de considérer l'instance comme active : elle se retrouverait dans un état incohérent entre son statut réel et celui connu par la plateforme. Dans le cas où un nœud resterait bloqué dans un état intermédiaire après une extinction effectuée depuis l'OS, vous pouvez le débloquer en forçant son arrêt puis son démarrage avec les commandes `openstack baremetal node power off <node>` et `openstack baremetal node power on <node>`.

Suppression d'une instance

Vous pouvez supprimer une instance grâce à la commande suivante :

```
openstack server delete $INSTANCE_ID
```

Votre nœud passera en état `Cleaning`. Cette étape consiste à la réinitialisation matérielle du serveur physique et de l'effacement des données présentes sur les disques.

Durant cette étape, vous ne verrez plus l'instance dans la liste des instances, cependant le nœud ne sera pas disponible immédiatement pour une nouvelle installation. N'oubliez pas de prendre en compte ce délai lors de vos opérations de maintenance. L'opération peut prendre plusieurs minutes avant que le nœud soit de nouveau `Available` et disponible pour une nouvelle installation.

Références

- OpenStack Official Documentation - Client.
- OpenStack Official Documentation - Network.

Aller plus loin

Si vous avez besoin d'une formation ou d'une assistance technique pour la mise en oeuvre de nos solutions, contactez votre commercial ou cliquez sur ce lien pour obtenir un devis et demander une analyse personnalisée de votre projet à nos experts de l’équipe Professional Services.

Échangez avec notre communauté d’utilisateurs.

Comment configurer LACP sur un nœud

Apprenez à configurer un nœud dans OpenStack pour utiliser LACP (Link Aggregation Control Protocol)

Objectif

La configuration de LACP doit être appliquée sur le nœud avant de déployer une instance, afin que les interfaces réseau soient correctement agrégées.

Ce guide explique comment configurer un nœud (serveur physique) dans OPCP pour activer LACP (Link Aggregation Control Protocol).

Nous verrons également comment configurer le **bonding** (association logique de plusieurs interfaces réseau pour former une seule interface virtuelle) au niveau de votre instance, afin de tirer pleinement parti du **LACP**.

> **Warning**
> Un utilisateur standard ne peut pas configurer LACP lui-même. Vous devez être **admin**, ou disposer de **nœud disponible** dans votre projet OpenStack.
>
> Il est recommandé de configurer LACP **avant** le déploiement d'une instance. Ce guide **ne couvre pas** la configuration d'un nœud déjà en production.

Pourquoi utiliser LACP ?

LACP peut être utilisé dans deux cas d'usage précis :

- **Augmenter la capacité réseau :** En agrégeant plusieurs cartes réseau, vous pourrez ajouter la bande passante de chaque carte réseau ajoutée à cet agrégat, pour les mêmes réseaux.
- **Augmenter la résilience :** Chaque serveur de OPCP bénéficie de 4 interfaces 25G. Chaque serveur est relié à deux switchs réseau (A&B) pour assurer la résilience en cas de panne matérielle de l'un d'entre eux. LACP permet de créer des interfaces virtuelles résilientes en cas de panne matérielle en aggrégant deux cartes réseau reliées sur des switchs distincts.

Prérequis

Avant de commencer, assurez-vous de disposer des éléments suivants :

- Disposer d'un service OPCP actif.
- Un accès **OpenStack CLI configuré** avec les droits nécessaires (`clouds.yaml` ou variables d'environnement).
- Le rôle **admin** et/ou ou des nœuds transférés dans votre projet.

LACP est une configuration réseau spécifique, nécessitant des connaissances réseau et système avancées. Nous vous conseillons d'appliquer ce guide si vous connaissez déjà l'un ou plusieurs des concepts suivants : configuration de nœuds au sein de OpenStack Ironic, gestion des ports au sein de OpenStack Neutron, et la connaissance de la CLI OpenStack.

En pratique

Configuration du nœud

1. Lister les nœuds

La liste des nœuds disponibles dans votre projet peut être affichée avec la commande suivante :

```
openstack baremetal node list
```

Exemple de sortie :

```
+--------------------------------------+---------------+--------------------------------------+-------------+-------
-------------+-------------+
| UUID                                 | Name          | Instance UUID                        | Power State |
Provisioning State | Maintenance |
+--------------------------------------+---------------+--------------------------------------+-------------+-------
-------------+-------------+
| 88830859-5b16-4935-8f41-d381b754cbe5 | SERVER-1      | None                                 | power off   |
available          | False       |
| 726bb7e9-3b20-4a44-99d7-2af747983781 | SERVER-2      | None                                 | power off   |
available          | False       |
| af711edf-a579-491e-b474-3c03639ed99b | SERVER-3      | 83b7083b-bbdd-4327-941c-ee91197818c5 | None        | active
| True        |
+--------------------------------------+---------------+--------------------------------------+-------------+-------
-------------+-------------+
```

2. Transférer la propriété d’un nœud (admin uniquement)

Un admin peut transférer la propriété d’un nœud à un projet donné :

```
openstack baremetal node set <node-id> --owner <project-id>
```

3. Lister les ports réseau

Chaque carte réseau physique, appelée Network Interface Card (NIC), d’un nœud est représentée dans OpenStack par un port.

Pour afficher la liste des ports associés à un nœud :

```
openstack baremetal port list --node <node-id>
```

Exemple de sortie :

```
+--------------------------------------+-------------------+
| UUID                                 | Address           |
+--------------------------------------+-------------------+
| 068a06b2-ebf9-48c9-a3c3-94016ca5e3da | 85:32:f2:87:29:da |
| 4937d704-7517-4525-86d1-abecb94a7ce9 | 85:32:f2:87:29:db |
| 422eece6-dcfa-40cd-975d-ba8bb11c774e | 85:32:f2:89:66:f8 |
| 34073903-92ad-47d1-a751-15aa96991415 | 85:32:f2:89:66:f9 |
+--------------------------------------+-------------------+
```

Pour visualiser les détails d'un port, y compris les informations sur la connexion physique.
Cela est particulièrement utile si vous souhaitez configurer un **bonding LACP 2x2**, en répartissant les NIC sur deux switches différents pour assurer une **meilleure redondance et tolérance aux pannes**.

```
openstack baremetal port show <port-id>
```

Exemple de sortie :

```
openstack baremetal port show 71899d54-546d-4fdd-8d8b-52ad986bf425
+----------------------+--------------------------------------------------------------------------------------------------
----+
| Field                | Value
|
+----------------------+--------------------------------------------------------------------------------------------------
----+
| address              | 85:32:f2:89:66:f9
|
| created_at           | 2025-01-06T10:43:38.020574+00:00
|
| extra                | {}
|
| internal_info        | {}
|
| is_smartnic          | False
|
| local_link_connection | {'switch_id': 'c0:00:15:8c:33:78', 'port_id': 'Ethernet25/2', 'switch_info': 'demo-tor1b-
a70'} |
| node_uuid            | ddc7c763-74fc-4900-b9e6-5463233836b0
|
| physical_network     | None
|
| portgroup_uuid       | c3d3fcb3-7a92-4257-b944-736d222e8c4a
|
| pxe_enabled          | False
|
| updated_at           | 2025-11-06T09:52:46.355883+00:00
|
| uuid                 | 71899d54-546d-4fdd-8d8b-52ad986bf425
|
+----------------------+--------------------------------------------------------------------------------------------------
----+
```

4. Activer le mode maintenance

Avant toute modification de configuration réseau, le nœud doit être placé en **mode maintenance**. Cela assure que ce nœud ne puisse pas être déployé durant toute l'opération :

```
openstack baremetal node maintenance set <node-id>
```

5. Créer un groupe de ports (LACP Bond)

Le **groupe de ports** permet d’activer l’agrégation LACP entre plusieurs interfaces réseau.

> **Tip**
>
> Vous pouvez créer :
>
> - un **groupe de ports unique** pour un bond 1×4, ou
> - deux **groupes de ports** pour des bonds 2×2.

Utilisez le paramètre `--mode 802.3ad` pour activer LACP.

Le paramètre `--address <MAC>` doit être égal à l'adresse MAC du port PXE s'il est utilisé. Sinon, vous pouvez omettre le paramètre ou définir la valeur MAC à partir de l'une des interfaces physiques que vous utiliserez.

Vous pouvez lister tous les ports avec `openstack baremetal port list --node <node-id> --long` et vérifier si PXE est utilisé ou non.

Remarque : il est recommandé de préfixer le nom du groupe de ports avec le nom du nœud (ex. : `<node-name>-<name>`) pour faciliter l'identification.

Exemple :

```
openstack baremetal port group create \
  --node 88830859-5b16-4935-8f41-d381b754cbe5 \
  --name node_name-pg-lacp \
  --mode 802.3ad \
  --address 00:00:00:20:00:01
```

Exemple de sortie :

```
+---------------------------+------------------------------------------+
| Champ                     | Valeur                                   |
+---------------------------+------------------------------------------+
| uuid                      | d082c2ab-5960-44e3-920d-3d6dfb6811e9     |
| address                   | 00:00:00:20:00:01                        |
| node_uuid                 | 88830859-5b16-4935-8f41-d381b754cbe5     |
| name                      | node_name-pg-lacp                        |
| mode                      | 802.3ad                                  |
| standalone_ports_supported | True                                    |
+---------------------------+------------------------------------------+
```

6. Associer les ports au groupe

Chaque port du nœud doit être associé au groupe de ports créé :

```
openstack baremetal port set --port-group `<port-group-id>` `<port-id>`
```

Exemple :

```
openstack baremetal port set --port-group d082c2ab-5960-44e3-920d-3d6dfb6811e9 068a06b2-ebf9-48c9-a3c3-94016ca5e3da
openstack baremetal port set --port-group d082c2ab-5960-44e3-920d-3d6dfb6811e9 4937d704-7517-4525-86d1-abecb94a7ce9
openstack baremetal port set --port-group d082c2ab-5960-44e3-920d-3d6dfb6811e9 422eece6-dcfa-40cd-975d-ba8bb11c774e
openstack baremetal port set --port-group d082c2ab-5960-44e3-920d-3d6dfb6811e9 34073903-92ad-47d1-a751-15aa96991415
```

7. Désactiver le mode maintenance

Une fois la configuration terminée, désactivez le mode maintenance :

```
openstack baremetal node maintenance unset  `<node-id>`
```

8. Créer une instance sur le nœud configuré

Une fois votre nœud configuré avec LACP, vous pouvez déployer une instance.

Par défaut, OpenStack sélectionne un hôte en fonction du **flavor** choisi et des **règles du scheduler**, ce qui ne garantit pas que le nœud que vous venez de configurer sera utilisé.

Pour vous assurer que votre instance sera déployée sur ce nœud précis, vous pouvez cibler ce dernier utilisant sa **zone de disponibilité** :

```
openstack server create --availability-zone "nova::`<node-id>`"
```

Exemple :

```
openstack server create --image `<image-name>` \
   --nic net-id=NETWORK1,v4-fixed-ip=198.18.56.200 \
   --flavor `<flavor-id>` \
   --key-name `<keypair-name>`  \
   --availability-zone "nova::`<node-id>`" \
   `<instance-name>`
```

Résumé des étapes

Étape	Action	Commande
1	Lister les nœuds	`openstack baremetal node list`
2	Transférer la propriété	`openstack baremetal node set <node-id> --owner <tenant-id>`
3	Lister les ports	`openstack baremetal port list --node <node-id>`
4	Activer le mode maintenance	`openstack baremetal node maintenance set <node-id>`
5	Créer un groupe LACP	`openstack baremetal port group create --node <node-id> --name <port-group-name> --mode 802.3ad`
6	Associer les ports	`openstack baremetal port set --port-group <port-group-id> <port-id>`
7	Désactiver le mode maintenance	`openstack baremetal node maintenance unset <node-id>`
8	Créer une instance	`openstack server create --image <image-name> --nic net-id=<network-1> --flavor <flavor-id> --key-name <keypair-name> --availability-zone "nova::<node-id>" <instance-name>`

Configuration du système d’exploitation de l’instance

Après avoir configuré votre nœud dans OpenStack et déployé un système d'exploitation, il reste à configurer le réseau afin de bénéficier du network **bonding**, qui permet d'agréger plusieurs interfaces réseau pour plus de performance et de redondance.

Vérifier la configuration du bonding

Sur certaines images (comme **Debian 12** ou **Ubuntu 22.04**), la configuration du **bonding** est automatiquement détectée et configurée.
Cependant, d'autres distributions peuvent nécessiter un ajustement manuel.

1. Vérifier les bonds actifs

```
ls /proc/net/bonding/
bond0
```

2. Vérifier les interfaces membres

```
cat /proc/net/bonding/bond0 | grep Interface
Slave Interface: ens22f1np1
Slave Interface: ens22f0np0
Slave Interface: ens21f1np1
Slave Interface: ens21f0np0
```

3. Vérifier la politique de hachage (`Transmit Hash Policy`)

```
cat /proc/net/bonding/* | grep Trans
Transmit Hash Policy: layer2 (0)
```

> **Warning**
> Pour exploiter toute la bande passante, configurez la politique `layer3+4` (par défaut, certains OS utilisent `layer2` , moins performante). Plus de détails : Ubuntu Bonding Documentation

Modifier la configuration

1. Changement à chaud (non persistant)

Si vous souhaitez tester votre configuration manuellement, vous pouvez utiliser la commande suivante :

```
sudo ip link set bond0 type bond xmit_hash_policy layer3+4
```

Attention, cette configuration sera réinitialisée si votre machine redémarre.

2. Changement persistant (exemple via Netplan et cloud-init)

Créez votre fichier de configuration (ex. `/etc/cloud/cloud.cfg.d/99-custom-network.cfg`) pour y inclure :

```
network:
  version: 2
  ethernets:
    ens21f0np0: {}
    ens21f1np1: {}
    ens22f0np0: {}
    ens22f1np1: {}
  bonds:
    bond0:
      dhcp4: true
      interfaces:
        - ens21f0np0
        - ens21f1np1
        - ens22f0np0
        - ens22f1np1
      macaddress: 00:00:00:20:00:23
      parameters:
        mode: 802.3ad
        transmit-hash-policy: layer3+4
        lacp-rate: fast
        mii-monitor-interval: 100
```

Puis appliquez la configuration en rédémarrant l'instance.

3. Tester la bande passante avec `iperf3`

Pour tester correctement LACP, vous devez disposer de **2 nœuds** dans le **même réseau**, tous deux configurés avec LACP.

Serveur Iperf3 (nœud 1)

```
iperf3 -s
```

Client Iperf3 (nœud 2)

Utilisez `-P` pour générer plusieurs flux parallèles, afin d'atteindre la bande passante maximale.

```
iperf3 -c <ip-du-nœud> -P 64
```

Exemple de résultat

```
[SUM] 0.0000-10.0121 sec   110 GBytes  94.0 Gbits/sec
```

Avec un lien 4×25 Gbps, vous devriez atteindre environ **100 Gbps**. Il peux être nécessaire d'ajuster certains paramètres système pour exploiter pleinement cette capacité.

Conclusion

Vous avez configuré :

- Le **LACP (802.3ad)** au niveau du nœud Baremetal OpenStack ;
- Le **paramétrage du bonding** dans l'OS invité ;
- Et validé la **performance réseau** via `iperf3`.

Votre instance est désormais prête à exploiter toute la bande passante disponible du lien agrégé.

Aller plus loin

Échangez avec notre communauté d’utilisateurs.

Comment configurer les ports Trunk sur un nœud

Apprenez à configurer les ports Trunk Neutron dans OPCP pour une connectivité multi-réseau vlan sur des instances baremetal ou machines virtuelles

Objectif

Les ports Trunk permettent à une seule instance (bare metal ou machine virtuelle) d'envoyer et de recevoir du trafic sur plusieurs réseaux Neutron via du vlan tagging, à travers une seule interface physique ou un bond LACP.

Ce guide explique comment configurer les ports Trunk Neutron dans OPCP pour activer la connectivité multi-réseau (vlan) sur un nœud bare metal ou une machine virtuelle.

Ce guide montre également comment configurer des **sous-interfaces vlan** au sein de votre instance pour accéder à chaque réseau rattaché au trunk.

> **Warning**
> La création de trunk nécessite le rôle **admin**. Un utilisateur projet ne peut pas créer de trunks.
> L'ajout de sous-ports à un trunk nécessite également les droits admin par défaut, mais cela peut être délégué par votre administrateur.
>
> Il est recommandé de configurer le trunk **avant** le déploiement d'une instance.
> Ce guide **ne couvre pas** la configuration d'un trunk sur une instance déjà en production.

Pourquoi utiliser les ports Trunk ?

Les ports Trunk peuvent être utilisés dans trois cas d'usage précis :

- **Accès multi-réseau depuis une seule instance :** Les ports Trunk permettent à un serveur bare metal ou une machine virtuelle de communiquer sur plusieurs réseaux Neutron isolés via du vlan tagging, sans nécessiter de ports séparés pour chaque réseau.
- **Dépasser la limite d'interfaces physiques sur bare metal :** Sur un serveur bare metal, le nombre de réseaux Neutron est normalement limité par le nombre d'interfaces réseau physiques. Grâce aux ports Trunk, vous pouvez connecter plus de réseaux que d'interfaces physiques disponibles en multiplexant plusieurs vlans sur une seule interface ou un bond LACP.
- **Gestion réseau simplifiée :** Au lieu de provisionner plusieurs ports et de les attacher individuellement, vous créez un seul trunk avec des sous-ports, chacun taggé avec un vlan ID spécifique. Cela permet de garder la topologie réseau claire et maintenable.

Prérequis

Avant de commencer, assurez-vous de disposer des éléments suivants :

- Un service OPCP actif.

- Un accès **OpenStack CLI configuré** avec les droits nécessaires (`clouds.yaml` ou variables d'environnement).
- Le rôle **admin** (nécessaire pour la création de trunks et la gestion des sous-ports).
- Au moins **deux réseaux Neutron** déjà créés dans votre projet (un pour le port parent et un ou plusieurs pour les sous-ports).
- Un nœud bare metal ou un projet de machine virtuelle disponible.

La configuration des ports Trunk est une fonctionnalité réseau avancée nécessitant une bonne connaissance des concepts réseau OpenStack Neutron, du vlan tagging et de la CLI OpenStack.

En pratique

Configuration réseau et Trunk

1. Identifier vos réseaux

Avant de créer le trunk, identifiez les réseaux auxquels votre instance doit accéder. Listez les réseaux disponibles dans votre projet :

```
openstack network list
```

Exemple de sortie :

```
+--------------------------------------+--------------------+--------------------------------------+
| ID                                   | Name               | Subnets                              |
+--------------------------------------+--------------------+--------------------------------------+
| 3fa85f64-5717-4562-b3fc-2c963f66afa6 | primary-network    | a1b2c3d4-e5f6-7890-abcd-ef1234567890 |
| 7c9e6679-7425-40de-944b-e07fc1f90ae7 | network-1          | b2c3d4e5-f6a7-8901-bcde-f12345678901 |
| 9b1deb4d-3b7d-4bad-9bdd-2b0d7b3dcb6d | network-2          | c3d4e5f6-a7b8-9012-cdef-123456789012 |
+--------------------------------------+--------------------+--------------------------------------+
```

2. Créer le port parent

Créez un port Neutron qui servira de **port parent** du trunk. Ce port est requis par le modèle trunk de Neutron pour ancrer le trunk à l'instance.

```
openstack port create --network <nom-reseau> <nom-port-parent>
```

Exemple :

```
openstack port create --network primary-network primary-port
```

Exemple de sortie :

```
+-------------------------+-------------------------------------+
| Field                   | Value                               |
+-------------------------+-------------------------------------+
| id                      | f47ac10b-58cc-4372-a567-0e02b2c3d479 |
| mac_address             | fa:16:3e:aa:bb:cc                   |
| name                    | primary-port                        |
| network_id              | 3fa85f64-5717-4562-b3fc-2c963f66afa6 |
| status                  | DOWN                                |
+-------------------------+-------------------------------------+
```

> **Warning**
>
> Sur les **instances bare metal**, le port parent est un **port factice**. Il existe dans la base de données Neutron mais n'a **aucun effet sur le fabric réseau**. Le réseau assigné au port parent ne transportera **aucun** trafic vers l'instance. Toute la connectivité réseau réelle doit être configurée via les **sous-ports** (voir étapes 4 et 5).
>
> Sur les **machines virtuelles**, le port parent transporte le réseau parent en trafic **non taggé** sur l'interface de base. Les réseaux des sous-ports sont délivrés en trafic vlan taggé.

3. Créer le Trunk

Créez un trunk Neutron en utilisant le port parent créé à l'étape précédente :

```
openstack network trunk create --parent-port <nom-port-parent> <nom-trunk>
```

Exemple :

```
openstack network trunk create --parent-port primary-port my-trunk
```

Exemple de sortie :

```
+----------------+--------------------------------------+
| Field          | Value                                |
+----------------+--------------------------------------+
| id             | 550e8400-e29b-41d4-a716-446655440000 |
| name           | my-trunk                             |
| parent_port_id | f47ac10b-58cc-4372-a567-0e02b2c3d479 |
| status         | DOWN                                 |
| sub_ports      |                                      |
+----------------+--------------------------------------+
```

> **Info**
>
> À ce stade, le trunk existe mais n'est attaché à aucun serveur. Le port parent est un port Neutron standard qui sera référencé lors de la création de l'instance.

4. Créer un sous-port

Créez un port Neutron sur chaque réseau que vous souhaitez rendre accessible via le trunk :

```
openstack port create --network <nom-reseau> <nom-sous-port>
```

Exemple :

```
openstack port create --network network-1 sub-port-1
```

5. Ajouter le sous-port au Trunk

Attachez le sous-port au trunk en spécifiant le type de segmentation (`vlan`) et l'identifiant de segmentation correspondant au tag vlan du réseau :

```
openstack network trunk set \
  --subport port=`<nom-sous-port>`,segmentation-type=vlan,segmentation-id=`<vlan-id>` \
  `<nom-trunk>`
```

Exemple :

```
openstack network trunk set \
  --subport port=sub-port-1,segmentation-type=vlan,segmentation-id=100 \
  my-trunk
```

> **Warning**
>
> Le comportement du `segmentation-id` diffère selon le type d'instance :
>
> - **Bare metal :** le `segmentation-id` **doit correspondre** à l'identifiant de segmentation du réseau assigné au sous-port. Neutron ne vérifie pas cette valeur, mais si elle ne correspond pas, le trafic n'atteindra pas l'instance.
> - **Machines virtuelles :** le `segmentation-id` peut être **n'importe quelle valeur** de votre choix. L'hyperviseur gère la traduction entre le tag vlan du sous-port et l'identifiant de segmentation réel du réseau.

> **Info**
>
> Pour ajouter d'autres réseaux, répétez les étapes 4 et 5 pour chaque réseau supplémentaire. Pour les instances bare metal, utilisez le `segmentation-id` correspondant à chaque réseau.

6. Vérifier la configuration du Trunk

Confirmez que le trunk est correctement configuré avec tous les sous-ports attendus :

```
openstack network trunk show <nom-trunk>
```

Exemple :

```
openstack network trunk show my-trunk
```

Exemple de sortie :

```
+----------------+-------------------------------------------------------------------------------------------+
| Field          | Value                                                                                     |
+----------------+-------------------------------------------------------------------------------------------+
| id             | 550e8400-e29b-41d4-a716-446655440000                                                      |
| name           | my-trunk                                                                                  |
| parent_port_id | f47ac10b-58cc-4372-a567-0e02b2c3d479                                                      |
| status         | DOWN                                                                                      |
| sub_ports      | [{"port_id": "...", "segmentation_id": 100, "segmentation_type": "vlan"}]                 |
+----------------+-------------------------------------------------------------------------------------------+
```

7. Déployer une instance avec le Trunk

Créez l'instance en référençant le **port parent**. OpenStack configurera le trunk lors du provisionnement.

```
openstack server create \
  --image <nom-image> \
  --flavor <flavor> \
  --port <nom-port-parent> \
  --key-name <nom-keypair> \
  <nom-instance>
```

Exemple bare metal :

```
openstack server create \
  --image ubuntu-22.04 \
  --flavor baremetal \
  --port primary-port \
  --key-name my-keypair \
  --availability-zone "nova::88830859-5b16-4935-8f41-d381b754cbe5" \
  my-trunk-instance
```

Exemple machine virtuelle :

```
openstack server create \
  --image ubuntu-22.04 \
  --flavor m1.large \
  --port primary-port \
  --key-name my-keypair \
  my-trunk-instance
```

> **Warning**
>
> Vous **devez** utiliser `--port` (en référençant le port parent) plutôt que `--nic net-id=...`. Utiliser `--nic` créerait un nouveau port et contournerait entièrement la configuration trunk.

