Magic Castle

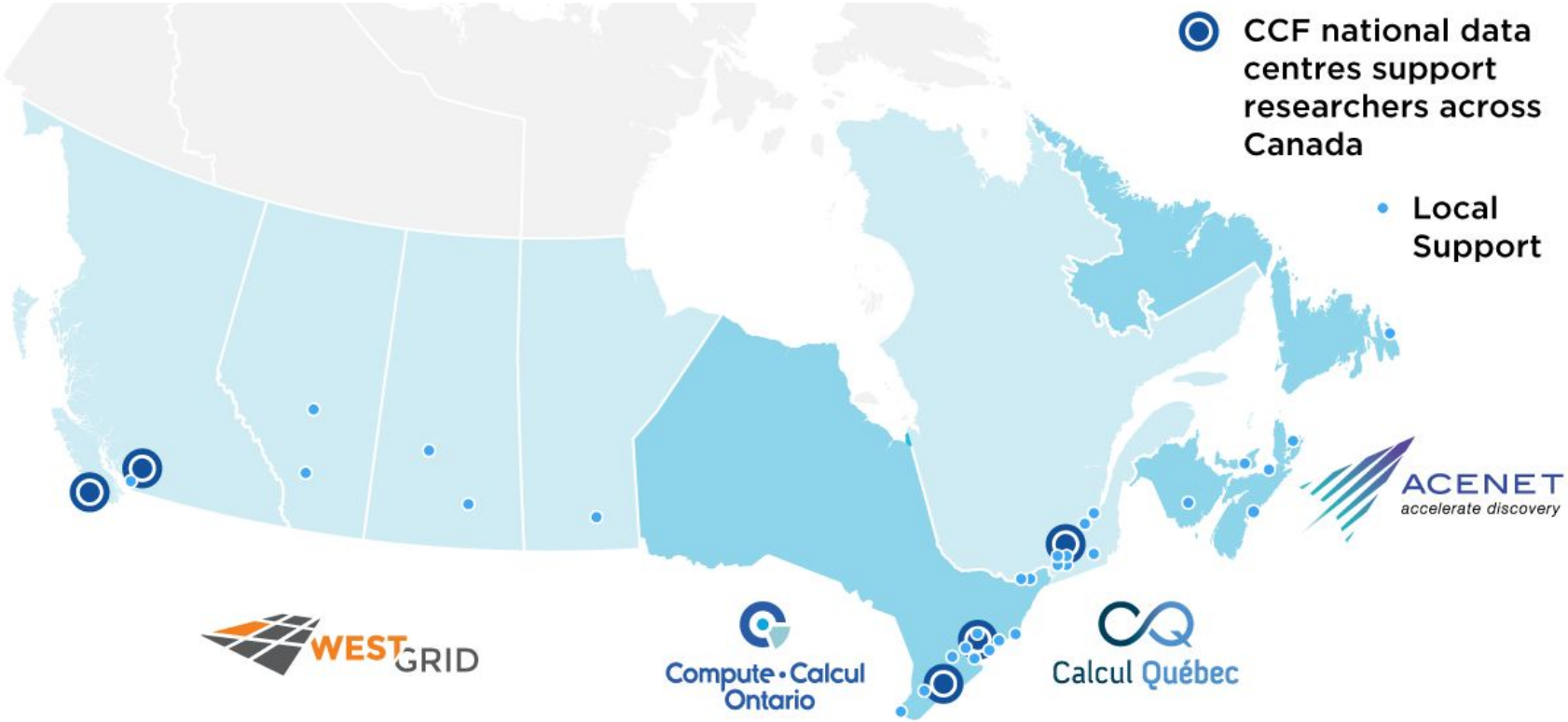
Terraforming the Cloud for HPC

Félix-Antoine Fortin, FOSDEM20

Why are there more wizards in Harry Potter than in Lord of the Rings?

Context

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?

So what is the difference between HP and LotR?

?

