

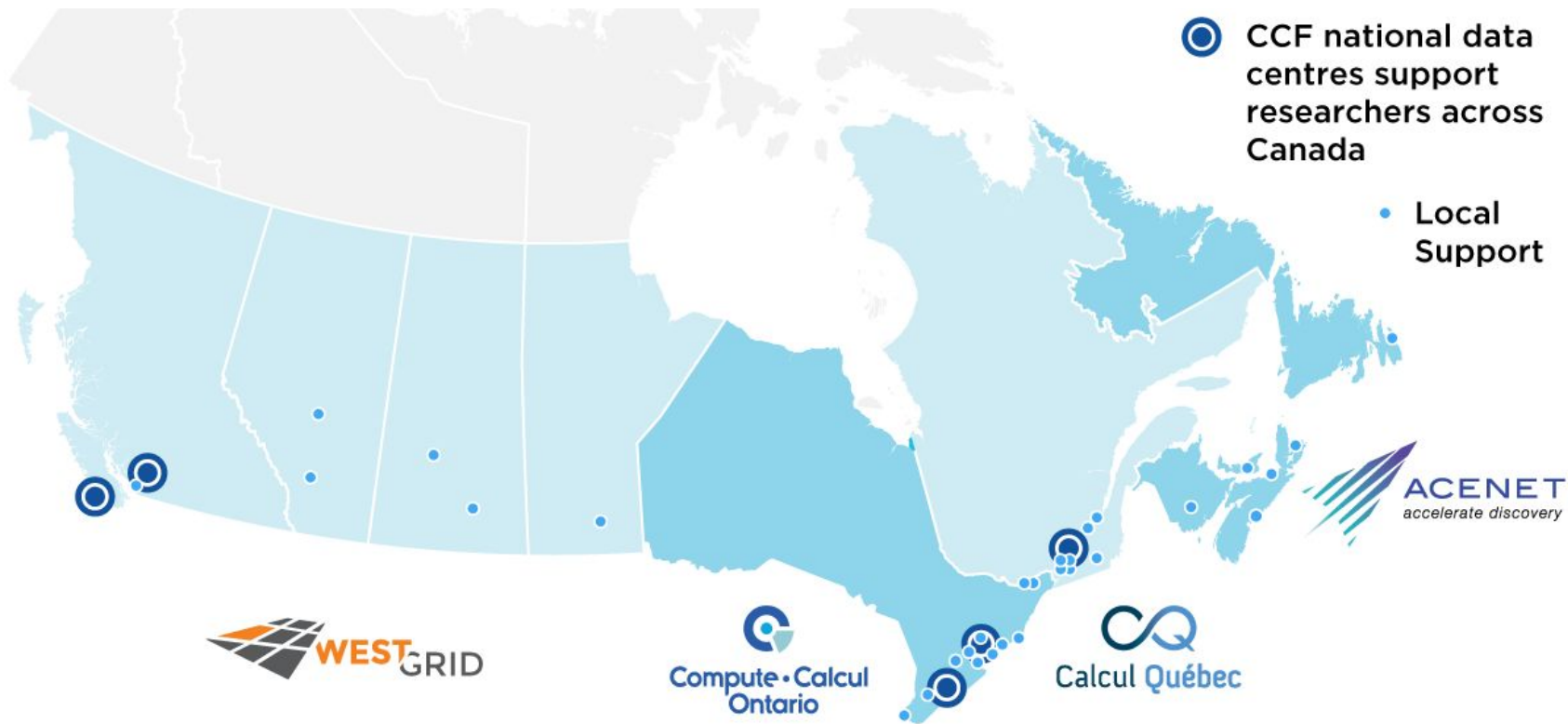
# Magic Castle

## Terraforming the Cloud for HPC

Félix-Antoine Fortin, 6th EasyBuild User Meeting



# Canada Digital Research Infrastructure



# Education and Training in Compute Canada

---

- Over 150 workshops / year
- Most workshops use the HPC software environment
- HPC clusters require an account
- Account creation process can take a few days

Could we replicate the HPC environment for training?



**Solution**

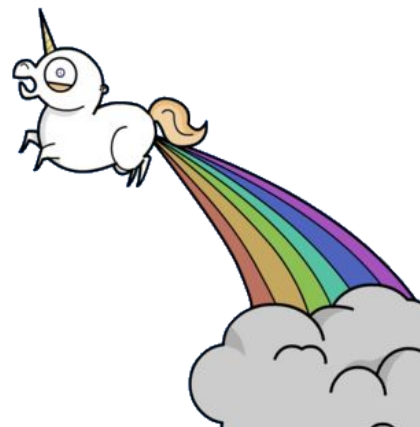
# Magic Castle

---

Open source project that instantiates a Compute Canada cluster replica in any major cloud with Terraform and Puppet

- Create instances
  - Management nodes
  - Login nodes
  - Compute nodes
- Create volumes, network, network acls
- Create certificates, dns records, passwords
- Configuration done via input parameters

[https://github.com/computecanada/magic\\_castle](https://github.com/computecanada/magic_castle)



# Terraform



- Tool for building, changing, and versioning infrastructure
- Infrastructure is described using a high-level configuration syntax.
- Create resources that can then be setup by a config management tool.

# Puppet



- Config management tool used for deploying, configuring and managing servers.
- Define configurations for each host
- Continuously check whether the required configuration is in place and is not altered

# Other Cloud HPC Cluster Projects

---

- `AWS ParallelCluster` [AWS]
- `Cluster in the cloud` [Ansible - AWS, GCP, Oracle]
- `Elasticcluster` [AWS, GCP, OpenStack]
- `Slurm on Google Platform` [GCP]
- `NVIDIA DeepOps` [Ansible only]
- `StackHPC Ansible Role OpenHPC` [Ansible only]

# Magic Castle Founding Guidelines

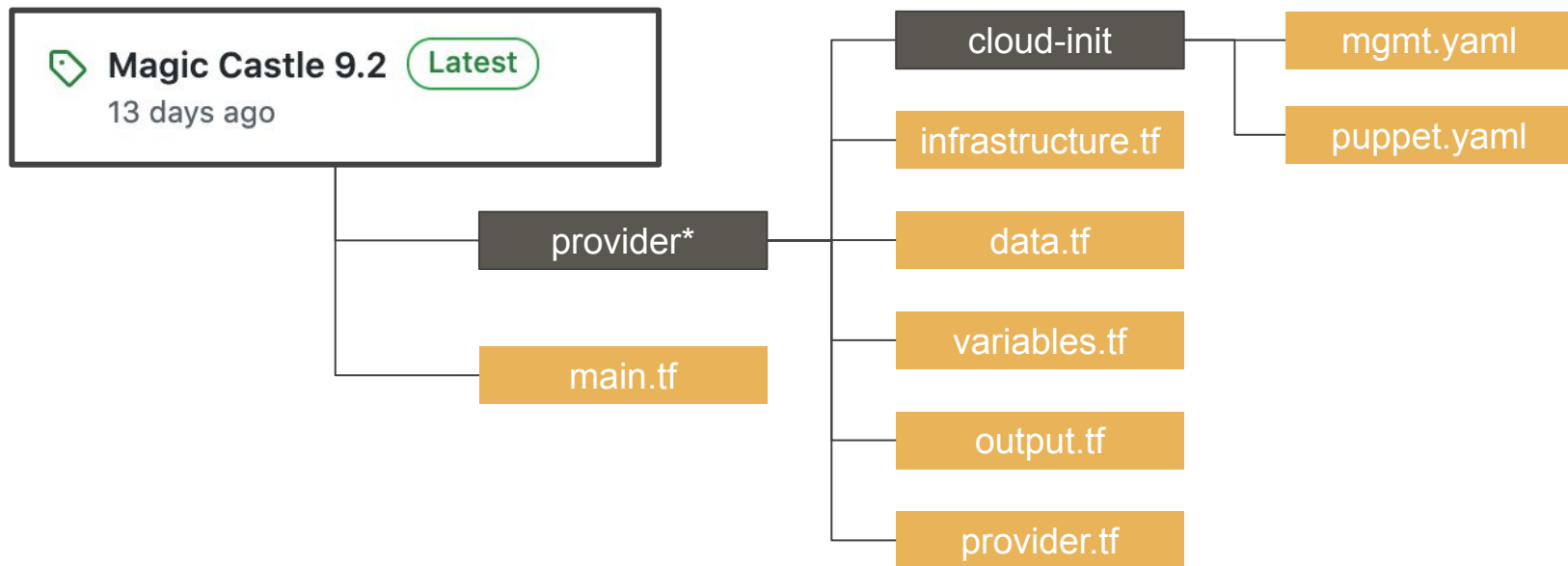
---

1. No custom command-line interface, Terraform is the CLI
2. Manage configuration with Puppet to encourage reuse of modules within Compute Canada
3. SELinux should always be enforced
4. Maintain an extensive user documentation