Résumé des étapes

Étape	Action	Commande
1	Lister les réseaux	`openstack network list`
2	Créer le port parent	`openstack port create --network <nom-reseau> <nom-port-parent>`
3	Créer le trunk	`openstack network trunk create --parent-port <nom-port-parent> <nom-trunk>`
4	Créer un sous-port	`openstack port create --network <nom-reseau> <nom-sous-port>`
5	Ajouter le sous-port au trunk	`openstack network trunk set --subport port=<nom-sous-port>,segmentation-type=vlan,segmentation-id=<vlan-id> <nom-trunk>`
6	Vérifier le trunk	`openstack network trunk show <nom-trunk>`
7	Déployer l'instance	`openstack server create --port <nom-port-parent> --flavor <flavor> ...`

Configuration du système d'exploitation de l'instance

Après le déploiement de votre instance, vous devez configurer des **sous-interfaces vlan** dans l'OS invité pour accéder à chaque réseau rattaché via les sous-ports du trunk.

> **Warning**
>
> La configuration automatique du trunk via cloud-init n'est **pas possible**. OpenStack ne transmet pas les métadonnées trunk au userdata de l'instance. Vous devez configurer les sous-interfaces vlan manuellement ou via un outil de provisionnement post-déploiement.

> **Warning**
>
> Sur les **instances bare metal**, le port parent étant un port factice sans effet sur le fabric réseau, l'interface réseau de base n'aura **aucune** connectivité réseau par défaut. Tous les réseaux doivent être accessibles via des sous-interfaces vlan correspondant au `segmentation-id` assigné à chaque sous-port.
>
> Sur les **machines virtuelles**, l'interface de base transporte le réseau parent en trafic non taggé. Seuls les réseaux des sous-ports nécessitent des sous-interfaces vlan.

1. Identifier l'interface réseau principale

Connectez-vous à votre instance et identifiez l'interface réseau principale :

```
ip link show
```

Repérez l'interface principale (par exemple `ens3`, `ens21f0np0`, ou `bond0` si LACP est configuré). C'est l'interface physique qui transporte le trunk.

2. Créer des sous-interfaces vlan (temporaire)

Pour chaque sous-port, créez une sous-interface vlan correspondant au `segmentation-id` que vous avez assigné. Il s'agit d'une méthode non persistante pour les tests :

```
sudo ip link add link <interface-principale> name <interface-principale>.<vlan-id> type vlan id <vlan-id>
sudo ip link set <interface-principale>.<vlan-id> up
sudo ip addr add <adresse-ip>/<cidr> dev <interface-principale>.<vlan-id>
```

Exemple :

```
sudo ip link add link ens3 name ens3.100 type vlan id 100
sudo ip link set ens3.100 up
sudo ip addr add 192.168.1.10/24 dev ens3.100
```

> **Warning**
> Cette configuration ne survivra pas à un redémarrage. Consultez l'étape suivante pour une configuration persistante.

3. Configuration persistante (exemple Netplan)

Pour une configuration persistante des sous-interfaces vlan avec Netplan (Ubuntu/Debian avec cloud-init), créez un fichier de configuration (par exemple `/etc/netplan/60-vlans.yaml`) :

```
network:
  version: 2
  vlans:
    ens3.100:
      id: 100
      link: ens3
      addresses:
        - 192.168.1.10/24
    ens3.200:
      id: 200
      link: ens3
      addresses:
        - 10.0.0.10/24
```

Puis appliquez la configuration :

```
sudo netplan apply
```

> **Info**
> Si votre instance utilise le bonding LACP (voir le guide LACP), remplacez `ens3` par le nom de votre interface bond (par exemple `bond0`). Les sous-interfaces vlan deviennent alors `bond0.100`, `bond0.200`, etc.

4. Vérifier la connectivité

Vérifiez que vos sous-interfaces vlan sont actives et possèdent les bonnes adresses IP :

```
ip addr show <interface-principale>.<vlan-id>
```

Puis testez la connectivité :

```
ping <passerelle-ou-pair-sur-vlan>
```

Exemple :

```
ip addr show ens3.100
ping 192.168.1.1
```

> **Tip**
> Si le ping réussit, votre sous-interface vlan est correctement configurée et le trunk transporte le trafic du réseau correspondant.

Conclusion

Vous avez configuré avec succès :

- Les **ports Trunk Neutron** au niveau OpenStack, connectant une instance à plusieurs réseaux via du vlan tagging ;
- Les **sous-interfaces vlan** dans l'OS invité pour accéder à chaque réseau rattaché via les sous-ports du trunk ;
- Et vérifié la **connectivité réseau** sur chaque vlan.

Votre instance peut désormais communiquer sur plusieurs réseaux isolés grâce à une seule configuration trunk.

Aller plus loin

Si vous avez besoin d'une formation ou d'une assistance technique pour la mise en œuvre de nos solutions, contactez votre commercial ou cliquez sur ce lien pour obtenir un devis et demander une analyse personnalisée de votre projet à nos experts du service Professional Services.

Échangez avec notre communauté d'utilisateurs.

Comment configurer un RAID logiciel sur un nœud

Apprenez à configurer et gérer un RAID logiciel sur un nœud OpenStack Ironic dans OPCP

Objectif

Ce guide vous explique comment configurer et gérer un **RAID logiciel** (RAID software) sur un nœud OpenStack Ironic dans votre environnement OPCP.

Le RAID logiciel permet de créer une configuration de redondance au niveau logiciel, sans nécessiter de contrôleur RAID matériel dédié. Cette solution est particulièrement utile pour améliorer la disponibilité et les performances de stockage de vos instances.

Ce guide couvre :

- La configuration du RAID logiciel via l'interface agent d'Ironic
- La vérification de la configuration RAID
- Les bonnes pratiques pour la gestion du RAID logiciel

> **Warning**
> La configuration du RAID logiciel doit être effectuée **avant** le déploiement d'une instance sur le nœud. Une fois une instance déployée, la modification de la configuration RAID nécessite la suppression de l'instance et la reconfiguration du nœud. La création ou la modification d'un RAID efface les données présentes sur les disques utilisés.
>
> La supervision du RAID logiciel est à la charge du client final, nous n'avons pas d'accès à l'instance déployée pour gérer le monitoring. Il est recommandé de mettre en place un système d'alerte (par exemple en utilisant la commande `mdadm --monitor`) et d'intégrer cette supervision à vos outils de monitoring existants.

Prérequis

Avant de commencer, assurez-vous de disposer des éléments suivants :

- Un service OPCP actif.
- Un accès **OpenStack CLI configuré** avec les droits nécessaires (`clouds.yaml` ou variables d'environnement).
- Le rôle **admin** et/ou les nœuds transférés dans votre projet.
- Un nœud disponible (statut `available`) ou en mode maintenance.
- Une image système **Linux** (type GNU/Linux) est requise pour l'instance. Cette image doit inclure le paquet `mdadm` ou permettre son installation. Les appliances VMware et les systèmes d'exploitation Windows, par exemple, ne sont pas compatibles avec cette procédure.
- Connaissances de base sur OpenStack Ironic et la gestion des nœuds baremetal.

Pourquoi utiliser un RAID logiciel ?

Le RAID logiciel offre plusieurs avantages dans un environnement OPCP :

- **Redondance des données** : Protection contre la perte de données en cas de défaillance d'un disque.
- **Amélioration des performances** : Répartition des opérations de lecture/écriture sur plusieurs disques.
- **Coût réduit** : Pas besoin de contrôleur RAID matériel dédié.

En pratique

1. Vérifier les disques disponibles sur le nœud

Avant de configurer le RAID, vous devez identifier les disques disponibles sur votre nœud.

Lister les nœuds disponibles :

```
openstack baremetal node list
```

Vérifier les propriétés hardware du nœud :

```
openstack baremetal node show <node-id>
```

2. Activer le mode maintenance

Avant toute modification de configuration, placez le nœud en **mode maintenance** :

```
openstack baremetal node maintenance set <node-id> --reason "Configuration RAID logiciel"
```

3. Niveaux de RAID supportés

Ironic supporte plusieurs niveaux de RAID logiciel. Les valeurs suivantes sont acceptées dans la configuration JSON :

Niveau RAID	Valeur JSON	Description	Nombre minimum de disques
RAID 1	`"1"`	Mirroring (miroir)	2

> **Warning**
>
> **Contrainte importante** : Le premier disque logique avec `is_root_volume: true` **doit obligatoirement être en RAID 1**. Les autres niveaux RAID (RAID 0, 5, 6, 10, etc.) ne sont pas autorisés pour le volume racine.

Seul le RAID 1 est donc utilisable pour le déploiement de l'instance.

4. Configurer le RAID logiciel

Ironic permet de configurer le RAID logiciel via l'interface **agent**. Cette configuration est appliquée automatiquement lors du déploiement d'une instance.

4.1. Vérifier et activer l'interface agent si nécessaire

Avant de configurer le RAID, vérifiez l'interface RAID actuellement configurée sur le nœud :

```
openstack baremetal node show `<node-id>` -f json | jq '.raid_interface'
```

Si la sortie est `null` ou différente de `"agent"`, activez l'interface agent pour le RAID :

```
openstack baremetal node set <node-id> --raid-interface=agent
```

4.2. Créer le fichier de configuration RAID

Créez un fichier JSON contenant la configuration RAID souhaitée. Voici un exemple pour créer un RAID 1 (mirroring) avec deux disques :

```
cat > /tmp/raid1.json <<EOF {
  "logical_disks": [{
    "controller": "software",
    "size_gb": "MAX",
    "raid_level": "1",
    "is_root_volume": true,
    "physical_disks": [
      {"size": "<1000"},
      {"size": "<1000"}
    ]
  }]
}
EOF
```

> **Info**
>
> Le paramètre `"size": "<1000"` dans `physical_disks` permet de sélectionner automatiquement des disques de moins de 1000 Go. Vous pouvez ajuster cette valeur selon la taille de vos disques ou utiliser d'autres critères de sélection.

4.3. Appliquer la configuration RAID

Une fois le fichier de configuration créé, appliquez-le au nœud :

```
openstack baremetal node set `<node-id>` --target-raid-config /tmp/raid1.json
```

4.4. Vérifier la configuration RAID

Pour vérifier la configuration RAID appliquée sur un nœud :

```
openstack baremetal node show `<node-id>` -f json | jq '.target_raid_config'
```

5. Désactiver le mode maintenance

Une fois la configuration RAID terminée, désactivez le mode maintenance :

```
openstack baremetal node maintenance unset `<node-id>`
```

6. Déployer une instance sur le nœud configuré

Une fois le RAID configuré, vous pouvez déployer une instance sur le nœud :

```
openstack server create \
  --image `<image-name>` \
  --flavor `<flavor-id>` \
  --key-name `<keypair-name>` \
  --nic net-id=`<network-id>` \
  --availability-zone "nova::`<node-id>`" \
  `<instance-name>`
```

> **Tip**
> Pour vous assurer que votre instance est déployée sur le nœud configuré avec RAID, utilisez la zone de disponibilité `nova::<node-id>`.

7. Vérifier la configuration RAID après déploiement

Une fois l'instance déployée, vous pouvez vérifier la configuration RAID depuis l'instance :

Vérifier les périphériques RAID :

```
cat /proc/mdstat
```

Exemple de sortie :

```
Personalities : [raid1] [raid10] [linear] [multipath] [raid0] [raid6] [raid5] [raid4]
md0 : active raid1 sda[0] sdb[1]
      500G 0 blocks super 1.2 [2/2] [UU]

unused devices: <none>
```

Vérifier les détails d'un périphérique RAID :

```
mdadm --detail /dev/md0
```

Désactivation du RAID

Dans certains cas, vous pouvez avoir besoin de désactiver le RAID sur un nœud. Pour cela, définissez l'interface RAID sur `no-raid` et effacez la configuration RAID cible.

1. Désactiver l'interface RAID

Définissez l'interface RAID sur `no-raid` :

```
openstack baremetal node set `<node-id>` --raid-interface=no-raid
```

2. Effacer la configuration RAID cible

Effacez la configuration RAID cible :

```
openstack baremetal node set `<node-id>` --target-raid-config "{}"
```

> **Warning**
> La désactivation du RAID effacera toutes les données présentes sur les disques utilisés pour la configuration RAID. Assurez-vous d'avoir sauvegardé toutes les données importantes avant de procéder.

Résumé des commandes principales

Action	Commande
Lister les nœuds	`openstack baremetal node list`
Activer le mode maintenance	`openstack baremetal node maintenance set <node-id>`
Vérifier l'interface RAID	`openstack baremetal node show <node-id> -f json | jq '.raid_interface'`
Activer l'interface agent RAID	`openstack baremetal node set <node-id> --raid-interface=agent`
Appliquer la configuration RAID	`openstack baremetal node set <node-id> --target-raid-config /tmp/raid1.json`
Vérifier la configuration RAID	`openstack baremetal node show <node-id> -f json | jq '.target_raid_config'`
Désactiver le mode maintenance	`openstack baremetal node maintenance unset <node-id>`
Déployer une instance	`openstack server create --image <image-name> --flavor <flavor-id> --key-name <keypair-name> --nic net-id=<network-id> --availability-zone "nova::<node-id>" <instance-name>`
Vérifier l'état RAID (depuis l'instance)	`cat /proc/mdstat`
Désactiver l'interface RAID	`openstack baremetal node set <node-id> --raid-interface=no-raid`
Effacer la configuration RAID cible	`openstack baremetal node set <node-id> --target-raid-config "{}"`

Bonnes pratiques

- **Toujours configurer le RAID avant le déploiement** : La configuration doit être effectuée sur un nœud sans instance active.
- **Utiliser le mode maintenance** : Placez toujours le nœud en maintenance avant toute modification.
- **Sauvegardes régulières** : Le RAID n'est pas une solution de sauvegarde, effectuez des sauvegardes régulières de vos données.
- **Supervision du RAID** : Mettez en place un monitoring du RAID (par exemple via `mdadm --monitor`) et intégrez les alertes à vos outils de supervision pour détecter rapidement toute dégradation ou panne de disque.

Limitations et considérations

- **Performance** : Le RAID logiciel peut avoir un impact sur les performances CPU par rapport au RAID matériel.
- **Compatibilité** : Tous les systèmes d'exploitation ne supportent pas tous les niveaux de RAID logiciel.
- **Reconstruction** : La reconstruction d'un RAID après remplacement d'un disque peut prendre du temps et consommer des ressources.

Dépannage

Erreur	Cause	Solution
`Driver redfish does not support raid (disabled or not implemented). (HTTP 404).`	L'interface RAID n'est pas configurée sur `agent`.	Voir la section 4.1 - Vérifier et activer l'interface agent si nécessaire.
`Software RAID Configuration requires RAID-1 for the first logical disk.`	Le premier disque logique avec `is_root_volume: true` doit être en RAID 1.	Utiliser RAID 1 pour le volume racine. Voir la section 3 - Niveaux de RAID supportés.

Références

- OpenStack Ironic Documentation - RAID Configuration
- mdadm Documentation
- Cycle de vie d'un nœud

Aller plus loin

Si vous avez besoin d'une formation ou d'une assistance technique pour la mise en œuvre de nos solutions, contactez votre commercial ou cliquez sur ce lien pour demander un devis et faire analyser votre projet par nos experts de l'équipe Professional Services.

Échangez avec notre communauté d'utilisateurs.

Comment visualiser l'inventaire des nœuds

Découvrez comment consulter l'inventaire des nœuds OPCP

Objectif

Ce guide explique comment consulter l’inventaire matériel, les ressources consommées et les ressources disponibles sur une plateforme **OVHcloud On-Prem Cloud Platform (OPCP)**, à l’aide de :

- **Grafana** : monitoring en temps réel de l’infrastructure
- **NetBox** : inventaire physique et logique
- **OpenStack Horizon** : inventaire Ironic via Horizon
- **OpenStack CLI** : inventaire Ironic via CLI

> **Info**
>
> Dans OpenStack, un nœud correspond à un serveur physique du rack OPCP.
>
> Dans ce guide, le terme **nœud** est donc utilisé pour désigner un serveur physique.

Prérequis

- Être administrateur de l'infrastructure OPCP et avoir accès à l'interface d'administration `admin.dashboard`.
- Un accès **OpenStack CLI configuré** avec les droits nécessaires (`clouds.yaml` ou variables d’environnement).

En pratique

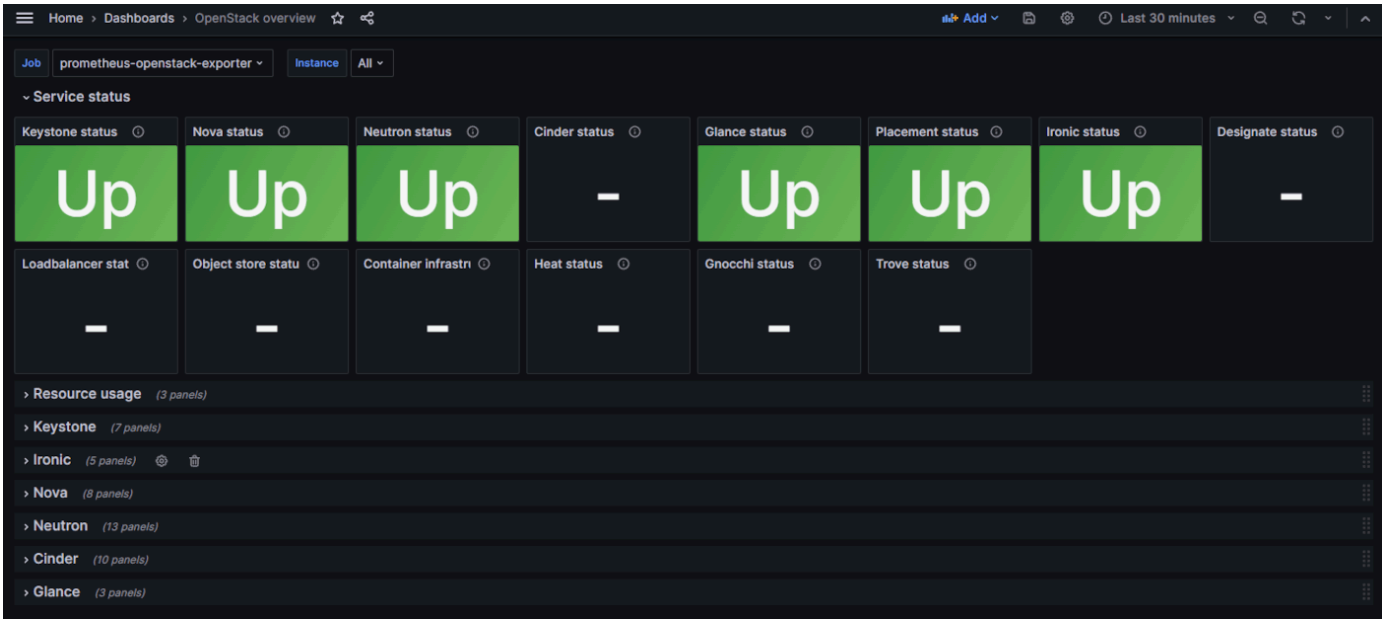
1. Inventaire et monitoring via Grafana

Grafana fournit une vue de monitoring en temps réel et permet de suivre facilement l'évolution de votre parc OPCP.

1.1 Accéder au dashboard Home

1. Connectez-vous à l’URL d'administration `admin.dashboard`.
2. Cliquez sur **Grafana**.
3. Cliquez sur **Dashboards** et recherchez le dashboard **OpenStack overview**.

Vous devriez voir un dashboard similaire à l'image suivante :



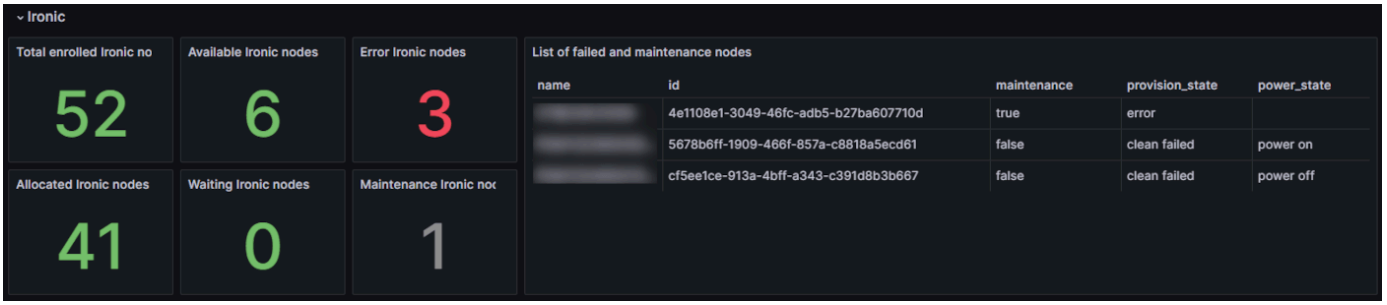
1.2 Comprendre le dashboard OpenStack overview

Le dashboard permet de visualiser :

- L'état des services OpenStack : **Service status**
- L'utilisation des ressources : **Resource usage**
- Une vue détaillée du service Keystone : **Keystone**
- Une vue détaillée du service Ironic : **Ironic**
- Une vue détaillée du service Nova : **Nova**
- Une vue détaillée du service Neutron : **Neutron**
- Une vue détaillée du service Cinder : **Cinder**
- Une vue détaillée du service Glance : **Glance**

1.3 Comprendre et utiliser la section Ironic

Nous allons nous focaliser sur la partie « Ironic », c'est ici que nous allons pouvoir voir l'état des différents nœuds.



Cette section permet de visualiser :

- Le nombre total de nœuds : **Total enrolled Ironic nodes**
- Le nombre de nœuds disponibles et pas en maintenance : **Available Ironic nodes**
- Le nombre de nœuds en erreur : **Error Ironic nodes**
- La liste des nœuds en erreur ou en maintenance : **List of failed and maintenance nodes**
- Le nombre de nœuds utilisés : **Allocated Ironic nodes**

- Le nombre de nœuds en attente : **Waiting Ironic nodes**
- Le nombre de nœuds en maintenance : **Maintenance Ironic nodes**

2. Inventaire matériel et logique avec NetBox

NetBox est la référence pour gérer l'inventaire physique, réseau et logique de votre infrastructure.

L'inventaire de NetBox est automatiquement renseigné, aucune action manuelle n'est requise pour l'ajout ou la modification des nœuds.

2.1 Accéder à NetBox

1. Connectez-vous à l'URL d'administration `admin.dashboard`.
2. Cliquez sur **NetBox**.

2.2 Consulter l'inventaire des nœuds

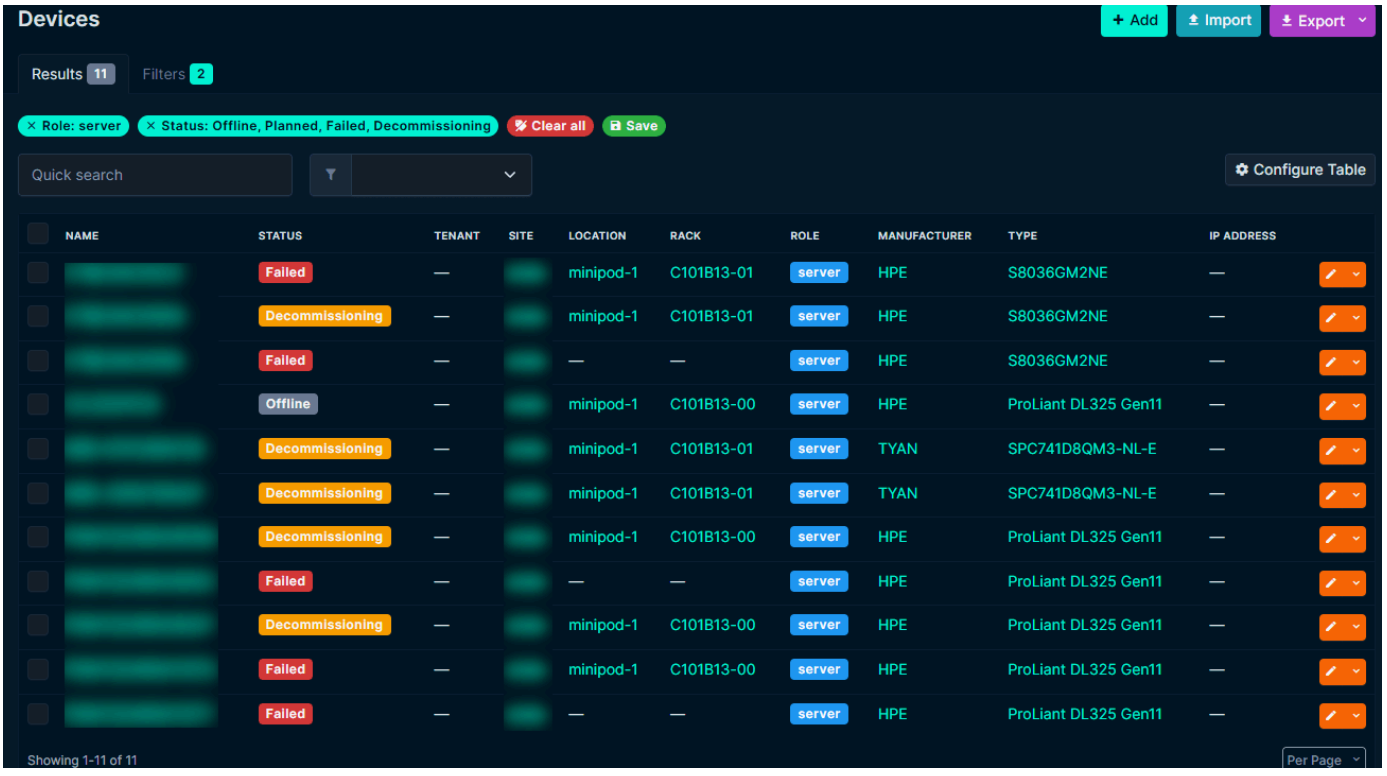
Dans **Appareils**, vous pouvez voir la liste complète des équipements de votre rack OPCP (nœuds, équipements réseau, etc.).