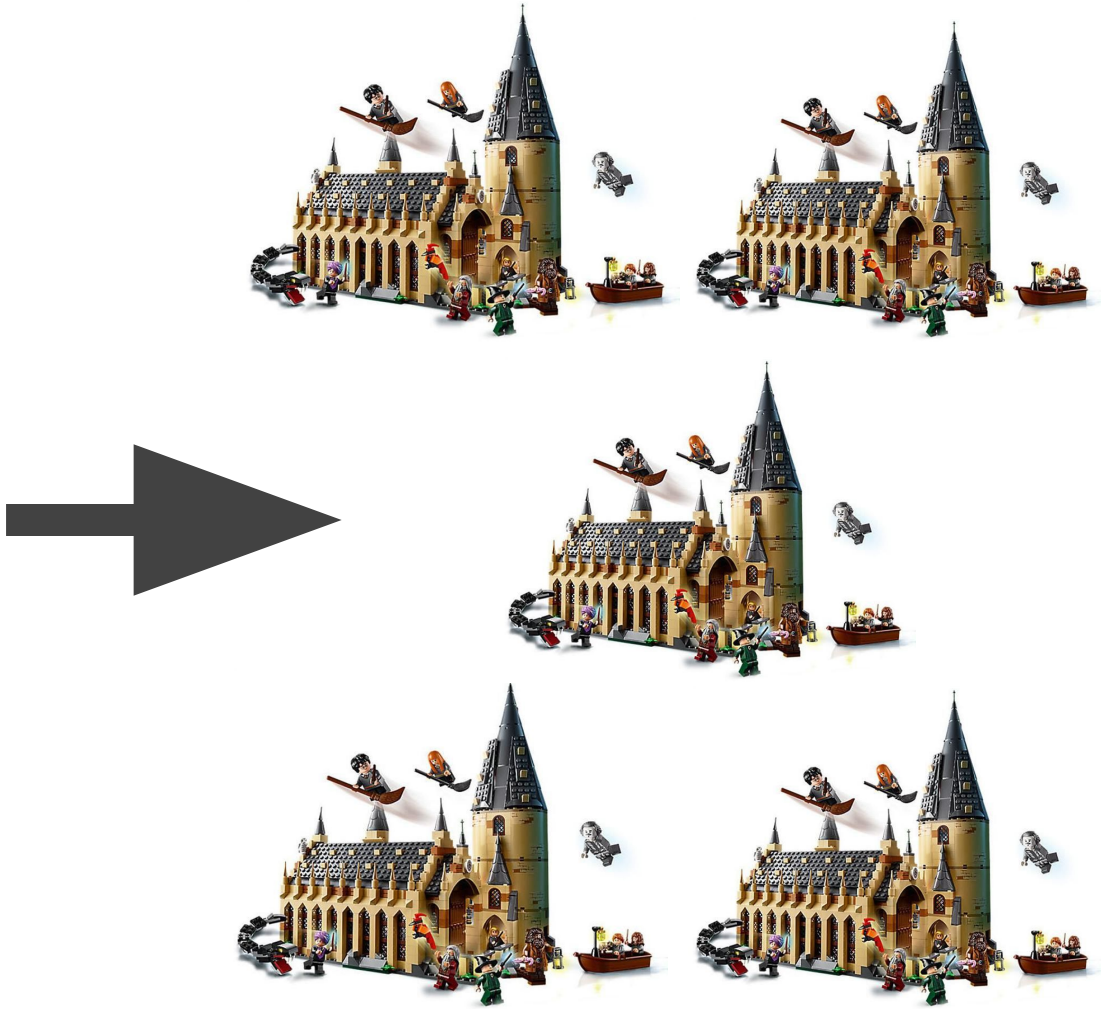
So what is the difference between HP and LotR?