# Overview of a Magic Castle Release

---

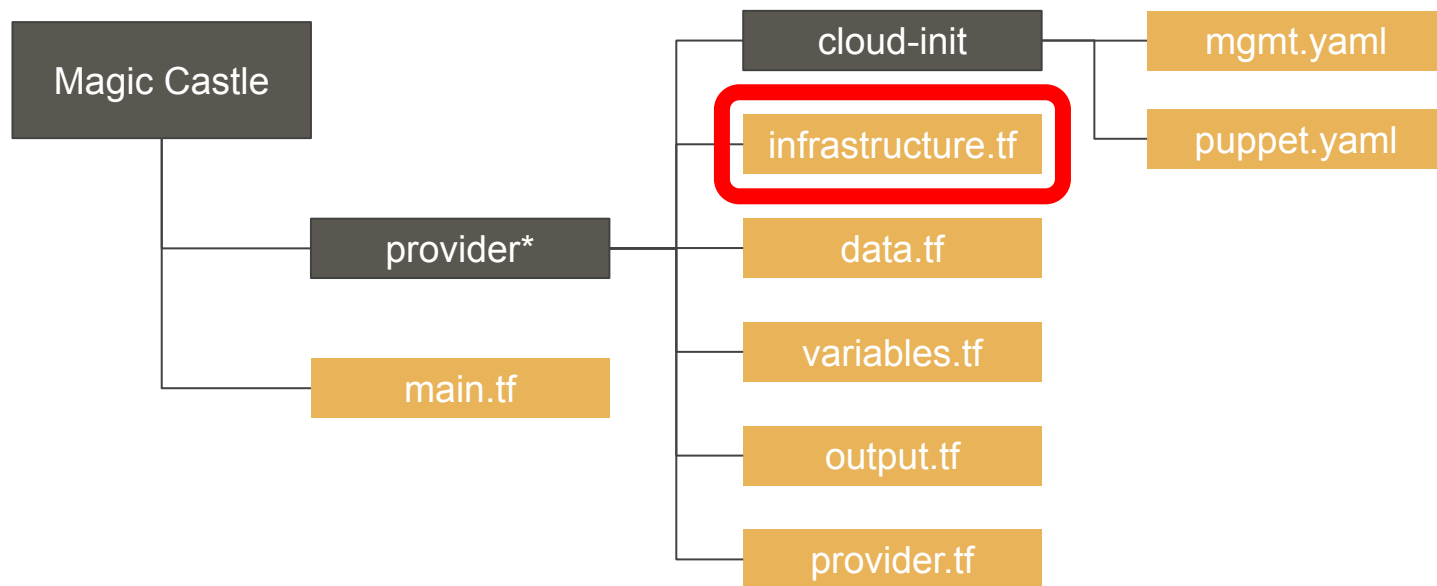


\*could be any in [aws, azure, gcp, openstack, ovh]

