Slurm referencia architektúra az ELKH Cloudon

Emődi Márk
emodi.mark@sztaki.hu

SZTAKI
Wigner
ELKH Eötvös Loránd Research Network

Emődi Márk


SZTAKI Párhuzamos és Elosztott Rendszerek Kutatólaboratórium – tudományos segédmunkatárs, kutató


Óbudai Egyetem – PhD hallgató, egyetemi tanársegéd


emodi.mark@sztaki.hu

Tartalomjegyzék

Mi az a Slurm?

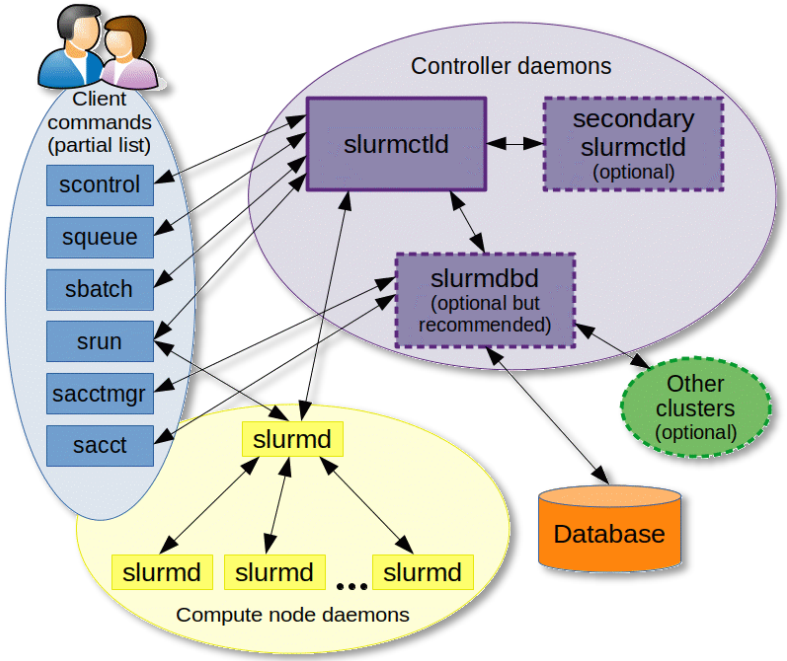
- **SLURM:** Slurm Workload Manager (Simple Linux Utility for Resource Management)
- A Slurm egy nyílt forráskódú, hibatűrő és nagymértékben skálázható erőforrás kezelő és feladatütemező rendszer.
  - Szuperszámítógépeken, elosztott erőforrásokon alkalmazzák
- Három fő funkcionalitást kínál:
  - Kizárólagos erőforrást képes rendelni adott felhasználóhoz adott időre
  - Keretrendszert (interfészt) biztosít a job-ok indításához, törléséhez és felügyeletéhez.
  - Felügyeli a feladatotokat
- Alaprendszer moduláris felépítésű
  - MPI, Konténeres környezet, saját logikán működő ütemező, stb.

A Slurm architektúrája

- **slurmd démon:** minden egyes (worker) node-on fut
- **slurmctld démon:** a master node-on fut (management node)
- **slurmdbd démon:** adatbázis, mely a felhasználókezelésért felel (opcionális)
- **Slurm parancsok:** a slurm-ön belüli parancsok a teljes klaszteren belül (master és worker) node-okon egyaránt futtathatóak

Azonos jelentéssel bírnak:

- Master node = Management node = Controller node
- Worker node = Compute node



Referencia architektúrák célja

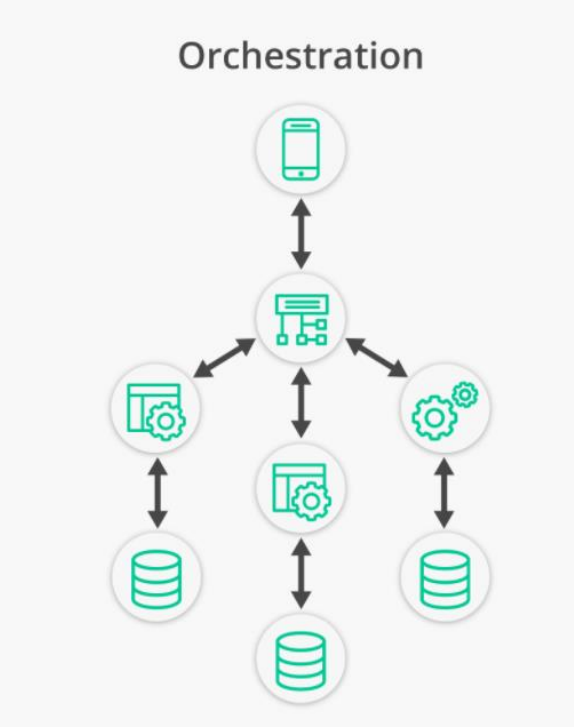
- Komplex virtuális infrastruktúrák kiépítése automatizált módon az ELKH Cloud-on. Az alábbi kritériumoknak magas minőségben való megfelelés:
  - Elosztott
  - Hibatűrő
  - Biztonságos
  - Skálázható
- Egyszerűsített igénybevétel
  - Minimális LINUX és felhő ismereti tudás szükséges