Vous pouvez :

- utiliser la recherche rapide pour trouver un élément par nom ;
- utiliser **Filtres** pour une recherche avancée.

Par exemple, vous pouvez lister tous les nœuds hors production en appliquant les filtres suivants :

- **Statut** : Démonté, Inventaire, Échoué, Planifié
- **Rôle** : server



Vous pouvez enregistrer la recherche pour la réutiliser plus tard.

3. OpenStack Horizon – Ironic UI

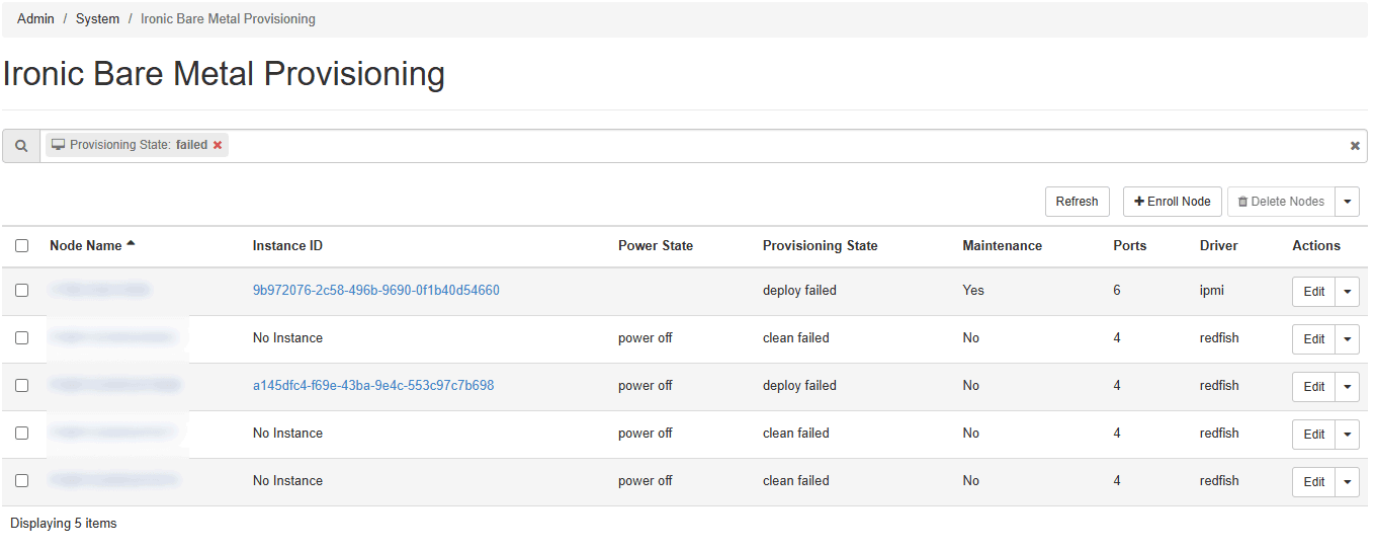
3.1 Accéder à l'interface Ironic

1. Connectez-vous à l'URL d'administration `admin.dashboard`
2. Cliquez sur **Horizon**
3. Cliquez sur **Admin**
4. Cliquez sur **System**, puis sur **Ironic Bare Metal Provisioning**

3.2 Lister tous les nœuds

Dans **Ironic Bare Metal Provisioning**, vous pouvez voir tous les nœuds.

Vous pouvez filtrer les nœuds selon leur statut, par exemple : `Provisioning State = failed`.



4. État des ressources via OpenStack CLI

Le CLI OpenStack permet d'obtenir l'état **en temps réel** des ressources allouées et disponibles.

4.1 Lister tous les nœuds

```
openstack baremetal node list
```

Exemple de sortie :

```
+--------------------------------------+---------------+--------------------------------------+-------------+-------
-------------+-------------+
| UUID                                 | Name          | Instance UUID                        | Power State |
Provisioning State | Maintenance |
+--------------------------------------+---------------+--------------------------------------+-------------+-------
-------------+-------------+
| 88830859-5b16-4935-8f41-d381b754cbe5 | SERVER-1      | None                                 | power off   |
available          | False       |
| 726bb7e9-3b20-4a44-99d7-2af747983781 | SERVER-2      | None                                 | power off   |
available          | False       |
| af711edf-a579-491e-b474-3c03639ed99b | SERVER-3      | 83b7083b-bbdd-4327-941c-ee91197818c5 | power on    | active
| False       |
+--------------------------------------+---------------+--------------------------------------+-------------+-------
-------------+-------------+
```

4.2 Filtrer les nœuds

Vous pouvez filtrer les résultats à l'aide de **--provision-state** pour voir les nœuds dans un statut spécifique, ou encore **--maintenance** pour voir les nœuds en maintenance.

Tous les filtres sont disponibles via `--help`.

Exemple de sortie :

```
$ openstack baremetal node list --provision-state "clean failed"
+--------------------------------------+---------------+--------------+-------------+--------------------+---------
----+
| UUID                                 | Name          | Instance UUID | Power State | Provisioning State |
Maintenance |
+--------------------------------------+---------------+--------------+-------------+--------------------+---------
----+
| 37f96160-16cf-47e6-8bee-0ee42a59fafe | SERVER-1      | None          | power off   | clean failed       | False
|
| 5678b6ff-1909-466f-857a-c8818a5ecd61 | SERVER-2      | None          | power off   | clean failed       | False
|
| cf5ee1ce-913a-4bff-a343-c391d8b3b667 | SERVER-3      | None          | power off   | clean failed       | False
|
+--------------------------------------+---------------+--------------+-------------+--------------------+---------
----+
```

```
$ openstack baremetal node list --maintenance
+--------------------------------------+-------------+--------------------------------------+-------------+---------
-----------+-------------+
| UUID                                 | Name        | Instance UUID                        | Power State |
Provisioning State | Maintenance |
+--------------------------------------+-------------+--------------------------------------+-------------+---------
-----------+-------------+
| 4e1108e1-3049-46fc-adb5-b27ba607710d | SERVER-4    | 9b972076-2c58-496b-9690-0f1b40d54660 | None        | deploy
failed     | True        |
+--------------------------------------+-------------+--------------------------------------+-------------+---------
-----------+-------------+
```

4.3 Détails d'un nœud

Pour récupérer toutes les informations concernant un nœud, par exemple pour identifier la dernière erreur et comprendre pourquoi le déploiement a échoué :

```
openstack baremetal node show `<node-id>`
```

Aller plus loin

Si vous avez besoin d'une formation ou d'une assistance technique pour la mise en oeuvre de nos solutions, contactez votre commercial ou cliquez sur ce lien pour obtenir un devis et demander une analyse personnalisée de votre projet à nos experts de l’équipe Professional Services.

Échangez avec notre communauté d’utilisateurs.

Gérer l'élévation de rack dans NetBox

Découvrez comment l'élévation de rack est gérée dans NetBox pour votre infrastructure OPCP

Objectif

NetBox est l'outil de référence pour l'inventaire physique et logique de votre infrastructure **OVHcloud On-Prem Cloud Platform (OPCP)**. Parmi ses fonctionnalités, l'**élévation de rack** offre une représentation visuelle et à l'échelle d'un rack et des équipements qui y sont installés.

Ce guide explique comment l'élévation de rack est gérée dans NetBox et quelles informations elle vous apporte sur votre rack OPCP.

> **Info**
>
> L'inventaire NetBox est alimenté automatiquement depuis le plan de contrôle de l'OPCP ; les nœuds OPCP sont ajoutés et maintenus à jour sans aucune action manuelle. Leur **position** dans le rack est toutefois définie manuellement pour l'instant (voir la section 4).
>
> Dans OpenStack, un nœud correspond à un serveur physique du rack OPCP. Dans ce guide, le terme **nœud** désigne donc un serveur physique.

Prérequis

- Être administrateur de l'infrastructure OPCP et disposer d'un accès à l'interface d'administration `admin.dashboard`.
- Savoir consulter l'inventaire NetBox (voir le guide « Comment visualiser l'inventaire des nœuds »).

Instructions

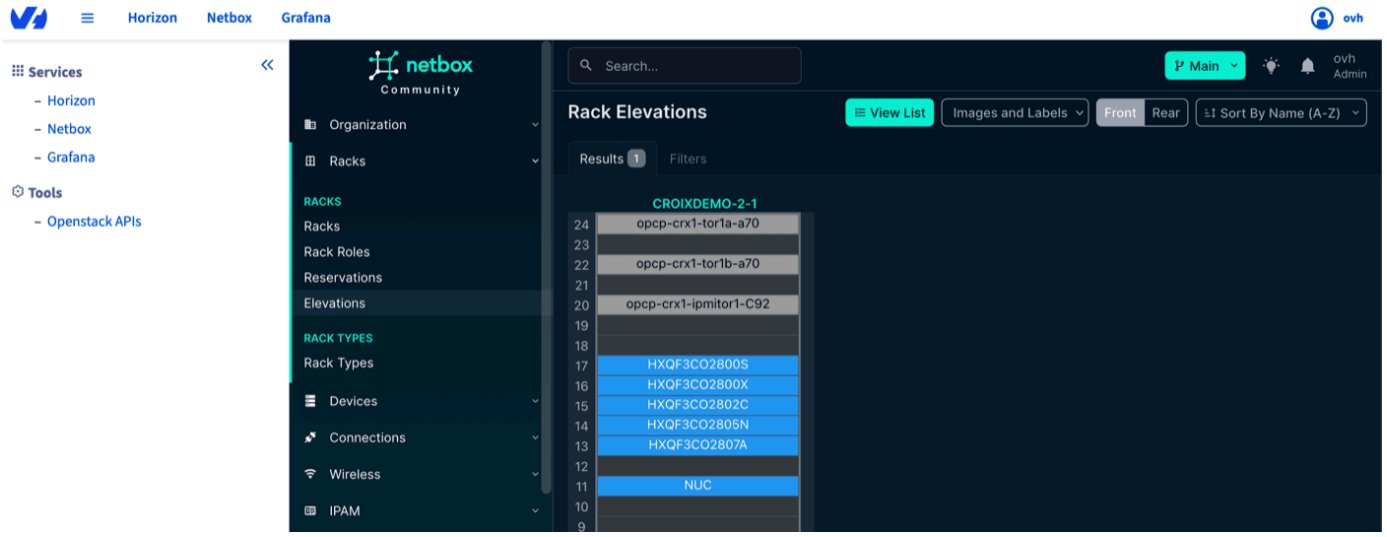
1. Qu'est-ce qu'une élévation de rack

Dans NetBox, un **rack** représente une armoire physique d'équipements située dans un datacenter. Sa capacité se mesure en **unités de rack (U)** — un rack standard fait généralement entre 42U et 48U de hauteur.

L'**élévation de rack** est un schéma à l'échelle de ce rack. Il indique, unité par unité, quel équipement occupe quelle position, ce qui vous permet de comprendre l'agencement physique de votre infrastructure OPCP sans être sur site.

2. Accéder à l'élévation de rack dans NetBox

1. Connectez-vous à l'URL d'administration `admin.dashboard`.
2. Cliquez sur **NetBox**. L'élévation de votre rack OPCP s'ouvre directement.



Chemin alternatif : si vous devez accéder à l'élévation depuis une autre partie de NetBox, ouvrez la section **DCIM**, sélectionnez **Racks**, puis sélectionnez votre rack OPCP pour ouvrir sa page de détail et accéder à la vue **Elevation**.

3. Lire la vue d'élévation

La vue d'élévation fournit plusieurs informations immédiatement visibles :

- **Faces avant et arrière** : l'élévation peut être affichée depuis l'avant ou l'arrière du rack, afin de voir les équipements montés de chaque côté.
- **Position et hauteur des équipements** : chaque équipement est représenté à l'unité de rack (U) où il est monté et s'étend sur le nombre d'unités qu'il occupe physiquement.
- **Numérotation des unités** : les unités de rack sont numérotées le long de l'élévation. Selon la configuration du rack, la numérotation peut aller du bas vers le haut ou du haut vers le bas.
- **Couleurs par rôle** : les équipements sont colorés selon leur rôle (par exemple serveurs ou équipements réseau), ce qui permet de distinguer facilement les types d'équipements.
- **Espace libre** : les unités de rack vides sont affichées comme espace disponible, ce qui donne un aperçu rapide de la capacité restante.

4. Comment les positions des nœuds sont gérées

Aujourd'hui, la position de chaque nœud dans l'élévation de rack est définie **manuellement**. Il s'agit du mode de fonctionnement standard : un administrateur définit la position d'un équipement en le modifiant dans NetBox et en lui attribuant un rack et une unité de rack (U). Une position définie de cette manière est persistante : elle n'est pas modifiée automatiquement et reste en place tant que vous ne la changez pas vous-même.

Vous pouvez également ajouter des **équipements personnalisés** (custom devices) à l'élévation de rack — des équipements qui ne sont pas gérés en tant que nœud OPCP, par exemple des appliances tierces que vous installez vous-même. Ils sont créés et positionnés manuellement dans NetBox de la même façon. Avant de placer un équipement personnalisé, contactez le support OVHcloud pour confirmer quelle(s) unité(s) de rack vous pouvez utiliser, afin de ne pas entrer en conflit avec les unités réservées à la plateforme.

> **Info**
>
> **Le positionnement automatique via des modèles arrivera prochainement.** OVHcloud travaille sur le placement automatique des nœuds à partir de leur câblage. Une fois cette fonctionnalité disponible, OVHcloud fournira des **modèles de position** (templates) qui associent un ensemble donné de raccordements de ports de switch à une unité de rack (U) précise, par **type de rack** (lequel dépend de la manière dont la plateforme a été installée). Lors de la synchronisation Ironic → NetBox, un nœud dont le câblage correspond à une entrée de modèle sera alors placé automatiquement à l'unité de rack correspondante, et cette position automatique sera prioritaire sur les modifications manuelles. Lorsqu'aucun modèle ne correspond au type de rack, le positionnement manuel restera le mécanisme de repli. Les équipements personnalisés, qui n'ont pas de câblage Ironic, continueront toujours d'être positionnés manuellement.

5. Ce que l'élévation de rack vous indique

Pour une infrastructure OPCP, l'élévation de rack vous aide à :

- **localiser physiquement un nœud** dans le rack pour des opérations sur site telles que la maintenance ou le câblage.
- **évaluer la capacité** en voyant quelles unités sont occupées et lesquelles sont libres.
- **mettre en correspondance les vues physique et logique** en recoupant un équipement affiché dans l'élévation avec son entrée d'inventaire et son statut OpenStack/Ironic.

Pour aller plus loin avec l'inventaire et les statuts des nœuds, consultez les guides ci-dessous :

- Comment visualiser l'inventaire des nœuds
- Cycle de vie d'un nœud OPCP

Références

- Documentation NetBox - Racks
- Documentation NetBox - Rack Types

Aller plus loin

Pour une formation ou une assistance technique sur la mise en œuvre de nos solutions, contactez votre commercial ou consultez la page Professional Services pour obtenir un devis et faire analyser votre projet par nos experts.

Échangez avec notre communauté d'utilisateurs.

Fonctionnement de la synchronisation Ironic vers NetBox

Découvrez comment les nœuds OPCP sont importés depuis Ironic vers NetBox et ce que la synchronisation maintient à jour

Objectif

Sur une **OVHcloud On-Prem Cloud Platform (OPCP)**, l'inventaire physique et logique affiché dans NetBox n'est pas maintenu manuellement : il est alimenté automatiquement depuis OpenStack Ironic, qui fait référence pour les nœuds bare-metal de votre plateforme.

Ce guide explique le fonctionnement de la synchronisation Ironic vers NetBox et les informations qu'elle maintient à jour.

> **Info**
>
> L'inventaire NetBox est alimenté automatiquement depuis le plan de contrôle de l'OPCP ; aucune action manuelle n'est nécessaire pour ajouter, mettre à jour ou supprimer des nœuds.
>
> Dans OpenStack, un nœud correspond à un serveur physique du rack OPCP. Dans ce guide, le terme **nœud** désigne donc un serveur physique.

Prérequis

- Être administrateur de l'infrastructure OPCP et disposer d'un accès à l'interface d'administration `admin.dashboard`.
- Savoir consulter l'inventaire NetBox (voir le guide « Comment visualiser l'inventaire des nœuds »).

Instructions

1. Vue d'ensemble de la synchronisation

La synchronisation s'exécute automatiquement et réconcilie l'inventaire NetBox avec l'état courant d'Ironic. À chaque exécution, elle :

1. récupère la liste des nœuds et de leurs ports réseau depuis Ironic.
2. crée ou met à jour les **équipements** correspondants dans NetBox.
3. maintient à jour les interfaces, les adresses MAC et le câblage vers les switches.
4. calcule et met à jour la **position en rack** de chaque nœud (voir section 3).
5. met hors service les équipements qui n'existent plus dans Ironic (voir section 4).

Comme NetBox reste synchronisé avec Ironic, il reflète la même réalité que la vue OpenStack/Ironic de vos nœuds. Les statuts décrits dans le guide « Cycle de vie d'un nœud OPCP » sont donc répercutés dans NetBox.

2. Ce que la synchronisation importe

Pour chaque nœud, la synchronisation importe et maintient à jour :

- **L'identité de l'équipement** : le nom de l'équipement, son site et son rack.
- **Le type et le fabricant de l'équipement** : déduits du modèle matériel et du fournisseur du nœud remontés par Ironic.
- **Les identifiants** : l'UUID du nœud Ironic est stocké à la fois comme **asset tag** de l'équipement et dans un champ dédié `baremetal_node_id`, afin qu'un équipement NetBox puisse toujours être relié à son nœud Ironic.
- **Les informations BMC** : l'adresse MAC du BMC et l'adresse du BMC sont stockées sur l'équipement.
- **Le statut** : déduit de l'état de provisionnement (provision state) d'Ironic.
- **Les interfaces et les adresses MAC** : une interface par port Ironic, avec son adresse MAC.
- **Le câblage** : les câbles entre chaque interface de nœud et le port de switch correspondant, d'après les informations de liaison remontées par Ironic.
- **La position en rack** : l'unité de rack (U) du nœud, lorsqu'elle peut être déterminée (voir section 3).

3. Position en rack et élévation

La position en rack importée par la synchronisation est ce qui alimente la vue d'**élévation de rack** dans NetBox. La position n'est pas saisie à la main : elle est calculée à partir du câblage du nœud et des modèles de position qu'OVHcloud fournit pour votre type de rack.

La façon dont la position est déterminée, les cas où le placement automatique s'applique et ceux où le positionnement manuel est conservé comme mécanisme de repli sont expliqués en détail dans le guide « Gérer l'élévation de rack dans NetBox ».

4. Mise hors service des nœuds supprimés

Lorsqu'un nœud qui existait précédemment dans Ironic n'est plus renvoyé par Ironic, la synchronisation ne supprime pas son équipement dans NetBox. Elle réalise plutôt les actions suivantes :

- elle passe le statut de l'équipement à **Decommissioning** ;
- elle efface sa position en rack et sa face, de sorte qu'il n'apparaît plus dans l'élévation de rack ;
- elle supprime les câbles raccordés à ses interfaces.

Cela conserve une trace de l'équipement dans NetBox tout en indiquant clairement qu'il ne fait plus partie de l'inventaire actif.

Références

- Comment visualiser l'inventaire des nœuds
- Cycle de vie d'un nœud OPCP
- Gérer l'élévation de rack dans NetBox
- Documentation NetBox - Devices

Aller plus loin

Pour une formation ou une assistance technique sur la mise en œuvre de nos solutions, contactez votre commercial ou consultez la page Professional Services pour obtenir un devis et faire analyser votre projet par nos experts.

Échangez avec notre communauté d'utilisateurs.

Comment utiliser Terraform

Découvrez comment générer une Application Credential depuis Horizon et automatiser le déploiement de vos ressources OPCP avec Terraform

Objectif

Terraform est un outil open source d'**Infrastructure as Code (IaC)** développé par HashiCorp. Il permet de décrire et de provisionner votre infrastructure de manière déclarative à partir de fichiers de configuration, écrits en *HashiCorp Configuration Language* (HCL).

L'offre **OPCP** reposant sur **OpenStack**, vous pouvez utiliser le **provider Terraform OpenStack** afin d'automatiser le déploiement de vos ressources : instances, réseaux, volumes, paires de clés, etc.

> **Info**
>
> Ce guide est également valable avec **OpenTofu**, le fork open source de Terraform maintenu par la Linux Foundation. OpenTofu est compatible avec la syntaxe HCL et les providers Terraform : il vous suffit de remplacer la commande `terraform` par `tofu` dans les exemples ci-dessous.

Ce guide détaille les étapes nécessaires pour générer une *Application Credential* depuis Horizon, configurer Terraform et déployer un premier serveur sur votre infrastructure OPCP.

Prérequis

- Disposer d'un service OPCP actif.
- Posséder un **compte utilisateur** avec les droits suffisants pour se connecter à Horizon sur l'offre OPCP.
- Avoir installé Terraform (version >= 1.0) ou OpenTofu sur votre poste de travail.
- Disposer d'une **paire de clés SSH** sur votre poste local pour accéder à votre instance.
- Avoir préalablement créé un **réseau privé** dans votre projet OPCP (voir le guide « Comment installer une instance depuis l'interface Horizon »).

Sommaire

En pratique

1. Création d'une Application Credential depuis Horizon

Pour permettre à Terraform de communiquer avec votre infrastructure OPCP, il est nécessaire de générer un couple **Application Credential** (`id` / `secret`) depuis l'interface Horizon. Ce mécanisme évite l'utilisation directe de vos identifiants Keycloak et fournit une authentification dédiée à vos automatisations, avec un périmètre de droits limité au projet courant.

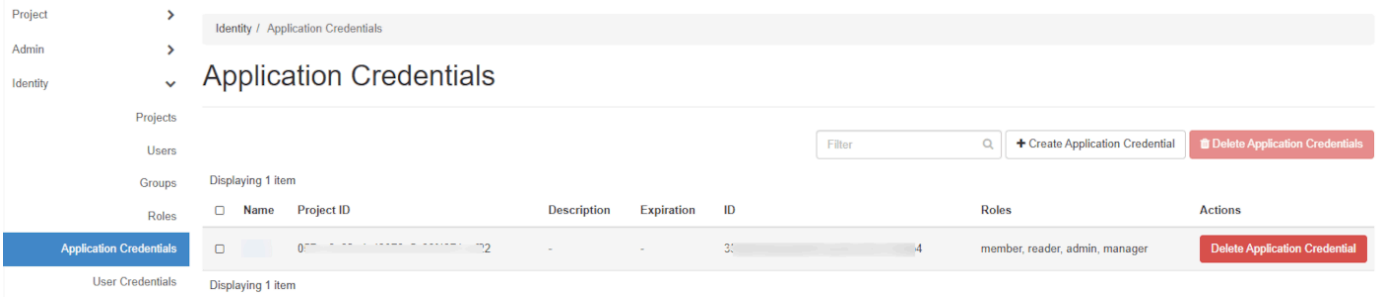
> **Warning**
>
> Une `Application Credential` est automatiquement supprimée lorsque l'utilisateur qui l'a créée est révoqué. Pour éviter toute perte d'accès dans un workflow d'automatisation, ne générez pas d'Application Credential depuis un compte utilisateur nominatif ou facilement révocable ; privilégiez un compte dédié.