# Infrastructure

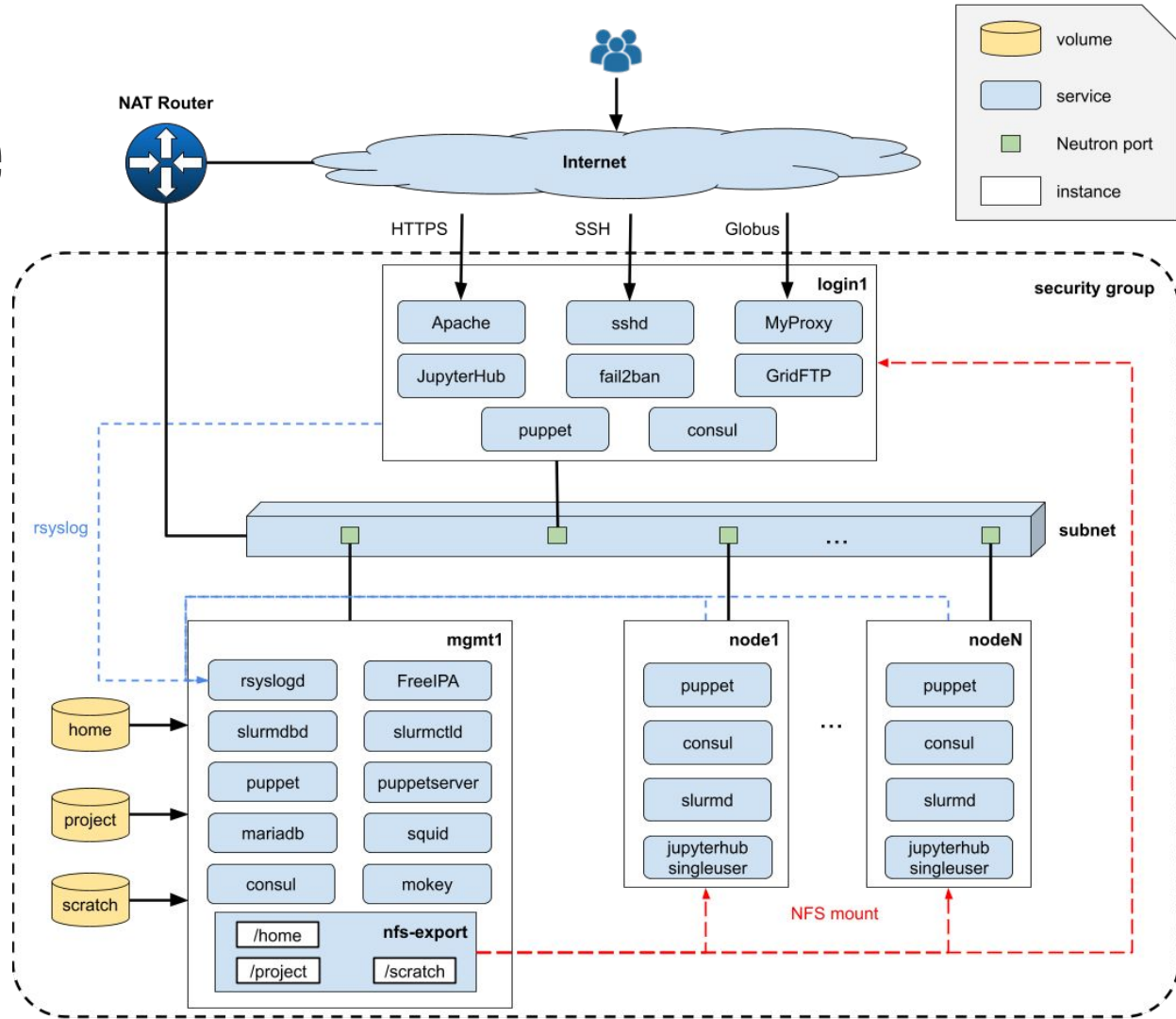
# Overview of a Magic Castle Release

---

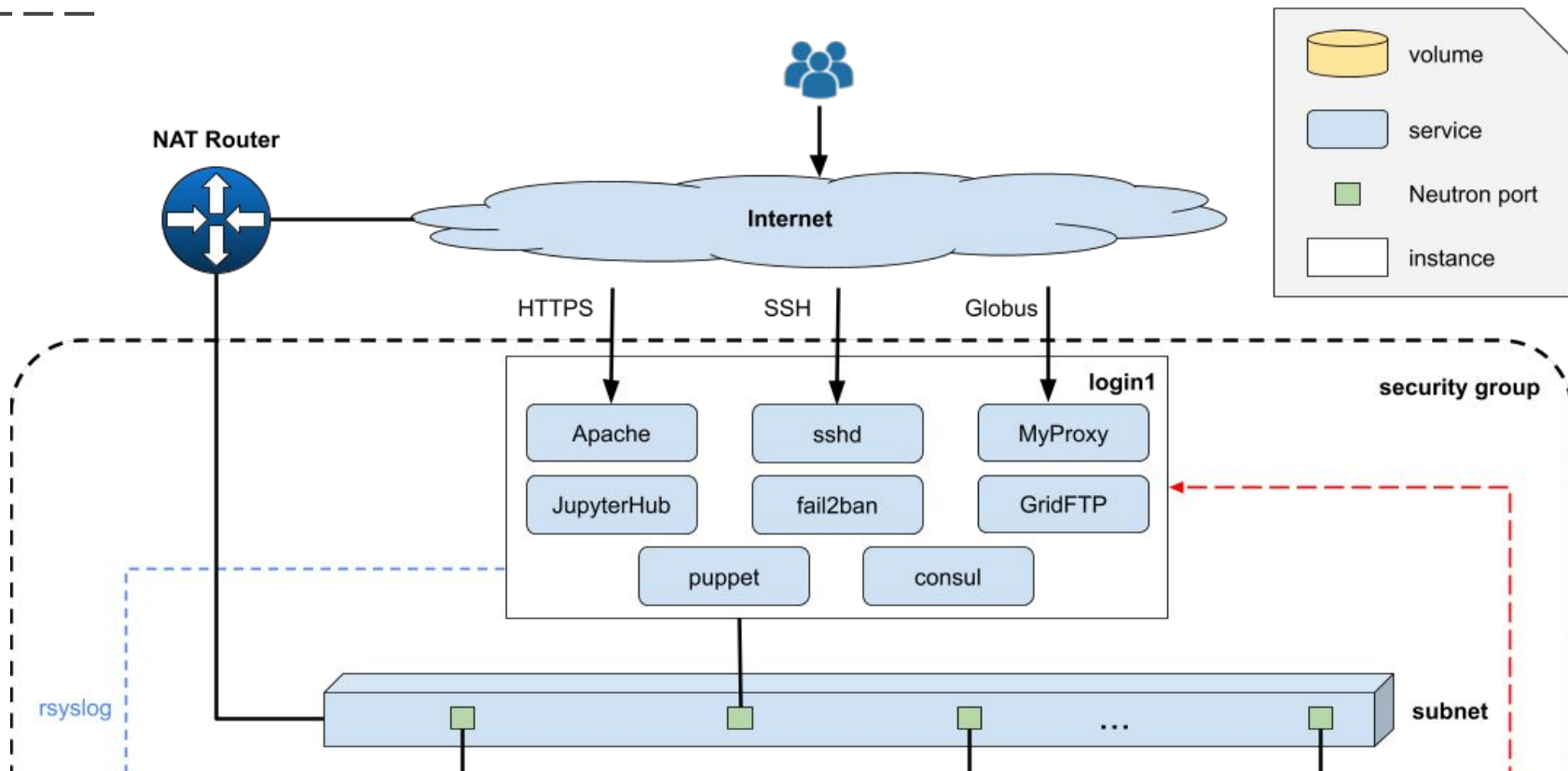


\*could be any in [aws, azure, gcp, openstack, ovh]

