# Hadoop és Spark referencia architektúra az ELKH Cloudon

Emődi Márk

emodi.mark@sztaki.hu

# Hadoop és Spark referencia architektúra

- Cél
  - Komplex virtuális infrastruktúrák kiépítése automatizált módon az ELKH Cloud-on. Az alábbi kritériumoknak magas minőségben való megfelelés:
    - Elosztott
    - Hibatűrő
    - Biztonságos
    - Skálázható
  - Egyszerűsített felhasználás tesztrendszer és élesrendszer felépítésére
    - Minimális LINUX és felhő ismereti tudás szükséges!

# Occopus

- SZTAKI PERL labor által fejlesztett nyílt forráskódú hibrid orkesztrációs eszköz
- „OCCO” – „One Click Cloud Orchestrator”
- Orkesztráció: virtuális gépek (VM) és infrastruktúrák rugalmas létrehozása és menedzselése felhő rendszerekben (előtérbe kerül IaaC, IaaS rétegben)
- Kontextualizáció: gépek személyre szabása
  - Szoftvercsomagok telepítése, konfigurációja, finomhangolása
- Hordozható leírók
- Skálázási lehetőség
- Felhőfüggetlen megoldás

ELKH Cloud

openstack CLOUD SOFTWARE

CloudSigma

amazon web services

Microsoft Azure

CloudBroker

Open Nebula

# Occopus leírók

Infrastruktúra leíró (infrastructure description)

- Csomópontok
- Változók
- Skálázás
- Függőségek

```
infra_name: spark-cluster
user_id: somebody@somewhere

nodes:
  - &M
    name: spark-master
    type: spark_master_node
  - &W
    name: spark-worker
    type: spark_worker_node
    scaling:
      min: 2
      max: 10

variables:
  HADOOP_VERSION: 3.3.0
  SPARK_VERSION: 3.0.1
  SPARK_HADOOP_VERSION: 3.2
  CONSUL_VERSION: 1.9.3
  CONSUL_TEMPLATE_VERSION: 0.25.1

dependencies:
  -
    connection: [ *W, *M ]
```

# Occopus leírók

Infrastruktúra leíró (infrastructure description)

- Csomópontok
- Változók
- Skálázás
- Függőségek

Csomópont leíró (Node definition)

- Erőforrás definiálás
- Kontextualizáció
- Egészség-ellenőrzés
- Konfigurációs menedzsment

```
'node_def:spark_master_node':
  -
    resource:
      type: nova
      endpoint: replace_with_endpoint_of_nova_interface_of_your_cloud
      project_id: replace_with_projectid_to_use
      user_domain_name: Default
      image_id: replace_with_id_of_your_image_on_your_target_cloud
      network_id: replace_with_id_of_network_on_your_target_cloud
      flavor_name: replace_with_id_of_the_flavor_on_your_target_cloud
      key_name: replace_with_name_of_keypair_or_remove
      security_groups:
        - replace_with_security_group_to_add_or_remove_section
      floating_ip: add_yes_if_you_need_floating_ip_or_remove
      floating_ip_pool: replace_with_name_of_floating_ip_pool_or_remove
    contextualisation:
      type: cloudinit
      context_template: !yaml_import
        url: file://cloud_init_spark_master.yaml
    health_check:
      ports:
        - 8080
      timeout: 1000
...
```

```
...
'node_def:spark_worker_node':
  -
    resource:
      type: nova
      endpoint: replace_with_endpoint_of_nova_interface_of_your_clou
      project_id: replace_with_projectid_to_use
      user_domain_name: Default
      image_id: replace_with_id_of_your_image_on_your_target_cloud
      network_id: replace_with_id_of_network_on_your_target_cloud
      flavor_name: replace_with_id_of_the_flavor_on_your_target_clou
      key_name: replace_with_name_of_keypair_or_remove
      security_groups:
        - replace_with_security_group_to_add_or_remove_section
    contextualisation:
      type: cloudinit
      context_template: !yaml_import
        url: file://cloud_init_spark_worker.yaml
    health_check:
      ping: False
```

# Occopus leírók

Infrastruktúra leíró (infrastructure description)

- Csomópontok
- Változók
- Skálázás
- Függőségek

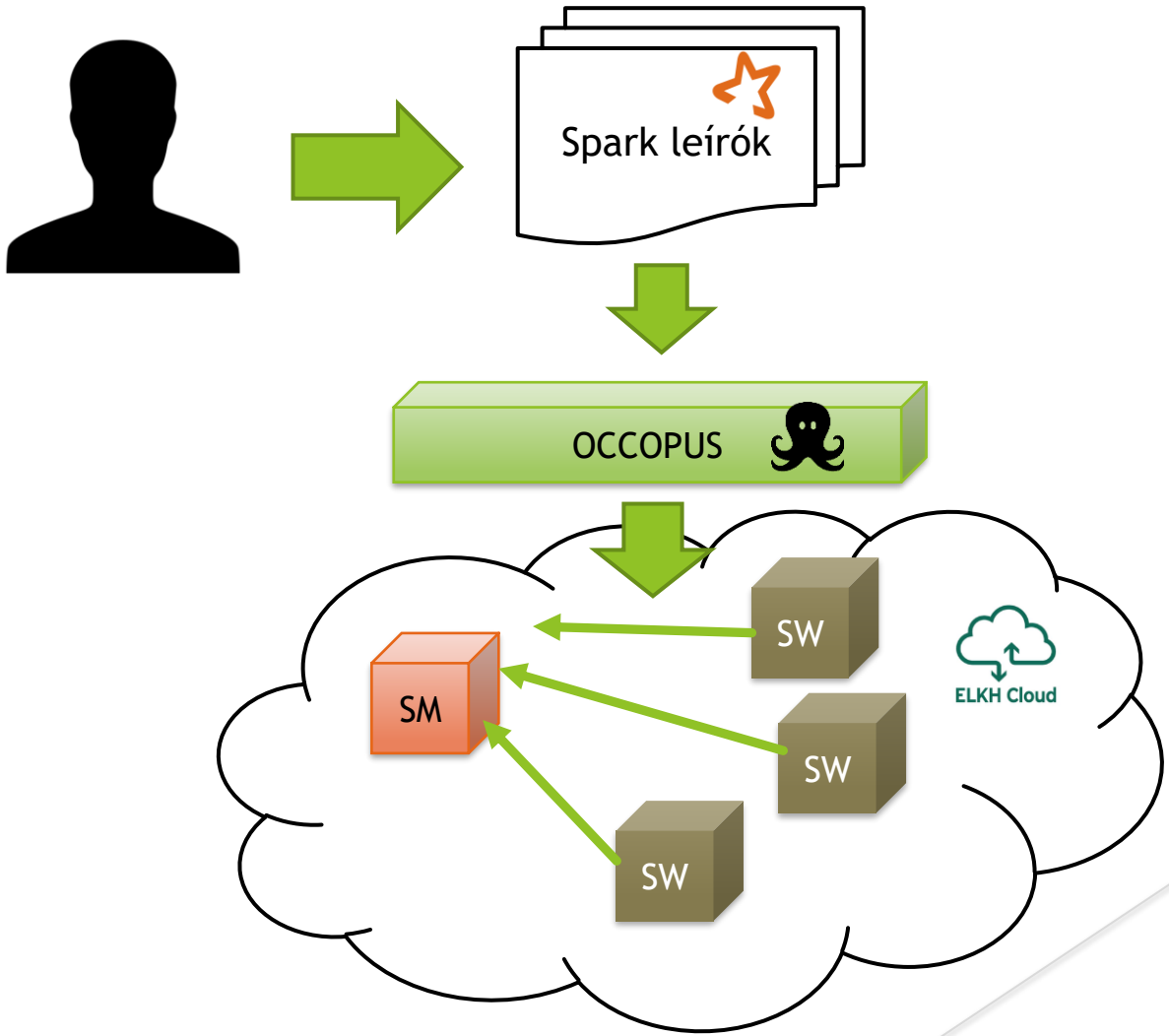
Csomópont leíró (Node definition)

- Erőforrás definiálás
- Kontextualizáció
- Egészség-ellenőrzés
- Konfigurációs menedzsment

Cloud-init fájlok

- A virtuális gép konfiguráláshoz szükséges lépések:
  - Felhasználó kezelés
  - Komponensek telepítése/beállítása
  - Szolgáltatások telepítése/konfigurálása/indítása
  - Szolgáltatások elindítása

# A megoldás architektúrája



# Megoldás használatának lépései

**Felhasználó feladatköre:**

0. Lépés: **Előkészítés** (ELKH Cloud projekt, Üres Ubuntu VM elindítás)

1. Lépés: **Occopus telepítés/konfigurálás**

2. Lépés: **Leírók letöltése a virtuális gépre**
Occopus/ELKH Cloud weboldala
3. Lépés: **Tűzfalszabályok létrehozása**

ELKH Cloud OpenStack felületén

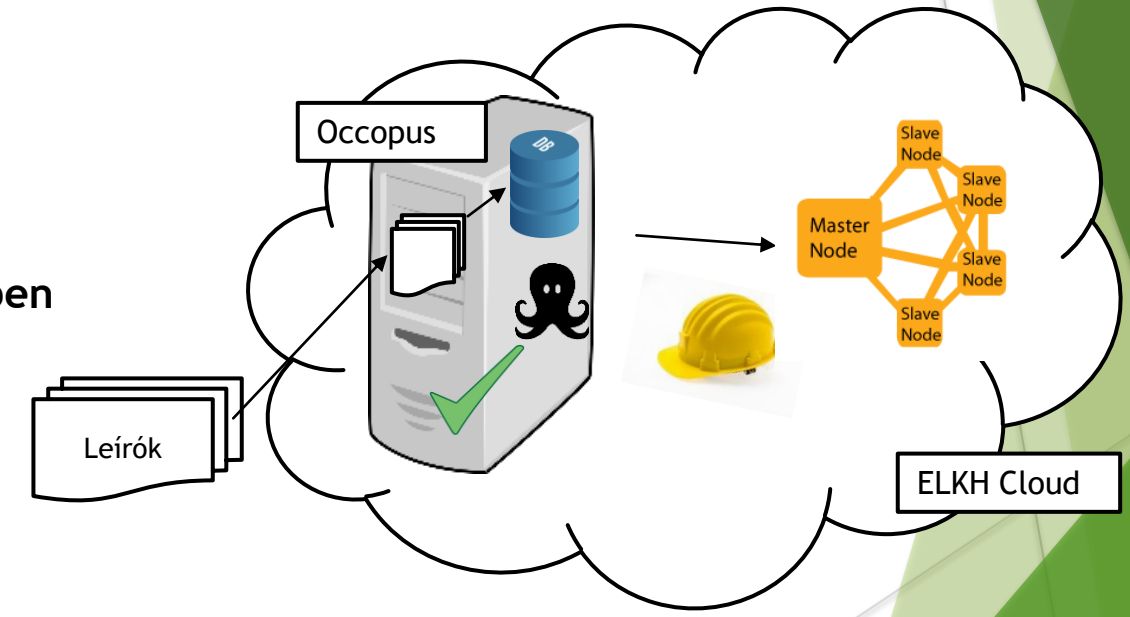
4. Lépés: **Leírók személyre szabása a virtuális gépen**
5. Lépés: **Occopus aktiválása**
    $ source ~/occopus/bin/activate
6. Lépés: Leírók importálása Occopus számára

    $ occopus-import nodes/node_definitions.yaml

7. Lépés: **Infrastruktúra kiépítése**

    $ occopus-build --parallelize infra-spark-cluster.yaml

8. Lépés: **Infrastruktúra használata**



# A megoldás használata

1. Occopus telepítése
2. Leírók letöltése
3. Tűzfalszabályok létrehozása
4. Leírók személyre szabása
5. Occopus aktiválása
6. Leírók importálása
7. Klaszter kiépítése
8. Infrastruktúra használata

**1x** (Aktuális VM-en az aktuális referencia architektúra esetén)

**egy-egy sor kód**

Rövid és hosszú felhasználás is támogatott

# 1. lépés: Occopus telepítése

ELKH Cloud

```
# Rendszerszintű dependenciák telepítése:
$ sudo apt update && \
sudo apt install -y python3-pip virtualenv redis-server libssl-dev

# Virtuális környezet előkészítése:
$ virtualenv -p python3 $HOME/occopus
$ source $HOME/occopus/bin/activate

# Occopus telepítése
$ pip install --no-index --find-links https://pip3.lpds.sztaki.hu/packages OCCO_API

# Occopus alapértelmezett konfigurációjának letöltése
$ mkdir -p $HOME/.occopus
$ curl https://raw.githubusercontent.com/occopus/docs/master/tutorials/.occopus/occopus_config.yaml
-o $HOME/.occopus/occopus_config.yaml

# Occopus alapértelmezett authentikációs fájljának letöltése
$ curl https://raw.githubusercontent.com/occopus/docs/master/tutorials/.occopus/auth_data.yaml -o $HOME/.occopus/auth_data.yaml
```

Telepítési leírás lépésről-lépésre:
https://occopus.readthedocs.io/en/latest/user-doc-setup.html

# 1. lépés: Occopus telepítése – Authentikáció beállítása

A felhasználóknak helyesen kell beállítaniuk a hitelesítési információkat a használni kívánt erőforrás hitelesítési fájljukban ( nano $HOME/.occopus/auth_data.yaml ).