Connexion à Horizon

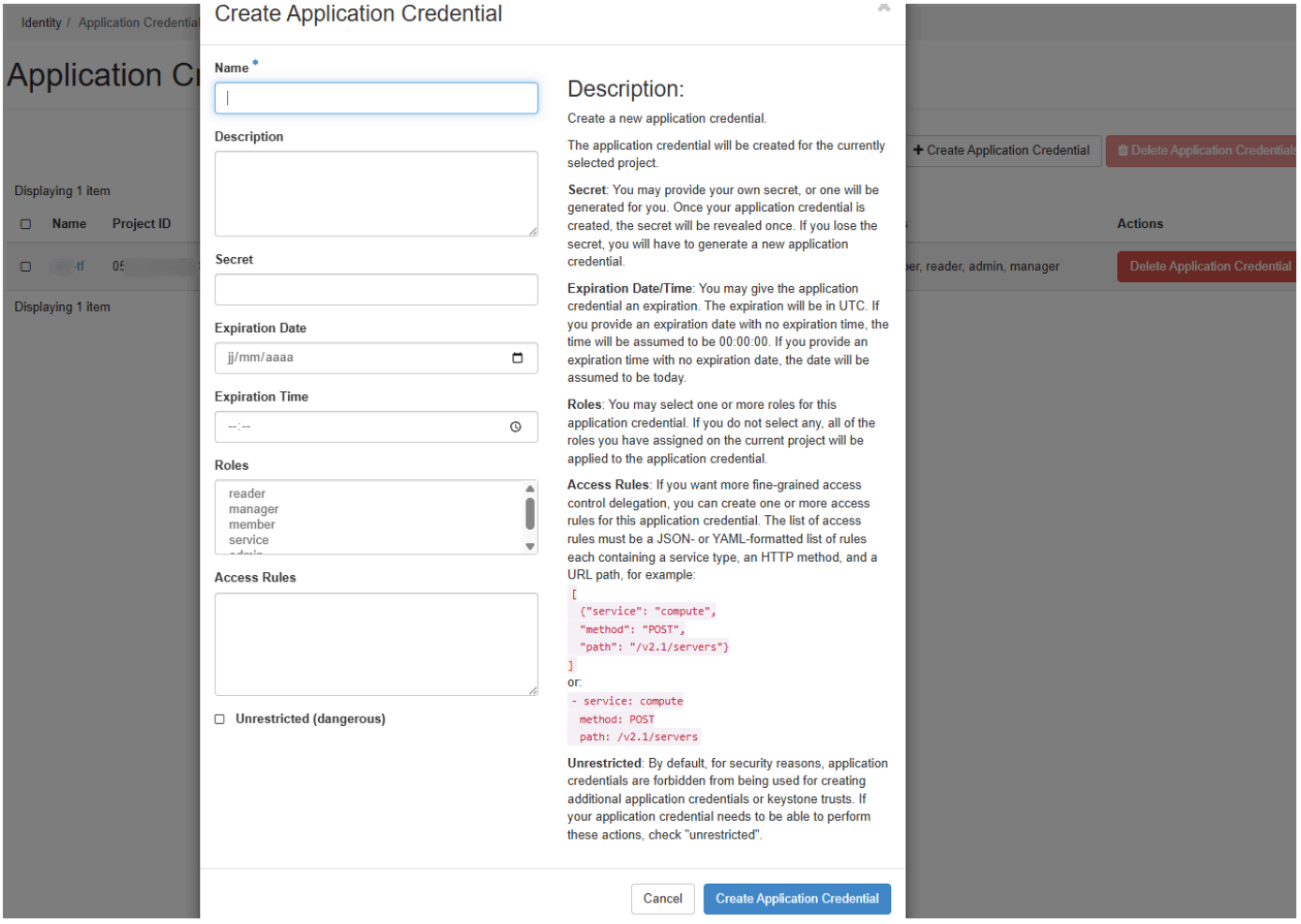
Connectez-vous à l'interface **Horizon** de votre environnement OPCP, puis sélectionnez le **projet** dans lequel vous souhaitez déployer vos ressources via Terraform. Pour plus d'informations, consultez le guide « Mise en route de votre OPCP ».

Création de l'Application Credential

Dans le menu de gauche, cliquez sur **Identity**, puis sur **Application Credentials**.



Cliquez sur **+ Create Application Credential**.

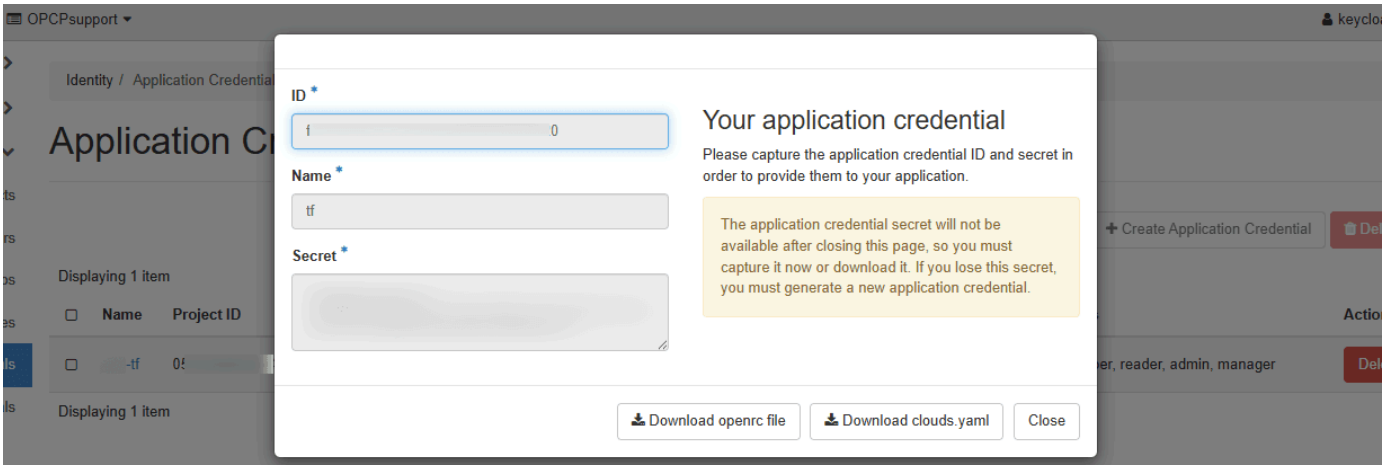


Renseignez les champs suivants :

Champ	Description
Name	Saisissez un nom pour votre Application Credential (ex. : *terraform-cred*).
Description	Optionnel. Ajoutez une description si nécessaire.
Secret	Optionnel. Si non renseigné, un secret sera généré automatiquement.
Expiration Date / Expiration Time	Optionnel. Définissez une date d'expiration.
Roles	Sélectionnez les rôles à associer à la credential (ex. : *member*, *admin*).
Access Rules	Optionnel. Permet de restreindre les actions autorisées.
Unrestricted	Laissez décoché pour limiter les actions possibles.

Cliquez sur **Create Application Credential**.

> **Warning**
>
> Une fois la fenêtre fermée, le **secret ne sera plus accessible**. Téléchargez le fichier `clouds.yaml` ou `openrc` proposé par Horizon, ou copiez les valeurs `id` et `secret` dans un emplacement sécurisé.



> **Info**
>
> Pour la suite de ce tutoriel, **téléchargez le fichier** `openrc` : il sera utilisé à l'étape suivante pour authentifier Terraform auprès de votre infrastructure OPCP.

2. Préparation de l'environnement Terraform

Création du dossier de travail

Créez un répertoire dédié à votre projet Terraform :

```
mkdir terraform-opcp && cd terraform-opcp
```

Définition du provider OpenStack

Dans un fichier nommé `provider.tf`, ajoutez les lignes suivantes :

```
terraform {
  required_version = ">= 1.0"
  required_providers {
    openstack = {
      source  = "terraform-provider-openstack/openstack"
      version = ">= 3.0.0"
    }
  }
}

provider "openstack" {

}
```

Aucun paramètre n'est nécessaire dans le bloc **provider** : le provider OpenStack récupère automatiquement les informations d'authentification depuis les variables d'environnement `OS_*`.

Chargement des variables d'environnement OpenStack

Lors de la création de votre Application Credential, Horizon vous a proposé le téléchargement d'un fichier `clouds.yaml` ou `openrc`. Le plus simple est de charger ce fichier `openrc.sh` dans votre shell avant d'exécuter Terraform :

```
source openrc.sh
```

> **Info**
>
> Le fichier `openrc.sh` exporte notamment `OS_AUTH_URL`, `OS_REGION_NAME`, `OS_APPLICATION_CREDENTIAL_ID` et `OS_APPLICATION_CREDENTIAL_SECRET`. Ces variables seront automatiquement utilisées par le provider OpenStack de Terraform.

Initialisation

Téléchargez les plugins du provider OpenStack :

```
terraform init
```

3. Création d'un serveur

Dans un fichier `main.tf`, déclarez les ressources nécessaires pour créer une instance attachée à un réseau privé existant :

```
# Variables liées à l'instance
variable "instance_name" {
  description = "Nom de l'instance qui sera créée dans OPCP"
  type        = string
  default     = "instance-terraform-name"
}

variable "image_name" {
  description = "Nom de l'image à utiliser pour l'instance (ex: Debian 12 OPCP)"
  type        = string
  default     = "Debian 12 OPCP"
}

variable "flavor_name" {
  description = "Flavor baremetal définissant les ressources matérielles de l'instance (ex: scale-1, scale-2, etc.)"
  type        = string
  default     = "scale-1"
}

variable "network_name" {
  description = "Nom du réseau existant auquel l'instance sera rattachée"
  type        = string
  default     = "<your-network-name>"
}

variable "key_name" {
  description = "Nom de la paire de clés SSH à associer à l'instance"
  type        = string
  default     = "terraform-key"
}

# Création d'une instance
resource "openstack_compute_instance_v2" "terraform_instance" {
  name        = var.instance_name
  image_name  = var.image_name
  flavor_name = var.flavor_name
  key_pair    = var.key_name

  network {
    name = var.network_name
  }

  lifecycle {
    # Évite que Terraform détecte une dérive lorsque l'image de base est mise à jour
    ignore_changes = [
      image_name
    ]
  }

  # Métadonnées utiles
  metadata = {
    environment = "production"
    managed_by  = "terraform"
  }

  # Timeouts plus longs (le provisioning baremetal est lent)
  timeouts {
    create = "60m"
    delete = "30m"
  }
}

# Sorties utiles après création
output "instance_id" {
```

```
  value = openstack_compute_instance_v2.terraform_instance.id
}

output "instance_ip" {
  value = openstack_compute_instance_v2.terraform_instance.access_ip_v4
}
```

> **Info**
>
> Les noms d'images, de flavors et de réseaux disponibles peuvent être listés depuis Horizon ou avec la CLI OpenStack (`openstack image list` , `openstack flavor list` , `openstack network list`). Pour configurer la CLI, consultez le guide « Comment utiliser les API et obtenir les informations d'identification ».

Vérification du plan

Avant tout déploiement, prévisualisez les actions qui seront effectuées :

```
terraform plan
```

Application de la configuration

Déployez l'instance avec la commande suivante :

```
terraform apply
```

Confirmez avec `yes` lorsque Terraform vous le demande. Une fois la création terminée, l'instance apparaît dans la section **Compute** > **Instances** de l'interface Horizon.

4. Configurations complémentaires sur le nœud (RAID, LACP)

Certaines configurations doivent être appliquées **sur le nœud baremetal** avant le déploiement de l'instance et ne sont pas couvertes par le provider Terraform OpenStack. Elles nécessitent des droits **admin** Ironic (ou des nœuds transférés dans votre projet) et restent à effectuer via la CLI OpenStack.

> **Info**
>
> **RAID logiciel** : pour configurer un RAID logiciel sur un nœud baremetal, consultez le guide « Comment configurer un RAID logiciel sur un nœud ». L'attribut `target_raid_config` n'est pas exposé par la ressource `openstack_baremetal_node_v1` . Cette opération est à réaliser **avant** le `terraform apply` qui déploie l'instance, en ciblant ensuite le nœud configuré via `availability_zone = "nova::<node-id>"` .

> **Info**
>
> **LACP / bonding** : pour agréger plusieurs interfaces réseau d'un nœud, consultez le guide « Comment configurer LACP sur un nœud ». La configuration des ports baremetal et du bonding n'est pas gérable de manière déclarative par le provider Terraform OpenStack. Cette opération est à réaliser **avant** le `terraform apply` qui déploie l'instance.

5. Suppression de l'infrastructure

Pour supprimer l'ensemble des ressources créées via Terraform :

```
terraform destroy
```

> **Warning**
> `terraform destroy` ne réinitialise pas les configurations RAID ou LACP appliquées sur le nœud. Pour les retirer, suivez la section dédiée du guide correspondant via la CLI OpenStack.

6. Dépannage

Problème	Cause possible	Solution
`Authentication failed`	Mauvais `application_credential_id` ou `secret`	Vérifiez les identifiants dans Horizon (**Identity** > **Application Credentials**).
`Could not find image`	Le nom de l'image ne correspond pas	Listez les images disponibles via `openstack image list` ou depuis Horizon.
`No suitable endpoint could be found`	Mauvaise URL `auth_url` ou région inexistante	Vérifiez l'URL Keystone et la région dans **Project** > **API Access** .
`Network not found`	Réseau privé absent dans le projet	Créez un réseau privé au préalable depuis Horizon (**Network** > **Networks**).

7. Références

- Documentation officielle Terraform
- Documentation officielle OpenTofu
- Provider Terraform OpenStack
- Ressource openstack_compute_instance_v2
- OpenStack Application Credentials

Aller plus loin

Pour une formation ou une assistance technique sur la mise en œuvre de nos solutions, contactez votre commercial ou consultez la page Professional Services pour obtenir un devis et faire analyser votre projet par nos experts.

Échangez avec notre communauté d'utilisateurs.

Comment mettre à jour les buckets S3 de sauvegarde

Découvrez comment changer les buckets de stockage objet S3 utilisés par les sauvegardes de la plateforme OPCP

Objectif

Votre plateforme **OVHcloud On-Prem Cloud Platform (OPCP)** stocke ses sauvegardes dans un **stockage objet compatible S3**[1] externe. Ces sauvegardes couvrent les composants critiques nécessaires à la restauration du plan de contrôle, à savoir :

- les **bases de données** internes (`db`),
- les **volumes persistants** (`pv`),
- l'**état Terraform** (`tfState`),
- les **parts de l'auto-unsealer KMS** (`kmsShares`).

Vous pouvez être amené à changer les buckets de stockage objet ou l'endpoint utilisés pour ces sauvegardes dans plusieurs situations : migration vers un nouvel endpoint ou une nouvelle région S3, changement de fournisseur de stockage objet, renommage des buckets, ou ajustement de la politique de rétention.

Ce guide explique comment mettre à jour les buckets S3 de sauvegarde dans la configuration OPCP, redéployer le plan de contrôle et réinitialiser les dépôts de sauvegarde **Velero** / **Kopia** afin qu'ils pointent vers le nouvel emplacement de stockage objet.

> **Warning**
> Cette procédure modifie la configuration de sauvegarde de la plateforme et réinitialise les dépôts de sauvegarde. Les sauvegardes existantes stockées dans les **anciens** buckets ne sont **pas** migrées automatiquement. Assurez-vous que les anciennes sauvegardes ne sont plus nécessaires (ou qu'elles ont été recopiées) avant de commencer, et réalisez cette opération pendant une fenêtre de maintenance.

Prérequis

- Disposer d'une infrastructure OPCP opérationnelle.
- Disposer d'un accès administrateur au contrôleur OPCP avec les outils `opcp-cli` , `opcp-diag` , `kubectl` , `flux` , `tfctl` et `velero` installés et configurés.
- Disposer des informations de connexion du stockage objet S3 **cible** : URL de l'endpoint, région et noms des buckets à utiliser pour les sauvegardes.
- Prévoir une fenêtre de maintenance, le plan de contrôle étant redéployé pendant l'opération.

En pratique

1. Lancer un diagnostic préalable

Avant toute modification, vérifiez que la plateforme est en bonne santé. L'outil `opcp-diag` exécute une série de contrôles de la plateforme :

```
opcp-diag run -p
```

Ne poursuivez que lorsque le diagnostic est au vert. Vous disposez ainsi d'un état de référence sain à comparer une fois l'opération terminée.

2. Mettre à jour la configuration de sauvegarde

Ouvrez la configuration OPCP dans votre éditeur :

```
opcp-cli config edit
```

Repérez la section `backups` et mettez-la à jour avec le nouvel endpoint de stockage objet et les nouveaux noms de buckets :

```
...
backups:
  endpoint:
    region: yourRegion
    url: "url.of.your.s3.service"
  db:
    enabled: true
    retention: "30d"
    bucket: "opcp01-backups-db-region"
  pv:
    enabled: true
    retention: "720h0m0s"
    bucket: "opcp01-backups-pv-region"
    tfState:
      retention: "720h0m0s"
    kmsShares:
      retention: "720h0m0s"
...
```

Les champs à vérifier sont :

- `endpoint.region` / `endpoint.url` : la région et l'URL du service S3 compatible cible.
- `db.bucket` : le bucket qui stocke les sauvegardes des bases de données, avec sa `retention` (ici `30d`).
- `pv.bucket` : le bucket qui stocke les sauvegardes des volumes persistants, avec sa `retention` (ici `720h0m0s`, soit 30 jours).
- `pv.tfState.retention` / `pv.kmsShares.retention` : la rétention appliquée aux sauvegardes de l'état Terraform et des parts KMS.

> **Info**
>
> Les valeurs de rétention peuvent être exprimées soit sous forme de durée en jours (par exemple `30d`), soit sous forme de durée Go (par exemple `720h0m0s`, ce qui équivaut également à 30 jours). Conservez le format utilisé par la configuration existante.

Enregistrez et fermez le fichier. Cette commande se contente de **mettre à jour** la configuration enregistrée ; elle ne l'**applique** pas encore. La modification est appliquée à l'étape suivante avec `opcp-cli deploy`. La CLI indique lorsque l'édition est terminée :

```
✅ Command 'edit' completed in 1m12.081606602s
```

3. Appliquer la nouvelle configuration

Appliquez la configuration mise à jour :

```
opcp-cli deploy
```

Cette commande ne redéploie **pas** l'intégralité du plan de contrôle : elle exécute uniquement la convergence nécessaire pour appliquer votre modification, c'est-à-dire la mise à jour de la configuration de sauvegarde Velero et de la configuration de sauvegarde des bases de données.

L'opération peut prendre plusieurs minutes. Attendez de voir la confirmation :

```
...
✅ Deployment of control plane terminated
✅ Command 'deploy' completed in 10m19.19421549s
```

Vous pouvez suivre le déploiement en temps réel dans un autre terminal en affichant le journal de la CLI :

```
tail -F /var/log/opcp-cli/opcp-cli.log | jq .msg
```

4. Attendre la réconciliation

La plateforme s'appuie sur **Flux** (réconciliation GitOps) et **Terraform** pour converger vers le nouvel état souhaité. Attendez que toutes les ressources soient réconciliées.

Listez les ressources qui ne sont pas encore prêtes :

```
flux get all -A --status-selector ready=false
flux get all -A --status-selector ready=unknown
```

Vérifiez l'état des réconciliations Terraform :

```
tfctl get
```

Attendez que ces commandes ne signalent plus aucune ressource en attente ou en échec. Si une ressource reste bloquée, il peut parfois être nécessaire de redémarrer les pods concernés — soyez patient, la réconciliation peut prendre un certain temps.

5. Vérifier l'emplacement de stockage des sauvegardes

Une fois la réconciliation terminée, confirmez que Velero pointe désormais vers le nouveau bucket et que l'emplacement de stockage est `Available` :

```
velero backup-location get
```

```
NAME              PROVIDER   BUCKET/PREFIX              PHASE       LAST VALIDATED                 ACCESS MODE
DEFAULT
backup-location   aws        opcp01-backups-velero-region   Available   2026-07-07 12:30:41 +0000 UTC   ReadWrite
true
```

La colonne `PHASE` doit indiquer `Available`. Après le déploiement, quelques minutes peuvent être nécessaires avant que l'emplacement de stockage soit validé et passe à `Available` : patientez un instant et relancez la commande si besoin. S'il reste dans un autre état, revérifiez la configuration de l'endpoint et du bucket à l'étape 2 ainsi que les identifiants du service S3.

6. Réinitialiser les dépôts de sauvegarde Velero

Les dépôts de sauvegarde **Kopia** utilisés par Velero sont liés à l'ancien emplacement de stockage objet. Pour que Velero les recrée sur les nouveaux buckets, vous devez supprimer les dépôts existants et redémarrer les composants Velero.

Listez d'abord les dépôts de sauvegarde actuels :

```
kubectl -n velero get backuprepositories -A
```

```
NAMESPACE   NAME                                  AGE    REPOSITORY TYPE
velero      glance-backup-location-kopia-jqj4d    215d   kopia
velero      ironic-backup-location-kopia-w46h4    215d   kopia
velero      kms-backup-location-kopia-jxdvz       215d   kopia
velero      monitoring-backup-location-kopia-hqjnt   215d   kopia
```

Supprimez-les tous :

```
kubectl -n velero delete backuprepositories --all
```

```
backuprepository.velero.io "glance-backup-location-kopia-jqj4d" deleted from velero namespace
backuprepository.velero.io "ironic-backup-location-kopia-w46h4" deleted from velero namespace
backuprepository.velero.io "kms-backup-location-kopia-jxdvz" deleted from velero namespace
backuprepository.velero.io "monitoring-backup-location-kopia-hqjnt" deleted from velero namespace
```

Confirmez qu'il ne reste plus aucun dépôt :

```
kubectl -n velero get backuprepositories -A
```

```
No resources found
```

Redémarrez le déploiement Velero et le daemonset `node-agent` afin qu'ils recréent les dépôts sur les nouveaux buckets :

```
kubectl -n velero rollout restart deployment/velero
kubectl -n velero rollout restart daemonset/node-agent
```

> **Info**
>
> Un message `Warning` peut s'afficher lors du redémarrage de ces charges de travail. C'est normal et vous pouvez l'ignorer sans risque.

Vérifiez que les pods Velero et `node-agent` ont redémarré et sont à l'état `Running` :

```
kubectl get pod -n velero
```

```
NAME                                                  READY   STATUS      RESTARTS   AGE
...
node-agent-p92tl                                      1/1     Running     0          62s
node-agent-pkhwn                                      1/1     Running     0          58s
node-agent-qk6j2                                      1/1     Running     0          60s
...
velero-5f7fd4c8b8-rqprb                               1/1     Running     0          68s
```

7. Déclencher une première sauvegarde pour chaque planification

La plateforme définit plusieurs planifications de sauvegarde. Listez-les pour confirmer qu'elles sont activées :

```
velero get schedule
```

```
NAME                            STATUS    CREATED                         SCHEDULE    BACKUP TTL   LAST BACKUP
SELECTOR                                  PAUSED
kms-autounsealer-shares-backup  Enabled   2025-12-03 13:30:21 +0000 UTC   0 * * * *   720h0m0s     34m ago
kms-autounsealer-shares-backup=enabled    false
pvs-backup                      Enabled   2025-12-03 13:30:21 +0000 UTC   0 * * * *   720h0m0s     34m ago
backup=true                               false
tfstate-backup                  Enabled   2025-12-03 13:30:21 +0000 UTC   0 * * * *   720h0m0s     34m ago
tfstate=true                              false
```

Plutôt que d'attendre la prochaine exécution planifiée, déclenchez une sauvegarde immédiate à partir de chaque planification. L'option `--wait` fait patienter la commande jusqu'à la fin de la sauvegarde.