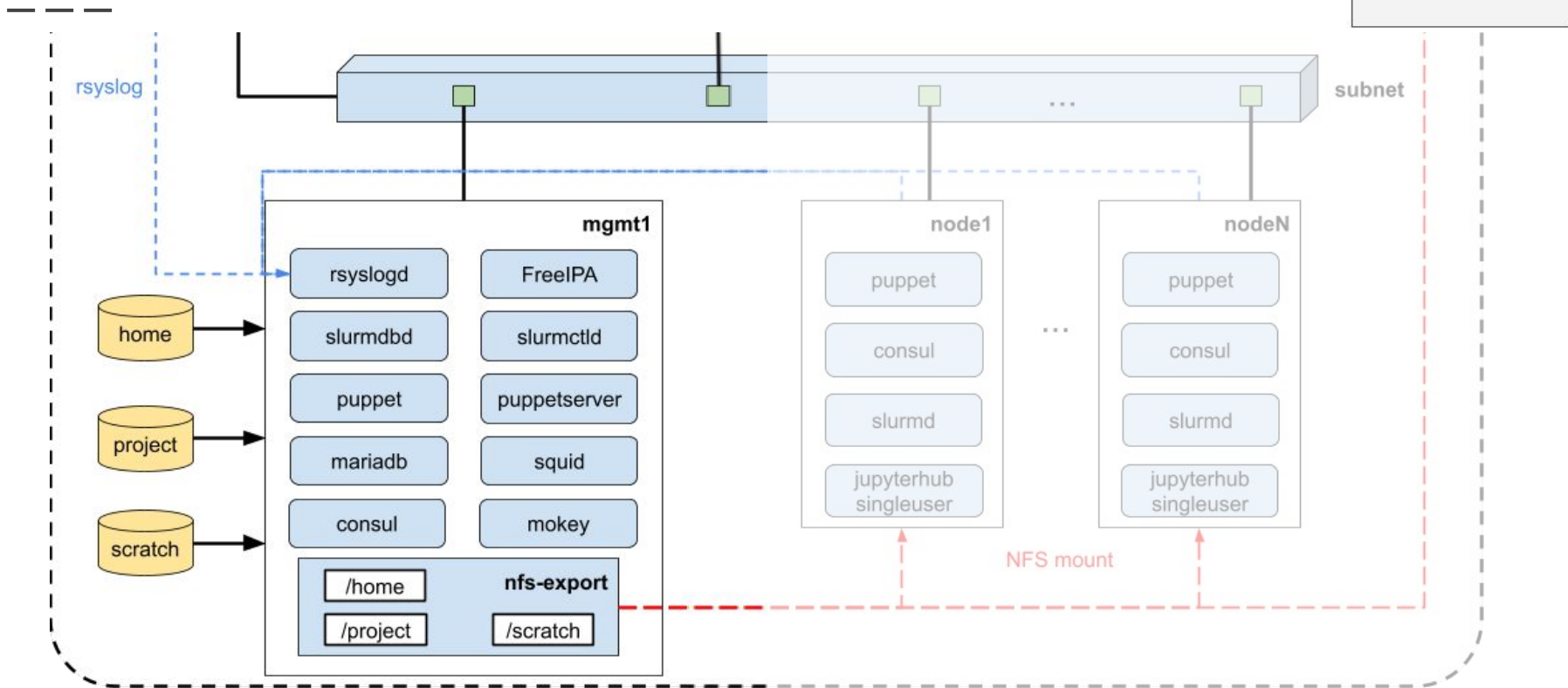
# Architecture



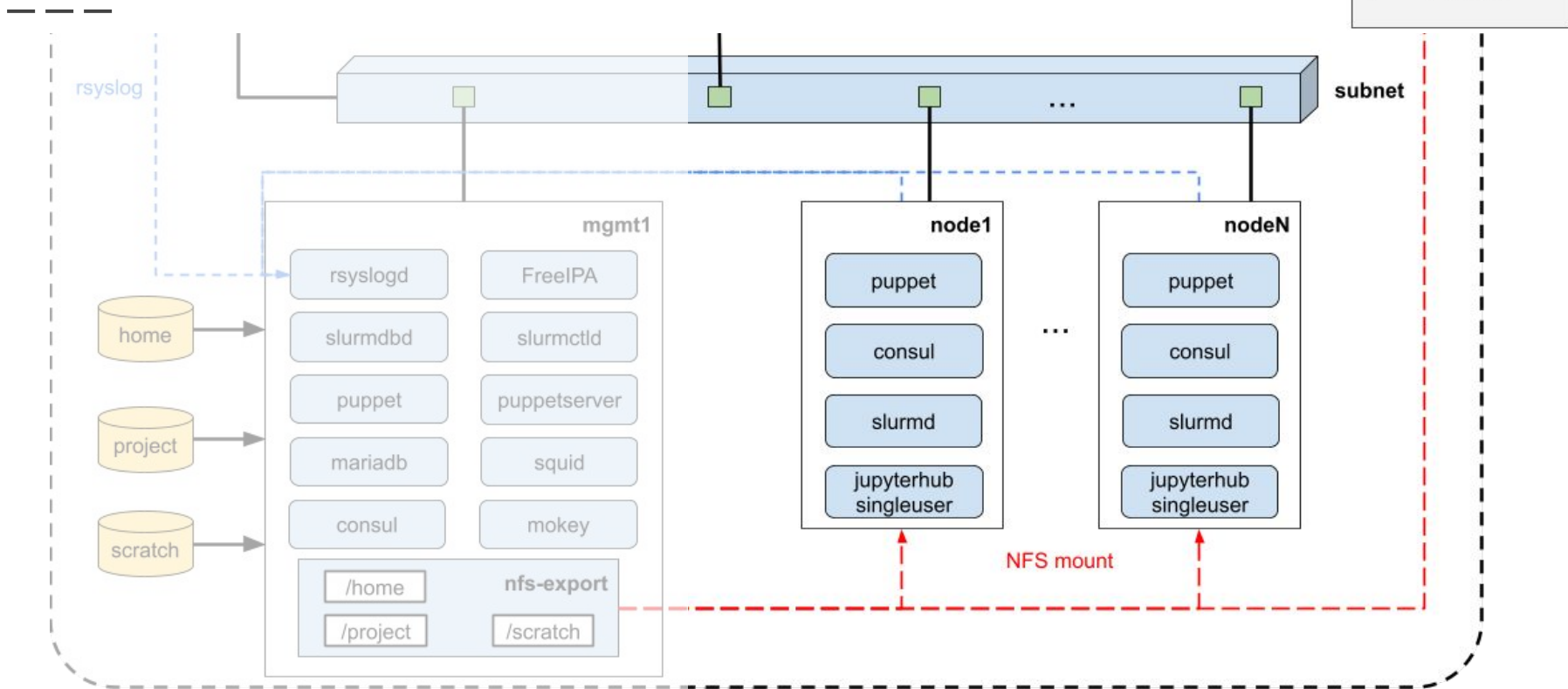
# Architecture - login nodes



# Architecture - management nodes



# Architecture - compute nodes

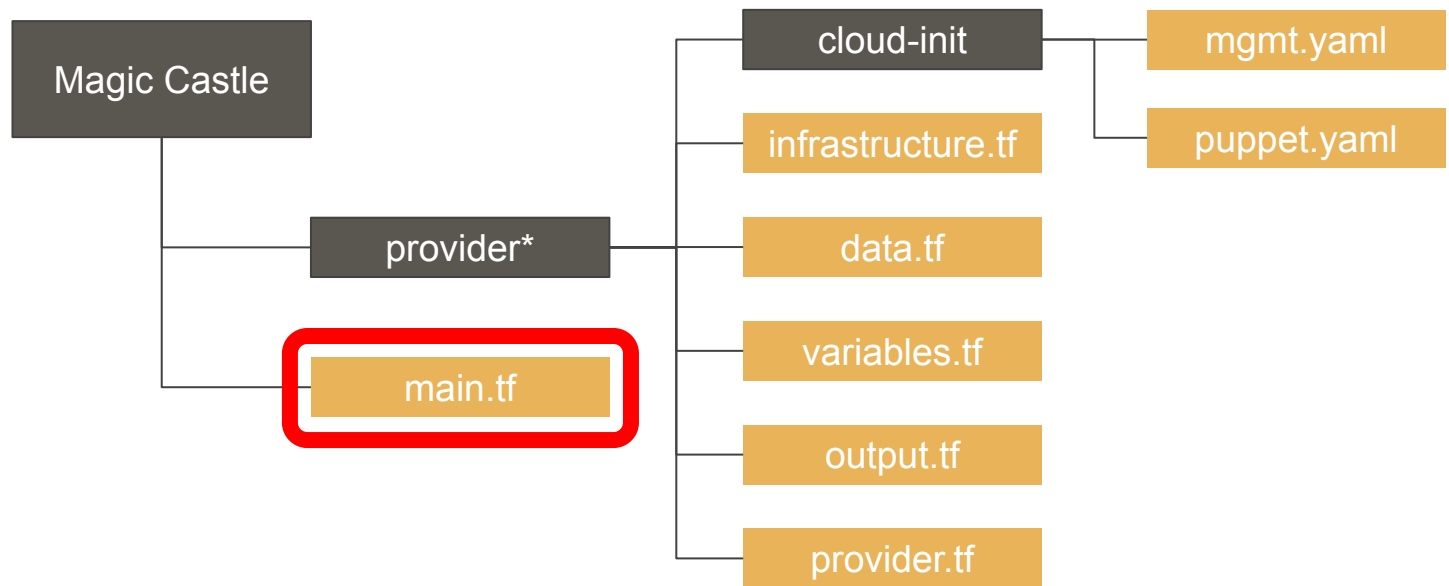


# Main Interface



# Overview of a Magic Castle Release

---



\*could be any in [aws, azure, gcp, openstack, ovh]

# Magic Castle Terraform Main Module

---

4 sections

1. Cloud provider selection
2. Infrastructure customization
3. Cloud Provider specifics inputs
4. DNS Configuration (optional)

# MC Module - 1. source

```
source = "./provider"
```



# MC Module - 2.1 Infrastructure customization

---

```
cluster_name = "eum21"  
domain       = "computecanada.dev"  
image        = "CentOS-7-x64-2020-03"  
nb_users     = 100  
public_keys  = [file("~/ssh/id.pub")]
```

# MC Module - 2.2 Instance definition

---

```
instances = {  
  mgmt = { type = "p4-6gb", count = 1 },  
  login = { type = "p2-3gb", count = 1 },  
  node = [  
    { type = "p2-3gb", count = 1 },  
  ]  
}
```

# MC Module - 2.3 Storage definition

---

```
storage = {  
    type          = "nfs"  
    home_size     = 100  
    project_size  = 50  
    scratch_size  = 50  
}
```

# MC Module - 3. Cloud Provider Specific Inputs

---

Examples:

- OpenStack list of floating ips
- Google GPU attachment for compute nodes
- AWS / Azure / Google Cloud region

# MC Module - 4. DNS Configuration (optional)

---

```
source           = "./dns/cloudflare"  
name             = module.provider.cluster_name  
domain          = module.provider.domain  
email           = "you@example.com"  
public_ip       = module.provider.ip  
rsa_public_key  = module.provider.rsa_public_key  
sudoer_username = module.provider.sudoer_username
```



# Apply Plan

---

```
$ terraform apply
```

```
Apply complete! Resources: 30 added, 0 changed, 0 destroyed.
```

```
Outputs:
```

```
admin_username = centos
```

```
guest_passwd = **redacted**
```

```
guest_usernames = user[01-10]
```