For `nova` resources:

In case of username/password authentication:

```
resource:
    -
        type: nova
        auth_data:
            username: your_username
            password: your_password
```

https://occopus.readthedocs.io/en/latest/user-doc-setup.html?highlight=auth_data#authentication

# 2. lépés: leírók letöltése

1


## Apache Spark klaszter Python stack-el

2

**Áttekintés:**

Ez a bemutató áttekintés ad arról, hogy hogyan lehet létrehozni egy skálázható Apache Spark infrastruktúrát az Occopus eszköz segítségével. Az Apache Spark egy gyors és általános célú klaszter keretrendszer. Magas szintű API-kat biztosít Java, Scala, Python és R programnyelvekhez. Továbbá számos magas szintű eszközt támogat, többet között a Spark SQL-t a strukturált adatfeldolgozáshoz, MLlib-et a gépi tanuláshoz, GraphX-et a gráf feldolgozáshoz, és Spark Streaming-et a nagy mennyiségű adatok valós idejű feldolgozásához. További információkért látogasson el az Apache Spark hivatalos weboldalára.

Az Apache Spark klaszter a HDFS-el (Hadoop Distributed File System) együtt a Big Data és a a gépi tanulási alkalmazások egyik legfontosabb eszköze, amely lehetővé teszi a nagy adatállományok párhuzamos feldolgozását több virtuális gépen, amelyek a Spark Workerek. Azonban, egy Spark klaszter létrehozása a HDFS-el a felhőben nem egyszerű, a felhő rendszerek és az Apache Spark architektúrájának mély ismeretét igényli. Azért, hogy a kutatókat megóvjuk ettől a munkától, létrehoztuk és közzétettük azokat a szükséges infrastruktúra leírókat, amelyek segítségével az Occopus automatikusan építi a Spark klasztert, a felhasználó által megadott Workerek számával. A Spark egy "MLlib" nevű speciális könyvtárat biztosít a gépi tanulási alkalmazások támogatására. Hasonlóképpen, az R-orientált Spark környezethez, kifejlesztettük az infrastruktúra-leírókat a gépi tanulási környezet létrehozásához a felhőben. Itt, a programozási nyelv a Python és a felhasználói programozási környezet a Jupyter Notebook. A teljes gépi tanulási környezet a következő összetevőkből áll: Jupyter, Python, Spark és HDFS. Ezt a gépi tanulási környezetet az Occopus automatikusan építi ki és a Spark Workerek számát a felhasználó határozhatja meg.

Ez a bemutató egy teljes Apache Spark infratstruktúrát hoz létre, amely integrálva van a HDFS, a Python és a Jupyter Notebook-al. Tartalmaz egy Spark Master csomópontot és Spark Worker csomópontokat, amelyek számát felfelé vagy lefelé lehet skálázni.

**Használati és telepítési útmutató**

**https://occopus.readthedocs.io/en/latest/tutorial-bigdata-ai.html#apache-spark-cluster-with-jupyter-notebook-and-pyspark**

https://science-cloud.hu/felhasznalast-segito-szolgaltatasok

# 2. lépés: leírók letöltése

ELKH Cloud

Apache Spark klaszter Jupyter notebook és PySpark Stack leírása:
https://occopus.readthedocs.io/en/latest/tutorial-bigdata-ai.html#apache-spark-cluster-with-jupyter-notebook-and-pyspark

Apache Spark R Stack leírása:
https://occopus.readthedocs.io/en/latest/tutorial-bigdata-ai.html#apache-spark-cluster-with-rstudio-stack

## Apache Spark cluster with RStudio Stack

This tutorial sets up a complete Apache Spark (version **3.0.1**) infrastructure with HDFS (Hadoop Distributed File System) (version **3.3.0**) and RStudio server. Apache Spark is a fast and general-purpose cluster computing system. It provides high-level APIs in Java, Scala, Python and R, and an optimized engine that supports general execution graphs. It also supports a rich set of higher-level tools including Spark SQL for SQL and structured data processing, MLlib for machine learning, GraphX for graph processing, and Spark Streaming. For more information visit the official Apache Spark page .

This tutorial sets up a complete Apache Spark infrastructure integrated with HDFS, R, RStudio and sparklyr. It contains a Spark Master node and Spark Worker nodes, which can be scaled up or down.

### Features

- creating two types of nodes through contextualisation
- utilising health check against a predefined port
- using scaling parameters to limit the number of Spark Worker nodes

### Download

You can download the example as tutorial.examples.spark-cluster-with-r .

# Apache Spark klaszter referencia architektúrák

Apache Spark klaszter Jupyter notebook és PySpark Stack leírása:
https://occopus.readthedocs.io/en/latest/tutorial-bigdata-ai.html#apache-spark-cluster-with-jupyter-notebook-and-pyspark

Apache Spark R Stack leírása:
https://occopus.readthedocs.io/en/latest/tutorial-bigdata-ai.html#apache-spark-cluster-with-rstudio-stack

Main UI port list:

| Port | Description |
|---|---|
| 4040 | Application port (active only if a Spark application is running) |
| 6066 | Submit job to cluster via REST API |
| 7077 | Submit job to cluster/Join to the cluster |
| 8080 | Master UI |
| 8081 | Worker UI |
| 9870 | HDFS NameNode UI |

5. Load the node definitions into the database. Make sure the proper virtualenv is activated!

**Important**

Occopus takes node definitions from its database when builds up the infrastructure, so importing is necessary whenever the node definition or any imported (e.g. contextualisation) file changes!

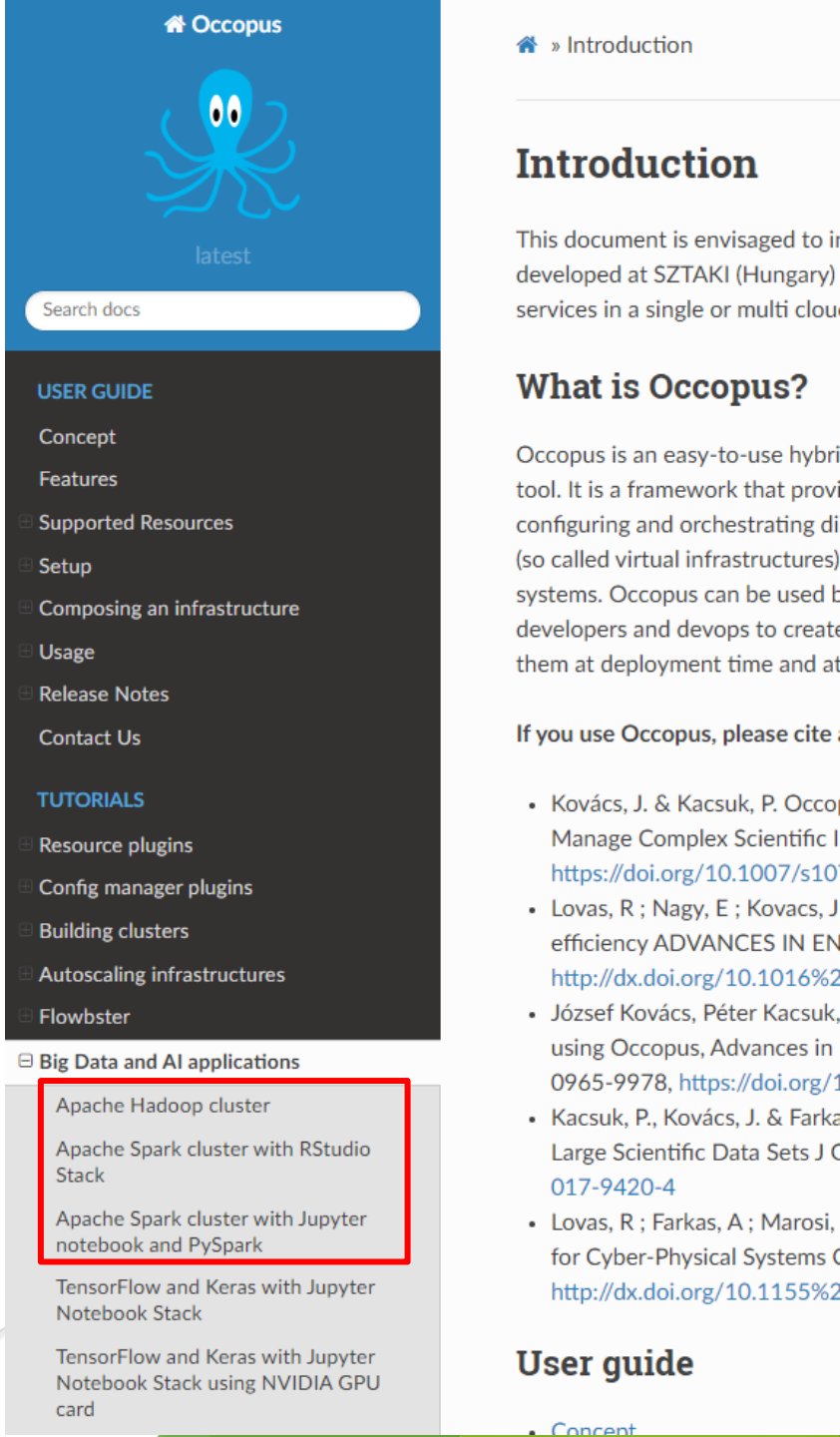
```
occopus-import nodes/node_definitions.yaml
```

6. Start deploying the infrastructure.

```
occopus-build infra-spark-cluster.yaml
```

# További referencia architektúrák elérhetősége

- Tutorials szekció alatt megtalálható
  - https://occopus.readthedocs.io/en/latest/index.html
  - Részletes leírások



# 3. lépés: Tűzfalszabályok létrehozása

ELKH Cloud

1

openstac

Project

COMPUTE

Overview

Instances

Volumes

Images

Access & Security

NETWORK

ORCHESTRATION

Identity

2

Access & Security

Compute/Access&Security/Create Security Group

Security Groups | Key Pairs | Floating IPs | API Access

+ CREATE SECURITY GROUP | DELETE SECURITY GROUPS

| Name | Description | Actions |
|---|---|---|
| All_open | | |
| default | | |
| ssh | | |

3

Create Security Group

Name *

Spark

Description

This is a sample Spark Security Group

Description:

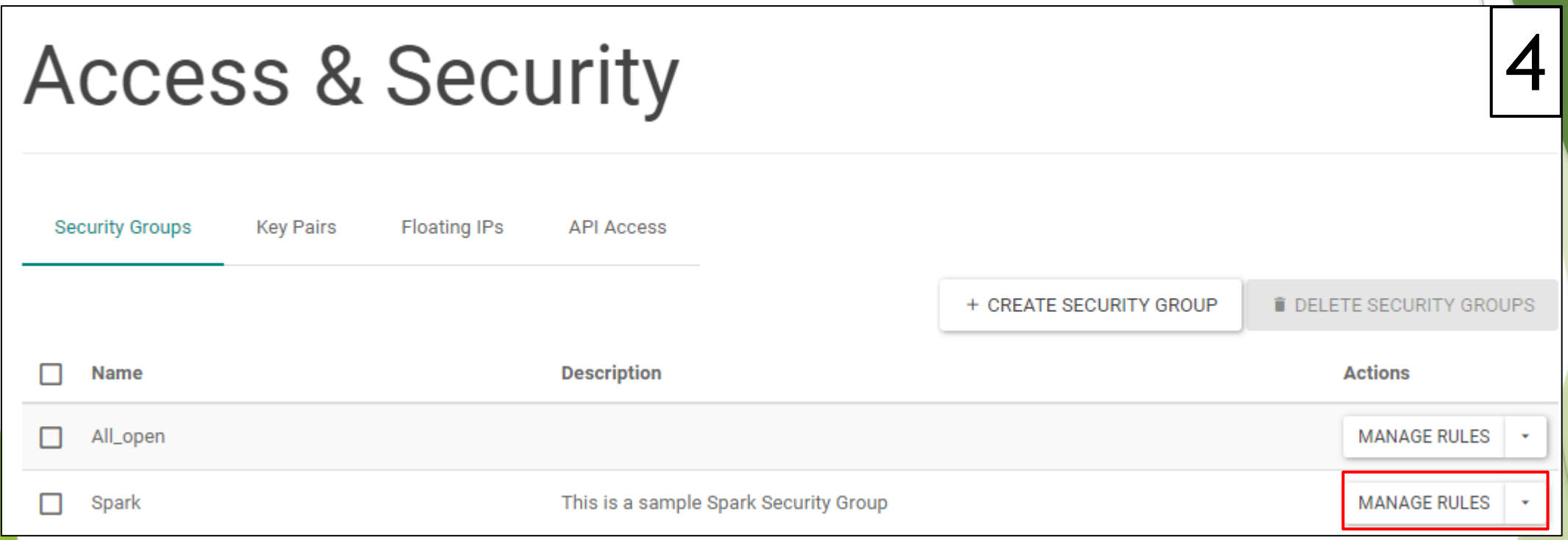
Security groups are sets of IP filter rules that are applied to the network settings for the VM. After the security group is created, you can add rules to the security group.

CANCEL | CREATE SECURITY GROUP

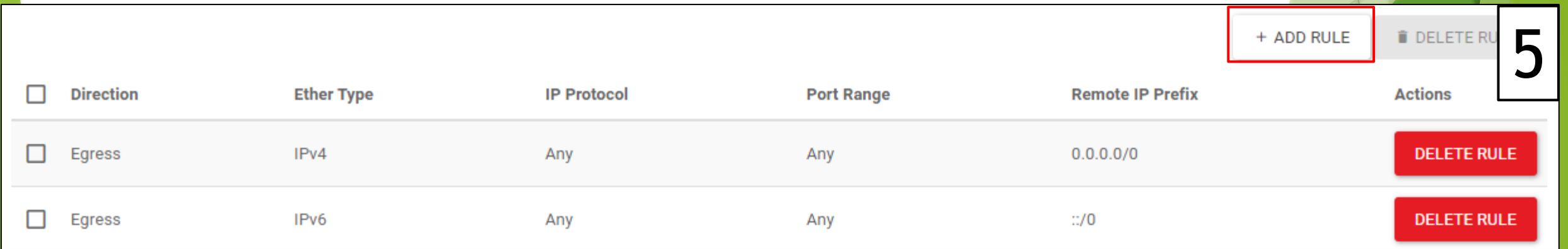
# 3. lépés: Tűzfalszabályok létrehozása

ELKH Cloud

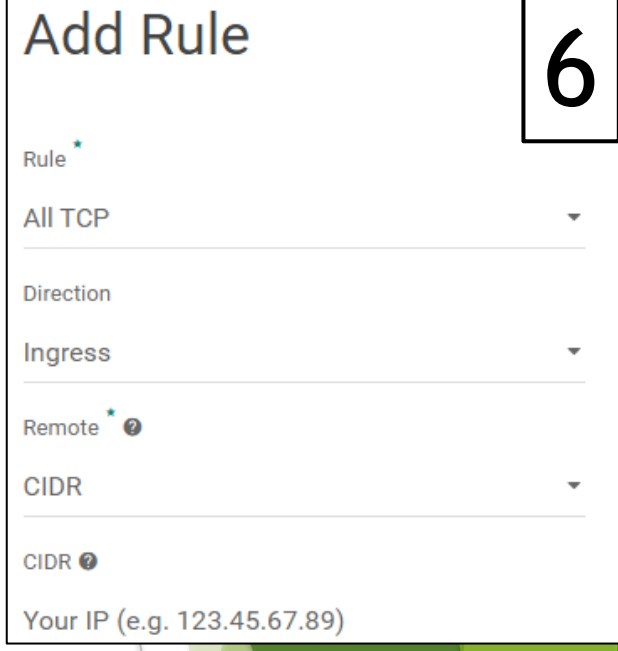
4

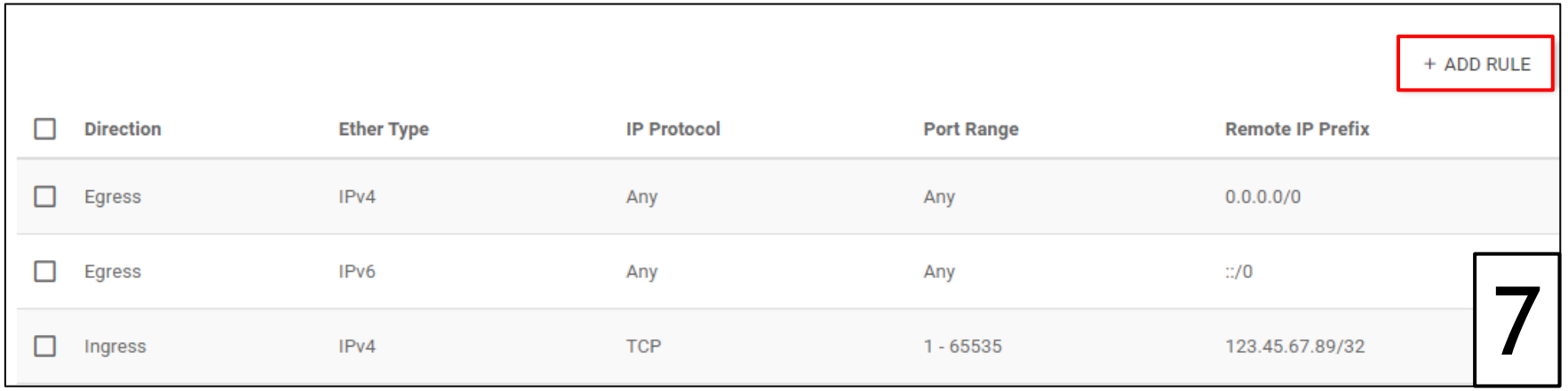



5



# 3. lépés: Tűzfalszabályok létrehozása



# 3. lépés: Tűzfalszabályok létrehozása

ELKH Cloud

9

openstack Project name

Project

COMPUTE

NETWORK

Network Topology

Networks

Routers

## Networks

| Name | Subnets Associated |
|---|---|
| Project name | projectname_subnet 192.168.0.0/24 |

8

## Add Rule

Rule *

All TCP

Direction

Ingress

Remote *

CIDR

CIDR

192.168.0.0/24

### Description:

Rules define which traffic is allowed to instances assigned to the security group. A security group rule consists of three main parts:

**Rule:** You can specify the desired rule template or use custom rules, the options are Custom TCP Rule, Custom UDP Rule, or Custom ICMP Rule.

**Open Port/Port Range:** For TCP and UDP rules you may choose to open either a single port or a range of ports. Selecting the "Port Range" option will provide you with space to provide both the starting and ending ports for the range. For ICMP rules you instead specify an ICMP type and code in the spaces provided.

**Remote:** You must specify the source of the traffic to be allowed via this rule. You may do so either in the form of an IP address block (CIDR) or via a source group (Security Group). Selecting a security group as the source will allow any other instance in that security group access to any other instance via this rule.

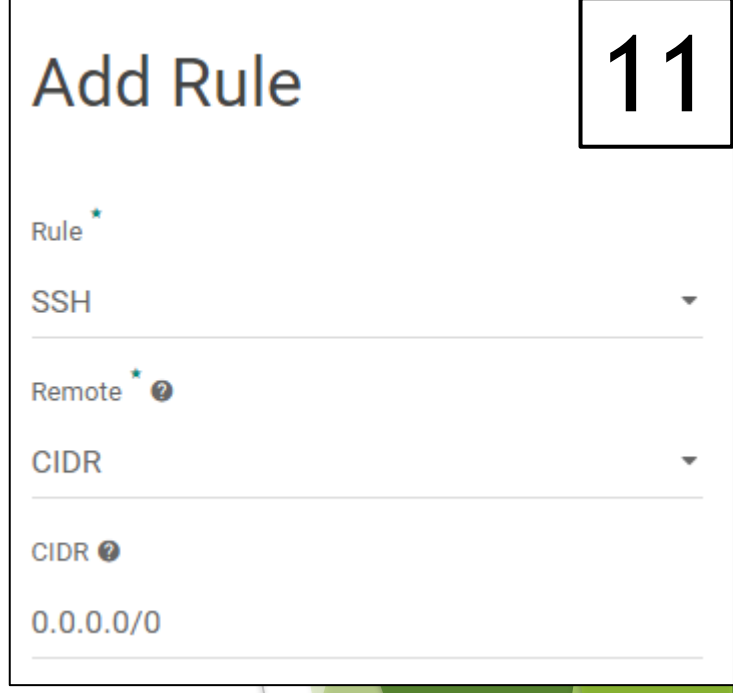
CANCEL ADD

10

| | Ether Type | IP Protocol | Port Range | Remote IP Prefix |
|---|---|---|---|---|
| | IPv4 | Any | Any | 0.0.0.0/0 |
| Egress | IPv6 | Any | Any | ::/0 |
| Ingress | IPv4 | TCP | 1 - 65535 | 123.45.67.89/32 |
| Ingress | IPv4 | TCP | 1 - 65535 | 192.168.0.0/24 |