– – –

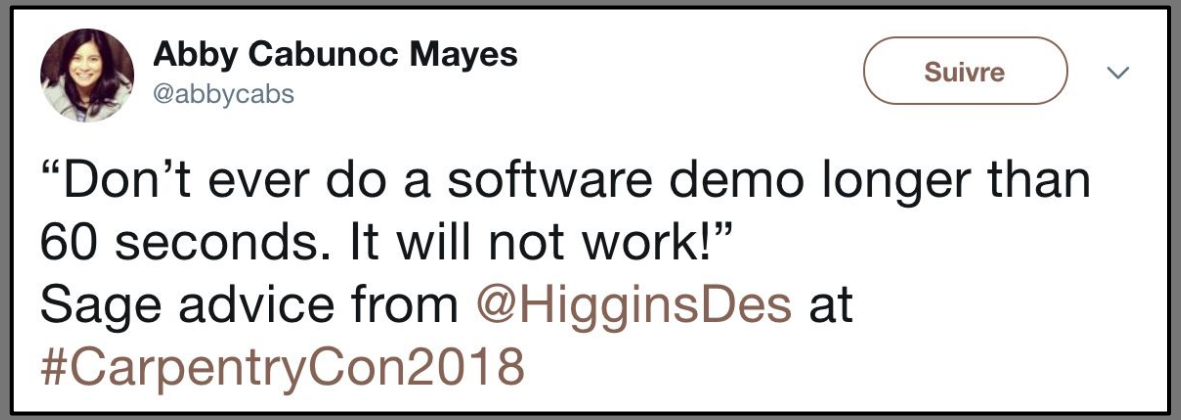
Wizardry Schools

Proposal

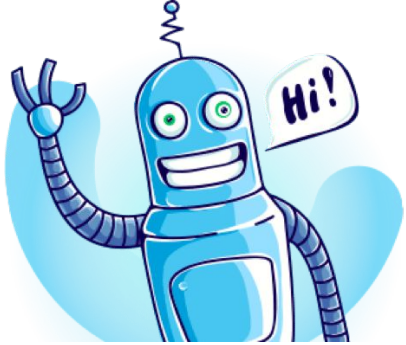
HPC Wizard Tower by
Simon Guilbault



demo


Abby Cabunoc Mayes
@abbycabs
Suivre
“Don’t ever do a software demo longer than 60 seconds. It will not work!”
Sage advice from @HigginsDes at #CarpentryCon2018