Terraform

ELKH Cloud SZTAKI Wigner ELKH Eötvös Loránd Research Network

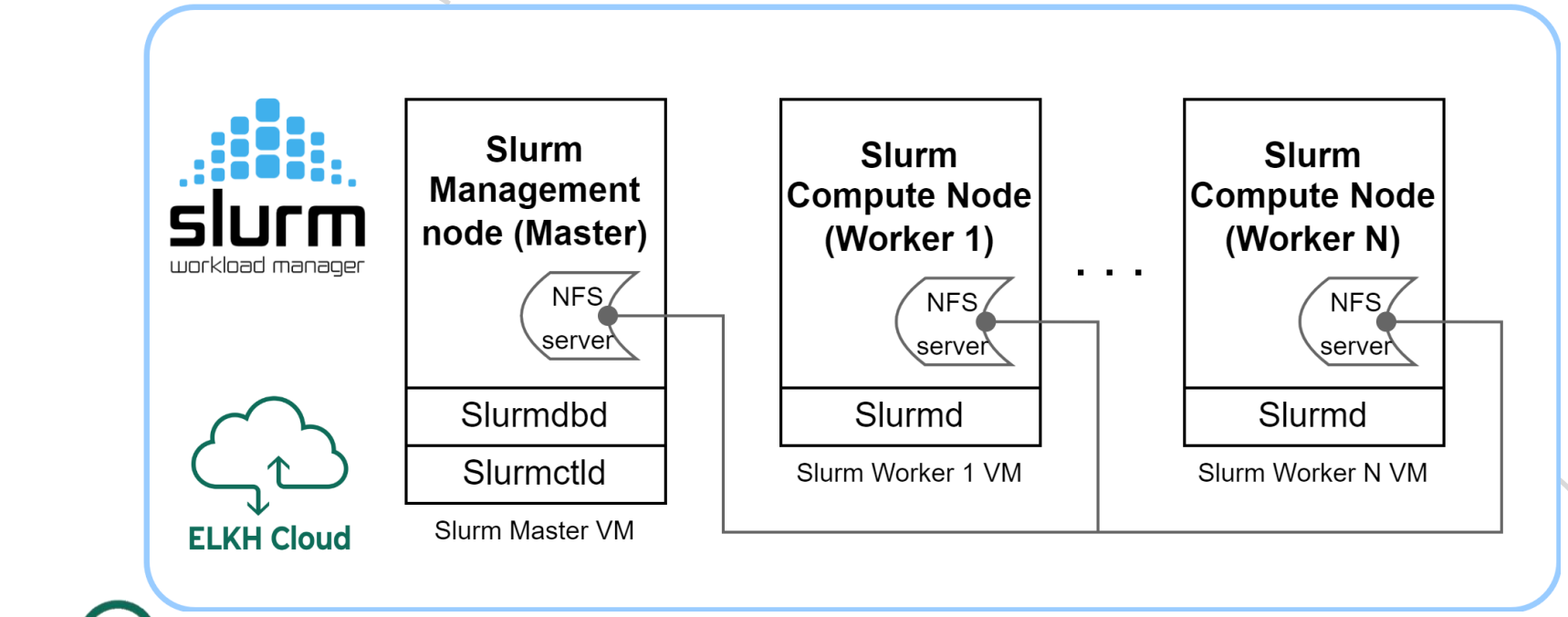
- Felhő objektumok menedzselése
- Komplex infrastruktúrák kiépítése a felhőben
- Nyílt forráskódú szoftver
- Tervezése és vizualizálása az infrastruktúráknak
- Változás nyomkövetése
- IaC (Infrastucture as Code)
- Moduláris felépítés
  - 2659 bővítmény (2022.11.23)
  - A ismertebb felhő interfészekhez nyújt támogatást

Terraform



Forrás: solace.com

A Slurm architektúrája



NFS: Network File Server

Megoldás használatának lépései

ELKH Cloud SZTAKI Wigner ELKH Eötvös Loránd Research Network

Felhasználó feladatköre:

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

1. Lépés: Terraform telepítés/konfigurálás a virtuális/saját gépen

0-1. lépés

Csak első alkalommal kell beállítani.

2. Lépés: Leírók letöltése
ELKH Cloud weboldala

3. Lépés: Tűzfalszabályok létrehozása (opcionális)
ELKH Cloud OpenStack felületén vagy leíró fájlban

4. Lépés: Leírók személyre szabása (konfigurálása)

5. Lépés: Terraform inicializálása az adott ref. architektúrán

```
$ terraform init
```

Referencia architektúránként az Terraformos gépen 1x kell beállítani.