# 3. lépés: Tűzfalszabályok létrehozása

11



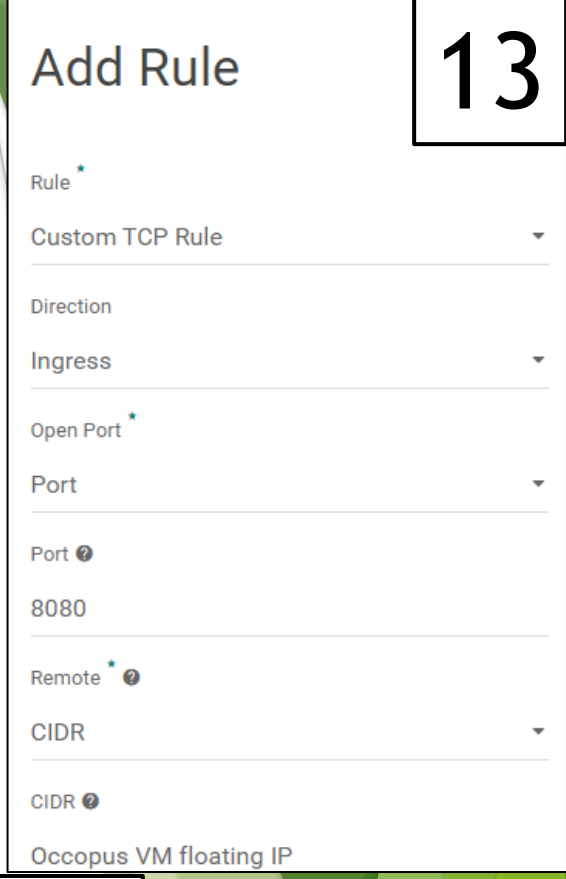

12

| ction | Ether Type | IP Protocol | Port Range | Remote IP Prefix |
|---|---|---|---|---|
| ess | IPv4 | Any | Any | 0.0.0.0/0 |
| Egress | IPv6 | Any | Any | ::/0 |
| Ingress | IPv4 | TCP | 1 - 65535 | 123.45.67.89/32 |
| Ingress | IPv4 | TCP | 1 - 65535 | 192.168.0.0/24 |
| Ingress | IPv4 | TCP | 22 (SSH) | 0.0.0.0/0 |

# 3. lépés: Tűzfalszabályok összegzése

Ajánlott tűzfalszabály Hadoop és Spark architektúrákra

13




14

| | Direction | IP Protocol | Port Range | Remote IP Prefix |
|---|---|---|---|---|
| 1 | Egress | Any | Any | 0.0.0.0/0 |
| 2 | Egress | Any | Any | ::/0 |
| 3 | Ingress | TCP | 1 - 65535 | Your IP address 123.45.67.89/32 |
| 4 | Ingress | TCP | 1 - 65535 | 192.168.0.0/24 |
| 5 | Ingress | TCP | 22 (SSH) | 0.0.0.0/0 |
| 6 | Ingress | TCP | 8080 | Occopus VM - Floating IP |

Kimenő forgalom

Pl. lekérdezés (harmadik féltől)

$ curl ifconfig.me
92.21.242.26

# 4. lépés: Leírók személyre szabása

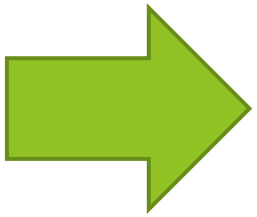
## Csomópont definíciós fájl (**nodes/node_definition.yaml**)

- Nova erőforrás szekció:

```
'node_def:spark_master_node':
  -
    resource:
      type: nova
      endpoint: replace_with_endpoint_of_nova_interface_of_your_cloud
      project_id: replace_with_projectid_to_use
      user_domain_name: Default
      image_id: replace_with_id_of_your_image_on_your_target_cloud
      network_id: replace_with_id_of_network_on_your_target_cloud
      flavor_name: replace_with_id_of_the_flavor_on_your_target_cloud
      security_groups:
        - replace_with_security_group_to_add_or_remove_section
    contextualisation:
      type: cloudinit
      context_template: !yaml_import
        url: file://cloud_init_spark_master.yaml
    health_check:
      ports:
        - 8080
      timeout: 1000
```

# 4. lépés: Leírók személyre szabása

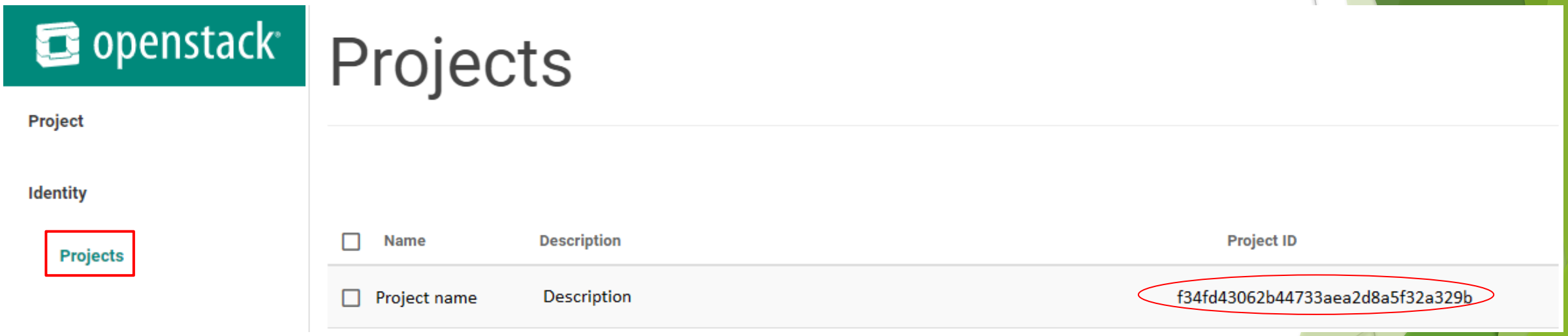
```
'node_def:spark_master_node':
  -
    resource:
      type: nova
      endpoint: replace_with_endpoint_of_nova_interface_of_your_cloud
      project_id: replace_with_projectid_to_use
      user_domain_name: Default
      image_id: replace_with_id_of_your_image_on_your_target_cloud
      network_id: replace_with_id_of_network_on_your_target_cloud
      flavor_name: replace_with_id_of_the_flavor_on_your_target_cloud
      key_name: replace_with_name_of_keypair_or_remove
      security_groups:
        - replace_with_security_group_to_add_or_remove_section
      floating_ip: add_yes_if_you_need_floating_ip_or_remove
      floating_ip_pool: replace_with_name_of_floating_ip_pool_or_remove
    contextualisation:
      type: cloudinit
      context_template: !yaml_import
        url: file://cloud_init_spark_master.yaml
    health_check:
      ports:
        - 8080
      timeout: 1000
...
```



```
'node_def:spark_master_node':
  -
    resource:
      type: nova
            endpoint: https://sztaki.cloud.mta.hu:5000/v3
            project_id: caml6db63ddf47a98045ef9c726vgqbp
            user_domain_name: Default
            image_id: zgsf1dc3-b6d5-4b15-942e-61e0ef2l8dk
            network_id: 3yqqqe1c-858c-4047-a48a-e2fab0nd547
            flavor_name: 3
            key_name: key_name
            security_groups:
              - f7d8dc12-fd7a-4d69-ba8d-c1f11c9b5b73
            floating_ip: yes
          contextualisation:
            type: cloudinit
            context_template: !yaml_import
              url: file://cloud_init_spark_master.yaml
          health_check:
            ports:
              - 8080
            timeout: 1000
...
```

https://occopus.readthedocs.io/en/latest/user-doc-collecting-resources.html#openstack-horizon-nova

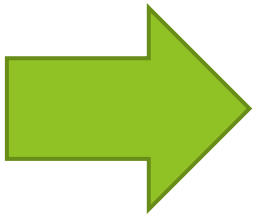
# 4. lépés: Leírók személyre szabása - segítség




https://occopus.readthedocs.io/en/latest/user-doc-collecting-resources.html#openstack-horizon-nova

# 4. lépés: Leírók személyre szabása

```
'node_def:spark_master_node':
  -
    resource:
      type: nova
      endpoint: replace_with_endpoint_of_nova_interface_of_your_cloud
      project_id: replace_with_projectid_to_use
      user_domain_name: Default
      image_id: replace_with_id_of_your_image_on_your_target_cloud
      network_id: replace_with_id_of_network_on_your_target_cloud
      flavor_name: replace_with_id_of_the_flavor_on_your_target_cloud
      key_name: replace_with_name_of_keypair_or_remove
      security_groups:
        - replace_with_security_group_to_add_or_remove_section
      floating_ip: add_yes_if_you_need_floating_ip_or_remove
      floating_ip_pool: replace_with_name_of_floating_ip_pool_or_remove
    contextualisation:
      type: cloudinit
      context_template: !yaml_import
        url: file://cloud_init_spark_master.yaml
    health_check:
      ports:
        - 8080
      timeout: 1000
...
```



```
'node_def:spark_master_node':
  -
    resource:
      type: nova
            endpoint: https://sztaki.cloud.mta.hu:5000/v3
            project_id: f34fd43062b44733aea2d8a5f32a329b
            user_domain_name: Default
            image_id:
            network_id:
            flavor_name:
            key_name: key_name
            security_groups:
              -
            floating_ip: yes
          contextualisation:
            type: cloudinit
            context_template: !yaml_import
              url: file://cloud_init_spark_master.yaml
          health_check:
            ports:
              - 8080
            timeout: 1000
...
```

https://occopus.readthedocs.io/en/latest/user-doc-collecting-resources.html#openstack-horizon-nova

openstack

Project

COMPUTE

- Overview
- Instances
- Volumes
- Images
- Access & Security

NETWORK

ORCHESTRATION

Identity

# Images

PROJECT (7) | SHARED WITH ME (0) | PUBLIC (12)

| Image Name | Type | Status | Public | Protected | Format |
|---|---|---|---|---|---|
| CentOS 7 Cloud image | Image | Active | Yes | No | Raw |
| DCI Bridge | Image | Active | Yes | No | Raw |
| DCI Bridge + Python | Image | Active | Yes | No | Raw |
| gUSE | Image | Active | Yes | No | Raw |
| gUSE-10 | Image | Active | Yes | No | Raw |
| Ubuntu 14.04 LTS Cloud image | Image | Active | Yes | No | Raw |
| Ubuntu 16.04 LTS Cloud image | Image | Active | Yes | No | Raw |
| Ubuntu 16.04 LTS for Heat | Image | Active | Yes | No | Raw |
| Ubuntu 16.04 LTS with MPI v2 | Image | Active | Yes | No | Raw |
| Ubuntu 18.04 LTS Cloud image | Image | Active | Yes | No | Raw |

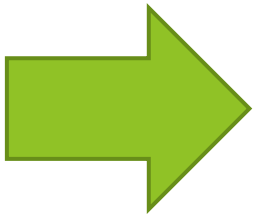
ELKH Cloud

Images / Ubuntu 18.04 LTS Cloud image

| | |
|---|---|
| Name | Ubuntu 18.04 LTS Cloud image |
| ID | 6bba6dc3-b6d5-4b15-942e-61e0ef2f93cb |
| Owner | 045cebb7b44d48418d7bfa11a77687fa |
| Status | Active |
| Public | Yes |
| Protected | No |
| Checksum | a5245b9680cc59454f8435b6e4e04e86 |
| Created | Aug. 6, 2018, 2:48 p.m. |
| Updated | Aug. 6, 2018, 2:49 p.m. |