CC Wizard: Magic Castle Voice Assistant



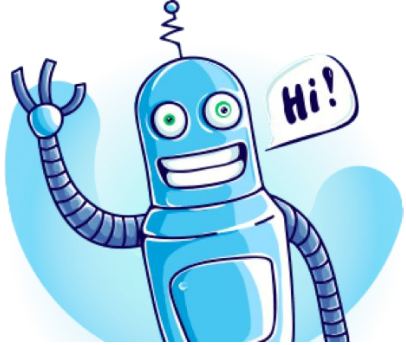
Dialogflow

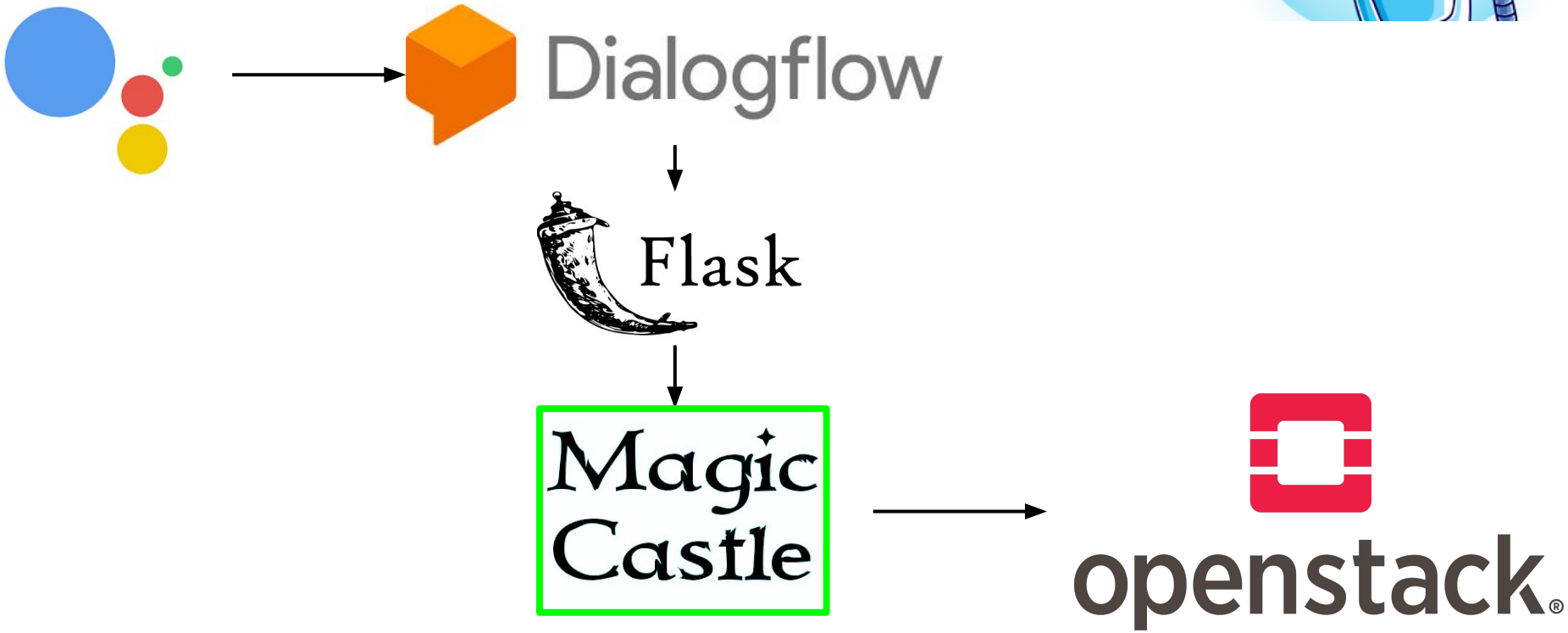
Flask

Magic Castle

openstack®