2-4. lépés

6. Lépés: Infrastruktúra kiépítése (létrehozandó objektumok ellenőrzése)

```
$ terraform apply (--auto-approve)
```

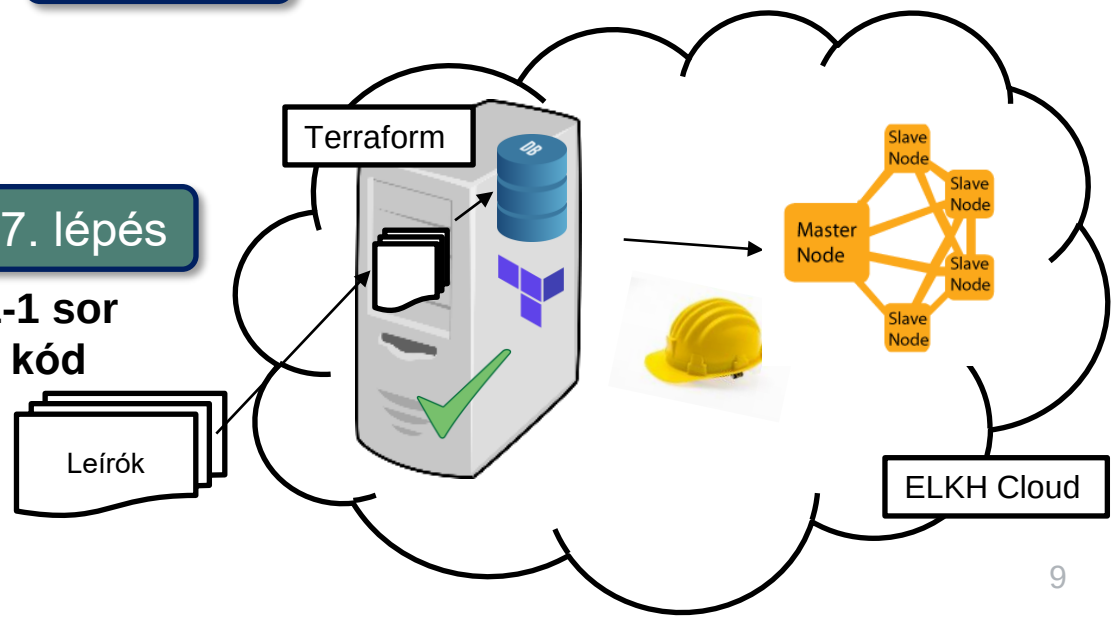
7. Lépés: Infrastruktúra használata

8. Lépés: Infrastruktúra leállítása

```
$ terraform destroy (--auto-approve)
```

5-7. lépés

1-1 sor kód



Terraform leírók

Authentikációs adatok
(auth_data.auto.tfvars)

- Felhő authentikációhoz szükséges adatok

```
auth_data = ({
    auth_url                      = "SET_YOUR_AUTH_URL"
    application_credential_id     = "SET_YOUR_APPLICATION_CREDENTIAL_ID"
    application_credential_secret = "SET_YOUR_APPLICATION_CREDENTIAL_SECRET"
})
```

Terraform leírók

ELKH Cloud SZTAKI Wigner ELKH Eötvös Loránd Research Network

Authentikációs adatok
(auth_data.auto.tfvars)

- Felhő authentikációhoz szükséges adatok

Erőforrás leírások
(resources.auto.tfvars)

- Erőforrás méretének meghatározása
- IP cím allokálás
- Képfájl meghatározás
- SSH kulcs érték
- Elnevezés módosítás
- Kötetméret
- Csomópont darabszáma

```
master_node = ({
  flavor_name  = "SET_YOUR_FLAVOR_NAME"
  floating_ip  = "SET_YOUR_FLOATING_IP"
  image_id     = "SET_YOUR_IMAGE_ID"
  key_pair     = "SET_YOUR_KEY_PAIR_NAME"
  name         = "slurm-master"
  network_name = "SET_YOUR_NETWORK_NAME"
  volume_size  = 20
})

worker_node = ({
  count        = 2
  flavor_name  = "SET_YOUR_FLAVOR_NAME"
  image_id     = "SET_YOUR_IMAGE_ID"
  key_pair     = "SET_YOUR_KEY_PAIR_NAME"
  name         = "slurm-worker"
  network_name = "SET_YOUR_NETWORK_NAME"
  volume_size  = 20
})
```