# 4. lépés: Leírók személyre szabása

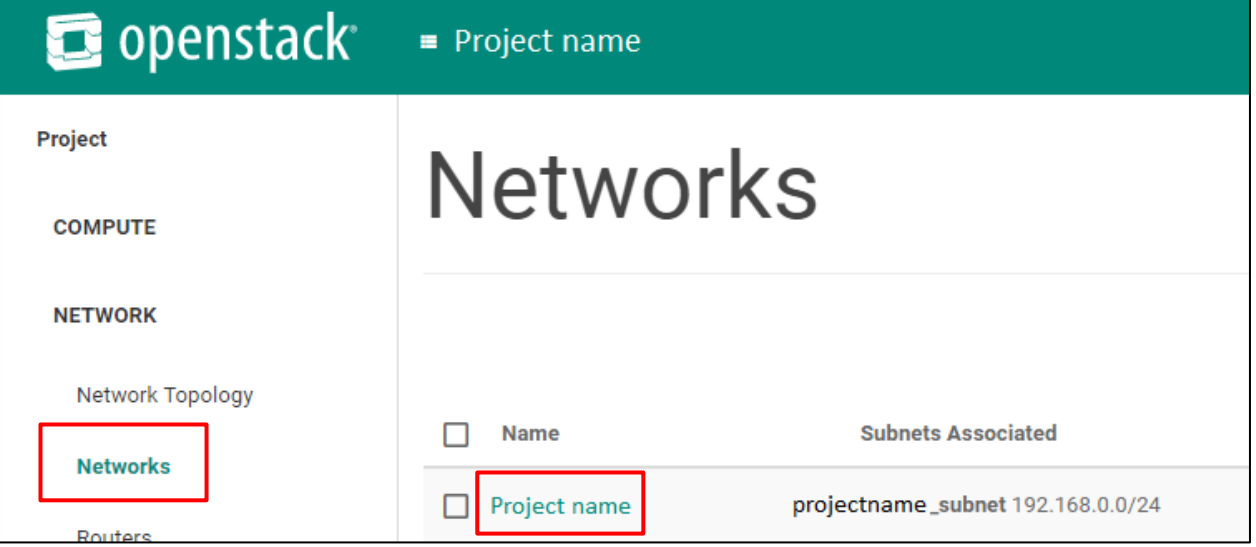
```
'node_def:spark_master_node':
  -
    resource:
      type: nova
      endpoint: replace_with_endpoint_of_nova_interface_of_your_cloud
      project_id: replace_with_projectid_to_use
      user_domain_name: Default
      image_id: replace_with_id_of_your_image_on_your_target_cloud
      network_id: replace_with_id_of_network_on_your_target_cloud
      flavor_name: replace_with_id_of_the_flavor_on_your_target_cloud
      key_name: replace_with_name_of_keypair_or_remove
      security_groups:
        - replace_with_security_group_to_add_or_remove_section
      floating_ip: add_yes_if_you_need_floating_ip_or_remove
      floating_ip_pool: replace_with_name_of_floating_ip_pool_or_remove
    contextualisation:
      type: cloudinit
      context_template: !yaml_import
        url: file://cloud_init_spark_master.yaml
    health_check:
      ports:
        - 8080
      timeout: 1000
...
```

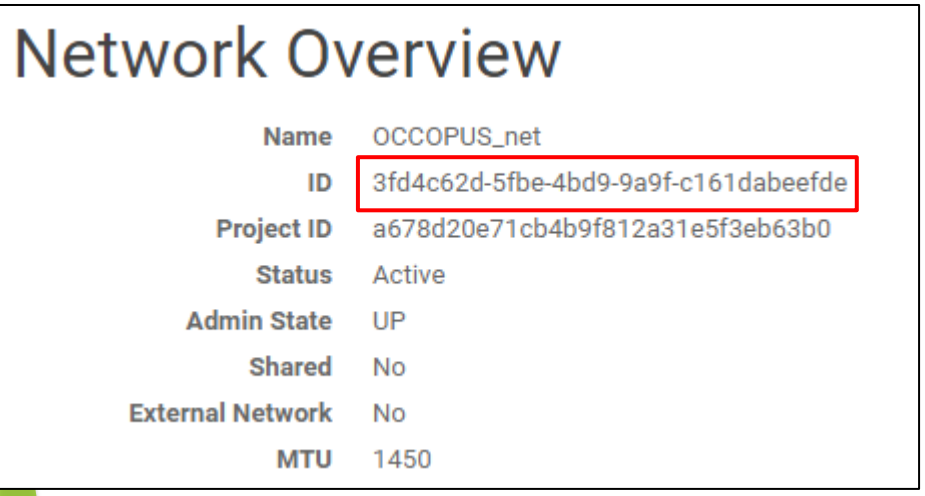


```
'node_def:spark_master_node':
  -
    resource:
      type: nova
            endpoint: https://sztaki.cloud.mta.hu:5000/v3
            project_id: f34fd43062b44733aea2d8a5f32a329b
            user_domain_name: Default
            image_id: 6bba6dc3-b6d5-4b15-942e-61e0ef2f93cb
            network_id:
            flavor_name:
            key_name:
            security_groups:
              -
            floating_ip: yes
          contextualisation:
            type: cloudinit
            context_template: !yaml_import
              url: file://cloud_init_spark_master.yaml
          health_check:
            ports:
              - 8080
            timeout: 1000
...
```

https://occopus.readthedocs.io/en/latest/user-doc-collecting-resources.html#openstack-horizon-nova

# Network ID az ELKH Cloudon



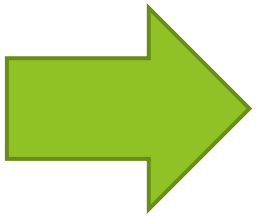




# Flavor ID-k az ELKH Cloudon

| Name | ID | VCPUS | RAM | Total Disk | Root Disk |
|---|---|---|---|---|---|
| m1.small | 4740c1b8-016d-49d5-a669-2b673f86317c | 1 | 2 GB | 20 GB | 20 GB |
| m1.medium | 3 | 2 | 4 GB | 40 GB | 40 GB |
| m1.large | 4 | 4 | 8 GB | 80 GB | 80 GB |
| m1.xlarge | 41316ba3-2d8b-4099-96d5-efa82181bb22 | 8 | 16 GB | 160 GB | 160 GB |

# 4. lépés: Leírók személyre szabása

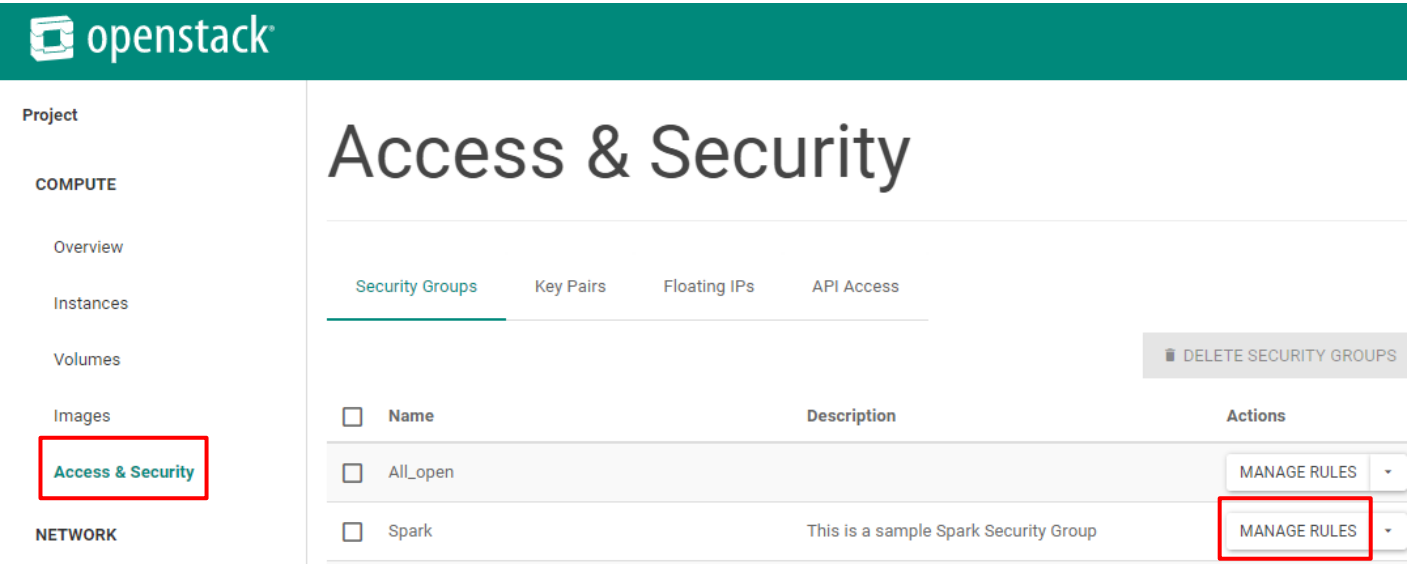
```
'node_def:spark_master_node':
  -
    resource:
      type: nova
      endpoint: replace_with_endpoint_of_nova_interface_of_your_cloud
      project_id: replace_with_projectid_to_use
      user_domain_name: Default
      image_id: replace_with_id_of_your_image_on_your_target_cloud
      network_id: replace_with_id_of_network_on_your_target_cloud
      flavor_name: replace_with_id_of_the_flavor_on_your_target_cloud
      key_name: replace_with_name_of_keypair_or_remove
      security_groups:
        - replace_with_security_group_to_add_or_remove_section
      floating_ip: add_yes_if_you_need_floating_ip_or_remove
      floating_ip_pool: replace_with_name_of_floating_ip_pool_or_remove
    contextualisation:
      type: cloudinit
      context_template: !yaml_import
        url: file://cloud_init_spark_master.yaml
    health_check:
      ports:
        - 8080
      timeout: 1000
...
```

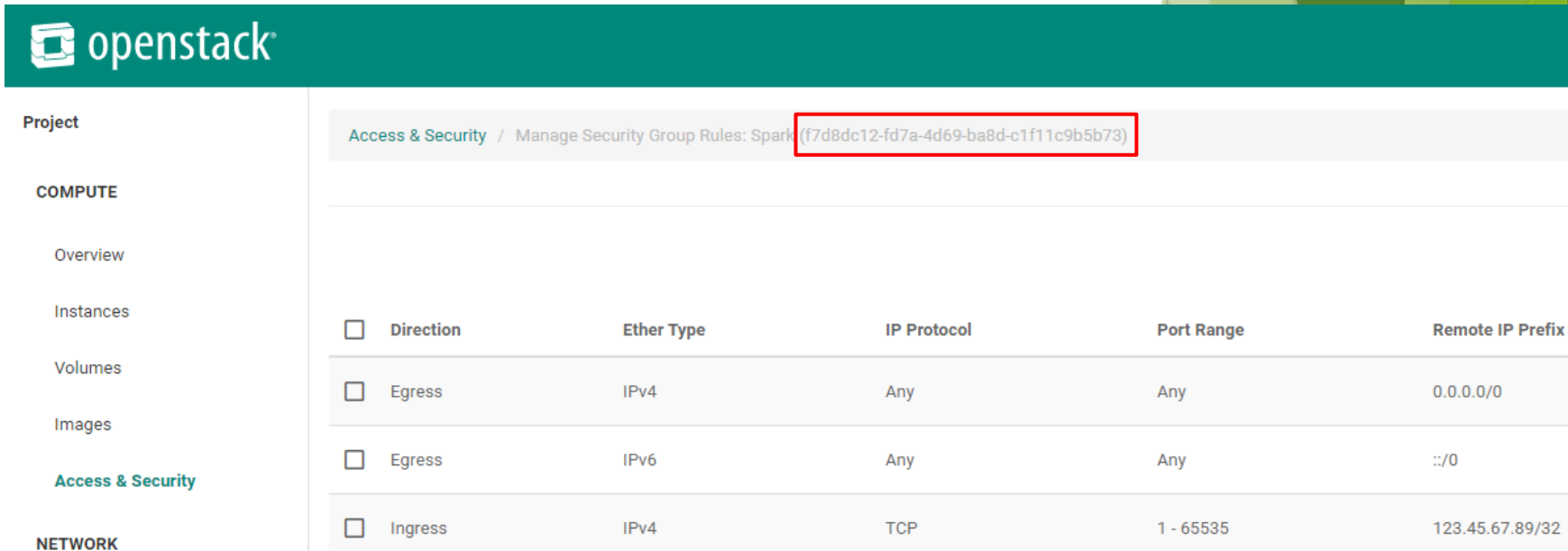


```
'node_def:spark_master_node':
  -
    resource:
      type: nova
            endpoint: https://sztaki.cloud.mta.hu:5000/v3
            project_id: f34fd43062b44733aea2d8a5f32a329b
            user_domain_name: Default
            image_id: 6bba6dc3-b6d5-4b15-942e-61e0ef2f93cb
            network_id: 3fd4c62d-5fbe-4bd9-9a9f-c161dabeefde
            flavor_name: 3
            key_name: key_name
            security_groups:
              -
            floating_ip: yes
          contextualisation:
            type: cloudinit
            context_template: !yaml_import
              url: file://cloud_init_spark_master.yaml
          health_check:
            ports:
              - 8080
            timeout: 1000
...
```

https://occopus.readthedocs.io/en/latest/user-doc-collecting-resources.html#openstack-horizon-nova