CC Wizard: Magic Castle Voice Assistant

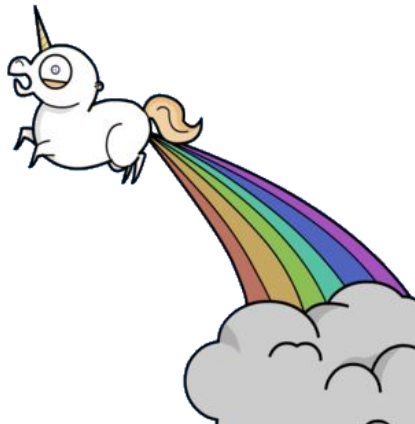




Magic Castle

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

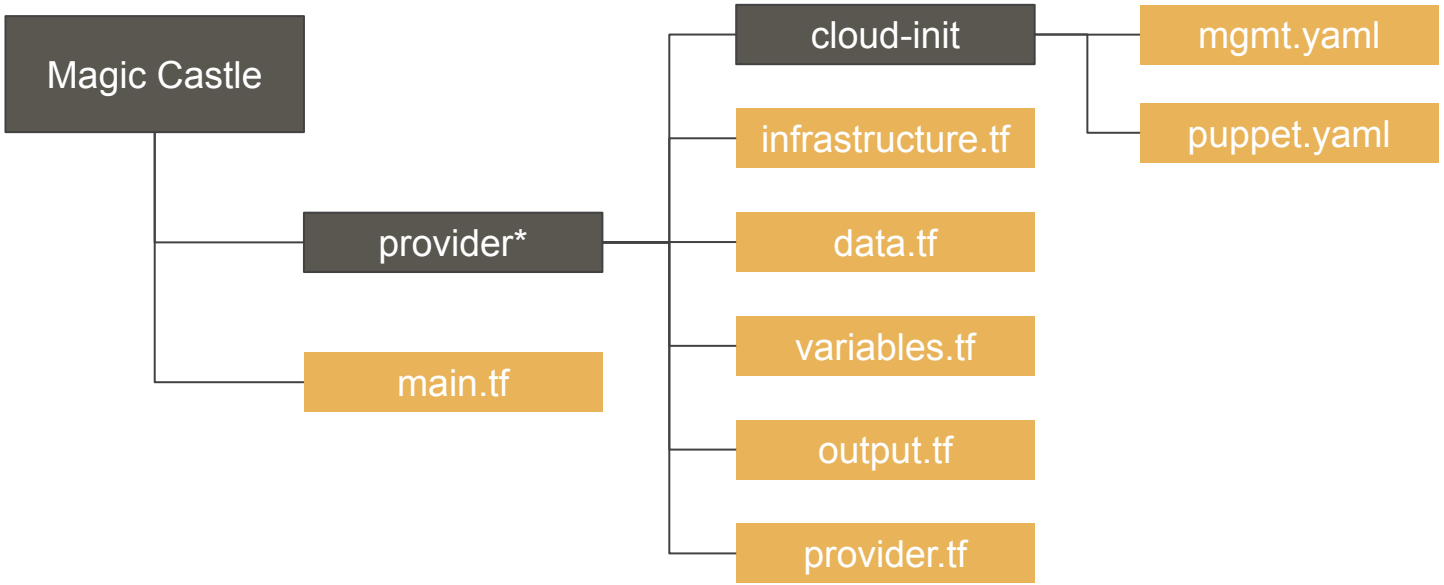
Terraform