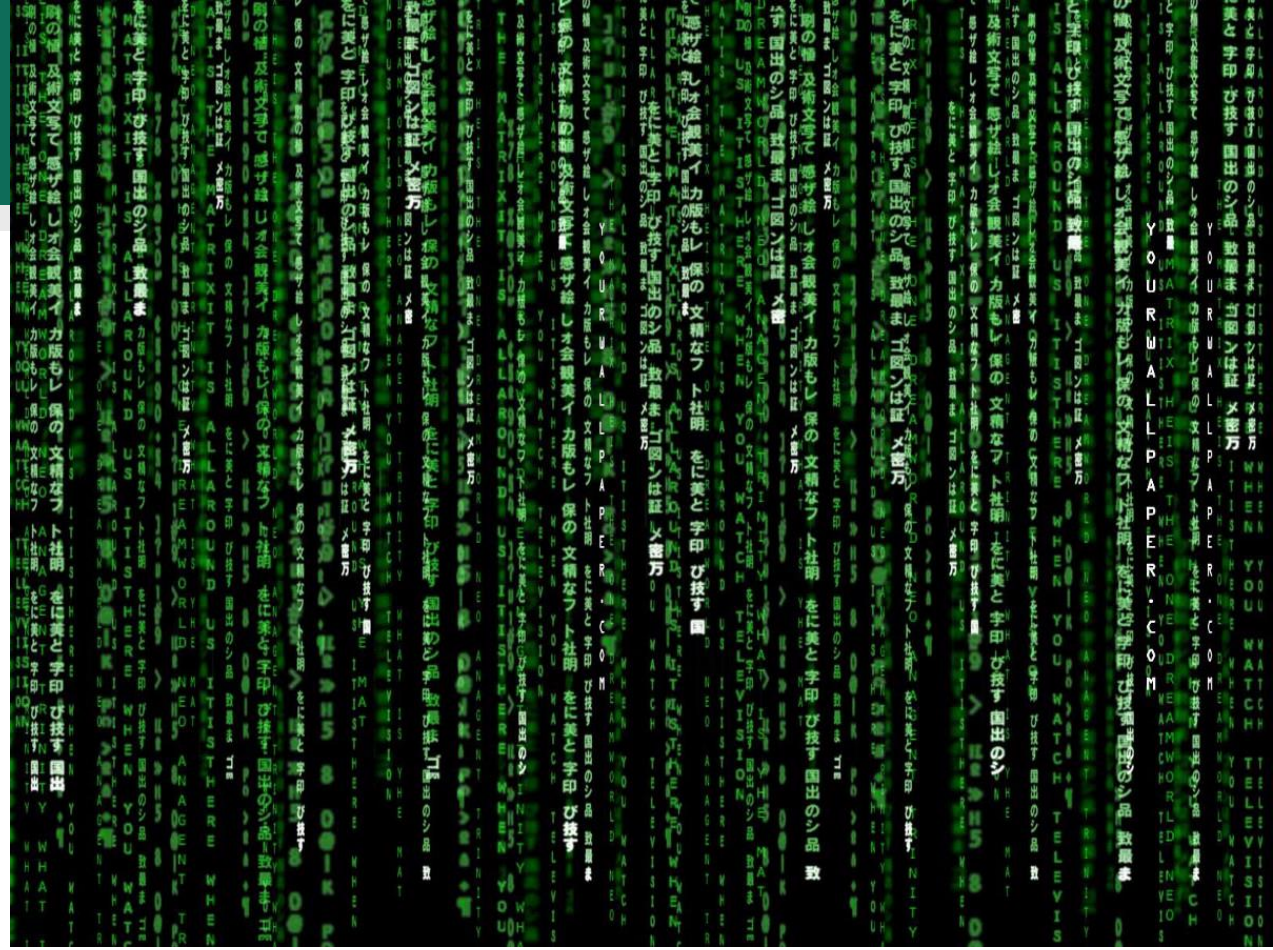
Terraform leírók

Authentikációs adatok
(auth_data.auto.tfvars)

- Felhő authentikációhoz szükséges adatok

Erőforrás leírások
(resources.auto.tfvars)

- Erőforrás méretének meghatározása
- IP cím allokálás
- Képfájl meghatározás
- SSH kulcs érték
- Elnevezés módosítás
- Kötetméret
- Csomópont darabszáma



Cloud-init fájlok
(scripts/)

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

Terraform leírók – Tűzfalszabályok hozzáadása (opcionális)

ELKH Cloud SZTAKI Wigner ELKH Eötvös Loránd Research Network

Tűzfalszabályok meghatározása (sec_groups.tf)

- A tűzfalszabályok létrehozásáért felelős leíró

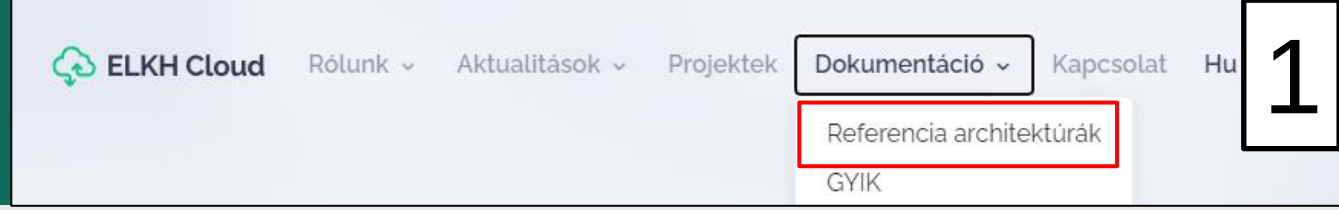
```
resource "openstack_compute_secgroup_v2" "ssh-public" {
  name        = "ssh-public"
  description = "Allow SSH (TCP/22) traffic from
anywhere (0.0.0.0/0). Managed by terraform."

  rule {
    from_port   = 22
    to_port     = 22
    ip_protocol = "tcp"
    cidr        = "0.0.0.0/0"
  }
}
```

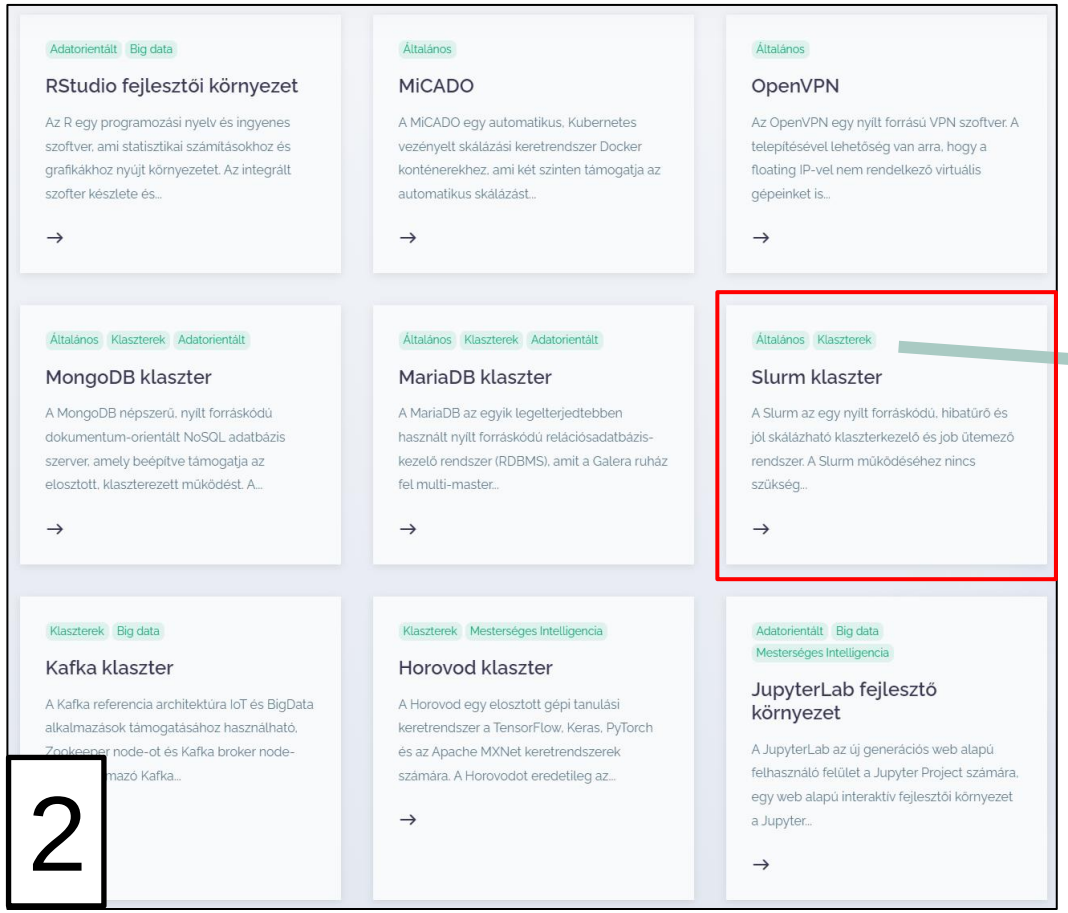
Infrastruktúra leírás (main.tf)