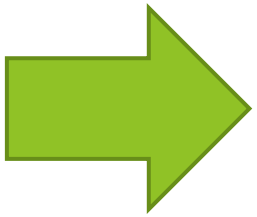
# Security Group ID az ELKH Cloudon



# 4. lépés: Leírók személyre szabása

```
'node_def:spark_master_node':
  -
    resource:
      type: nova
      endpoint: replace_with_endpoint_of_nova_interface_of_your_cloud
      project_id: replace_with_projectid_to_use
      user_domain_name: Default
      image_id: replace_with_id_of_your_image_on_your_target_cloud
      network_id: replace_with_id_of_network_on_your_target_cloud
      flavor_name: replace_with_id_of_the_flavor_on_your_target_cloud
      key_name: replace_with_name_of_keypair_or_remove
      security_groups:
        - replace_with_security_group_to_add_or_remove_section
      floating_ip: add_yes_if_you_need_floating_ip_or_remove
      floating_ip_pool: replace_with_name_of_floating_ip_pool_or_remove
    contextualisation:
      type: cloudinit
      context_template: !yaml_import
        url: file://cloud_init_spark_master.yaml
    health_check:
      ports:
        - 8080
      timeout: 1000
...
```



```
'node_def:spark_master_node':
  -
    resource:
      type: nova
            endpoint: https://sztaki.cloud.mta.hu:5000/v3
            project_id: caml6db63ddf47a98045ef9c726vgqbp
            user_domain_name: Default
            image_id: zgsf1dc3-b6d5-4b15-942e-61e0ef2l8dk
            network_id: 3yqqqe1c-858c-4047-a48a-e2fab0nd547
            flavor_name: 3
            key_name: key_name
            security_groups:
              - f7d8dc12-fd7a-4d69-ba8d-c1f11c9b5b73
            floating_ip: yes
          contextualisation:
            type: cloudinit
            context_template: !yaml_import
              url: file://cloud_init_spark_master.yaml
          health_check:
            ports:
              - 8080
            timeout: 1000
...
```

https://occopus.readthedocs.io/en/latest/user-doc-collecting-resources.html#openstack-horizon-nova

# 4. lépés: Leírók személyre szabása – klaszter méretének beállítása

Ha szükséges, frissítse a Spark dolgozó (worker) csomópontok számát!

Ehhez szerkessze át az infra-spark-cluster.yaml fájlt és módosítsa a „min” és „max” paramétereket a „scaling” kulcsszó alatt.

- A skálázás az az intervallum, amelyben a csomópontok száma megváltozhat (min, max). Jelenleg a minimális érték 2-re van állítva (ami az indításkor a kezdeti szám lesz), és a maximális értéke 10.
- Ne feledje, hogy az Occopusnak legalább egy csomópontot el kell indítania minden egyes csomóponttípusból, hogy az infrastruktúra megfelelően működjön, valamint a skálázás csak a Spark dolgozó (worker) csomópontokon alkalmazható ebben a példában!
- Később: manuális skálázás

```
nodes:
    - &M
        name: spark-master
        type: spark_master_node
    - &W
        name: spark-worker
        type: spark_worker_node
        scaling:
          min: 2
          max: 10
dependencies:
    -   connection: [ *W, *M ]
```

# 5. lépés: occopus aktiválása

Győződjön meg róla, hogy a megfelelő virtualenv aktiválva van!

Amennyiben ezt még nem tette volna meg korábban, az alábbi parancs segítségével aktiválható az Occopus virtuális környezete:

```
ubuntu@occo:~/spark-cluster-with-python$ source $HOME/occopus/bin/activate

(occopus) ubuntu@occo:~/spark-cluster-with-python$
```

# 6. lépés: leírók importálása

Importáljuk a személyre szabott leírókat az Occopus adatbázisába:

```
$ occopus-import nodes/node_definitions.yaml
Successfully imported nodes: spark_master_node, spark_worker_node
```

- **Megjegyzés:** A nodes mappában található cloud init fájlok szerkesztésével a haladó felhasználók személyre szabhatják a Spark konfigurációs fájljait (cloud_init_spark_master.yaml, cloud_init_spark_worker.yaml).
- **Fontos:** Az Occopus akkor veszi fel a csomópont definíciókat az adatbázisból, amikor felépíti az infrastruktúrát, így mindig importálásra van szükség, ha a node definíciós fájl, vagy bármelyik (pl.: kontextualizációs) fájl megváltozik!

# 7. lépés: infrastruktúra kiépítése

Az alábbi parancs segítségével megkezdhetjük a klaszter felépítését:
`$ occopus-build --parallelize infra-spark-cluster.yaml`

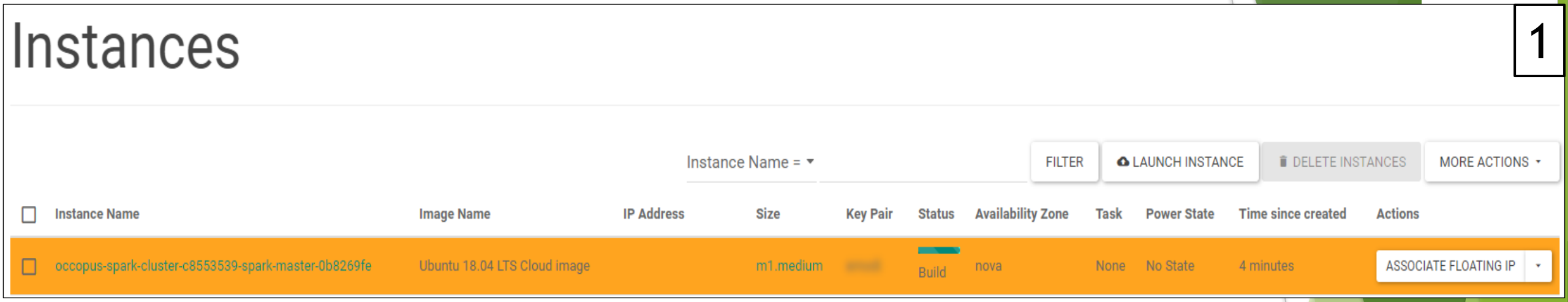
**Tipp:** occopus-build **--parallelize** infra-spark-cluster.yaml (párhuzamos VM kiépítés, az egymástól független VM-ek esetében)

```
$ occopus-build --parallelize infra-spark-cluster.yaml

** 2021-02-17 17:07:21,391      Creating node 'spark-master'/'0b8269fe-78ec-47fc-8cdb-7195209e5123’

...

** 2021-02-17 17:25:04,184      Health checking for node 'spark-master'/'0b8269fe-78ec-47fc-8cdb-7195209e5123'
** 2021-02-17 17:25:05,360        Checking node reachability (0b8269fe-78ec-47fc-8cdb-7195209e5123):
** 2021-02-17 17:25:05,371          193.224.59.67 => ready
** 2021-02-17 17:25:05,371        Checking port availability (0b8269fe-78ec-47fc-8cdb-7195209e5123):
** 2021-02-17 17:25:05,373          8080 => ready
** 2021-02-17 17:25:05,373      Health checking result: ready
** 2021-02-17 17:25:05,376      Node 'spark-master'/'0b8269fe-78ec-47fc-8cdb-7195209e5123' is ready.
** 2021-02-17 17:25:05,409      Creating node 'spark-worker'/'50c7cfe9-4c3c-4928-81e6-d58323b1e2b2'
** 2021-02-17 17:25:05,413      Creating node 'spark-worker'/'98a82e45-6bf2-4cb9-9ead-953be4435bd7'
```

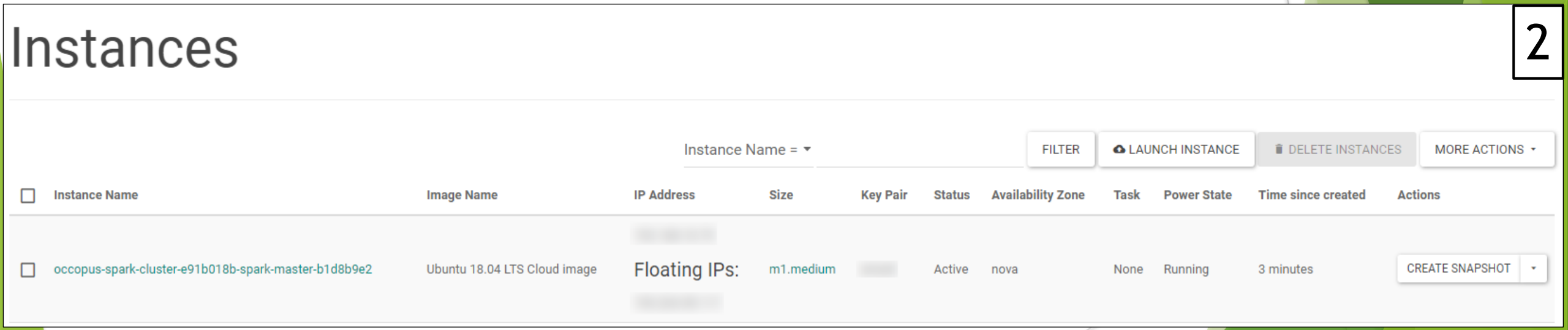
# 7. lépés: infrastruktúra kiépítése

ELKH Cloud

1




2



# 7. lépés: infrastruktúra kiépítése

ELKH Cloud

3

## Instances

Instance Name = | FILTER | LAUNCH INSTANCE | DELETE INSTANCES | MORE ACTIONS

| Instance Name | Image Name | IP Address | Size | Key Pair | Status | Availability Zone | Task | Power State | Time since created | Actions |
|---|---|---|---|---|---|---|---|---|---|---|
| occopus-spark-cluster-b68c5d40-spark-worker-7399ea46 | Ubuntu 18.04 LTS Cloud image | | m1.medium | | Active | nova | None | Running | 1 minute | CREATE SNAPSHOT |
| occopus-spark-cluster-b68c5d40-spark-worker-303b0625 | Ubuntu 18.04 LTS Cloud image | | m1.medium | | Active | nova | None | Running | 2 minutes | CREATE SNAPSHOT |
| occopus-spark-cluster-b68c5d40-spark-master-ac15312f | Ubuntu 18.04 LTS Cloud image | Floating IPs: | m1.medium | | Active | nova | None | Running | 11 minutes | CREATE SNAPSHOT |