Sauvegardez les parts de l'auto-unsealer KMS :

```
velero create backup --from-schedule kms-autounsealer-shares-backup --wait
```

Sauvegardez les volumes persistants :

```
velero create backup --from-schedule pvs-backup --wait
```

Sauvegardez l'état Terraform :

```
velero create backup --from-schedule tfstate-backup --wait
```

Chaque commande se termine par un message `Backup completed with status: Completed.`. Par exemple :

```
Backup request "pvs-backup-20260707123616" submitted successfully.
Waiting for backup to complete. You may safely press ctrl-c to stop waiting - your backup will continue in the
background.
.....................................................................................................
.................................
Backup completed with status: Completed. You may check for more information using the commands `velero backup
describe pvs-backup-20260707123616` and `velero backup logs pvs-backup-20260707123616`.
```

8. Vérifier les sauvegardes et les dépôts recréés

Confirmez que les trois sauvegardes se sont terminées sans erreur :

```
velero get backup
```

```
NAME                                          STATUS      ERRORS   WARNINGS   CREATED
EXPIRES   STORAGE LOCATION   SELECTOR
kms-autounsealer-shares-backup-20260707123604   Completed   0        0          2026-07-07 12:36:04 +0000 UTC   29d
backup-location    kms-autounsealer-shares-backup=enabled
pvs-backup-20260707123616                       Completed   0        0          2026-07-07 12:36:16 +0000 UTC   29d
backup-location    backup=true
tfstate-backup-20260707123915                   Completed   0        0          2026-07-07 12:39:15 +0000 UTC   29d
backup-location    tfstate=true
```

Vérifiez que Velero a recréé les dépôts de sauvegarde Kopia sur les nouveaux buckets (leur colonne `AGE` doit désormais être récente) :

```
kubectl -n velero get backuprepositories
```

```
NAME                                  AGE     REPOSITORY TYPE
glance-backup-location-kopia-6sdt6    5m20s   kopia
ironic-backup-location-kopia-hnbjh    5m18s   kopia
kms-backup-location-kopia-567f6       5m17s   kopia
monitoring-backup-location-kopia-kvmjv   5m15s   kopia
```

9. Sauvegarder les bases de données

Les bases de données internes sont gérées par **CloudNativePG** et sauvegardées séparément, vers le `barmanObjectStore` qui pointe désormais vers le nouveau bucket. Déclenchez une sauvegarde pour chaque base de données.

> **Info**
>
> Les commandes ci-dessous utilisent la base `placement-db` dans le namespace `placement` à titre d'exemple. Répétez-les pour les autres bases de données, en adaptant le nom de la base et son namespace en conséquence.

```
kubectl cnpg backup placement-db -n placement
```

```
backup/placement-db-20260707124646 created
```

Confirmez qu'elle atteint l'état `completed` :

```
kubectl get backups -n placement
```

```
...
placement-db-20260707124646   5s      placement-db   barmanObjectStore   completed
```

10. Vérifier que les données ont bien été écrites dans les nouveaux buckets

Les étapes précédentes indiquent que les sauvegardes se sont terminées du point de vue de la plateforme. Par précaution supplémentaire, vérifiez que les objets ont réellement été créés dans les **nouveaux** buckets S3.

À l'aide de votre client S3 habituel ou de l'interface de votre stockage objet, parcourez chaque bucket de sauvegarde et confirmez que de nouveaux objets sont apparus depuis l'opération. Vous devriez voir des objets récemment datés correspondant aux sauvegardes des bases de données, des volumes persistants, de l'état Terraform et des parts KMS que vous avez déclenchées. Cela confirme que les sauvegardes sont effectivement stockées sur le nouveau stockage objet.

11. Vérification finale

Relancez le diagnostic pour confirmer que la plateforme est en bonne santé après la modification :

```
opcp-diag run -p
```

Enfin, confirmez que les sauvegardes planifiées s'exécutent désormais automatiquement sur leur nouvel emplacement. Après la prochaine exécution planifiée, `velero get backup` doit afficher les sauvegardes nouvellement créées à côté de celles que vous avez déclenchées manuellement :

```
velero get backup
```

```
NAME                                          STATUS      ERRORS   WARNINGS   CREATED
EXPIRES   STORAGE LOCATION   SELECTOR
kms-autounsealer-shares-backup-20260707130021   Completed   0        0          2026-07-07 13:00:21 +0000 UTC   29d
backup-location    kms-autounsealer-shares-backup=enabled
kms-autounsealer-shares-backup-20260707123604   Completed   0        0          2026-07-07 12:36:04 +0000 UTC   29d
backup-location    kms-autounsealer-shares-backup=enabled
pvs-backup-20260707130021                       Completed   0        0          2026-07-07 13:00:22 +0000 UTC   29d
backup-location    backup=true
pvs-backup-20260707123616                       Completed   0        0          2026-07-07 12:36:16 +0000 UTC   29d
backup-location    backup=true
tfstate-backup-20260707130021                   Completed   0        0          2026-07-07 13:00:30 +0000 UTC   29d
backup-location    tfstate=true
tfstate-backup-20260707123915                   Completed   0        0          2026-07-07 12:39:15 +0000 UTC   29d
backup-location    tfstate=true
```

De la même manière, les sauvegardes de bases de données se poursuivent automatiquement :

```
kubectl get backups -n placement
```

```
...
placement-db-20260707124646   20m     placement-db   barmanObjectStore   completed
placement-db-20260707130000   6m54s   placement-db   barmanObjectStore   completed
```

Les sauvegardes de votre plateforme OPCP ciblent désormais les nouveaux buckets S3.

Aller plus loin

Pour une formation ou une assistance technique sur la mise en œuvre de nos solutions, contactez votre commercial ou consultez la page Professional Services pour obtenir un devis et faire analyser votre projet par nos experts.

Échangez avec notre communauté d'utilisateurs.

[1] : S3 est une marque déposée appartenant à Amazon Technologies, Inc. Les services de OVHcloud ne sont pas sponsorisés, approuvés, ou affiliés de quelque manière que ce soit.

Sécurité

Gestion des droits IAM

Découvrez comment gérer les utilisateurs, les rôles et les droits d'accès sur votre On-Prem Cloud Platform via Keycloak

Objectif

Ce guide explique comment gérer les droits et les accès IAM (Identity and Access Management) de votre **On-Prem Cloud Platform (OPCP)** via **Keycloak**, qui centralise l'authentification aux services applicatifs de la plateforme.

Prérequis

- Disposer d'un service On-Prem Cloud Platform livré et opérationnel
- Avoir accès à l'interface Keycloak avec un compte disposant du rôle `it_admin` ou supérieur
- Avoir pris connaissance du guide « Mise en route de votre OPCP »

En pratique

Architecture d'authentification

Toute l'authentification de l'**OPCP** repose sur deux niveaux distincts :

- **L'accès au control plane** : accès SSH aux contrôleurs qui composent l'infrastructure de la plateforme, remis lors de la mise à disposition du service
- **L'accès aux services** : centralisé dans **Keycloak** (Realm Master), qui sert de point d'entrée unique pour :
  - le **Dashboard** (tableau de bord unifié du service)
  - l'interface graphique **OpenStack Horizon**
  - les **APIs OpenStack** (Keystone, Nova, Neutron, Glance, Ironic, etc.)
  - les outils de supervision : **Grafana**, **Netbox**, **Prometheus**

Accès au control plane

Lors de la mise à disposition de votre **OPCP**, des identifiants vous sont remis pour vous connecter en SSH aux différents contrôleurs qui composent le control plane de la plateforme.

Le périmètre d'accès du compte administrateur qui vous est remis varie selon le mode de gestion choisi.

Mode managé par OVHcloud

En mode managé, le compte administrateur qui vous est remis dispose de droits étendus permettant d'exécuter les outils d'administration de la solution :

- `opcp-cli` : outil en ligne de commande pour administrer la plateforme

- `opcp-diag` : outil de diagnostic du control plane

La gestion du déploiement et des mises à jour du control plane est assurée par OVHcloud.

Mode non managé par OVHcloud

En mode non managé, le compte administrateur qui vous est remis dispose des **pleins privilèges** sur le control plane, permettant de :

- Déployer le control plane
- Mettre à jour le control plane
- Accéder à l'ensemble des outils d'administration

> **Warning**
> En mode non managé, OVHcloud ne prend pas en charge les opérations d'administration du control plane. Vous êtes responsable du déploiement et des mises à jour.

Identifiants initiaux

Deux types d'identifiants vous sont remis à l'initialisation de votre OPCP :

Compte Linux administrateur : permet la connexion SSH aux contrôleurs. Par défaut, **l'authentification par mot de passe est désactivée** sur les contrôleurs. Lors de l'installation de votre OPCP, une clé SSH publique de votre choix peut être fournie à OVHcloud afin d'être déployée et de vous permettre un premier accès. Un mot de passe vous est également remis, utile uniquement pour accéder directement au serveur via la console en cas d'impossibilité de connexion SSH.

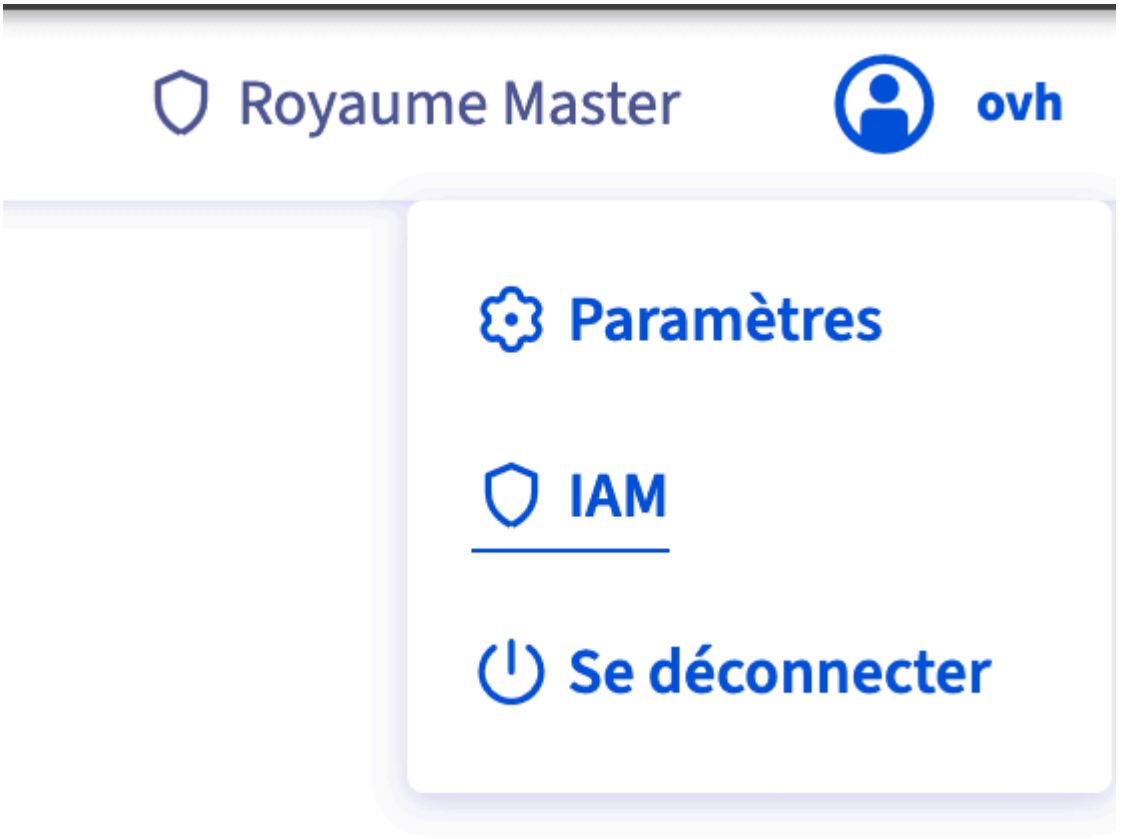
Compte Keycloak administrateur : fourni aussi bien en mode managé qu'en mode non managé, ce compte dispose des droits d'administration complets sur le Realm Master. Un mot de passe temporaire vous est attribué, que vous devez **changer dès la première connexion**.

> **Info**
> Ces identifiants initiaux sont distincts des comptes utilisateurs que vous créerez ensuite dans Keycloak pour vos équipes. Conservez-les en lieu sûr et changez les mots de passe temporaires immédiatement après la livraison.

Accéder à l'interface d'administration Keycloak

Deux méthodes permettent d'accéder à l'interface d'administration Keycloak de votre **OPCP**.

Via le Dashboard : connectez-vous à `admin.dashboard.<nomdedomaine>`, puis cliquez sur votre nom en haut à droite. Un menu apparaît avec un lien **IAM** qui vous redirige directement vers l'interface Keycloak.



Accès direct : vous pouvez également accéder à Keycloak directement via `admin.keycloak.<nomdedomaine>`.

> **Info**
>
> Remplacez `<nomdedomaine>` par le nom de domaine fourni lors de la livraison de votre OPCP.

Hiérarchie des rôles

Le modèle IAM de l'**OPCP** repose sur quatre rôles distincts, du plus restreint au plus privilégié. Ces rôles constituent un point de départ, mais Keycloak vous offre toute la flexibilité nécessaire pour composer vos propres rôles et les assigner à des groupes d'utilisateurs selon les besoins de votre organisation.

> **Info**
>
> Vous pouvez créer des rôles personnalisés dans Keycloak, en combinant les permissions disponibles, puis les assigner à des groupes d'utilisateurs. Référez-vous à la documentation officielle Keycloak pour la gestion avancée des rôles et des groupes.

> **Warning**
>
> Cette section s'applique à l'OPCP en version **3.0.6** ou supérieure.

Rôle `reader`

Profil supervision et audit. Ce rôle donne par défaut un accès en lecture seule, sans droit d'administration.

Il permet de :

- Consulter ses paramètres de compte dans Keycloak
- Accéder aux ressources des projets OpenStack selon les attributs de projet configurés dans Keycloak

> **Info**
>
> Le rôle Keycloak `reader` peut recevoir n'importe quel rôle OpenStack — `reader` (lecture seule), `member` (lecture/écriture) ou `admin` — via les attributs de projet configurés sur le compte ou le groupe Keycloak. Sans attribut de projet, aucune ressource OpenStack n'est visible.

Rôle `dc_operator`

Opérateur datacenter. Ce rôle est destiné aux équipes en charge de l'infrastructure réseau et physique.

Il donne les accès supplémentaires suivants par rapport au `reader` :

- Accéder à **Grafana** en lecture seule
- Accéder à **Netbox** en tant qu'opérateur (création et modification des ressources réseau)

> **Info**
>
> Le `dc_operator` n'a aucun droit d'administration des utilisateurs Keycloak. Comme le `reader`, il ne dispose d'aucun droit OpenStack par défaut : ceux-ci nécessitent des attributs de projet sur l'utilisateur ou son groupe. Il ne dispose pas non plus de l'iframe OpenStack du Dashboard par défaut — lorsque vous lui accordez un accès OpenStack via les attributs de projet, accordez-lui également cette iframe.

Rôle `it_admin`

Opérateur principal de la plateforme. Ce rôle est destiné aux équipes IT qui exploitent les ressources de l'OPCP au quotidien.

Il donne les accès supplémentaires suivants par rapport au `dc_operator` :

- Gérer les utilisateurs du Realm Master dans Keycloak (création de comptes, attribution de droits)
- Accéder à **Grafana** en édition (modification des dashboards)
- Accéder à **OpenStack** avec tous les droits (`reader`, `member`, `admin`)

> **Warning**
>
> Le rôle `it_admin` ne permet pas de modifier les paramètres globaux du realm Keycloak (politique de mot de passe, fédération d'identité, etc.). Ces actions sont réservées au `master_admin`.

Rôle `master_admin`

Administrateur de la plateforme. Ce rôle dispose de tous les droits, y compris l'administration complète du Realm Master Keycloak.

Il donne les accès supplémentaires suivants par rapport à l'`it_admin` :

- Modifier la configuration du Realm Master (politique de sécurité, fédération, etc.)
- Accéder à **Grafana** en administration
- Administrer **Netbox** avec tous les droits, y compris le rôle `admin`

> **Warning**
>
> Le rôle `master_admin` est attribué par OVHcloud lors de la livraison du service. Son usage doit être limité aux opérations d'administration de la plateforme qui ne peuvent pas être déléguées à un `it_admin`.

Matrice des droits

> **Warning**
>
> Cette matrice s'applique à l'OPCP en version **3.0.6** ou supérieure.

Le tableau ci-dessous récapitule les accès de chaque rôle sur les applications intégrées.

Légende : ✅ Autorisé | ❌ Non autorisé | 🟡 Configuration requise (sous réserve des attributs de projet)

Application	Permission	`reader`	`dc_operator`	`it_admin`	`master_admin`
Keycloak	Voir ses paramètres de compte	✅	✅	✅	✅
Keycloak	Gérer les utilisateurs du realm	❌	❌	✅	✅
Keycloak	Administrer le realm (politique, fédération)	❌	❌	❌	✅
OpenStack	reader (vue)	🟡	🟡	✅	✅
OpenStack	member (édition)	🟡	🟡	✅	✅
OpenStack	admin	🟡	🟡	✅	✅
Grafana	view (dashboards)	❌	✅	✅	✅
Grafana	edit (dashboards)	❌	❌	✅	✅
Grafana	admin (dashboards)	❌	❌	❌	✅
Netbox	reader	❌	✅	✅	✅
Netbox	operator	❌	✅	✅	✅
Netbox	admin	❌	❌	❌	✅
Prometheus	view	✅	✅	✅	✅
Dashboard	Keycloak — paramètres compte	✅	✅	✅	✅
Dashboard	Keycloak — administration utilisateurs	❌	❌	✅	✅
Dashboard	OpenStack (iframe)	✅	❌	✅	✅
Dashboard	Grafana (iframe)	❌	✅	✅	✅
Dashboard	Netbox (iframe)	❌	✅	✅	✅

> **Info**
>
> L'accès OpenStack des rôles Keycloak `reader` et `dc_operator` est conditionné par la présence d'**attributs de projet** sur l'utilisateur ou son groupe dans Keycloak. Sans attribut de projet configuré, aucune ressource OpenStack n'est accessible. N'importe quel rôle OpenStack peut techniquement être attribué ainsi, mais `member` est recommandé : le rôle `admin` n'est pas restreint au périmètre du projet.

Créer un utilisateur et lui assigner un rôle

Créer un utilisateur

Dans l'interface Keycloak, assurez-vous d'être connecté sur le **Realm Master**, puis accédez à **Users** > **Add user**.

Renseignez les champs suivants :

- **Username** : identifiant de connexion de l'utilisateur
- **Email** : adresse e-mail (recommandé)
- **First name / Last name** : nom et prénom

Cliquez sur **Create**, puis ouvrez l'onglet **Credentials** pour définir un mot de passe temporaire. Activez l'option **Temporary** afin que l'utilisateur soit contraint de le changer à sa première connexion.

Assigner un rôle à un utilisateur

Dans la fiche de l'utilisateur, ouvrez l'onglet **Role mapping**, puis cliquez sur **Assign role**.

Dans le filtre, sélectionnez **Filter by realm roles**, puis recherchez et sélectionnez le rôle souhaité (`reader`, `dc_operator`, `it_admin` ou `master_admin`). Cliquez sur **Assign**.

> **Info**
>
> Un utilisateur peut se voir assigner plusieurs rôles. Les permissions effectives sont l'union des droits de l'ensemble des rôles qui lui sont attribués.

> **Warning**
>
> Le rôle par défaut assigné à un nouvel utilisateur Keycloak est toujours `reader`. Bien que l'interface Keycloak permette de modifier ce rôle par défaut au niveau du realm, cette configuration est gérée par Terraform sur l'**OPCP** et sera systématiquement écrasée lors d'une réconciliation.

Assigner un rôle à un groupe

Assignez les rôles à des **groupes** plutôt qu'à des utilisateurs individuels pour simplifier la gestion des droits à grande échelle.

Dans Keycloak, accédez à **Groups** > **Create group**, nommez le groupe, puis ouvrez l'onglet **Role mapping** pour lui assigner le rôle souhaité.

Ajoutez ensuite les utilisateurs au groupe depuis leur fiche utilisateur, via **Groups** > **Join group**.

Configurer les droits OpenStack via les attributs Keycloak

Pour les rôles `reader` et `dc_operator`, les droits d'accès aux projets OpenStack se définissent directement dans les **attributs de l'utilisateur** (ou du groupe) dans Keycloak.

Ajouter un attribut de projet sur un utilisateur

Dans l'interface Keycloak, accédez à l'utilisateur concerné, puis ouvrez l'onglet **Attributes**.

Ajoutez un nouvel attribut avec les valeurs suivantes :

- **Key** : `project`
- **Value** : un objet JSON décrivant le projet et le rôle à attribuer

```
{
    "domain": {
        "name": "Default"
    },
    "name": "nom-du-projet",
    "roles": [
        {
            "name": "reader"
        }
    ]
}
```

> **Info**
>
> Pour les rôles `reader` et `dc_operator`, n'importe quel rôle OpenStack peut être attribué via les attributs de projet, mais `member` est recommandé plutôt qu'`admin`, qui n'est pas restreint au périmètre du projet. Les rôles `it_admin` et `master_admin` reposent également sur un attribut de projet, mais le leur est défini par défaut : aucune configuration n'est donc nécessaire.

Attribuer plusieurs projets à un utilisateur

Il est possible d'ajouter **plusieurs attributs** `project` pour un même utilisateur ou groupe. Ils seront automatiquement fusionnés en un attribut `projects` dans le token transmis à Keystone.

Configurer les attributs sur un groupe

La même configuration peut être appliquée à un **groupe Keycloak**. Tous les membres du groupe hériteront automatiquement des attributs de projet définis sur ce groupe.

Dans Keycloak, accédez à **Groups**, sélectionnez le groupe souhaité, puis renseignez les attributs `project` de la même manière que pour un utilisateur individuel.

> **Info**
>
> Les attributs définis sur un groupe et ceux définis directement sur l'utilisateur sont fusionnés. Un utilisateur peut ainsi bénéficier de projets issus de plusieurs groupes en plus de ses propres attributs.

Keycloak comme source unique de vérité

Nous recommandons de ne pas créer d'utilisateurs directement dans OpenStack, pour conserver Keycloak comme **source unique de vérité** sur les comptes de la plateforme. Tout compte créé directement dans OpenStack échapperait au cycle de vie géré par Keycloak et ne bénéficierait pas du nettoyage automatique ni de la gestion centralisée des droits.

Comptes de service pour l'automatisation

Pour les besoins d'automatisation (Terraform, OpenTofu, scripts, pipelines CI/CD, etc.), nous recommandons de créer un **compte dédié dans Keycloak** avec les permissions adaptées au périmètre de l'automate, puis de générer des **application credentials OpenStack** associés à ce compte.

Les application credentials permettent à vos outils d'automatisation de piloter OpenStack sans dépendre des credentials personnels d'un utilisateur. Un utilisateur connecté à OpenStack avec son compte Keycloak peut créer ses propres application credentials depuis l'interface Horizon, via **Identity** > **Application Credentials** > **Create Application Credentials**.

> **Info**
>
> En cas de suppression du compte Keycloak associé, les application credentials OpenStack seront automatiquement révoqués. Veillez à ne pas lier des automates critiques à un compte personnel.

Pour configurer l'authentification Keycloak avec la CLI OpenStack et obtenir vos credentials, référez-vous au guide « Comment utiliser les APIs et obtenir les credentials ».

Nettoyage automatique des comptes

La plateforme réagit aux événements de suppression d'utilisateur dans Keycloak : dès qu'un compte est supprimé du Realm Master, les utilisateurs et les **application credentials** OpenStack associés sont automatiquement supprimés.

> **Warning**
>
> La suppression d'un compte dans Keycloak entraîne la révocation immédiate des application credentials OpenStack associés. Anticipez cet impact avant toute suppression de compte, notamment si des automatisations ou des scripts utilisent ces credentials.

Configuration avancée de Keycloak

L'ensemble des fonctionnalités officielles de Keycloak est disponible sur votre **OPCP**. Pour toute configuration avancée (politiques de mots de passe, gestion fine des rôles, configuration des clients, etc.), référez-vous à la documentation officielle Keycloak.

Authentification multi-facteurs (MFA / OTP)

Nous recommandons fortement d'activer l'**authentification multi-facteurs** sur les comptes de votre plateforme, en particulier pour les rôles `it_admin` et `master_admin`. Keycloak supporte nativement le protocole TOTP (Time-based One-Time Password), compatible avec toute application d'authentification supportant ce standard ouvert, comme FreeOTP.

Le MFA se configure au niveau du flux d'authentification du realm. Consultez la documentation officielle Keycloak pour mettre en place un flux post-login avec OTP obligatoire ou conditionnel selon les rôles.

Fédération d'identité et Identity Providers

Keycloak permet de fédérer des annuaires externes (Active Directory, LDAP) ou de connecter des Identity Providers tiers (SAML, OIDC). Par exemple, pour mettre en place une fédération vers un Active Directory, consultez la section dédiée de la documentation officielle Keycloak.