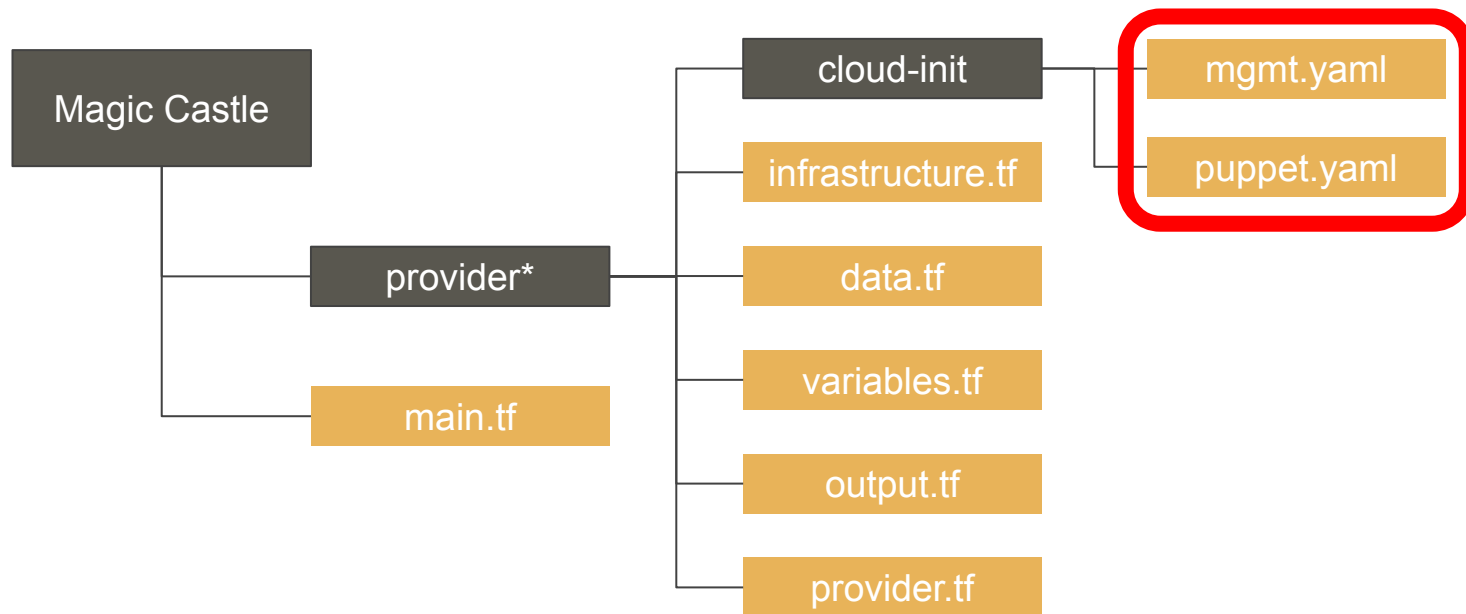
```
hostnames = [eum.computecanada.dev, login1.eum.computecanada.dev]
```

```
public_ip = [206.12.90.97]
```

# Configuration management

# Overview of a Magic Castle Release

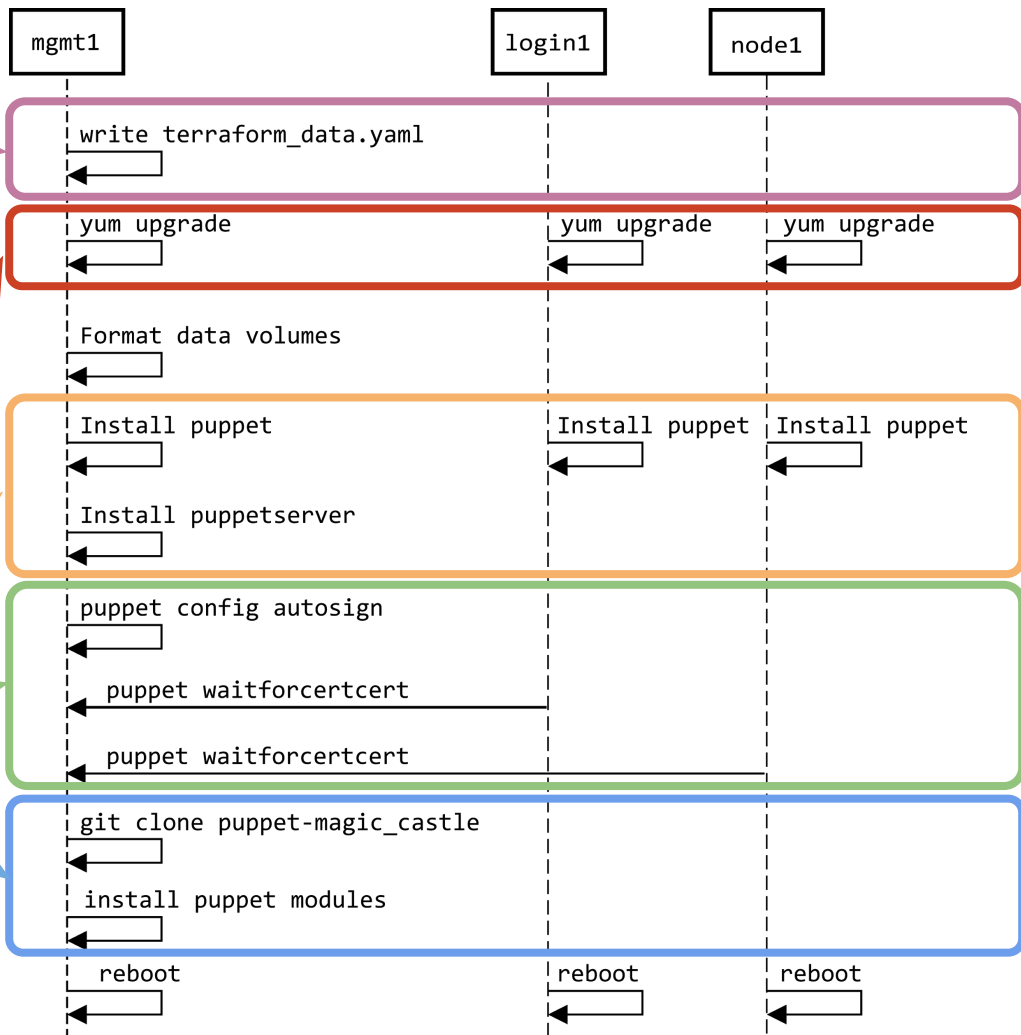
---



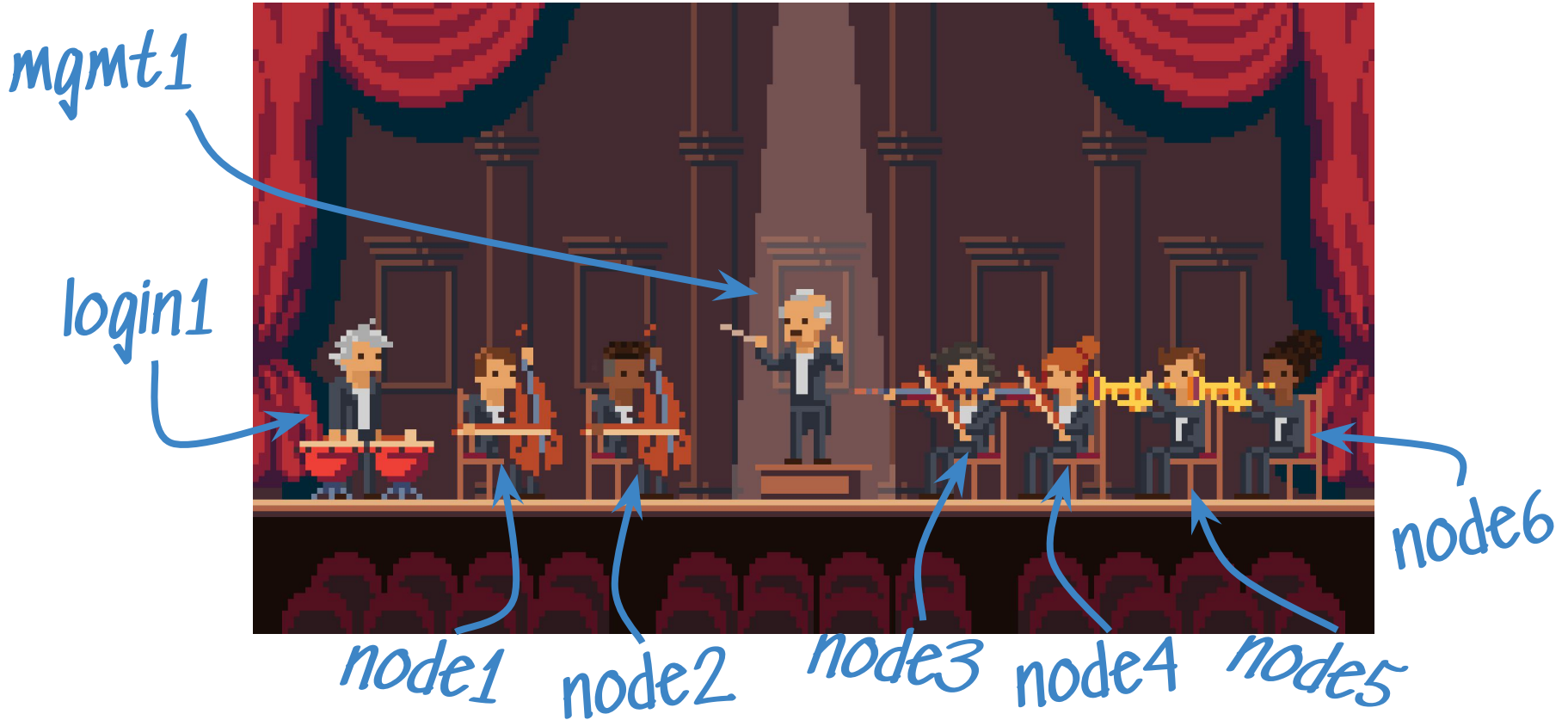
\*could be any in [aws, azure, gcp, openstack, ovh]