- 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

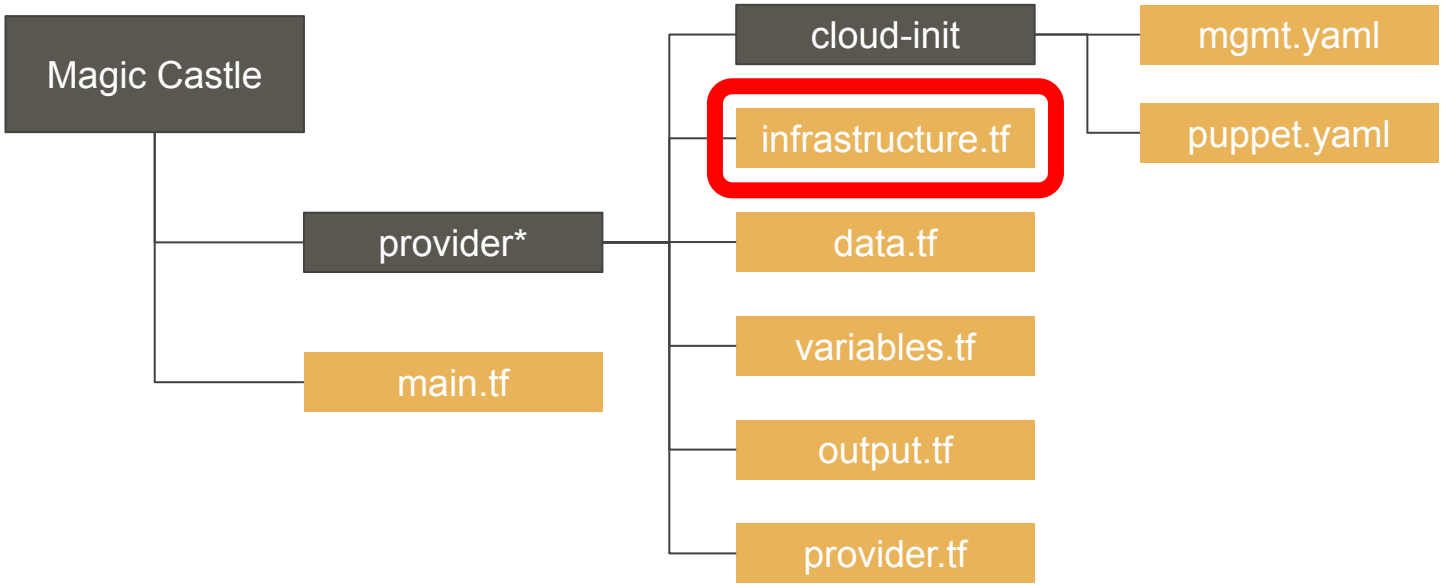
Overview of a Magic Castle Release



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