> **Warning**
> Dans le cas d'une fédération Keycloak ou de l'ajout d'un Identity Provider vers un endpoint externe, vous devez autoriser les flux réseau correspondants dans le fichier de configuration OPCP via la variable `keycloakAllowedEndpoints`.

```
keycloakAllowedEndpoints:
    - host: "myactivedirectory.corp.com"
      port: 636
    - host: "keycloak.corp.com"
      port: 443
```

Chaque entrée correspond à un endpoint externe que Keycloak doit être autorisé à joindre. Ajoutez autant d'entrées que nécessaire selon votre configuration.

Automatisation avec OpenTofu / Terraform

La gestion des utilisateurs, groupes et rôles dans Keycloak peut être automatisée grâce à des outils d'Infrastructure as Code comme **OpenTofu** ou **Terraform**, en utilisant le provider officiel Keycloak.

Cela permet de gérer vos accès de manière déclarative, reproductible et versionnée.

Exemple de ressource pour créer un utilisateur :

```
resource "keycloak_user" "john_doe" {
  realm_id = "master"
  username = "john.doe"
  enabled  = true

  email      = "john.doe@corp.com"
  first_name = "John"
  last_name  = "Doe"

  initial_password {
    value     = "changeme"
    temporary = true
  }
}
```

Pour l'ensemble des ressources disponibles (utilisateurs, groupes, rôles, fédération, etc.), référez-vous à la documentation du provider Keycloak.

Lister l'ensemble des droits

Via opcp-diag (vue consolidée)

Pour obtenir une vue complète de l'ensemble des comptes et de leurs droits sur la plateforme, connectez-vous en SSH à un contrôleur puis exécutez :

```
opcp-diag audit
```

Cet outil affiche l'ensemble des comptes déclarés dans **Keycloak**, sur les **contrôleurs**, et dans les différents outils qui composent le control plane.

Options disponibles

Option	Description
`-o, --format`	Format de sortie : `json` (défaut) ou `text`
`--only`	Restreindre l'audit à un ou plusieurs types, séparés par des virgules

Les types d'audit disponibles pour `--only` sont :

Type	Description
`keycloak_master`	Comptes et droits configurés dans le Realm Master Keycloak
`openstack`	Utilisateurs et rôles OpenStack
`linux`	Comptes présents sur les contrôleurs
`netbox`	Comptes dans Netbox, la CMDB de la plateforme
`nog`	Comptes dans NOG, le composant qui pilote la configuration des équipements réseau pour appliquer les réseaux Neutron

Par exemple, pour auditer uniquement Keycloak et OpenStack :

```
opcp-diag audit --only keycloak_master,openstack
```

Pour une sortie lisible en mode texte :

```
opcp-diag audit --format text
```

Manipuler la sortie JSON

La sortie par défaut est au format **JSON**, ce qui permet de la manipuler et de la filtrer avec des outils comme `jq`. Par exemple :

```
# Exporter le résultat complet dans un fichier daté
opcp-diag audit > audit-$(date +%Y%m%d).json

# Auditer uniquement Keycloak et filtrer sur un utilisateur précis
opcp-diag audit --only keycloak_master | jq '.keycloak_master.users[] | select(.username == "john.doe")'
```

> **Info**
>
> `opcp-diag audit` est la méthode recommandée pour auditer les accès de votre plateforme. Contrairement à la CLI OpenStack, elle couvre l'intégralité des comptes présents sur la plateforme, indépendamment de leur historique de connexion.

Via la CLI OpenStack

Pour lister les assignations de rôles actives dans OpenStack :

```
openstack role assignment list --names
```

Pour lister les rôles disponibles sur la plateforme :

```
openstack role list
```

> **Warning**
>
> Ces commandes ne retournent que les utilisateurs qui se sont **déjà authentifiés** au moins une fois via OpenStack. Les utilisateurs présents dans Keycloak mais n'ayant jamais effectué de connexion OpenStack n'apparaîtront pas dans ces résultats.

Aller plus loin

Pour une formation ou une assistance technique sur la mise en œuvre de nos solutions, contactez votre commercial ou consultez la page Professional Services pour obtenir un devis et faire analyser votre projet par nos experts.

Échangez avec notre communauté d'utilisateurs.

CloudStore

Premiers pas avec votre CloudStore

Découvrez comment vous connecter, gérer les comptes, déployer des services et configurer votre CloudStore

Objectif

Ce guide vous présente comment vous connecter à l'interface de gestion de votre CloudStore, comprendre ses concepts clés et effectuer les premières opérations : créer des comptes et déployer des services.

Le **CloudStore** est un framework d'infrastructure de haut niveau déployé sur On-Premise Cloud Platform (**OPCP**). Il fournit des services essentiels pour aider les fournisseurs cloud à déployer et gérer des solutions Cloud Native pour leurs clients via une plateforme basée sur un catalogue de services.

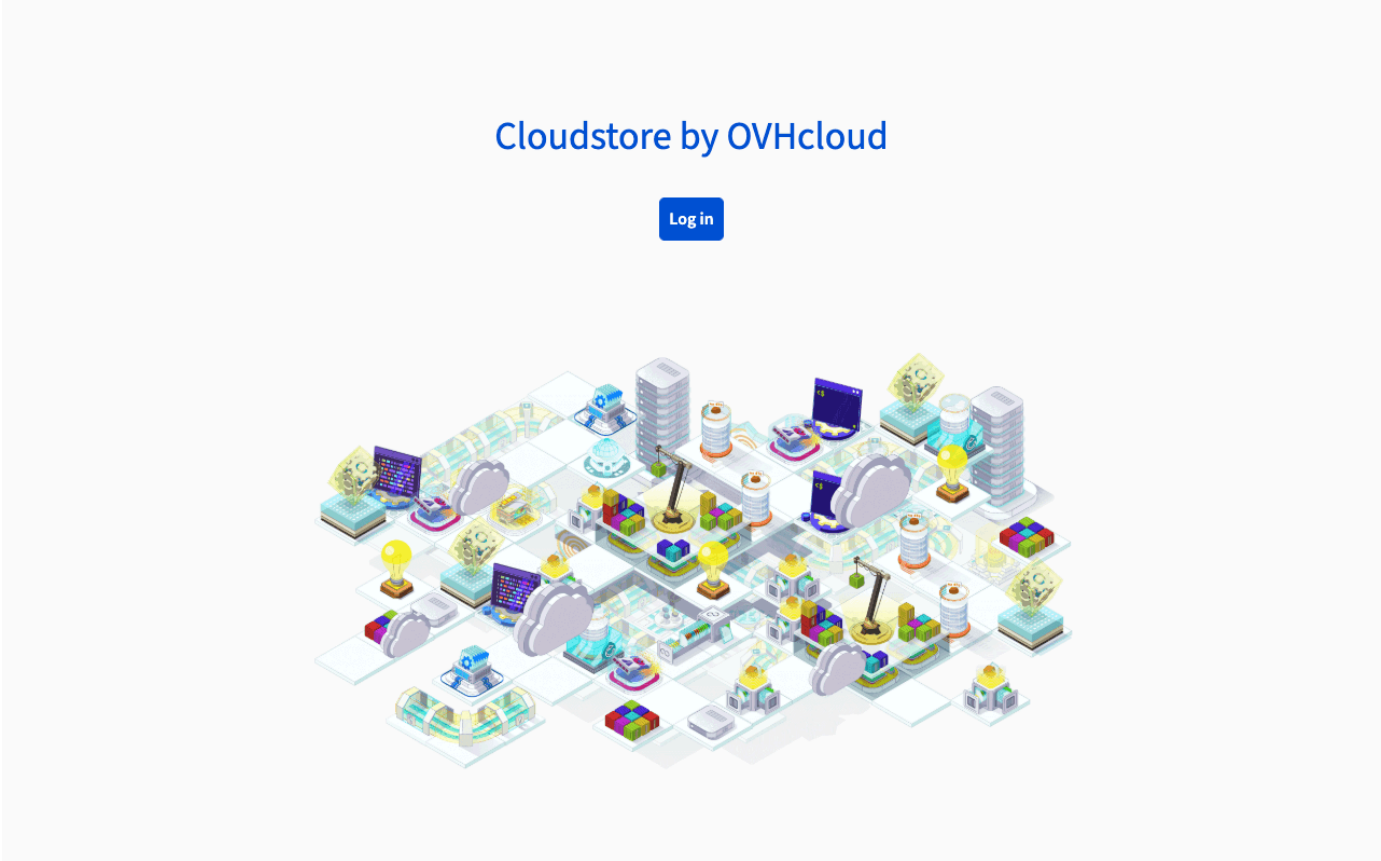
Prérequis

- L'**URL** de l'interface de gestion de votre CloudStore, fournie lors de la livraison du service.
- Les identifiants de connexion (nom d'utilisateur et mot de passe) fournis lors de la livraison du service. Ces identifiants sont gérés par **Keycloak**, la solution de gestion des identités et des accès utilisée par la plateforme.

En pratique

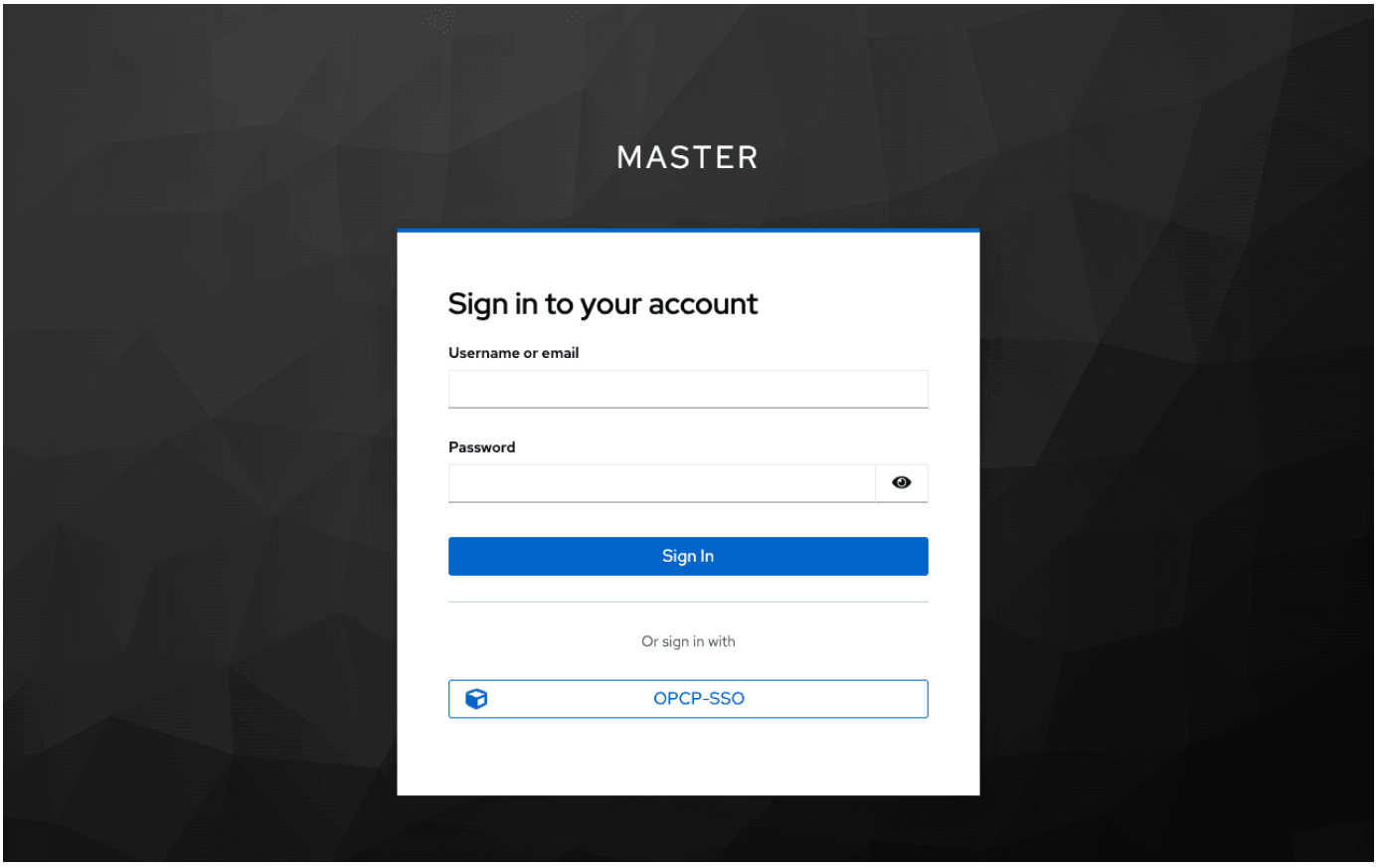
Connexion au CloudStore

Accédez à l'**URL** fournie pour votre instance CloudStore. Une page de connexion s'affiche.



Cliquez sur le bouton de connexion pour être redirigé vers la page d'authentification **Keycloak**. Deux options s'offrent à vous :

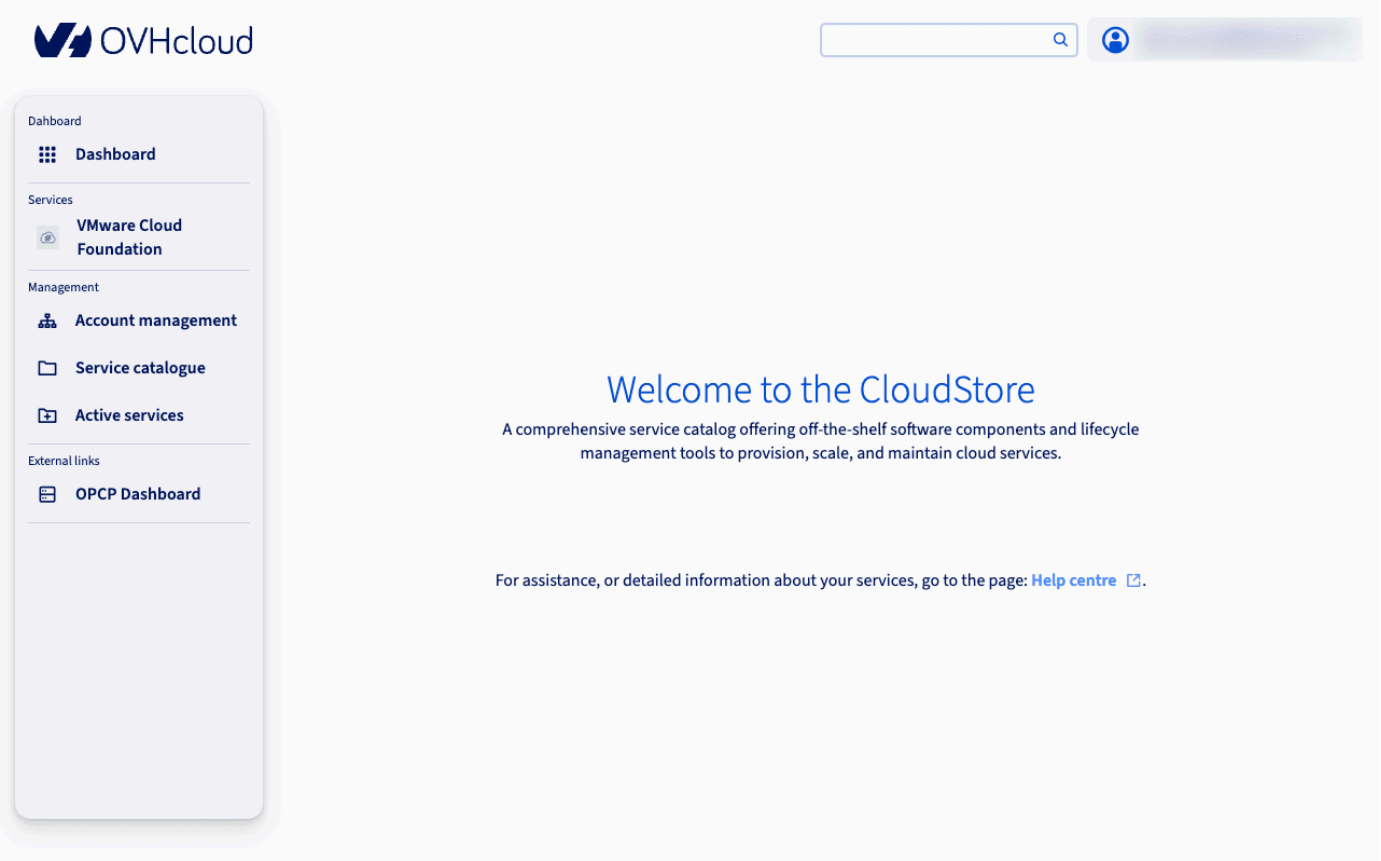
- Saisir vos identifiants directement dans le formulaire de connexion Keycloak.
- Utiliser le bouton **OPCP-SSO** pour vous authentifier via le Keycloak OPCP Core fédéré. Cette option permet aux utilisateurs déjà enregistrés dans OPCP Core de se connecter sans gérer un jeu d'identifiants supplémentaire.



Une fois authentifié, l'interface obtient un jeton d'accès OIDC utilisé pour toutes les opérations suivantes.

Tableau de bord CloudStore

Après connexion, vous êtes redirigé vers le tableau de bord du CloudStore.



Depuis le tableau de bord, vous pouvez accéder aux sections suivantes :

- **Catalogue de services** : Parcourir et activer les services disponibles (par exemple, VCF).
- **Comptes** : Créer et gérer les comptes locataires pour vos clients ou équipes internes.
- **Contrôleurs et Apps** : Déployer et gérer les composants d'infrastructure de chaque service.
- **Gestion IAM** : Configurer les utilisateurs et les permissions dans Keycloak.

Concepts clés

Avant d'utiliser le CloudStore, il est important de comprendre ses principaux concepts.

Personas

Le CloudStore distingue deux types d'utilisateurs :

- **Fournisseurs cloud** (IT admins, Service admins) : Clients OVHcloud qui exploitent la plateforme. Ils provisionnent l'infrastructure, déploient les services et gèrent les comptes.
- **Utilisateurs cloud** (Landing Zone Manager users) : Clients des fournisseurs cloud qui consomment les services déployés pour eux via la **Landing Zone**.

Persona	Responsabilités
IT admin	Créer des comptes, activer des services, déployer des apps
Service admin	Gérer un service spécifique, déployer son contrôleur et ses apps
Account admin	Gérer la configuration de son compte
Landing Zone Manager user	Accéder aux apps via la Landing Zone

Le modèle contrôleur/app

Chaque service dans le CloudStore suit une architecture **contrôleur/app** :

- Le **contrôleur** (controlplane) est déployé en premier. Il gère le cycle de vie des apps et orchestre les opérations multi-locataires. Un seul contrôleur peut gérer plusieurs apps réparties sur différents comptes.
- Les **apps** (dataplane) sont les charges de travail déployées pour un compte spécifique. Chaque app est isolée dans un seul compte et ne peut pas être partagée entre les comptes.

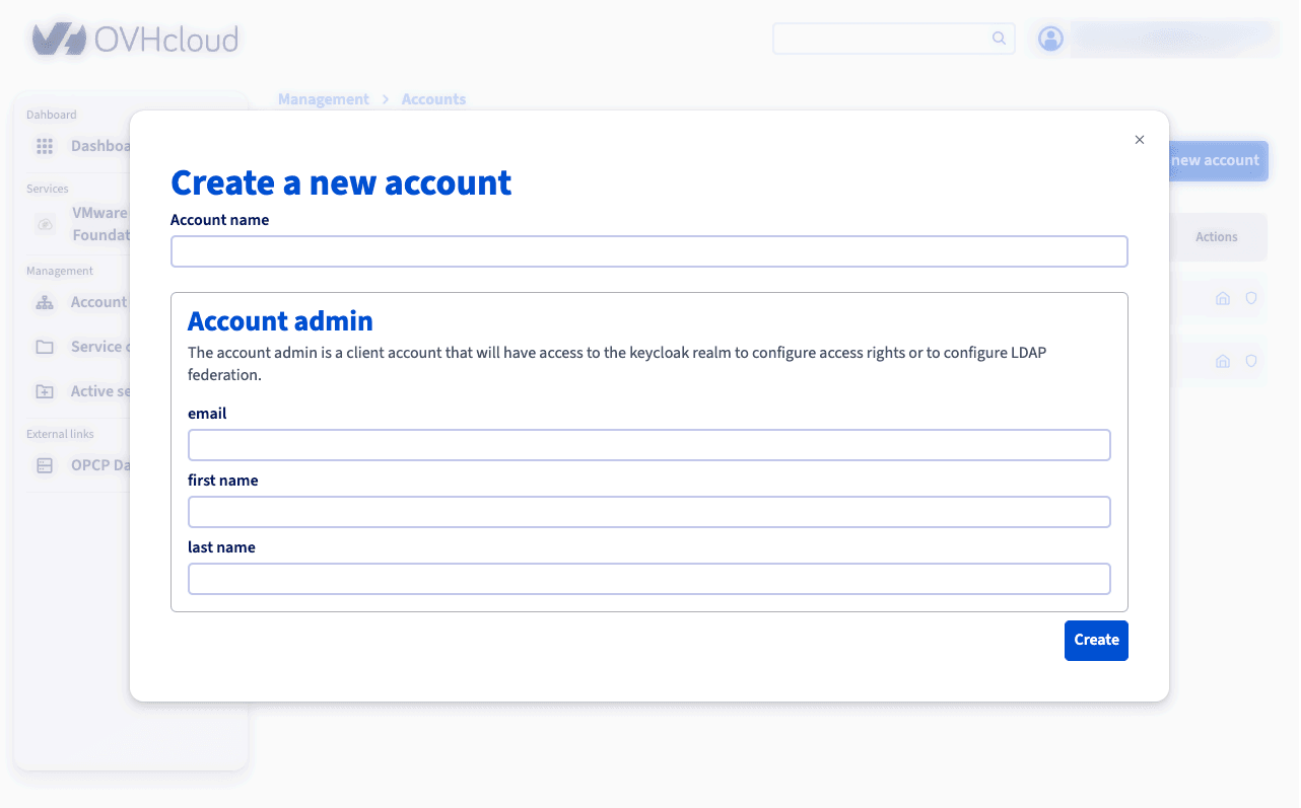
Ce modèle permet aux fournisseurs cloud d'offrir le même service à plusieurs clients, chacun disposant de sa propre instance d'app isolée tout en partageant la même infrastructure de contrôleur.

Création d'un compte

Un **compte** représente une entreprise ou un département. Chaque compte dispose de son propre realm Keycloak, garantissant une isolation IAM complète. Les comptes doivent être créés avant de déployer des apps, car celles-ci sont toujours rattachées à un compte.

Pour créer un compte :

1. Depuis le tableau de bord du CloudStore, accédez à la section **Comptes**.
2. Cliquez sur **Créer un compte**.
3. Remplissez les champs requis : nom du compte, nom complet de l'administrateur et adresse e-mail de l'administrateur.



4. Validez le formulaire.

La plateforme effectue automatiquement les actions suivantes :

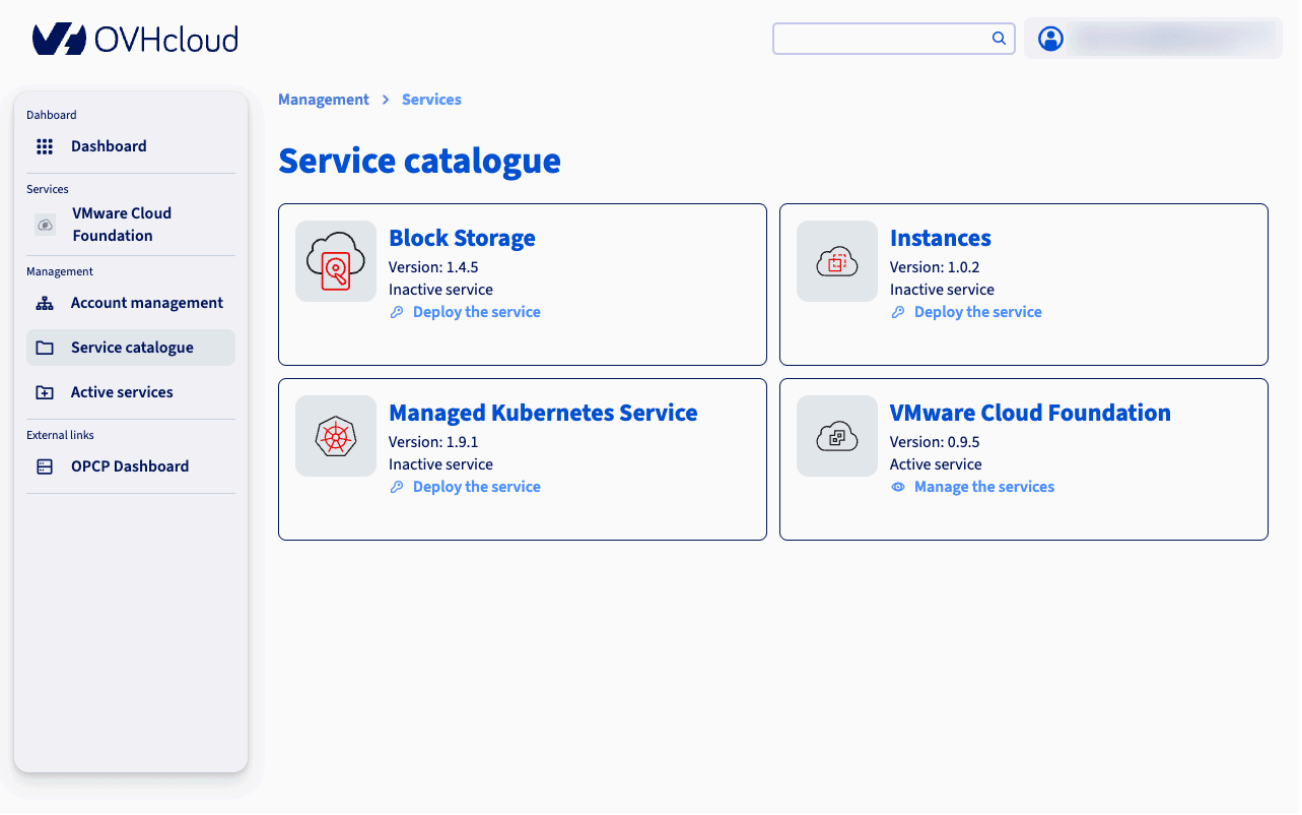
- Création d'un realm Keycloak dédié nommé `account-{name}`.
- Création d'un utilisateur administrateur avec le rôle `account-admin`.
- Configuration d'un mot de passe temporaire (l'administrateur du compte sera invité à le modifier lors de sa première connexion).
- Configuration d'un client `landing-zone` pour l'accès des utilisateurs Landing Zone Manager.