# Bootstrap Puppet

1. Inject data from TF
2. Upgrade CentOS
3. Install Puppet rpms
4. Configure Puppet certificates
5. Setup host configuration



# Configuration management with Puppet and Consul



# Why Consul?



- Consul is a service mesh solution [... for] service discovery, configuration [...]
- Services becoming online register in consul's key-value store
- Combined with consul-template it generates Slurm node configuration file (including automatic weight computation)
- Used to gather compute node CPU architectures to select a common set of modules (AVX2 vs AVX512)

# Automatic Slurm node registration and weight computation

---

```
NodeName=node1 CPUs=2 RealMemory=3006 Weight=1
NodeName=node2 CPUs=2 RealMemory=3006 Weight=1
NodeName=med-node1 CPUs=4 RealMemory=5965 Weight=2
NodeName=med-node2 CPUs=4 RealMemory=5965 Weight=2
NodeName=fat-node1 CPUs=4 RealMemory=15037 Weight=3
NodeName=fat-node2 CPUs=4 RealMemory=15037 Weight=3
NodeName=gpu-node1 CPUs=4 RealMemory=22093 Gres=gpu:1 Weight=4
```

[https://github.com/ComputeCanada/magic\\_castle-plugins](https://github.com/ComputeCanada/magic_castle-plugins)

**Software**



# Operating System

---



## CentOS 7

The Community ENTERprise Operating System



## CentOS 8<sup>\*</sup>

The Community ENTERprise Operating System

\* future depends on Compute Canada

# Batteries Included

---

- FreeIPA
  - Kerberos
  - BIND
  - 389 DS LDAP
- NFS
- Slurm
- Globus Endpoint
- JupyterHub
- LMOD
- noVNC Desktop
- singularity



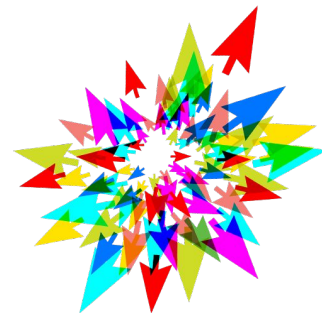
# Software Stack - CVMFS



easybuild

- CernVM File System (CVMFS) provides a scalable, reliable and low-maintenance software distribution service;
- Compute Canada CVMFS repo:
  - [600+ scientific applications](#)
  - 4,000+ permutations of version/arch/toolchain
  - All compiled with [EasyBuild](#)
- EESSI available since release 9.2

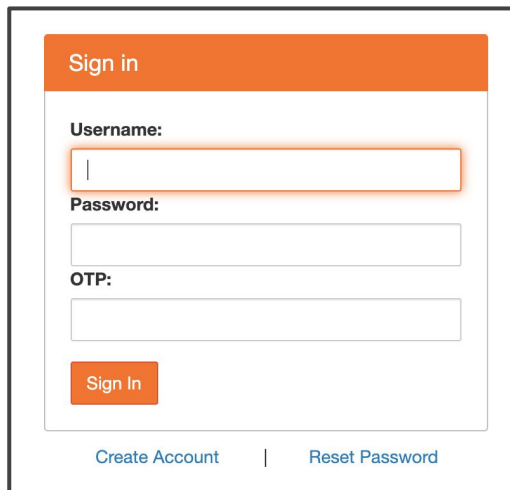
compute | calcul  
canada | canada



**E E S S I**  
EUROPEAN ENVIRONMENT FOR  
SCIENTIFIC SOFTWARE INSTALLATIONS

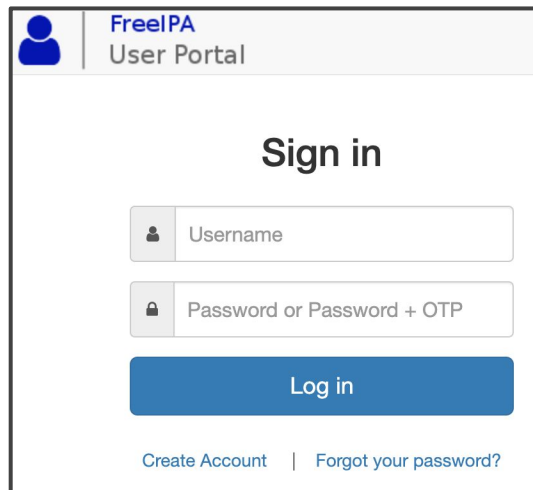
# User self-registration

---



The screenshot shows a sign-in form for JupyterHub. It features an orange header with the text "Sign in". Below the header, there are three input fields: "Username:" (with a cursor in the field), "Password:", and "OTP:". An orange "Sign in" button is positioned below the input fields. At the bottom of the form, there are two links: "Create Account" and "Reset Password".

JupyterHub



The screenshot shows a sign-in form for Mokey. It features a header with a blue user icon, the text "FreeIPA", and "User Portal". Below the header, there is a large "Sign in" heading. The form contains two input fields: "Username" (with a user icon) and "Password or Password + OTP" (with a lock icon). A blue "Log in" button is positioned below the input fields. At the bottom of the form, there are two links: "Create Account" and "Forgot your password?".