# 7. lépés: infrastruktúra kiépítése

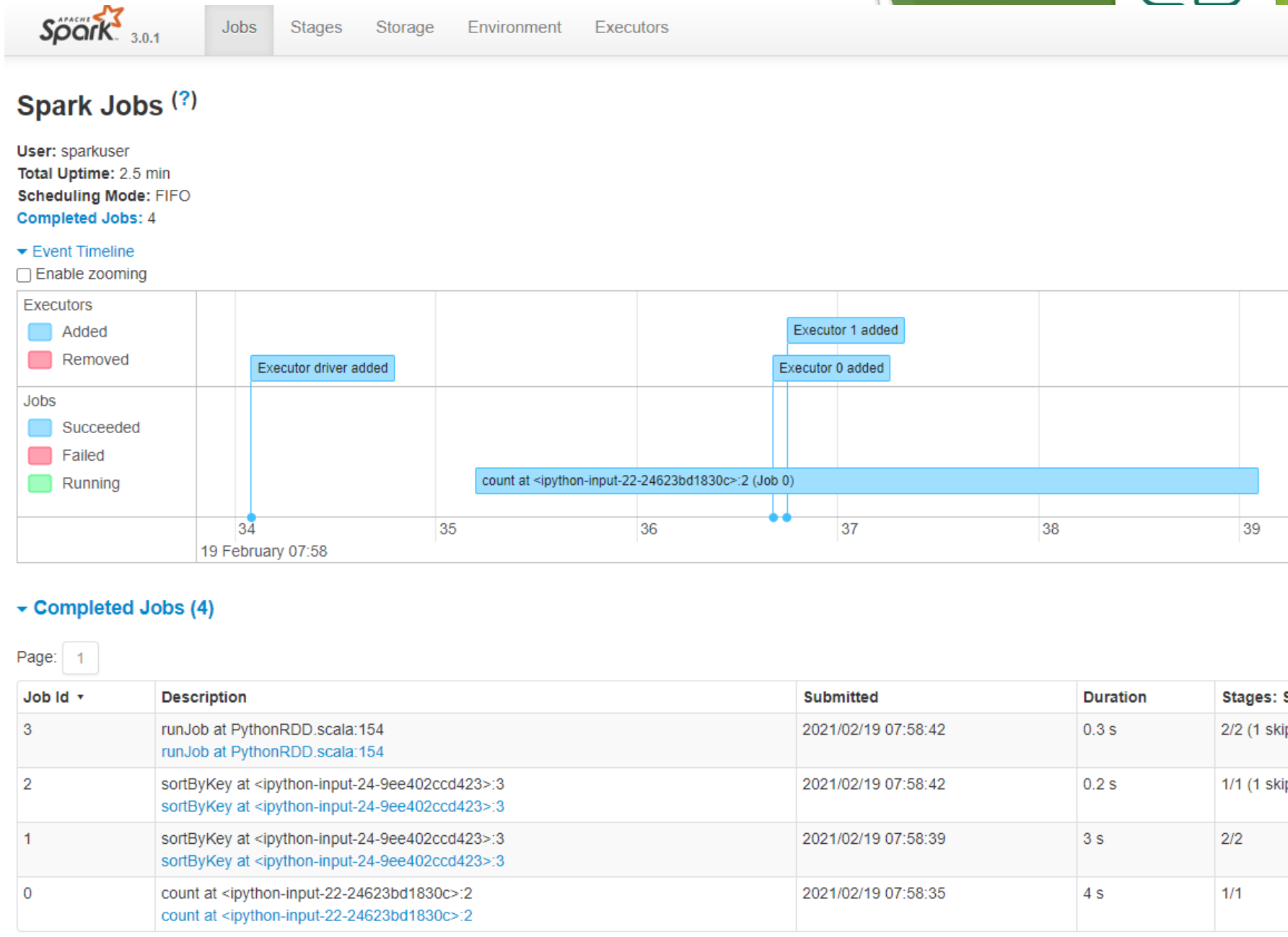
Sikeres lefutás után a virtuális gépek IP címei, node ID –jai, valamint az infrastruktúra azonosítója megjelenik a log üzenetek alján, listába szedve.

Az infrastruktúra azonosítója elmenthető, vagy lekérdezhető az occopus-maintain parancs segítségével:

```
** 2021-02-17 17:26:33,041      Submitted infrastructure: 'c8553539-42c9-40b1-97bc-d53ab8947ca7'
** 2021-02-17 17:26:33,127      List of nodes/instances/addresses:
** 2021-02-17 17:26:33,127      spark-worker:
** 2021-02-17 17:26:33,128        50c7cfe9-4c3c-4928-81e6-d58323b1e2b2:
** 2021-02-17 17:26:33,128          198.124.39.182
** 2021-02-17 17:26:33,128        98a82e45-6bf2-4cb9-9ead-953be4435bd7:
** 2021-02-17 17:26:33,128          198.124.39.144
** 2021-02-17 17:26:33,129      spark-master:
** 2021-02-17 17:26:33,129        0b8269fe-78ec-47fc-8cdb-7195209e5123:
** 2021-02-17 17:26:33,129          198.124.39.67
c8553539-42c9-40b1-97bc-d53ab8947ca7
```

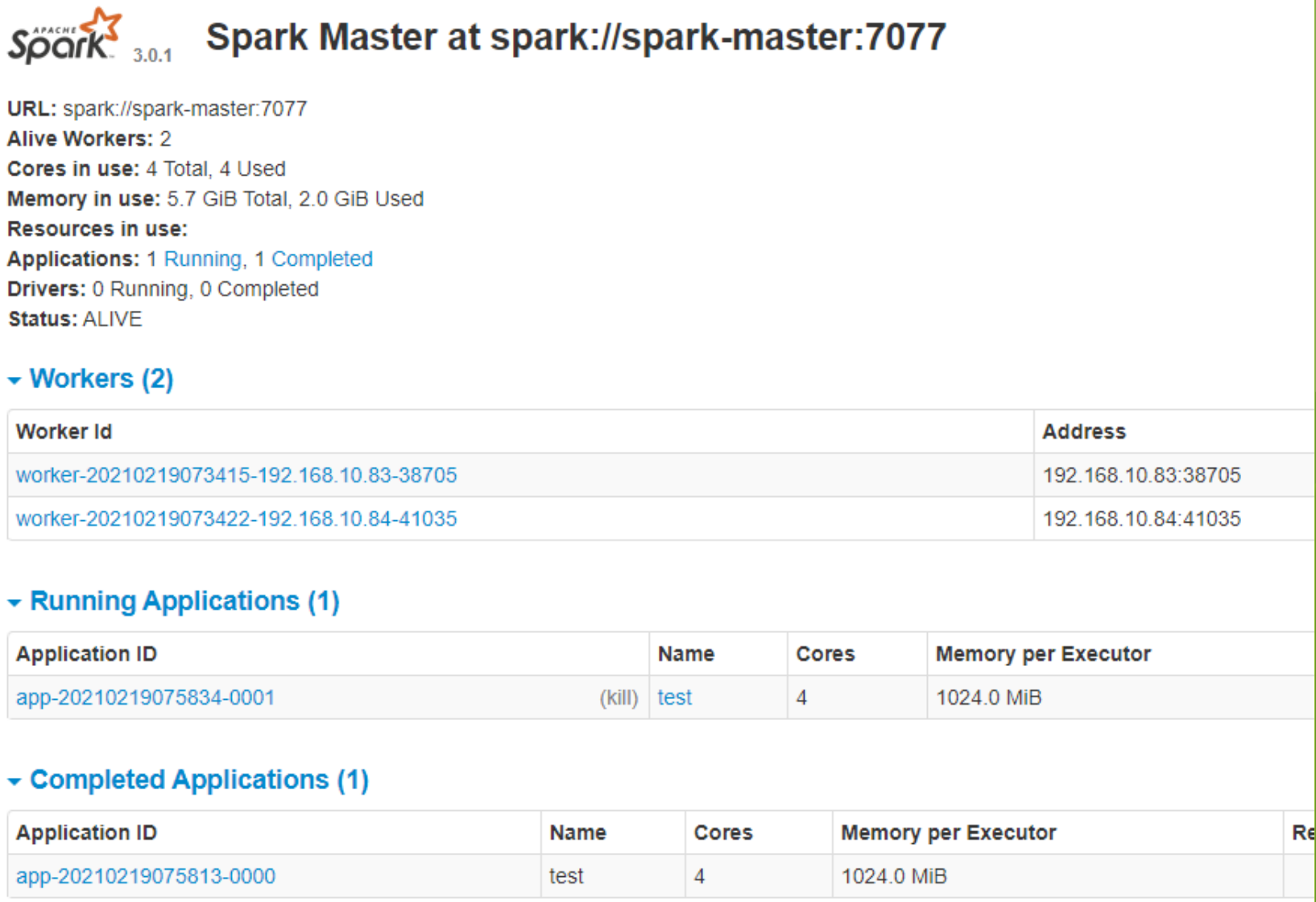
# 8. lépés: infrastruktúra használata

- Ellenőrizheti a Spark klaszter helyes működését, statisztikáit az alábbi weblapokon:
- Application UI: "http://<SparkMasterIP>:4040"
- Spark UI: "http://<SparkMasterIP>:8080"
- Jupyter UI: "http://<SparkMasterIP>:8888"
- HDFS UI: "http://<SparkMasterIP>:9870"



# 8. lépés: infrastruktúra használata

- Ellenőrizheti a Spark klaszter helyes működését, statisztikáit az alábbi weblapokon:
- Application UI: "http://<SparkMasterIP>:4040"
- Spark UI: "http://<SparkMasterIP>:8080"
- Jupyter UI: "http://<SparkMasterIP>:8888"
- HDFS UI: "http://<SparkMasterIP>:9870"



Spark 3.0.1 **Spark Master at spark://spark-master:7077**

**URL:** spark://spark-master:7077
**Alive Workers:** 2
**Cores in use:** 4 Total, 4 Used
**Memory in use:** 5.7 GiB Total, 2.0 GiB Used
**Resources in use:**
**Applications:** 1 Running, 1 Completed
**Drivers:** 0 Running, 0 Completed
**Status:** ALIVE

**Workers (2)**

| Worker Id | Address |
|---|---|
| worker-20210219073415-192.168.10.83-38705 | 192.168.10.83:38705 |
| worker-20210219073422-192.168.10.84-41035 | 192.168.10.84:41035 |

**Running Applications (1)**

| Application ID | | Name | Cores | Memory per Executor |
|---|---|---|---|---|
| app-20210219075834-0001 | (kill) | test | 4 | 1024.0 MiB |

**Completed Applications (1)**

| Application ID | Name | Cores | Memory per Executor | Re |
|---|---|---|---|---|
| app-20210219075813-0000 | test | 4 | 1024.0 MiB | |

# 8. lépés: infrastruktúra használata

- Ellenőrizheti a Spark klaszter helyes működését, statisztikáit az alábbi weblapokon:
- Application UI: "http://<SparkMasterIP>:4040"
- Spark UI: "http://<SparkMasterIP>:8080"
- Jupyter UI: "http://<SparkMasterIP>:8888"
- HDFS UI: "http://<SparkMasterIP>:9870"

## Test Spark Cluster

### Import python libraries

In [1]:
```
from pyspark import SparkContext, SparkConf
```

In [2]:
```
SparkMasterIP="192.168.10.8"
```

### Start Spark Application

In [3]:
```
# Start Spark local mode
# sc = SparkContext(appName="test", master="local")

# Start Spark cluster mode

sc = SparkContext(appName="test", master="spark://"+SparkMasterIP+":7077")
```

In [4]:
```
sc
```

Out[4]:

**SparkContext**

Spark UI

**Version**
`v3.0.1`

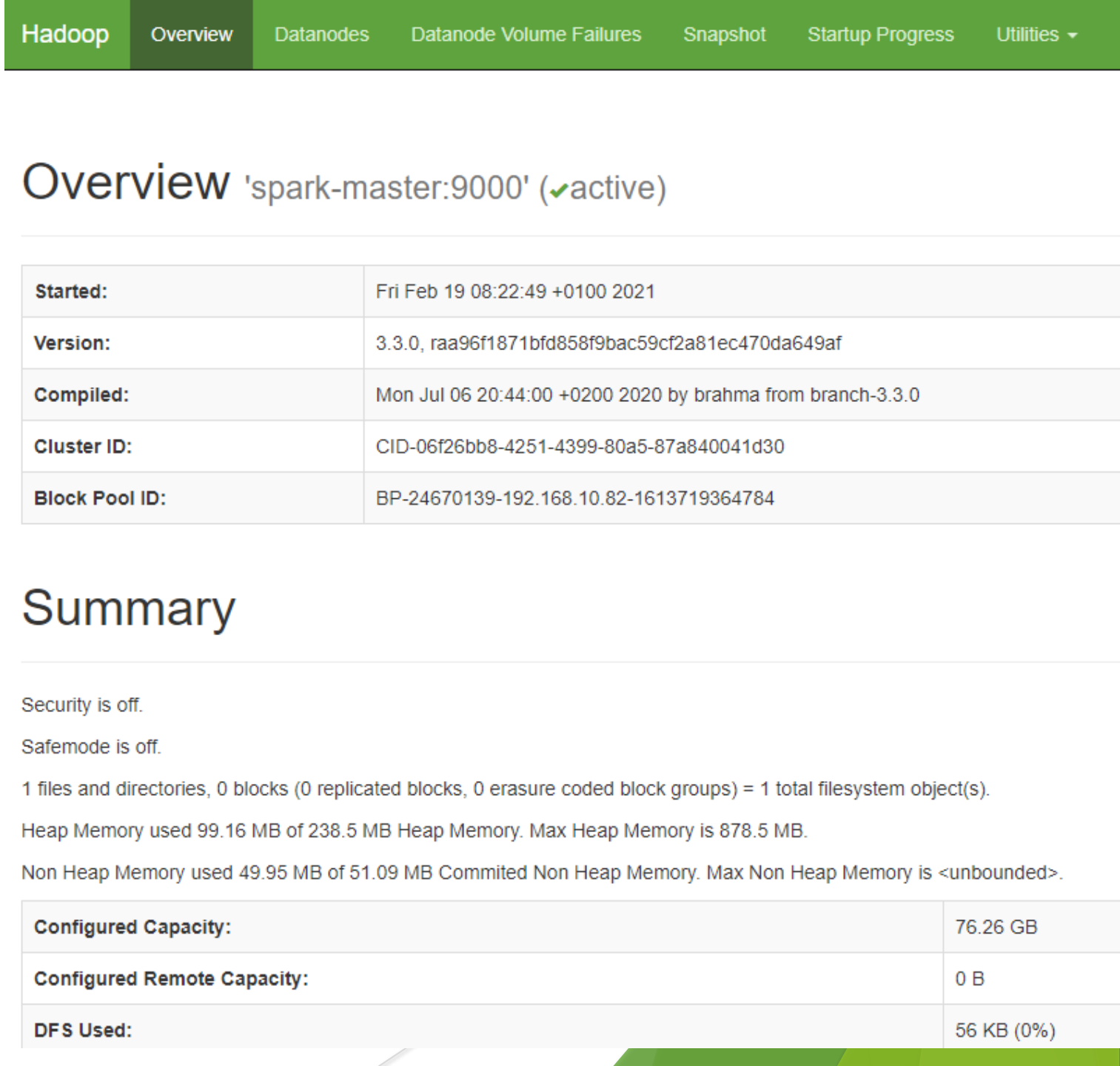
**Master**
`spark://192.168.10.8:7077`

**AppName**
`test`

In [5]:
```
import sys
from random import random
from operator import add
from pyspark.sql import SparkSession
```

# 8. lépés: infrastruktúra használata

- Ellenőrizheti a Spark klaszter helyes működését, statisztikáit az alábbi weblapokon:
- Application UI: "http://<SparkMasterIP>:4040"
- Spark UI: "http://<SparkMasterIP>:8080"
- Jupyter UI: "http://<SparkMasterIP>:8888"
- HDFS UI: "http://<SparkMasterIP>:9870"



# Megoldás használatának lépései

**Felhasználó feladatköre:**

0. Lépés: **Előkészítés** (ELKH Cloud projekt, Üres Ubuntu VM elindítás)

1. Lépés: **Occopus telepítés/konfigurálás**

2. Lépés: **Leírók letöltése a virtuális gépre**
Occopus/ELKH Cloud weboldala
3. Lépés: **Tűzfalszabályok létrehozása**

ELKH Cloud OpenStack felületén

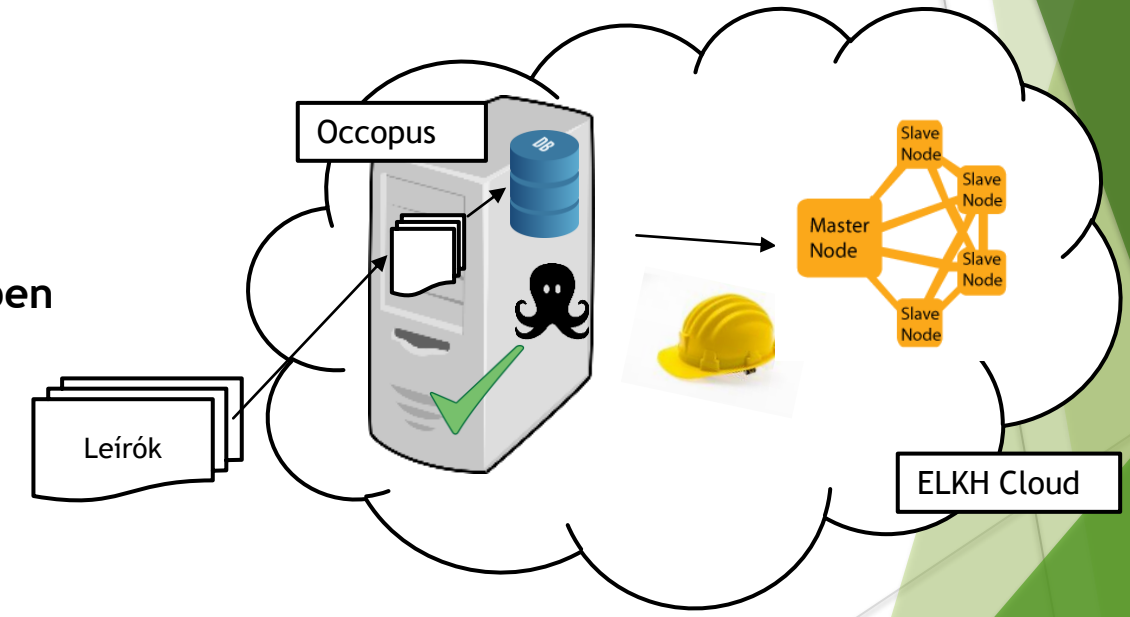
4. Lépés: **Leírók személyre szabása a virtuális gépen**
5. Lépés: **Occopus aktiválása**
    $ source ~/occopus/bin/activate
6. Lépés: Leírók importálása Occopus számára

    $ occopus-import nodes/node_definitions.yaml

7. Lépés: **Infrastruktúra kiépítése**

    $ occopus-build --parallelize infra-spark-cluster.yaml

8. Lépés: **Infrastruktúra használata**



# Infrastruktúra skálázás

Az Occopus eszköz lehetővé teszi a virtuális infrastruktúrák manuális fel- vagy leskálázását. A skálázás egy kétfázisú művelet: elsőként regisztráljuk a skálázási kérést, és ezután fel- vagy le skálázzuk a kiválasztott infrastruktúrát, új csomópontok építésével vagy meglévő csomópontok törlésével.

1. A skálázási kérelmet az **$ ocopus-scale** paranccsal tudjuk megadni.
   Pl.: `$ occopus-scale -n spark-slave -c COUNT -i INFRA_ID`, ahol:
   - -n (node): a skálázandó csomópont neve
   - -c (count): egy pozitív vagy negatív szám, amely megadja a skálázás irányát és annak nagyságát
   - -i (infrastructure): a cél infrastruktúra azonosítója ($occopus-maintain -l parancs segítségével lekérdezhető)
2. A skálázást az occopus-scale parancs kiadása után, az **$ occopus-maintain** paranccsal hajthatjuk végre a kéréseket. Példa: $ occopus-maintain – i INFRA_ID, ahol:
   - -i (infrastructure): a cél infrastruktúra azonosítója

# Infrastruktúra törlése

- Az infrastruktúra lebontásához szükségünk lesz az infrastruktúra ID-ra. Az ID-nak a beszerzésére az alábbi módokon van lehetőség:
  - Az infrastruktúra felépítés végénél az Occopus kiírja:

```
** 2021-02-17 17:26:33,127      spark-worker:
** 2021-02-17 17:26:33,128        50c7cfe9-4c3c-4928-81e6-d58323b1e2b2:
** 2021-02-17 17:26:33,128          198.124.39.182
** 2021-02-17 17:26:33,128        98a82e45-6bf2-4cb9-9ead-953be4435bd7:
** 2021-02-17 17:26:33,128          198.124.39.144
** 2021-02-17 17:26:33,129      spark-master:
** 2021-02-17 17:26:33,129        0b8269fe-78ec-47fc-8cdb-7195209e5123:
** 2021-02-17 17:26:33,129          198.124.39.67
c8553539-42c9-40b1-97bc-d53ab8947ca7
```

  - Az Occopus által menedzselt infrastruktúrák lekérése: $ occopus-maintain -l

```
(occopus) ubuntu@occo:~$ occopus-maintain -l
Using default configuration file: '/home/ubuntu/.occopus/occopus_config.yaml'
Using default authentication file: '/home/ubuntu/.occopus/auth_data.yaml'
** 2021-02-19 07:45:09,861      Starting up; PID = 12555
List of active infrastructure:
c8553539-42c9-40b1-97bc-d53ab8947ca7
```

# Infrastruktúra törlése

Az infrastruktúrát az **$ occopus-destroy -i <infraID>** paranccsal törölhetjük ki.

```
(occopus) ubuntu@occo:~$ occopus-destroy -i c8553539-42c9-40b1-97bc-d53ab8947ca7
Using default configuration file: '/home/ubuntu/.occopus/occopus_config.yaml'
Using default authentication file: '/home/ubuntu/.occopus/auth_data.yaml'
** 2021-02-17 19:26:21,112      Starting up; PID = 1787
** 2021-02-17 19:26:21,114      Start dropping infrastructure c8553539-42c9-40b1-97bc-d53ab8947ca7
** 2021-02-17 19:26:21,259      Dropping node 'spark-worker'/'50c7cfe9-4c3c-4928-81e6-d58323b1e2b2'
** 2021-02-17 19:26:22,440      Dropping node 'spark-worker'/'98a82e45-6bf2-4cb9-9ead-953be4435bd7'
** 2021-02-17 19:26:23,309      Dropping node 'spark-master'/'0b8269fe-78ec-47fc-8cdb-7195209e5123'
** 2021-02-17 19:26:24,184      Finished dropping infrastructure c8553539-42c9-40b1-97bc-d53ab8947ca7
```

# Apache Hadoop klaszter referencia architektúra

## Apache Hadoop cluster

This tutorial sets up a complete Apache Hadoop (version **3.3.0**) infrastructure. It contains a Hadoop Master node and Hadoop Slave worker nodes, which can be scaled up or down. To register Hadoop Slave nodes Consul is used.

### Features

- creating two types of nodes through contextualisation
- utilising health check against a predefined port
- using scaling parameters to limit the number of Hadoop Slave nodes
- manage cluster nodes with Consul

### Prerequisites

- accessing a cloud through an Occopus-compatible interface (e.g EC2, Nova, Azure, etc.)
- target cloud contains a base Ubuntu OS image with cloud-init support

### Download

You can download the example as tutorial.examples.hadoop-cluster .

Apache Hadoop klaszter leírása és letöltése:

https://occopus.readthedocs.io/en/devel/tutorial-bigdata-ai.html#apache-hadoop-cluster

# Összefoglalás

- Cél, hogy a magyar kutatók minél gyorsabban kezdhessék el az Big Data és MI-hez kapcsolódó kutató munkát az ELKH Cloud-on
- A különböző működőképes Big Data/MI környezetek létrehozását automatizáltan, az Occopus eszköz segítségével építjük ki
- A kiépítéshez nem szükséges mély informatikai, hálózati, vagy a szoftverek telepítéséhez és konfigurációjához szükséges tudás
- Az eddig összeállított MI környezetek (Hadoop, Spark, Jupyter, RStudio) referencia architektúra formájában elérhetők és kipróbálhatók az ELKH Cloud-on
- ELKH Cloud technikai support:

  info@science-cloud.hu

# Köszönöm a figyelmet!