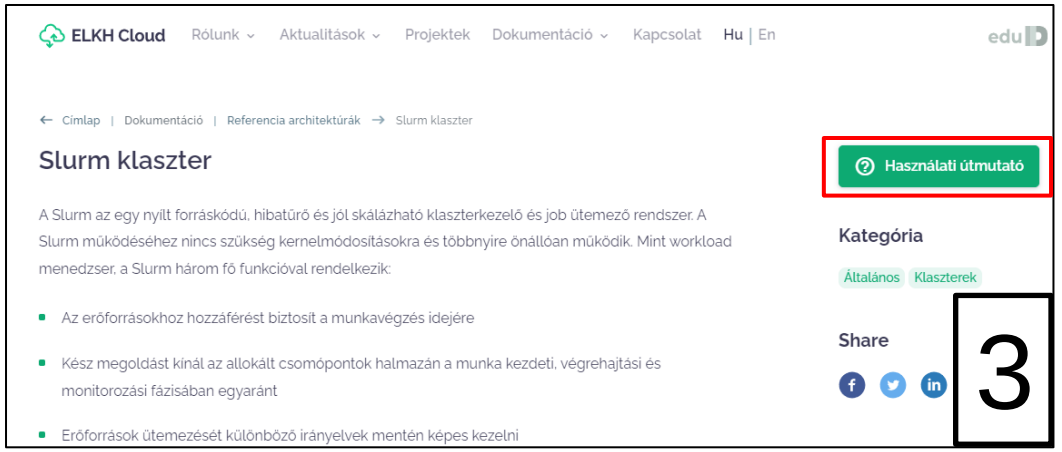
- Összefoglalja az architektúrához szükséges leírásokat

```
# resource specific configuration
resource "openstack_compute_instance_v2" "slurm-master"
{
  name            = var.master_node.name
  flavor_name     = var.master_node.flavor_name
  key_pair        = var.master_node.key_pair
  security_groups = [
    "${openstack_compute_secgroup_v2.ssh-public.id}",
  ]
}
```

2. lépés: leírók letöltése







https://science-cloud.hu/referencia-architekturak

2. lépés: leírók letöltése

ELKH Cloud

Slurm klaszter részletes (fejlesztői és felhasználói) leírása:
https://git.sztaki.hu/science-cloud/reference-architectures/slurm

README.md

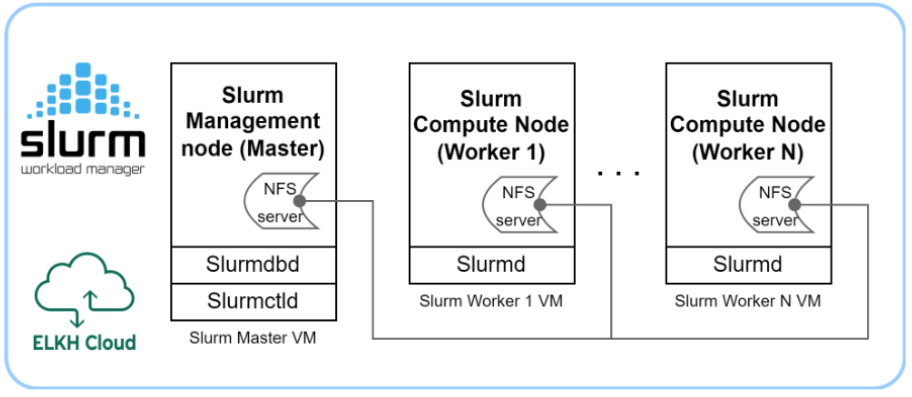
Slurm Reference Architecture

pipeline passed | Latest Release v0.1.0

Slurm is an open-source, fault-tolerant, and highly scalable cluster management and job scheduling system for large and small Linux clusters. Slurm requires no kernel modifications for its operation and is relatively self-contained. As a cluster workload manager, Slurm has three key functions:

First, it allocates exclusive and/or non-exclusive access to resources (compute nodes) to users for some duration of time so they can perform work. Second, it provides a framework for starting, executing, and monitoring work (normally a parallel job) on the set of allocated nodes. Finally, it arbitrates contention for resources by managing a queue of pending work. The reference architecture provides a Slurm (version 19.05.5) cluster. It contains a Slurm Management (master) node and Slurm Compute (worker) node(s).