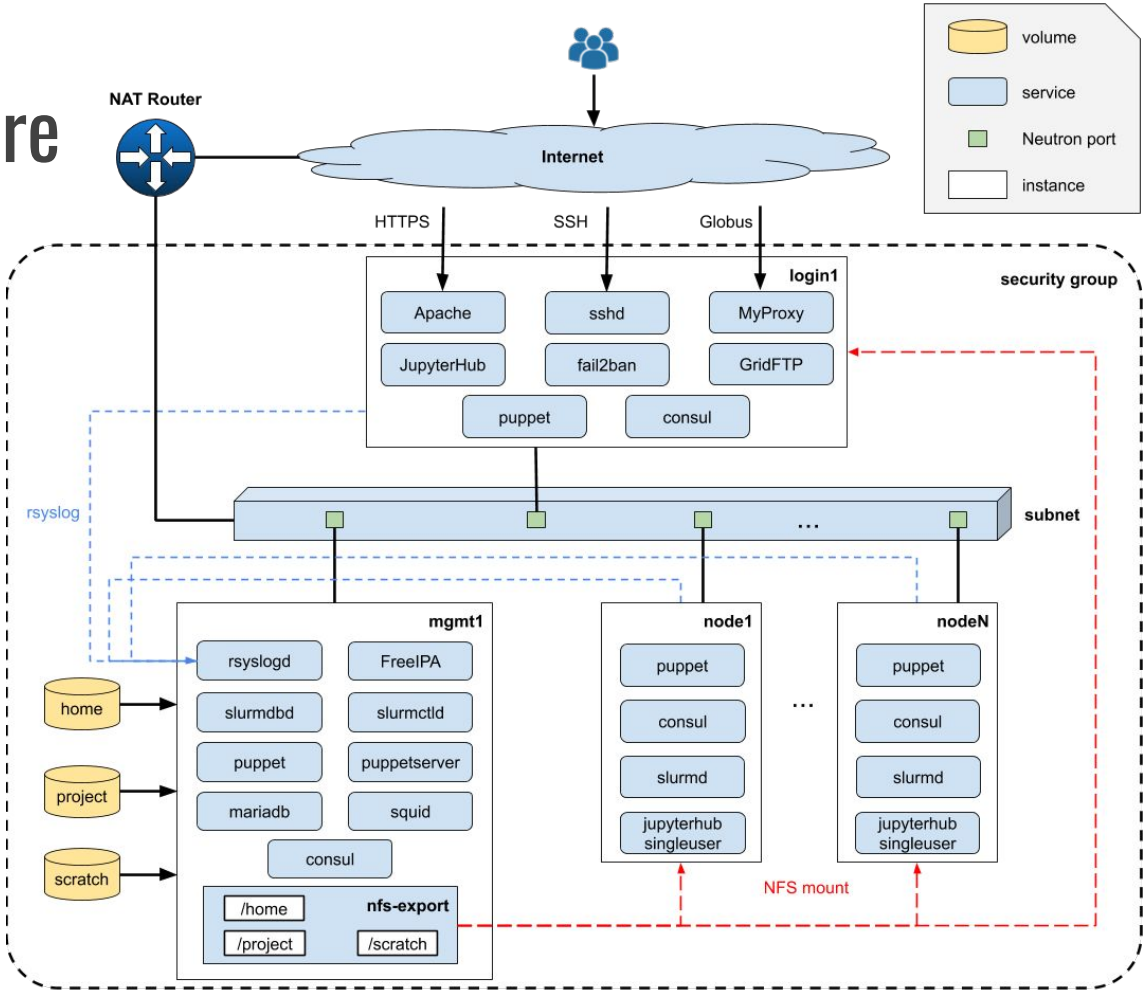
Infrastructure

Overview of a Magic Castle Release

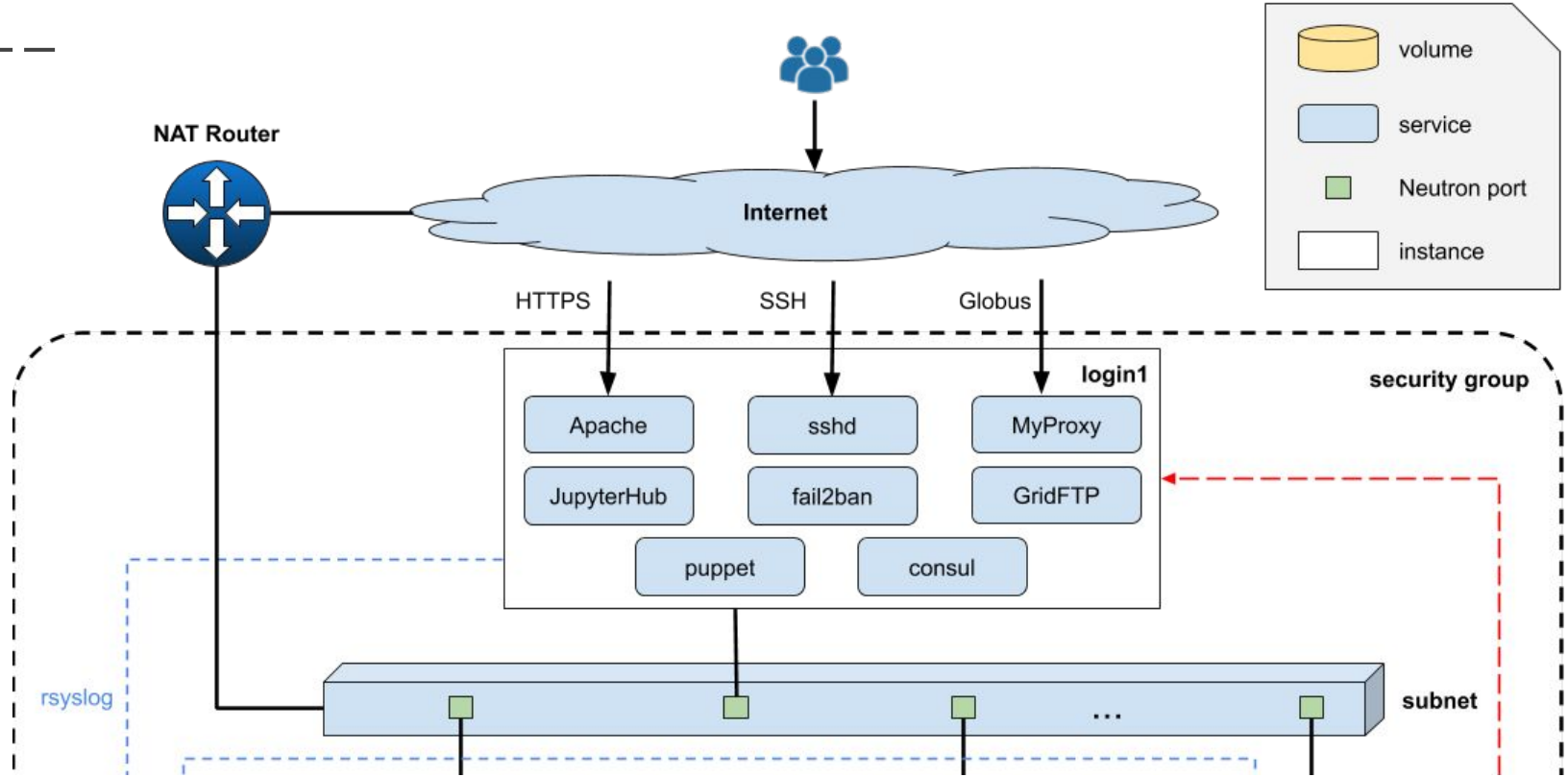


*could be any in [aws, azure, gcp, openstack, ovh]