Mokey

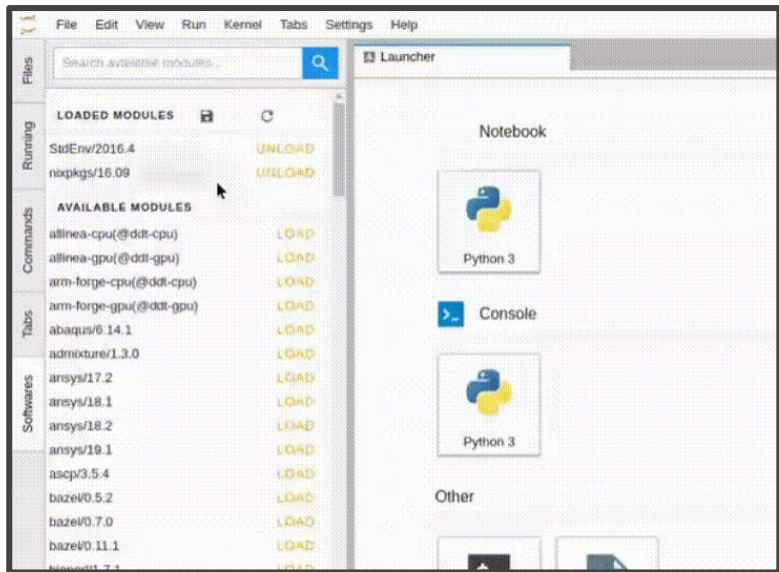
# Web interface for user management

— — —

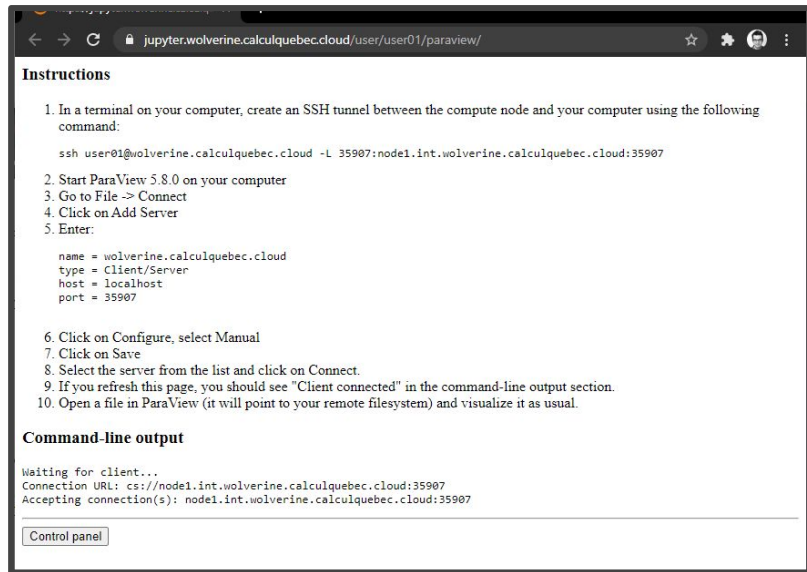
The screenshot shows a web browser window with the URL `calculquebec.cloud`. The page title is "Identity Management" and the user is logged in as "Administrator". The navigation menu includes "Identité", "Politique", "Authentification", "Services réseau", and "Serveur IPA". The main content area is titled "Utilisateurs actifs" and displays a table of active users. The table has columns for "Identifiant de connexion", "Prénom", "Nom", "État", "UID", "Adresse courriel", "Numéro de téléphone", and "Titre de poste". The "État" column shows "Activé(e)" for all users. The "Adresse courriel" column shows email addresses like `admin`, `mokeyapp@int.mrpink.calculquebec.cloud`, and `user01@int.mrpink.calculquebec.cloud`.

	Identifiant de connexion	Prénom	Nom	État	UID	Adresse courriel	Numéro de téléphone	Titre de poste
<input type="checkbox"/>	admin		Administrateur	✓ Activé(e)	60000			
<input type="checkbox"/>	mokeyapp	Mokey	App	✓ Activé(e)	60001	mokeyapp@int.mrpink.calculquebec.cloud		
<input type="checkbox"/>	user01	user01	user01	✓ Activé(e)	60003	user01@int.mrpink.calculquebec.cloud		
<input type="checkbox"/>	user02	user02	user02	✓ Activé(e)	60004	user02@int.mrpink.calculquebec.cloud		
<input type="checkbox"/>	user03	user03	user03	✓ Activé(e)	60005	user03@int.mrpink.calculquebec.cloud		
<input type="checkbox"/>	user04	user04	user04	✓ Activé(e)	60006	user04@int.mrpink.calculquebec.cloud		
<input type="checkbox"/>	user05	user05	user05	✓ Activé(e)	60007	user05@int.mrpink.calculquebec.cloud		
<input type="checkbox"/>	user06	user06	user06	✓ Activé(e)	60008	user06@int.mrpink.calculquebec.cloud		
<input type="checkbox"/>	user07	user07	user07	✓ Activé(e)	60009	user07@int.mrpink.calculquebec.cloud		
<input type="checkbox"/>	user08	user08	user08	✓ Activé(e)	60010	user08@int.mrpink.calculquebec.cloud		
<input type="checkbox"/>	user09	user09	user09	✓	60011	user09@int.mrpink.calculquebec.cloud		

# Dev. platform for JupyterHub use cases in HPC



[jupyter-lmod](#)



[pvserver-webproxy](#)

# Dev. platform for JupyterHub in HPC

---

## Spawner Options

**Reservation**

None

**Account**

def-sponsor00

**Time (hours)**

1000.00

**Number of cores**

2

**Memory (MB)**

3006

Enable core oversubscription? Recommended for interactive usage

**GPU configuration**

None

**User interface**

Jupyter Notebook

Spawn

[slurmformspawner](#)



[puppet-jupyterhub](#)

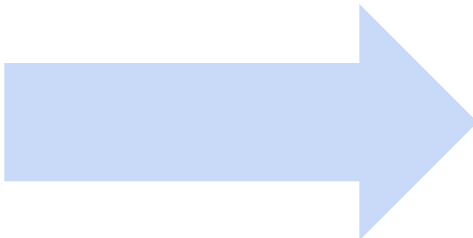
# MC-Hub

**When Terraform is too difficult**



# Improving the workflow with MC Hub

```
1 terraform {
2   required_version = ">= 0.12.21"
3 }
4
5 module "openstack" {
6   source = "git::https://github.com/ComputeCanada/magic_castle.git//openstack"
7
8   cluster_name = "phoenix"
9   domain       = "calculquebec.cloud"
10  image        = "CentOS-7-x64-2019-07"
11  nb_users     = 10
12
13  instances = {
14    mgmt = { type = "p4-6gb", count = 1 },
15    login = { type = "p2-3gb", count = 1 },
16    node = [
17      { type = "p2-3gb", count = 1 },
18    ]
19  }
20
21  storage = {
22    type       = "nfs"
23    home_size  = 100
24    project_size = 50
25    scratch_size = 50
26  }
27
28  public_keys = [file("~/..ssh/id_rsa.pub")]
29
30  # Shared password, randomly chosen if blank
31  guest_passwd = ""
32
33  # OpenStack specific
34  os_floating_ips = []
35 }
36
37 output "sudoer_username" {
38   value = module.openstack.sudoer_username
39 }
40
```



### Magic Castle Modification

Hostname  
magic.calculquebec.cloud Build success

General configuration

Image  
CentOS-7-x64-2019-07

Number of users  
10

Node Instances

Management	Type p4-6gb	Count 1
Login	Type p2-3gb	Count 1
Compute	Type c16-90gb-392	Count 2

Used Instances: 3% (4 / 122)

Used RAM: 78% (189 GB / 243 GB)

Used cores: 16% (38 / 242)

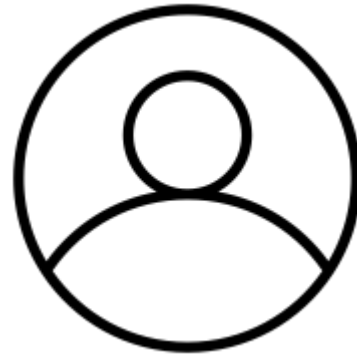
Storage

# What the user needs

Browser



An  
account



# Creating a Magic Castle configuration

- Format validation
- Quota validation

### Magic Castle Creation

General configuration

Cluster name  
phoenix

---

Domain  
calculquebec.cloud

---

Image  
CentOS-7-x64-2019-07

---

Number of users  
10

---

Node instances

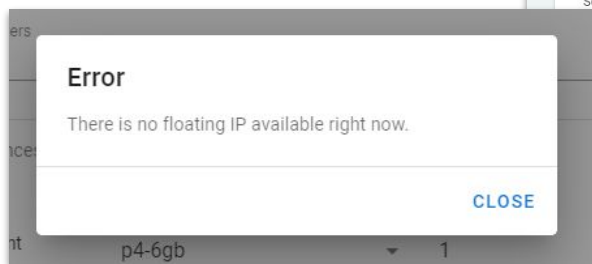
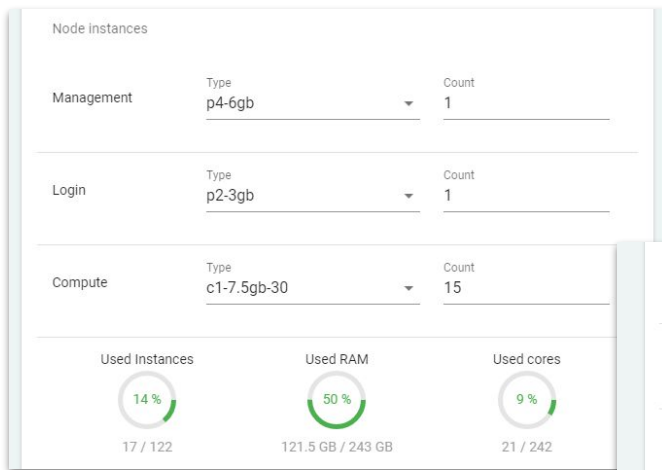
	Type	Count
Management	p4-6gb	1
Login	c2-7.5gb-31	1
Compute	p8-12gb	1
	<b>c2-15gb-31</b>	1
	c4-15gb-83	
	c4-30gb-83	