You can provision the reference architecture using Terraform.



Deployment using Terraform

Prerequisites

- Configuring an SSH key on the ELKH Cloud
- Terraform and Ansible are required
  - You can install them by running the commands below, or omit the installation and use the RefArch Toolset Docker image, which includes these tools

Deployment

1. Download and extract descriptor files:

```
wget https://git.sztaki.hu/science-cloud/reference-architectures/slurm/-/archive/main/slurm-main.tar.gz -O slurm-ra.tar.gz
tar -zxvf slurm-ra.tar.gz
```

2. Enter the directory of OpenStack descriptors:

```
cd slurm-main/terraform_openstack
```

3. Customize OpenStack descriptors:

- An **application credential** and your **authentication URL** is needed to enable Terraform to access your resources
  - You can create an application credential on the OpenStack web interface under **Identity** > **Application Credentials**. Please note that application credentials are valid only for the project selected at the time of their creation.
  - You can find your authentication URL under **Project** > **API Access**, as the endpoint of the entry labelled 'Identity'

3. lépés: Tűzfalszabályok összegzése (opcionális)

ELKH Cloud SZTAKI Wigner ELKH Eötvös Loránd Research Network

Tűzfalszabály Slurm referencia architektúrákra esetén (Opcionális)

```
# resource specific configuration
resource "openstack_compute_instance_v2" "slurm-master" {
  name            = var.master_node.name
  flavor_name     = var.master_node.flavor_name
  key_pair        = var.master_node.key_pair
  security_groups = [
    "SET_YOUR_FIREWALL_ROLE_NAME",
  ]
}
```

	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

Kimenő forgalom

(opcionális)
Pl. lekérdezés (harmadik féltől)

$ curl ifconfig.me
92.21.242.26

Slurm cluster oldali kommunikáció

SSH hozzáférés

4. lépés: Leírók személyre szabása – klaszter méretének beállítása

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

Ehhez szerkessze át az **resources.auto.tfvars** fájlt és módosítsa a count paramétert.

- A skálázás az az intervallum, amelyben a csomópontok száma megváltozhat. Jelenleg az alapértelmezett érték 2-re van állítva (ami az indításkor a kezdeti szám lesz).
- Ne feledje, hogy a Terraformnak legalább egy csomópontot el kell indítania minden egyes csomóponttípusból, hogy az infrastruktúra megfelelően működjön, valamint a skálázás csak a Slurm dolgozó (worker) csomópontokon alkalmazható ebben a példában!

Haladó felhasználók itt tudják átírni az eltérő verziójú Ubuntu rendszerekbe beépített package telepítők verziószámait. További információk: https://pkgs.org/download/slurmctld

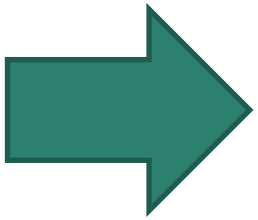
```
master_node = ({
  flavor_name  = "SET_YOUR_FLAVOR_NAME"
  floating_ip  = "SET_YOUR_FLOATING_IP"
  image_id     = "SET_YOUR_IMAGE_ID"
  key_pair     = "SET_YOUR_KEY_PAIR_NAME"
  name         = "slurm-master"
  network_name = "SET_YOUR_NETWORK_NAME"
  volume_size  = 20
})

worker_node = ({
  count        = 2
  flavor_name  = "SET_YOUR_FLAVOR_NAME"
  image_id     = "SET_YOUR_IMAGE_ID"
  key_pair     = "SET_YOUR_KEY_PAIR_NAME"
  name         = "slurm-worker"
  network_name = "SET_YOUR_NETWORK_NAME"
  volume_size  = 20
})
```

4. lépés: Leírók személyre szabása

Csomópont definíciós fájl (**terraform_openstack/resources.auto.tfvars**)
Támogatott operációs rendszer: Ubuntu 20.04

Felhőspecifikus azonosítók begyűjtése:

```
master_node = ({
  flavor_name  = "SET_YOUR_FLAVOR_NAME"
  floating_ip  = "SET_YOUR_FLOATING_IP"
  image_id     = "SET_YOUR_IMAGE_ID"
  key_pair     = "SET_YOUR_KEY_PAIR_NAME"
  name         = "slurm-master"
  network_name = "SET_YOUR_NETWORK_NAME"
  volume_size  = 20
})

worker_node = ({
  count        = 2
  flavor_name  = "SET_YOUR_FLAVOR_NAME"
  image_id     = "SET_YOUR_IMAGE_ID"
  key_pair     = "SET_YOUR_KEY_PAIR_NAME"
  name         = "slurm-worker"
  network_name = "SET_YOUR_NETWORK_NAME"
  volume_size  = 20
})
```



```
master_node = ({
  flavor_name  = "m2.medium"
  floating_ip  = ""
  image_id     = "8b693880-6273-44b0-91ab-f0e9403dff69"
  key_pair     = "emodi"
  name         = "slurm-master"
  network_name = "default"
  volume_size  = 50
})

worker_node = ({
  count        = 2
  flavor_name  = "m2.medium"
  image_id     = "8b693880-6273-44b0-91ab-f0e9403dff69"
  key_pair     = "emodi"
  name         = "slurm-worker"
  network_name = "default"
  volume_size  = 20
})
```