Déploiement d'un service

Le déploiement d'un service se fait en 2 étapes : commencez par déployer le **contrôleur**, puis déployez une ou plusieurs **apps** pour des comptes spécifiques.

Étape 1 — Déployer un contrôleur

1. Accédez au **Catalogue de services** depuis le tableau de bord.



2. Sélectionnez le service que vous souhaitez activer.
3. Cliquez sur **Activer le service**.
4. Choisissez la version et configurez les propriétés requises.
5. Sélectionnez les hôtes sur lesquels le contrôleur sera déployé.
6. Validez le formulaire.

> **Info**
>
> Le déploiement du contrôleur est **asynchrone**. L'interface confirmera que le déploiement a démarré, mais le provisionnement s'effectue en arrière-plan via Kubernetes et Terraform. Vous pouvez suivre l'état du déploiement depuis la page des contrôleurs.

Étape 2 — Déployer une app

Une fois le contrôleur actif et au moins un compte créé :

1. Accédez à la page du contrôleur.
2. Cliquez sur **Déployer une app**.
3. Sélectionnez le compte cible.
4. Choisissez la version et configurez les propriétés requises.
5. Sélectionnez les hôtes pour le déploiement de l'app.
6. Validez le formulaire.

Lors du déploiement d'une app, la plateforme crée automatiquement les ressources Keycloak nécessaires (client-id, rôles et groupes) dans le realm du compte cible, garantissant que seuls les utilisateurs autorisés de ce compte puissent accéder à l'app.

> **Info**
>
> Comme pour le contrôleur, le déploiement d'une app est **asynchrone**. Le provisionnement de l'infrastructure est géré par Kubernetes et Terraform en arrière-plan.

Mise à l'échelle de la capacité

Vous pouvez ajouter ou retirer des hôtes d'un contrôleur ou d'une app déployée pour ajuster la capacité.

1. Accédez à la page du contrôleur ou de l'app.
2. Cliquez sur **Étendre la capacité**.
3. Sélectionnez les hôtes à ajouter.
4. Validez le formulaire.

L'opération de mise à l'échelle est asynchrone. Kubernetes réconcilie la configuration et provisionne les ressources sur les hôtes mis à jour.

Authentification et niveaux d'accès

Le CloudStore utilise un modèle de fédération Keycloak en couches qui reflète l'architecture de la plateforme :

Niveau	Instance Keycloak	Utilisateurs	Rôle
L1	OPCP Core Keycloak	DC operators, Super admins	Identité au niveau infrastructure
L2	CloudStore Keycloak	IT admins, Service admins	Gestion de la plateforme
L3	Realms Keycloak par compte	Landing Zone Manager users	Accès aux applications

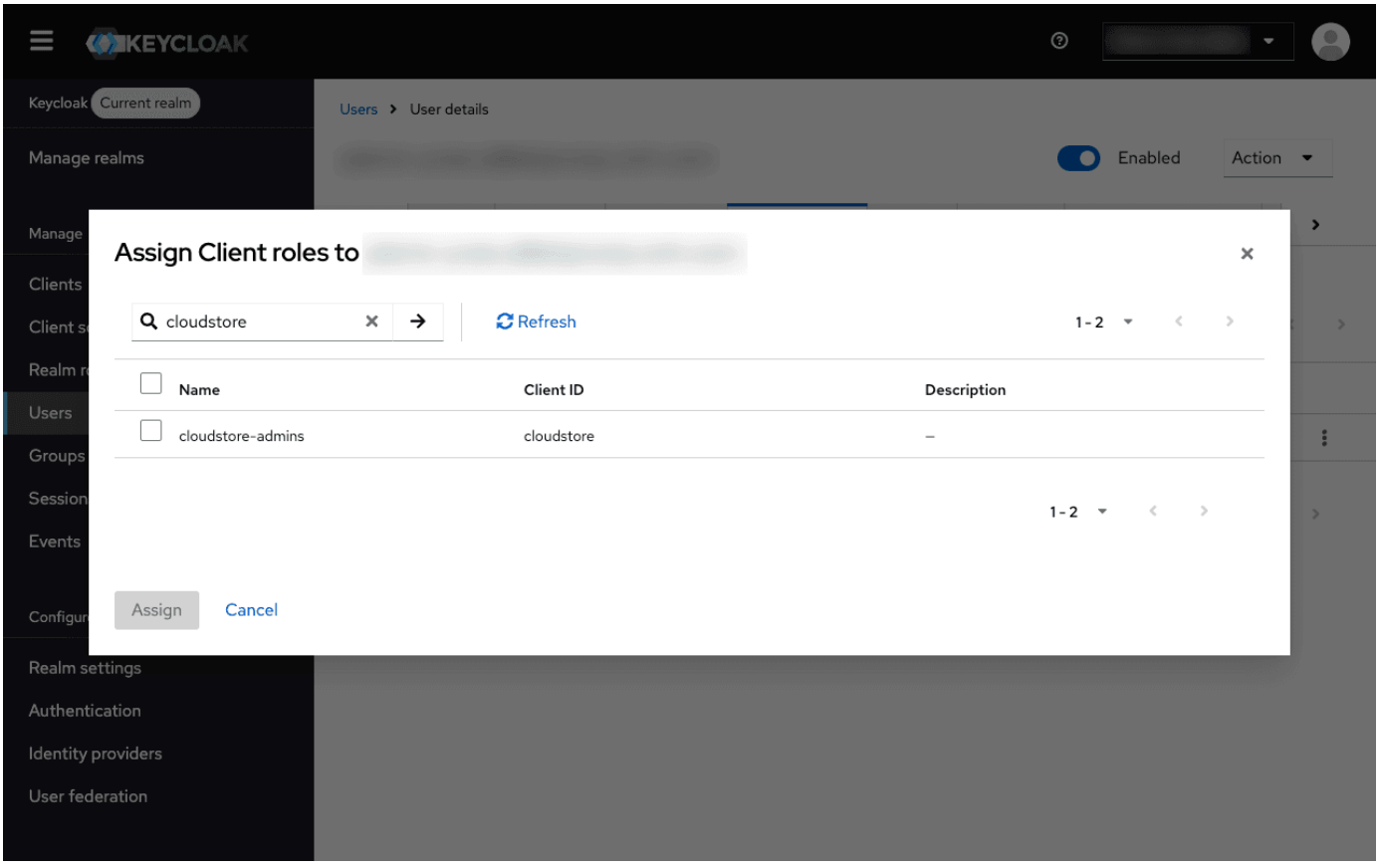
- **Keycloak L2** est fédéré avec **L1** (OPCP Core). Les droits accordés sur les projets OpenStack au niveau L1 sont ainsi propagés au niveau L2.
- **Keycloak L3** est une instance indépendante gérée par l'API CloudStore. Chaque compte dispose de son propre realm isolé.

> **Info**
>
> La **couche L3** est fournie par le Landing Zone Manager, un produit OPCP distinct en charge de la gestion des comptes utilisateurs Landing Zone Manager. Sa stack Keycloak **n'est pas fédérée** avec les instances Keycloak L1 (OPCP Core) et L2 (CloudStore).

Gestion de l'IAM sur le Keycloak CloudStore (L2)

Pour pouvoir gérer les utilisateurs, rôles et groupes sur le Keycloak CloudStore (L2), vous devez au préalable attribuer le rôle `cloudstore-admins` à votre utilisateur dans le **Keycloak OPCP Core (L1)**.

1. Connectez-vous à la console d'administration du Keycloak OPCP Core (L1).
2. Accédez à l'utilisateur auquel vous souhaitez accorder les droits de gestion IAM.
3. Dans l'onglet **Role mappings**, attribuez le rôle client `cloudstore-admins`.



Une fois ce rôle attribué, l'utilisateur disposera des permissions nécessaires pour administrer le Keycloak CloudStore (L2), y compris la gestion des realms, clients, utilisateurs et rôles.

La Landing Zone

La **Landing Zone** est l'interface destinée aux utilisateurs Landing Zone Manager (utilisateurs cloud). Elle offre une vue simplifiée des apps déployées pour leur compte.

Les utilisateurs Landing Zone Manager s'authentifient via le realm Keycloak spécifique à leur compte (L3) et ne peuvent accéder qu'aux apps déployées pour leur compte. La Landing Zone récupère la liste des apps accessibles et les filtre en fonction des permissions de l'utilisateur.

Aller plus loin

Si vous avez besoin d'une formation ou d'une assistance technique pour la mise en œuvre de nos solutions, contactez votre commercial ou cliquez sur ce lien pour obtenir un devis et demander une analyse personnalisée de votre projet à nos experts de l'équipe Professional Services.

Échangez avec notre communauté d'utilisateurs.

Landing Zone Manager

Landing Zone Manager - Créer un compte utilisateur

Découvrez comment créer un nouveau compte utilisateur dans le Landing Zone Manager et s'y connecter pour la première fois

Objectif

Le Landing Zone Manager est le portail OPCP qui permet à chaque équipe, département ou client de déployer et piloter ses workloads en toute autonomie, dans un espace dédié et cloisonné. Les administrateurs y provisionnent les utilisateurs et leur donnent accès à la plateforme. Chaque compte utilisateur s'appuie sur un realm dédié dans l'instance Keycloak sous-jacente, garantissant l'isolation multitenant des accès.

Ce guide explique comment créer un nouveau compte utilisateur dans le Landing Zone Manager et comment s'y connecter pour la première fois.

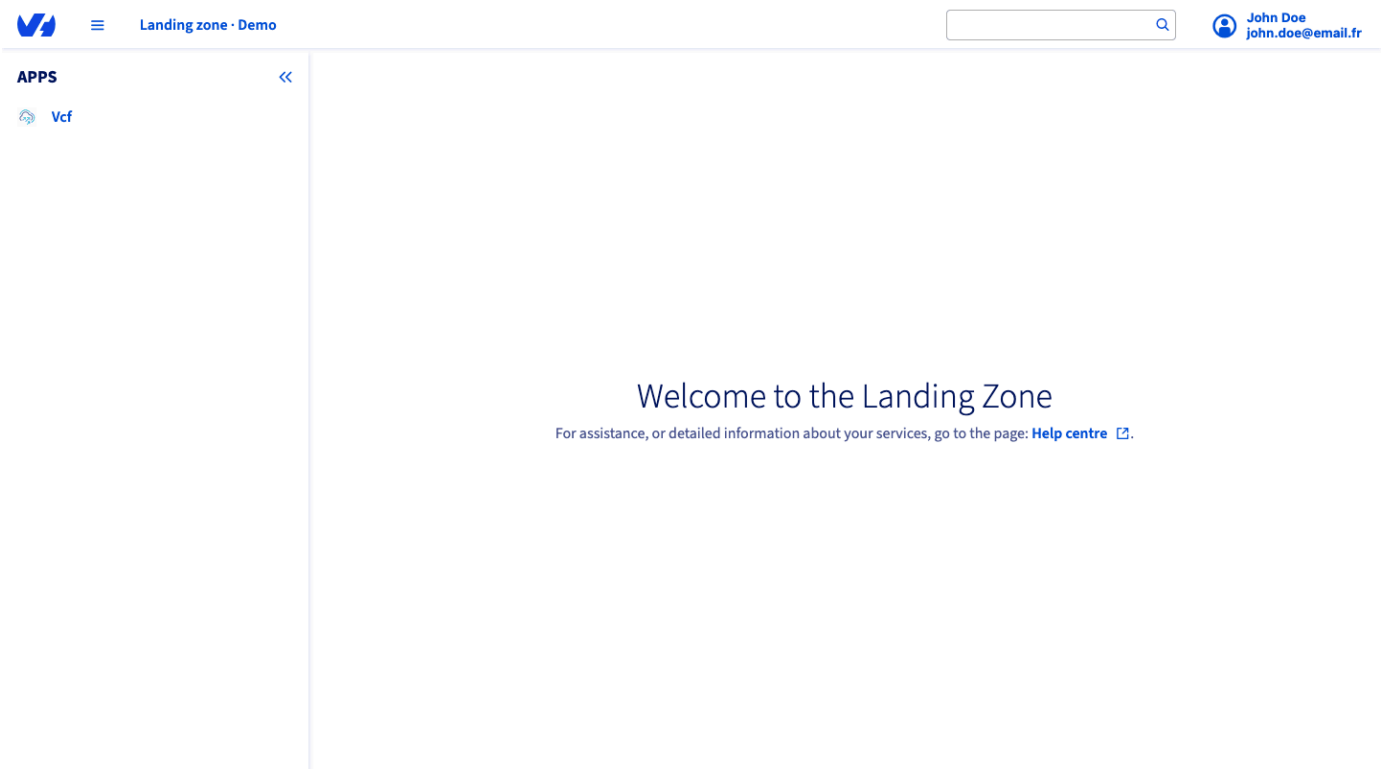
Prérequis

- Disposer d'un accès au Landing Zone Manager avec les privilèges d'administrateur permettant la gestion des comptes utilisateurs
- Connaître le mot de passe par défaut de première connexion défini lors du déploiement de votre Landing Zone Manager
- Disposer de l'URL du Landing Zone Manager (communiquée au préalable par votre administrateur)
- Disposer des informations utilisateur à provisionner (nom complet et adresse e-mail)

En pratique

Étape 1 : Accéder à la section de gestion des comptes utilisateurs

Connectez-vous au Landing Zone Manager avec un compte disposant des privilèges d'administrateur. Depuis la navigation principale, ouvrez la section **Account management**.



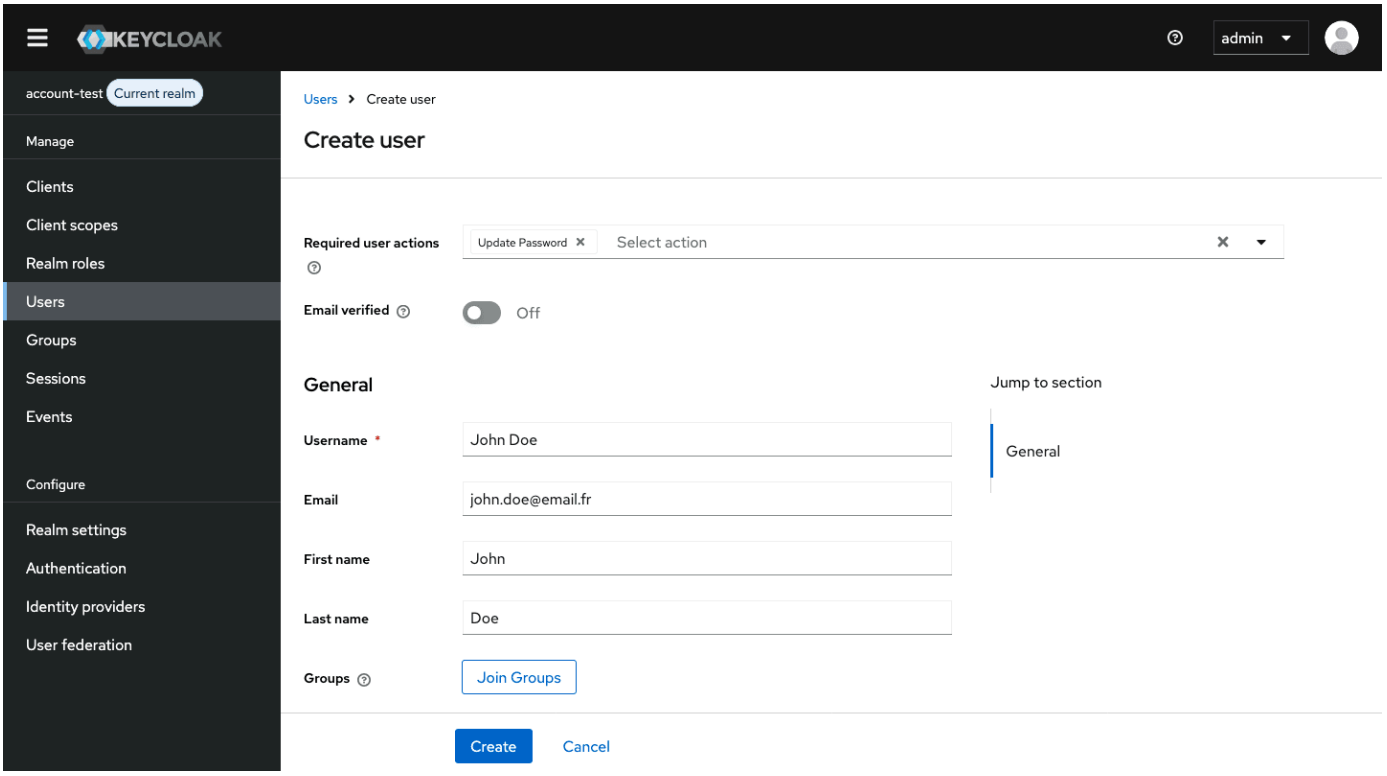
Étape 2 : Créer un nouveau compte utilisateur

Cliquez sur le bouton **+ Create new account** pour ouvrir le formulaire de création de compte utilisateur.

Renseignez les champs obligatoires suivants :

Champ	Description
Name	Le nom d'affichage du compte utilisateur
Email	L'adresse e-mail de l'utilisateur. Cette valeur sera également utilisée comme identifiant de **connexion**
First name	Le prénom de l'utilisateur
Last name	Le nom de famille de l'utilisateur

Une fois tous les champs remplis, confirmez la création du compte utilisateur.



> **Info**
>
> L'adresse e-mail fournie est utilisée comme identifiant de connexion du nouveau compte utilisateur. Assurez-vous qu'elle est correcte avant de valider, car les utilisateurs s'authentifieront avec cette valeur.

Étape 3 : Communiquer les identifiants de première connexion

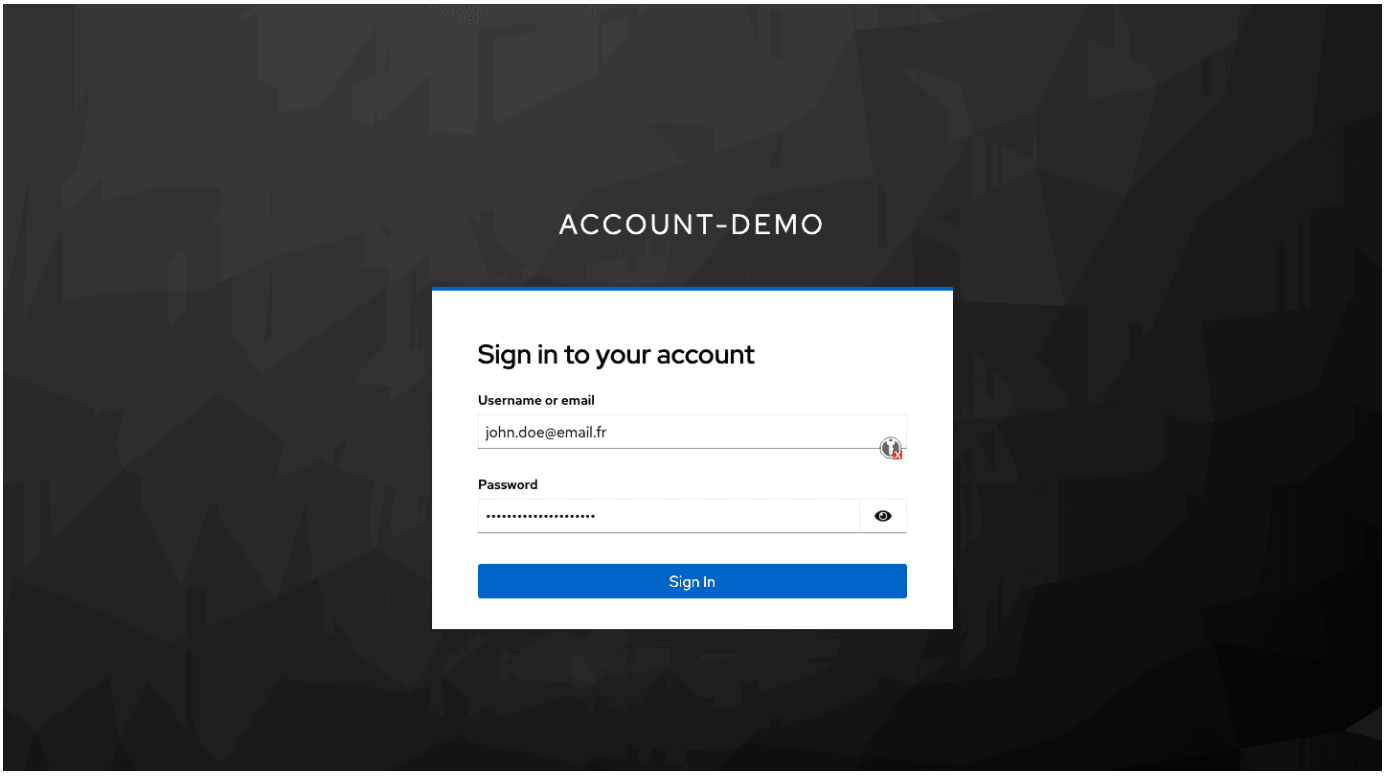
Une fois le compte utilisateur créé avec succès, communiquez l'URL du Landing Zone Manager à l'utilisateur ainsi que ses identifiants de première connexion :

- **Login** : l'adresse e-mail saisie lors de la création
- **Mot de passe** : le mot de passe par défaut de première connexion défini lors du déploiement d'OPCP et du CloudStore

> **Warning**
>
> Le mot de passe par défaut est partagé entre toutes les premières connexions et est défini au moment du déploiement. Pour des raisons de sécurité, il sera demandé aux utilisateurs de le changer immédiatement après leur première connexion.

Étape 4 : Première connexion au Landing Zone Manager

Une fois le compte utilisateur provisionné, l'utilisateur peut se connecter au Landing Zone Manager en utilisant l'URL communiquée précédemment.



Depuis la page de connexion :

1. Saisissez l'adresse e-mail utilisée lors de la création du compte comme identifiant.
2. Saisissez le mot de passe par défaut de première connexion.
3. Validez pour accéder au Landing Zone Manager.

Lors de cette première connexion, l'utilisateur doit définir un nouveau mot de passe et compléter les étapes d'authentification supplémentaires requises par la plateforme avant d'accéder au Landing Zone Manager.

Correspondance des comptes dans Keycloak

> **Info**
>
> Le Landing Zone Manager repose sur sa **propre stack Keycloak dédiée**, indépendante du Keycloak OPCP Core et de tout Keycloak CloudStore. Il n'est pas fédéré avec ces instances : les identités, realms et identifiants gérés dans le Landing Zone Manager sont totalement isolés des autres couches d'identité OPCP dédiées aux administrateurs.