Used Instances: 2%      Used cores: 9%      Used cores: 2%

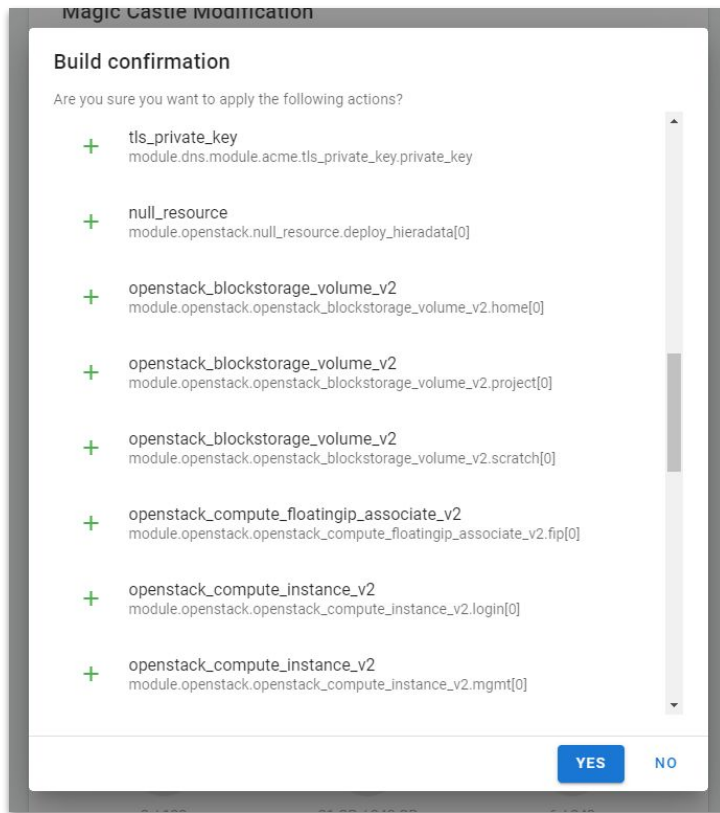
# Verifying the quotas

Easily watch quotas:

- Instance count
- RAM
- Virtual cores
- Volume count
- Volume storage
- Floating IPs



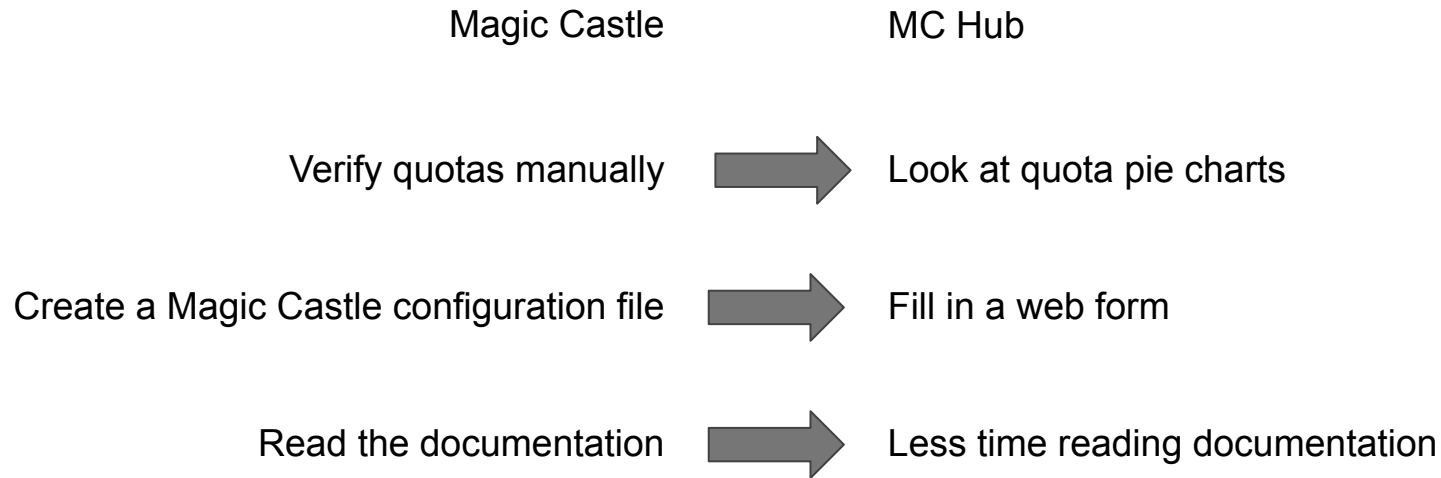
# Confirming the Terraform configuration



# Progress display

+ random_uuid	module.openstack.random_uuid.consul_token	✓ done
+ tls_private_key	module.openstack.tls_private_key.login_rsa	✓ done
+ tls_private_key	module.openstack.tls_private_key.ssh[0]	✓ done
+ cloudflare_record	module.dns.cloudflare_record.records[1]	⌋ running
+ cloudflare_record	module.dns.cloudflare_record.records[2]	⌋ running
+ cloudflare_record	module.dns.cloudflare_record.records[3]	⌋ running
+ cloudflare_record	module.dns.cloudflare_record.records[4]	⌋ running
+ cloudflare_record	module.dns.cloudflare_record.records[6]	⌋ running
+ acme_certificate	module.dns.module.acme.acme_certificate.certificate	⌋ running



# Workflow comparison for OpenStack



# Cluster management

## User's view







Your Magic Castles CREATE CLUSTER

Cluster name	Domain	Status	Owner	Actions
<a href="#">itworks</a>	calculquebec.cloud	Build success	mom	 

Rows per page: 10 1-3 of 3 < >

## Administrator view

Your Magic Castles CREATE CLUSTER

Cluster name ↑	Domain	Status	Owner	Actions
<a href="#">phoenix</a>	calculquebec.cloud	Build success	fredfc	 
<a href="#">itworks</a>	calculquebec.cloud	Build success	mom	 
<a href="#">johndoecluster</a>	calculquebec.cloud	Build running	johnd	 

Rows per page: 10 1-3 of 3 < >



# MC Hub Projects

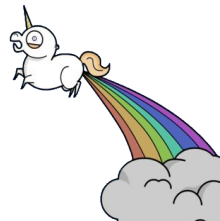
---

1. Web front-end (Vue.js) and backend (Flask+Terraform) that can create HPC cluster in OpenStack with Magic Castle.  
<https://github.com/ComputeCanada/mc-hub>
2. A Docker Container containing the UI and backend  
[https://hub.docker.com/r/fredericfc/magic\\_castle-ui](https://hub.docker.com/r/fredericfc/magic_castle-ui)
3. Ansible playbook to deploy a SAML authenticated MC Hub  
<https://github.com/ComputeCanada/ansible-mc-hub>

# Key Takeaways

---

1. Magic Castle is a mature project with a rich ecosystem that replicates an HPC cluster in the cloud with Terraform and Puppet
2. Once deployed, MC Hub can be used by anyone to deploy an HPC cluster on OpenStack



# Future directions and coming features

---

- OFED
- Lustre filesystem
- Compute instances automatic scaling
- Support external IdP

[https://github.com/computecanada/magic\\_castle/issues](https://github.com/computecanada/magic_castle/issues)