5. lépés: Terraform inicializálása

Az 5. lépésben inicializáljuk a Terraform bővítményeket és a környezetet:

```
ubuntu@:reference-arch:~/slurm-main/terraform_openstack$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding terraform-provider-openstack/openstack versions matching "1.48.0"...
- Finding latest version of hashicorp/random...
- Installing terraform-provider-openstack/openstack v1.48.0...
- Installed terraform-provider-openstack/openstack v1.48.0 (self-signed, key ID 4F80527A391BEFD2)
- Installing hashicorp/random v3.4.3...
- Installed hashicorp/random v3.4.3 (signed by HashiCorp)
…
Terraform has been successfully initialized!
```

6. lépés: Infrastruktúra kiépítése

ELKH Cloud SZTAKI Wigner ELKH Eötvös Loránd Research Network

Az 6. lépésben kiépítjük a referencia architektúrát:

```
ubuntu@:reference-arch:~/slurm-main/terraform_openstack$ terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated
with the
following symbols:
  + create
Terraform will perform the following actions:
  # openstack_compute_instance_v2.slurm-master will be created
  + resource "openstack_compute_instance_v2" "slurm-master" {
…
      + access_ip_v4        = (known after apply)
      + id                  = (known after apply)
      + image_id            = "72d56cf7-04da-4532-bf27-9047cdbee3f5"
      + key_pair            = "emodi"
      + name                = "slurm-master"
…
Plan: 3 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value:
```

6. lépés: Infrastruktúra kiépítése

ELKH Cloud SZTAKI Wigner ELKH Eötvös Loránd Research Network

- Sikeres lefutás után a virtuális gépek IP címei valamint elnevezései megjelennek a kimeneten
- Az infrastruktúra leíró adatok egy fájlban tárolódnak a mappában (state file).

```
openstack_compute_instance_v2.slurm-master: Creating...
openstack_compute_instance_v2.slurm-master: Still creating... [10s elapsed]
openstack_compute_instance_v2.slurm-master: Creation complete after 16s [id=3e464f98-4e32-44e8-b374-
68651e0488bb]
openstack_compute_instance_v2.slurm-worker[0]: Creating...
openstack_compute_instance_v2.slurm-worker[1]: Creating...
openstack_compute_instance_v2.slurm-worker[1]: Still creating... [10s elapsed]
openstack_compute_instance_v2.slurm-worker[0]: Still creating... [10s elapsed]
openstack_compute_instance_v2.slurm-worker[0]: Creation complete after 13s [id=56ecd361-5e4b-4b0d-bca2-
6f38219b9e4f]
openstack_compute_instance_v2.slurm-worker[1]: Creation complete after 16s [id=aa70a361-041b-488f-8d61-
6c929c24bf6d]

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.

Outputs:

slurm-master = "slurm-master"
slurm-master-ip = "192.168.0.55"
slurm-workers = [
  "slurm-worker-1",
  "slurm-worker-2",
]
slurm-workers-ips = [
  "192.168.0.166",
  "192.168.0.101",
]
```

6. lépés: infrastruktúra kiépítése

ELKH Cloud SZTAKI Wigner ELKH Eötvös Loránd Research Network

1

Instances

Instance ID = | Filter | Launch Instance | Delete Instances | More Actions

Displaying 3 items

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
slurm-master-3b804a5	Ubuntu 20.04	193.6.16.174	m1.medium	emodi	Build	nova	Spawning	No State	0 minutes	Associate Floating IP

2

Instances

Instance ID = | Filter | Launch Instance | Delete Instances | More Actions

Displaying 3 items

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
slurm-master-3b804a5	Ubuntu 20.04	193.6.16.174	m1.medium	emodi	Active	nova	None	Running	0 minutes	Create Snapshot

6. lépés: infrastruktúra kiépítése

ELKH Cloud SZTAKI Wigner ELKH Eötvös Loránd Research Network

3

Instances

Instance ID = | Filter | Launch Instance | Delete Instances | More Actions

Displaying 5 items

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
slurm-worker-d8911ce	Ubuntu 20.04	193.6.16.158	m1.medium	emodi	Active	nova	None	Running	0 minutes	Create Snapshot
slurm-worker-f284834	Ubuntu 20.04	193.6.16.159	m1.medium	emodi	Active	nova	None	Running	0 minutes	Create Snapshot
slurm-master-3b804a5	Ubuntu 20.04	193.6.16.174	m1.medium	emodi	Active	nova	None	Running	3 minutes	Create Snapshot

7. lépés: Az infrastruktúra használata

- A Slurm Master-re kapcsolódjunk rá SSH-n keresztül külső IP cím segítségével.
- Néhány alapvető parancs a Slurm-ben:
  - sinfo: áttekintést ad a klaszter által kínált erőforrásokról
    (akkor ad vissza kimenetet, ha van már legalább 1db worker állomás a klaszterben)
  - squeue: az erőforrások jelenleg mely job(ok)-hoz vannak hozzárendelve
- Az sinfo alapvetően a rendelkezésre álló partíciókat listázza.
  Egy partíció számítási állomások egy logikai csoportját foglalja magában.

```
ubuntu@slurm-master-61f21c0:~$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
primary*       up   infinite      2   idle slurm-worker-6c7eabe,slurm-worker-dff61eb
```

- Az ábrán a mi partíciónk a primary*. A csillag az alapértelmezett partícióra utal.

```
root@slurm-master-9b63b65:/storage# srun -J test.job -n 2 --ntasks-per-node 1 -o /storage/%j.txt hostname
root@slurm-master-9b63b65:/storage# ls
2.txt  slurm
root@slurm-master-9b63b65:/storage# cat 2.txt
slurm-worker-bfce264
slurm-worker-2cdcff9
```

Az infrastruktúra skálázása

A Terraform segítségével az infrastruktúrák felfelé vagy lefelé skálázhatóak.