Chaque compte créé dans le Landing Zone Manager génère automatiquement un **realm dédié dans Keycloak**. Cette conception garantit :

- une configuration indépendante des flux d'authentification par compte
- des bases d'utilisateurs et des mappings de rôles séparés par realm

> **Info**
>
> Comme chaque compte correspond à un realm Keycloak distinct, toute opération liée à l'identité (ajout d'utilisateurs, configuration de la fédération, définition de rôles) doit être effectuée au sein du realm associé au compte cible.

Aller plus loin

Pour une formation ou une assistance technique sur la mise en œuvre de nos solutions, contactez votre commercial ou consultez la page Professional Services pour obtenir un devis et faire analyser votre projet par nos experts.

Échangez avec notre communauté d'utilisateurs.

Ressources additionnelles

Matrice de compatibilité OPCP Core

Découvrez les images OS et les plateformes validées pour le provisionnement bare metal OPCP Core — fonctionnalités Ironic, qualification matérielle et documentation disponible

Objectif

OPCP Core offre du provisionnement bare metal via OpenStack Ironic. Parce que cela nécessite un support matériel spécifique et des versions de firmware validées, toutes les images OS ne fonctionnent pas sans configuration particulière — et des mises à jour de firmware peuvent rompre la compatibilité.

Ce guide est une **matrice de compatibilité publique** recensant les plateformes entièrement qualifiées sur OPCP Core. Pour chacune, il documente :

- Si la plateforme est **disponible à la livraison** (provisionnée directement par OVHcloud sans image personnalisée).
- Quelles **fonctionnalités Ironic** sont supportées : RAID logiciel, bonding Active/Backup, LACP (802.3ad).
- Si le **trunking Neutron** est disponible après installation.
- Des liens vers la **documentation de construction d'image** correspondante.

Matrice de compatibilité

Le tableau ci-dessous liste les plateformes dont le statut est **Ready** — celles qui ont été entièrement qualifiées sur le matériel OPCP Core.

Images intégrées

Plateforme	RAID logiciel Ironic	Bonding Active/Backup Ironic	LACP Ironic (802.3ad)	Trunking Neutron
Debian 12	Oui	Oui	Oui	Oui
Debian 13	Oui	Oui	Oui	Oui
CentOS Stream 9	Oui	Oui	Oui	Oui

Instructions de construction d'image

Plateforme	Guide de construction d'image	RAID logiciel Ironic	Bonding Active/Backup Ironic	LACP Ironic (802.3ad)	Trunking Neutron
Debian 12	Guide	Non	Oui	Oui	Oui
Debian 13	Guide	Non	Oui	Oui	Oui

Détail des fonctionnalités

RAID logiciel Ironic

Le RAID logiciel est configuré par Ironic lors du provisionnement bare metal. Lorsqu'il est activé, l'OS est déployé sur plusieurs disques (généralement RAID-1 pour des configurations à deux disques), offrant de la redondance de stockage local sans nécessiter de contrôleur RAID matériel.

> **Warning**
> Le RAID logiciel avec Ironic nécessite que GRUB soit compilé avec les modules `mdadm`. Cela doit être pris en compte lors de la construction d'une image OS personnalisée.

Bonding Active/Backup Ironic

Ironic peut configurer un bonding réseau Active/Backup lors du provisionnement. Cela assure une redondance de liens : une interface réseau est active tandis que l'autre est en attente, prête à prendre le relais automatiquement en cas de défaillance de l'interface primaire. Aucune configuration n'est requise sur le commutateur en amont.

LACP Ironic (802.3ad)

Le bonding LACP (IEEE 802.3ad) agrège deux interfaces réseau en un seul lien logique, offrant à la fois de la redondance et un débit accru. Ce mode requiert que le commutateur en amont supporte et soit configuré pour LACP.

Pour les instructions pas à pas, consultez le guide « Comment configurer LACP sur un nœud ».

Trunking Neutron

Le trunking Neutron permet à une instance bare metal de transporter plusieurs réseaux Neutron sur un seul port physique via le marquage VLAN. Il est configuré après l'installation via OpenStack Neutron, et non pendant la phase de provisionnement Ironic.

> **Info**
> OpenStack ne configure pas nativement le trunking VLAN via cloud-init. L'activation du trunking nécessite des étapes de configuration post-installation détaillées dans le guide « Comment configurer le trunk sur un nœud ».

Aller plus loin

- Comment installer une instance via la CLI OpenStack
- Comment configurer LACP sur un nœud
- Comment configurer le trunk sur un nœud
- Comment configurer le Softraid sur un nœud

Pour bénéficier d'une formation ou d'une assistance technique pour la mise en œuvre de nos solutions, contactez votre interlocuteur commercial ou consultez la page Professional Services pour obtenir un devis et faire analyser votre projet par nos experts.

Échangez avec notre communauté d'utilisateurs.

Fonctionnalités et spécifications d'Object Storage sur OPCP

Découvrez les spécifications d'Object Storage pour OVHcloud OPCP, avec informations sur l'accès aux buckets, les classes de performance, le chiffrement et le versioning.

> **Info**
>
> Veuillez noter que ces informations peuvent évoluer avant la disponibilité générale (GA). Selon nos estimations actuelles, la GA est prévue pour **mars 2026**. Nous nous engageons à vous tenir informés de toute modification, mais nous vous recommandons de consulter régulièrement ce guide pour obtenir les informations les plus récentes. Merci de votre compréhension.

Objectif

OVHcloud Object Storage dans **OPCP** fournit une solution évolutive, durable et à faible latence pour le stockage et l'accès aux données non structurées. Conçue pour répondre aux besoins de diverses applications, elle offre une plateforme robuste pour le stockage de données avec des optimisations spécifiques à OPCP afin d'assurer des performances et une disponibilité élevées.

Comment est géré l'accès au bucket ?

Toutes les clés d'accès d'un projet peuvent accéder à tous les buckets dans OPCP.

Classes de performance

- **Standard** : les classes de performance dans OPCP sont identiques aux performances standard d'OVHcloud, garantissant la cohérence dans le traitement des données et les vitesses d'accès.
- **High Performance** : les classes de stockage High Performance ne sont pas disponibles dans OPCP pour le moment. Seules les classes Standard sont prises en charge.

Le support des classes High Performance dans OPCP pourra être envisagé à l'avenir par OVHcloud.

Fonctionnalités

Les fonctionnalités d'Object Storage dans OPCP sont principalement les mêmes que dans les régions OVHcloud. Certaines différences importantes sont détaillées dans le tableau comparatif de ce guide.

Spécifications MultiAZ et MonoAZ

MultiAZ (Zone de multi-disponibilité)

- **Vue générale** : MultiAZ assure haute disponibilité et redondance en répartissant les données sur plusieurs zones de disponibilité au sein d'une même région, garantissant l'accès aux données même en cas de panne d'une zone.
- **Disponibilité** : les configurations MultiAZ sont disponibles dans les régions mais ne sont pas supportées dans OPCP, qui se concentre sur un accès à faible latence dans une zone unique.

MonoAZ (Zone de mono-disponibilité)

- **Vue générale** : MonoAZ stocke les données dans une seule zone de disponibilité, offrant **latence réduite et performances élevées** pour les applications ne nécessitant pas la redondance MultiAZ.
- **Disponibilité** : MonoAZ est pris en charge dans OPCP, adapté aux cas d'usage privilégiant performances et faible latence.

API de métadonnées (indisponible)

- **Vue générale** : l'API de métadonnées, qui permet de récupérer des informations sur le stockage, n'est pas disponible dans OPCP. Cela inclut le nombre de buckets et la taille totale via l'espace client ou l'API.
- **Impact** : les utilisateurs ne peuvent pas suivre ou gérer leur utilisation de stockage par programme dans OPCP.

Nos équipes travaillent à l'intégration future de cette API.

Chiffrement et versioning

Chiffrement

- **État actuel** : le chiffrement au repos n'est pas pris en charge dans OPCP.
- **Chiffrement pris en charge** : le chiffrement côté client (SSeC) est disponible, permettant aux utilisateurs de chiffrer leurs données avant envoi au service Object Storage.

Versioning

- **État actuel** : le versioning est pris en charge par l'API, permettant la gestion de plusieurs versions d'un objet dans un bucket.
- Une intégration plus complète du versioning sera proposée dans les futures mises à jour de l'API.

Stratégies utilisateur (indisponibles)

- **Vue générale** : les *user policies* peuvent théoriquement contrôler les accès, mais elles n'affectent pas le modèle OPCP.
- **Impact** : toutes les clés d'accès d'un projet ont un accès illimité à tous les buckets OPCP, ce qui peut ne pas répondre à certaines exigences de sécurité.

Des améliorations pour la gestion des stratégies utilisateur sont prévues dans de futures mises à jour.

Aller plus loin

Si vous avez besoin d'une formation ou d'une assistance technique pour la mise en oeuvre de nos solutions, contactez votre commercial ou cliquez sur ce lien pour obtenir un devis et demander une analyse personnalisée de votre projet à nos experts de l'équipe Professional Services.

Échangez avec notre communauté d'utilisateurs.

Création d'une image Debian personnalisée pour OPCP

Découvrez comment créer votre propre image Debian pour On-Prem Cloud Platform

Objectif

Ce guide explique comment construire des images disque Debian personnalisées pour OpenStack Ironic (bare metal) et Nova (machine virtuelle) en utilisant diskimage-builder.

C'est un bon point de départ pour comprendre le fonctionnement de diskimage-builder (DiB) avec un exemple pratique. Nous allons construire une image Debian 13 personnalisée à partir de l'image parent, puis la personnaliser avec Ansible.

À la fin, nous générerons une image disque complète `debian13.qcow2` (incluant le noyau et l'initramfs), prête à être importée dans OpenStack Glance.

Attentes pour OpenStack Ironic/Nova

Format de l'image

- **qcow2** (recommandé) : image disque compressée
- **raw** : image disque non compressée

L'image doit être une **image disque complète** qui inclut :

- Le secteur de boot / la partition système EFI
- La ou les partitions du système d'exploitation
- Le noyau et l'initramfs présents sur le disque

Partitionnement et exigences de démarrage

Pour un démarrage BIOS :

- Table de partition GPT ou MBR
- Partition de démarrage BIOS (1–2 Mo, type `ef02` pour GPT)
- Partition racine avec un chargeur de démarrage installé (GRUB2)
- Le chargeur de démarrage doit être installé dans le MBR / secteur de boot

Pour un démarrage EFI :

- Table de partition GPT obligatoire
- Partition système EFI (ESP) : 512 Mo, FAT32, montée sur `/boot/efi`
- Partition racine avec GRUB2 en mode EFI
- Entrées de démarrage EFI correctement configurées

Pour Ironic en bare metal (configuration recommandée) :

Schéma de partition standard (simple) :

```
# GPT partition table
- EFI System Partition (ESP): 512MiB, type EF00, FAT32, mounted at /boot/efi
- BIOS Boot Partition (BSP): 8MiB, type EF02 (for hybrid BIOS/EFI compatibility)
- Root partition: remaining space, type 8300, ext4, mounted at /
```

Prérequis

Pré-requis système :

- Droits root
- Au moins 10 Go d'espace disponible
- Quelques paquets nécessaires :

```
# Debian/Ubuntu
apt update -y && apt install -y \
  dosfstools \
  python3 \
  python3-venv \
  python3-pip \
  virtualenv \
  kpartx \
  debootstrap \
  lvm2 \
  squashfs-tools \
  qemu-utils
```

Installer diskimage-builder (DiB) dans un virtualenv :

```
mkdir -p diskimage-builder
python3 -m venv venv
. ./venv/bin/activate
pip install diskimage-builder
```

Comprendre les éléments diskimage-builder

Les *elements* sont des composants modulaires qui permettent de personnaliser votre image.

Dossier	Objectif
`environment.d/`	Variables d'environnement
`extra-data.d/`	Fichiers ajoutés avant l'installation
`pre-install.d/`	Scripts de pré-installation
`post-install.d/`	Scripts de post-installation
`finalise.d/`	Personnalisation finale
`cleanup.d/`	Tâches de nettoyage
`package-installs.yaml`	Déclarations de paquets
`block-device-default.yaml`	Partitionnement du disque

Les scripts dans ces répertoires sont exécutés dans l'ordre numérique / alphabétique.

Créer un élément de personnalisation avec Ansible

Créez un *element* qui utilise Ansible pour la personnalisation :

Structure de répertoires :

```
mkdir -p elements/os-custom/extra-data.d/ansible/{roles/customize/tasks,inventory/host_vars/sys_image}
mkdir -p elements/os-custom/extra-data.d/ansible/roles/customize/files
mkdir -p elements/os-custom/post-install.d
```

Scripts post-install :

`elements/os-custom/post-install.d/00-install-ansible` :

```
#!/bin/bash
set -eux
apt install --yes ansible
```

`elements/os-custom/post-install.d/01-apply-ansible` :

```
#!/bin/bash
set -eux
export LANG=C.UTF-8
export LC_ALL=C.UTF-8
cd /tmp/in_target.d/extra-data.d/ansible
ANSIBLE_STDOUT_CALLBACK=debug ansible-playbook -i inventory main.yml
```

`elements/os-custom/post-install.d/02-remove-ansible` :

```
#!/bin/bash
set -eux
apt remove --yes ansible
apt autoremove --yes
```

Rendre les scripts exécutables :

```
chmod +x elements/os-custom/post-install.d/*
```

Configuration Ansible :

`elements/os-custom/extra-data.d/ansible/main.yml` :

```
---
- hosts: all
  gather_facts: true
  roles:
    - name: customize
```

`elements/os-custom/extra-data.d/ansible/inventory/hosts` :

```
[all]
sys_image ansible_connection=local ansible_become=no
```

Dans cet exemple, nous utilisons `cloud-init` avec le moteur `netplan` pour configurer `systemd-networkd`. Il s'agit d'un exemple fonctionnel de personnalisation de la configuration réseau ; adaptez-le librement à vos besoins.

Note : lorsque Ironic configure le bare metal au premier démarrage, il fournit un manifeste `network_metadata` (configdrive) que `cloud-init` peut interpréter pour configurer automatiquement le réseau (IP statique, LACP, etc.).

`elements/os-custom/extra-data.d/ansible/roles/customize/tasks/main.yml` :

```
---
- name: Install additional packages
  apt:
    name:
      - cloud-init
    state: latest
    update_cache: yes

- name: Remove unwanted packages
  apt:
    name:
      - ifupdown
      - ifenslave
      - vlan
    state: absent
    purge: yes

# Allow Baremetal LACP auto-conf from Neutron
- name: Configure cloud-init for netplan
  copy:
    src: 50-netplan.cfg
    dest: /etc/cloud/cloud.cfg.d/50-netplan.cfg
    owner: root
    group: root
    mode: 0644
```

`elements/os-custom/extra-data.d/ansible/roles/customize/files/50-netplan.cfg` :

```
system_info:
  network:
    renderers: ['netplan']
```

Paquets additionnels :

`elements/os-custom/package-installs.yaml` :

```
man:
nano:
tcpdump:
iputils-ping:
netplan.io:
```

Dépendances de l'élément :

`elements/os-custom/element-deps` :

```
debian
```

Construire l’image

Créer un fichier d’environnement `debian13.env` :

```
export DIB_RELEASE=trixie
export DIB_CLOUD_INIT_DATASOURCES="ConfigDrive, OpenStack"
export DIB_GRUB_TIMEOUT=10
```

Construire l’image :

```
mkdir -p tmp-build-dir
export TMPDIR="$(pwd)/tmp-build-dir"
export ELEMENTS_PATH="$(pwd)/elements"

source debian13.env
disk-image-create -t qcow2 --image-size 16GB -a amd64 vm block-device-efi os-custom debian -o debian13
```

Vous devriez obtenir un fichier `debian13.qcow2`.

Si vous voulez des fichiers SBOM pour les variables d'environnement et les paquets :

```
cp debian13.d/dib-manifests/dib_environment debian13.env.sbom
cp debian13.d/dib-manifests/dib-manifest-dpkg-debian13 debian13.pkg.sbom
```

Tester l’image

Une façon rapide de tester l’image générée est d’utiliser qemu pour lancer une machine virtuelle à partir de l’image qcow2, puis un client VNC pour se connecter à la console. On peut tester aussi bien le démarrage EFI que BIOS.

Démarrage BIOS

```
qemu-system-x86_64 -enable-kvm -vnc 0.0.0.0:0,password=on -monitor stdio -m 2048 -drive
file=debian13.qcow2,if=virtio,format=qcow2
```

```
# Définir un mot de passe VNC personnalisé
QEMU 10.0.3 monitor - type 'help' for more information
(qemu) change vnc password
Password: ********
```

Utilisez n’importe quel client VNC pour vous connecter sur le port :5200.

Démarrage EFI

```
# Copier les variables OVMF pour éviter de modifier l'original
cp /usr/share/OVMF/OVMF_VARS_4M.fd /tmp/debian13-OVMF_VARS_4M.fd

qemu-system-x86_64 -enable-kvm \
  -machine q35,smm=on,accel=kvm \
  -drive if=pflash,format=raw,unit=0,file=/usr/share/OVMF/OVMF_CODE_4M.fd,readonly=on \
  -drive if=pflash,format=raw,unit=1,file=/tmp/debian13-OVMF_VARS_4M.fd \
  -vnc 0.0.0.0:0,password=on \
  -monitor stdio \
  -m 2048 \
  -drive file=debian13.qcow2,if=virtio,format=qcow2
```

```
# Définir un mot de passe VNC personnalisé
QEMU 10.0.3 monitor - type 'help' for more information
(qemu) change vnc password
Password: ********
```

Utilisez un client VNC pour vous connecter sur le port :5200.

Importer l'image dans OpenStack

```
openstack image create \
  --disk-format qcow2 \
  --container-format bare \
  --file debian13.qcow2 \
  debian13
```

C'est terminé ! Vous pouvez maintenant créer une instance bare metal ou une machine virtuelle avec cette nouvelle image.

Stockage Block Ceph RBD : Performance, Résilience et Scalabilité avec OpenStack

Découvrez le stockage Block Ceph RBD d'OVHcloud OPCP : performance, haute disponibilité et fonctionnalités avancées pour vos environnements OpenStack.

> **Info**
>
> Veuillez noter que ces informations peuvent évoluer avant la disponibilité générale (GA). Selon nos estimations actuelles, la GA est prévue pour **mars 2026**. Nous nous engageons à vous tenir informés de toute modification, mais nous vous recommandons de consulter régulièrement ce guide pour obtenir les informations les plus récentes. Merci de votre compréhension.

Objectif

Ce guide présente l'architecture de stockage Block mise en œuvre dans **OPCP**, basée sur OpenStack. Il explique comment le stockage persistant des machines virtuelles (VM) est assuré par **Ceph RADOS Block Device** (RBD), une solution open-source, distribuée et hautement scalable, conçue pour répondre aux besoins des entreprises modernes.

L'objectif est de montrer comment cette couche de stockage garantit performance, résilience et scalabilité, permettant aux équipes de se concentrer sur le développement et le déploiement d'applications sans être limitées par l'infrastructure.

1. Performance et Scalabilité : conçues pour évoluer

Contrairement aux solutions de stockage traditionnelles, l'infrastructure Ceph répartit les données et les charges de travail sur l'ensemble des nœuds physiques du cluster.

Cette approche élimine les goulots d'étranglement et permet une scalabilité linéaire adaptée aux environnements exigeants.

Fonctionnalité	Avantage client
Entrées/Sorties distribuées	**Élimination des goulots d'étranglement :** les opérations de lecture et d'écriture sont simultanément réparties sur de nombreux périphériques. Cette répartition assure un **débit élevé** et une **latence faible et constante**, indispensables aux applications et bases de données critiques.
Scalabilité horizontale	**Croissance sans interruption :** l'augmentation de la capacité de stockage se fait simplement en ajoutant des serveurs standards au cluster. Le système intègre automatiquement le nouveau matériel et rééquilibre les données en arrière-plan, **sans provoquer de temps d'arrêt**.
Provisionnement dynamique	**Efficacité optimisée :** les volumes sont alloués instantanément mais n'utilisent de l'espace physique que lorsque les données sont réellement écrites. Cette approche améliore considérablement l'**utilisation et l'efficacité** du pool de stockage physique.

2. Résilience et protection des donnée

L'architecture fondamentale de Ceph est conçue pour offrir une **tolérance maximale aux pannes**, garantissant que les défaillances matérielles n'entraînent ni interruption de service ni perte de données.

Réplication automatique (Haute disponibilité) :

- Chaque donnée est automatiquement copiée et stockée sur plusieurs disques et serveurs physiques distincts (typiquement 3 copies).
- En cas de panne d'un disque ou d'un serveur, les machines virtuelles continuent de fonctionner normalement, accédant aux copies restantes sans interruption.

Auto-réparation :

- Lorsqu'un périphérique tombe en panne, Ceph détecte immédiatement l'anomalie et réplique les données perdues sur les dispositifs sains restants.
- Ce processus restaure automatiquement le **niveau de protection des données défini**.

Absence de point de défaillance unique :

- Le système fonctionne sans contrôleur central ni composant matériel propriétaire, garantissant une **redondance réelle** et un **temps de disponibilité maximal** pour les services cloud.

3. Spécifications techniques et limites

Cette section détaille les principales caractéristiques techniques et exigences de compatibilité pour les utilisateurs déployant des applications sur la plateforme OVHcloud OPCP.

A. Compatibilité et accès

Élement	Détail	Description
Type de stockage	Stockage Block	Présenté à la VM comme un périphérique block SCSI ou virtio standard, équivalent à un disque physique pour des performances optimales.
Compatibilité POSIX	Oui (via OS invité)	Ceph RBD fournit le périphérique block brut ; le système de fichiers du système invité (XFS, ext4, NTFS, etc.) garantit la conformité POSIX.
Système de fichiers réseau	Non	Il ne s'agit pas d'un NFS ou SMB partagé. Chaque volume est dédié à une seule VM à un instant donné pour des performances maximales.

B. Limites et dimensionnement

Métrique	Valeur/par défaut	Notes
Taille maximale d'un volume	16 To	Limite pour assurer performance optimale et facilité de gestion sous OpenStack. Des pools personnalisés peuvent supporter des tailles supérieures sur demande.
Taille minimale d'un volume	1 Go	Plus petite unité de volume déployable via Cinder.
Disponibilité du volume	Instantanée	L'allocation et l'attachement des volumes se font immédiatement via l'API ou le tableau de bord OpenStack.

C. Performance et qualité de service (QoS)

La plateforme propose plusieurs **niveaux de performance** via les types de volumes Cinder (Storage Classes), avec des limites I/O garanties par le système QoS de Ceph RBD :

Classe de stockage	IOPS	Débit	Cas d'usage
Stockage distribué (Standard)	1 000 / 1 000	100 MB/s	VM générales, serveurs web, environnements de test et développement.
Stockage distribué (High)	3 000 / 3 000	200 MB/s	Bases de données, moteurs d'analytique, applications à fort trafic ou sensibles à la latence.
Stockage distribué (Insane)	7 500 / 7 500	300 MB/s	Workloads extrêmes, systèmes transactionnels critiques, infrastructure de deep learning.

> **Info**
>
> **Note :** ces seuils sont appliqués par le système QoS de Ceph. Les volumes peuvent parfois dépasser ces limites si les ressources système le permettent, mais les garanties assurent une performance prévisible, même en période de forte sollicitation.

4. Fonctionnalités avancées de gestion des données

Le backend **Ceph RBD** offre des outils avancés pour la gestion du cycle de vie des volumes, simplifiant et accélérant les opérations pour les utilisateurs cloud.

- **Snapshots instantanés :** Création de **copies ponctuelles efficaces en espace** de tout volume, permettant un rollback rapide ou la création de nouveaux environnements en quelques secondes.
- **Clonage rapide (Copy-on-Write) :** Provisionnement rapide de nouveaux volumes **lecture/écriture** à partir de snapshots ou d'images existantes. La technologie **CoW (Copy-on-Write)** économise de l'espace disque et accélère le déploiement des VM.
- **Redimensionnement "à chaud" des volumes :** Augmentation de la taille d'un volume attaché à une VM en fonctionnement, **sans éteindre ni redémarrer la VM.**
- **Support de la migration à chaud (Live Migration) :** Le stockage distribué permet la migration non disruptive des machines virtuelles actives entre les nœuds de calcul.
- **Démarrage depuis un volume (Boot from Volume) :** Lancement direct d'une VM à partir d'un volume Cinder, garantissant que le disque racine bénéficie immédiatement de **toutes les fonctionnalités de haute disponibilité et avancées de Ceph RBD.**
- **Prêt pour la reprise après sinistre (RBD Mirroring) :** L'architecture Ceph sous-jacente prend en charge le **mirroring asynchrone des volumes** vers un cluster Ceph distant, offrant une solution robuste pour le **plan de reprise après sinistre (DRP)**.