- **Felfelé skálázás:** amikor egy vagy több új node-ot adunk a klaszterhez
- **Lefelé skálázás:** amikor egy vagy több meglévő node-ot törlünk a klaszterből

Ehhez a **resources.auto.tfvars** fájlban a worker_nodeokhoz tartozó count paramétert szükséges módosítanunk.

A skálázási kérelem végrehajtása:

- `$ terraform apply` parancs kiadásával lehetséges

```
master_node = ({
  flavor_name  = "SET_YOUR_FLAVOR_NAME"
  floating_ip  = "SET_YOUR_FLOATING_IP"
  image_id     = "SET_YOUR_IMAGE_ID"
  key_pair     = "SET_YOUR_KEY_PAIR_NAME"
  name         = "slurm-master"
  network_name = "SET_YOUR_NETWORK_NAME"
  volume_size  = 20
})

worker_node = ({
  count        = 3
  flavor_name  = "SET_YOUR_FLAVOR_NAME"
  image_id     = "SET_YOUR_IMAGE_ID"
  key_pair     = "SET_YOUR_KEY_PAIR_NAME"
  name         = "slurm-worker"
  network_name = "SET_YOUR_NETWORK_NAME"
  volume_size  = 20
})
```

Az infrastruktúra skálázása

Felfelé skálázáskor a Slurm referencia architektúra működése:

- A Terraform létrehozza a kért worker node-okat
- A worker node-ok telepítése és beállítása után csatlakoznak a master node-hoz
- A master újraindítja a Slurm szolgáltatásokat, majd megjelenik az új node

Lefelé skálázáskor a Slurm referencia architektúra működése:

- A Terraform törli a kért worker node-okat
- A master pár perc múlva érzékeli, hogy nem érhetőek el a worker node-ok
  - törlésre kerülnek a node listából

Infrastruktúra törlése

ELKH Cloud SZTAKI Wigner ELKH Eötvös Loránd Research Network

- Az infrastruktúra lebontásához szükséges belépni az architektúra mappájába (slurm/terraform_openstack), ahol az állapot fájl kerül tárolásra.
- A `$ terraform destroy` parancs eltávolítja az erőforrásokat
  - Összegzés az aktuális és törlésre kerülő objektumokról

```
$ terraform destroy
openstack_compute_instance_v2.slurm-master: Refreshing state... [id=3e464f98-4e32-44e8-b374-68651e0488bb]
openstack_compute_instance_v2.slurm-worker[1]: Refreshing state... [id=aa70a361-041b-488f-8d61-6c929c24bf6d]
openstack_compute_instance_v2.slurm-worker[0]: Refreshing state... [id=56ecd361-5e4b-4b0d-bca2-6f38219b9e4f]

Terraform used the selected providers to generate the following execution plan. Resource actions are
indicated with the
following symbols:
  - destroy

Terraform will perform the following actions:

  # openstack_compute_instance_v2.slurm-master will be destroyed
  - resource "openstack_compute_instance_v2" "slurm-master" {
…
      - access_ip_v4        = "192.168.0.55" -> null
      - flavor_id           = "cf0f1cb2-43f5-4d36-99d0-8ca2e4380517" -> null
…
}
 # openstack_compute_instance_v2.slurm-worker[0] will be destroyed
  - resource "openstack_compute_instance_v2" "slurm-worker" {…}
 # openstack_compute_instance_v2.slurm-worker[1] will be destroyed
  - resource "openstack_compute_instance_v2" "slurm-worker" {…}
```

SZTAKI ÉS WIGNER ÁG KÖZTI KÜLÖNBSÉGEK

ELKH Cloud SZTAKI Wigner ELKH Eötvös Loránd Research Network

- VPN kapcsolat szükséges
  - Felhő API
  - Virtuális gépek elérése
- Floating IP (külső IP cím) megadása
- Provider szekció kiegészítése insecure paraméterrel
  - Kapcsolódás az interfészhez védett marad!
    - VPN, Self-signed certificate

```
provider "openstack" {
  auth_url                      = var.auth_data.auth_url
  application_credential_id     = var.auth_data.application_credential_id
  application_credential_secret = var.auth_data.application_credential_secret
  # If your cloud provider uses self-signed cert, use the insecure flag
  # insecure = true
}
```

Összefoglalás

- A Slurm referencia architektúra működése
- A referencia architektúra kiépítésének menetét
- A Slurm használata és alapvető parancsok
- A különböző működőképes Big Data/MI/konténer/workload menedzser környezetek létrehozását automatizáltan, a Terraform eszköz segítségével építjük ki
- A kiépítéshez nem szükséges mély informatikai, hálózati, vagy a szoftverek telepítéséhez és konfigurációjához szükséges tudás
- Az eddig összeállított környezetek (Hadoop, Spark, Tensorflow, Slurm, Horovod, Kafka, Kubernetes, MongoDB, MariaDB, OpenVPN, stb.) referencia architektúra formájában elérhetők és kipróbálhatók az ELKH Cloud-on
- ELKH Cloud technikai support:

  info@science-cloud.hu

Köszönöm a figyelmet!

SZTAKI

ELKH | Eötvös Loránd Research Network

Wigner

www.elkh.org