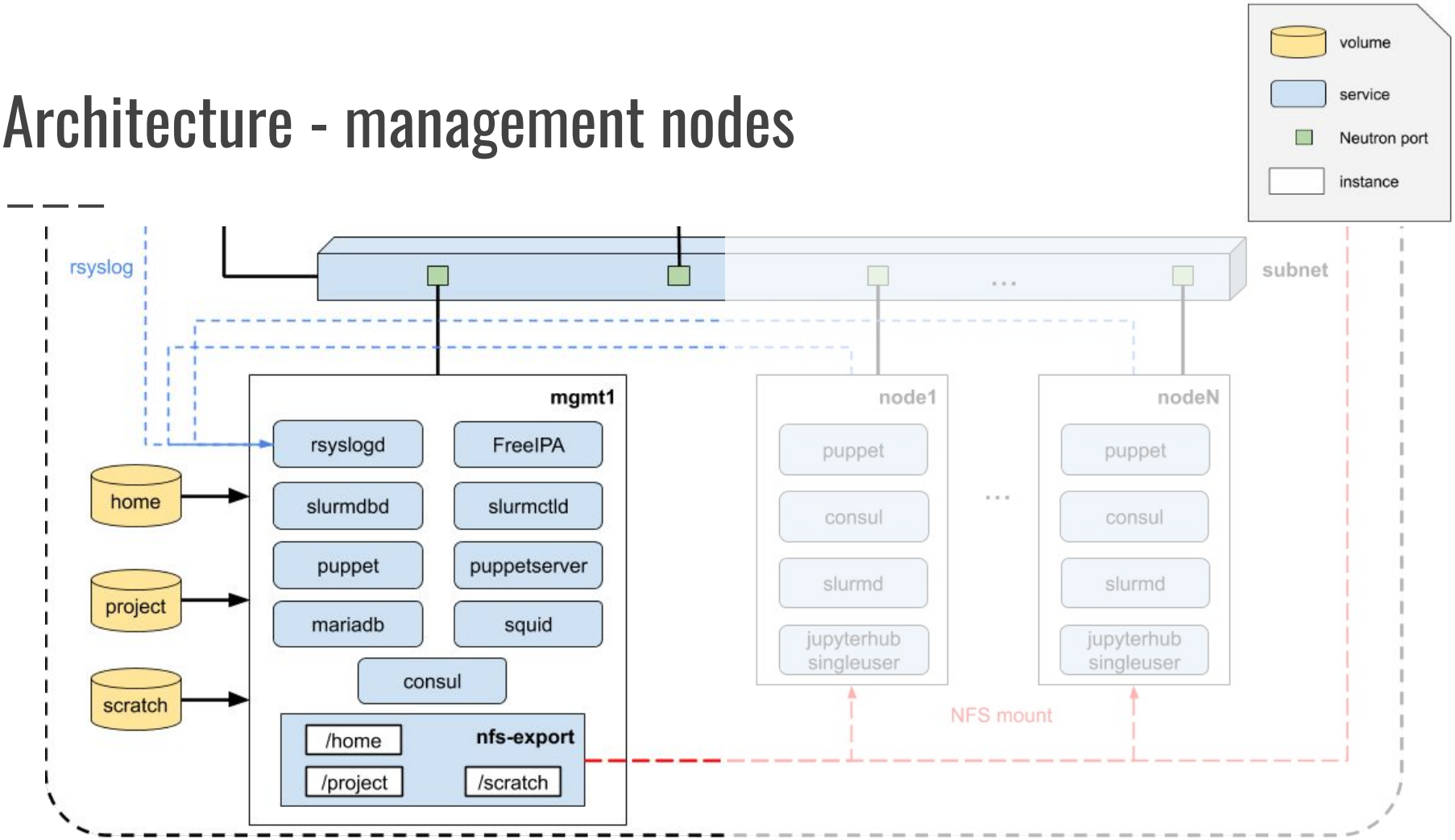
Architecture



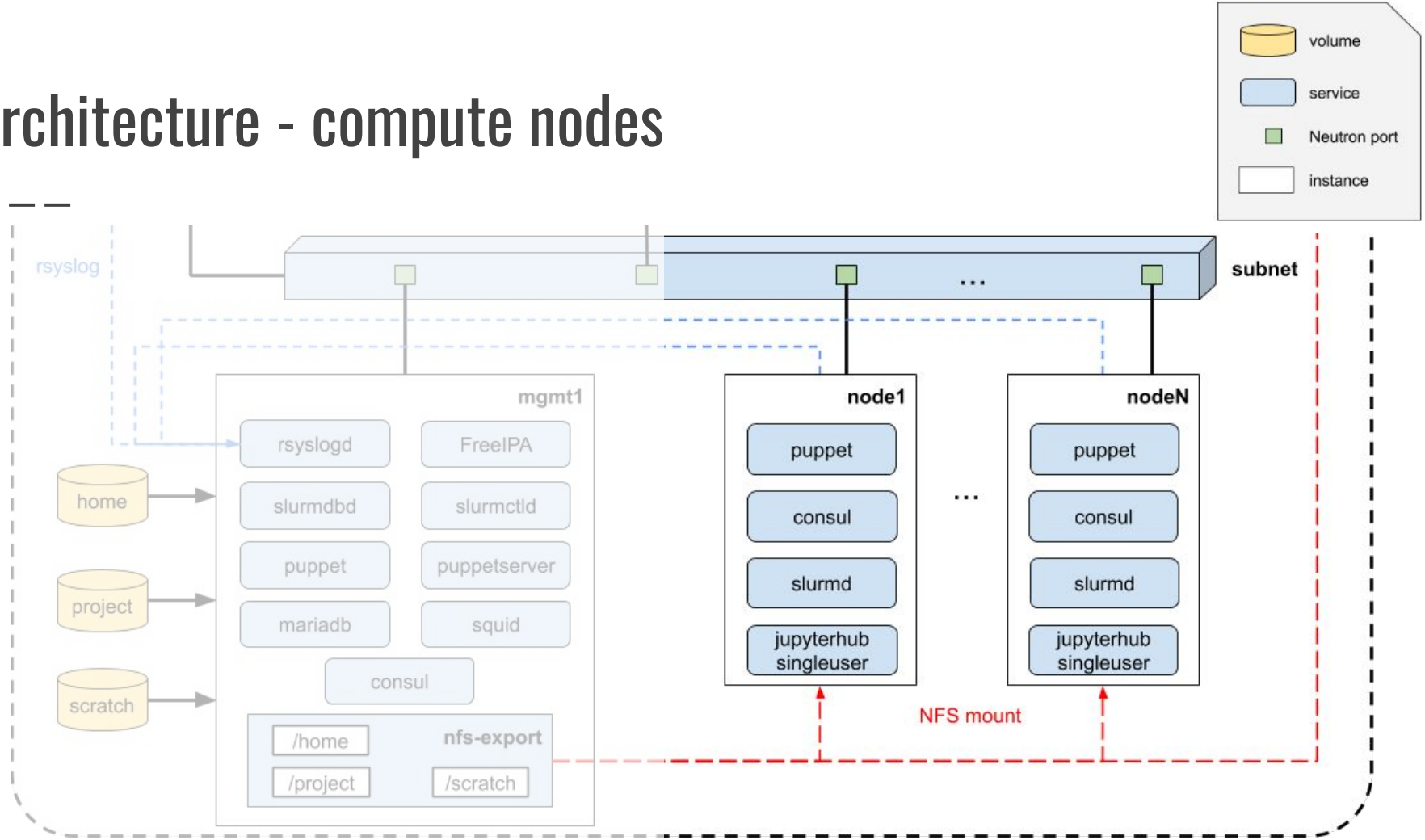
Architecture - login nodes



Architecture - management nodes

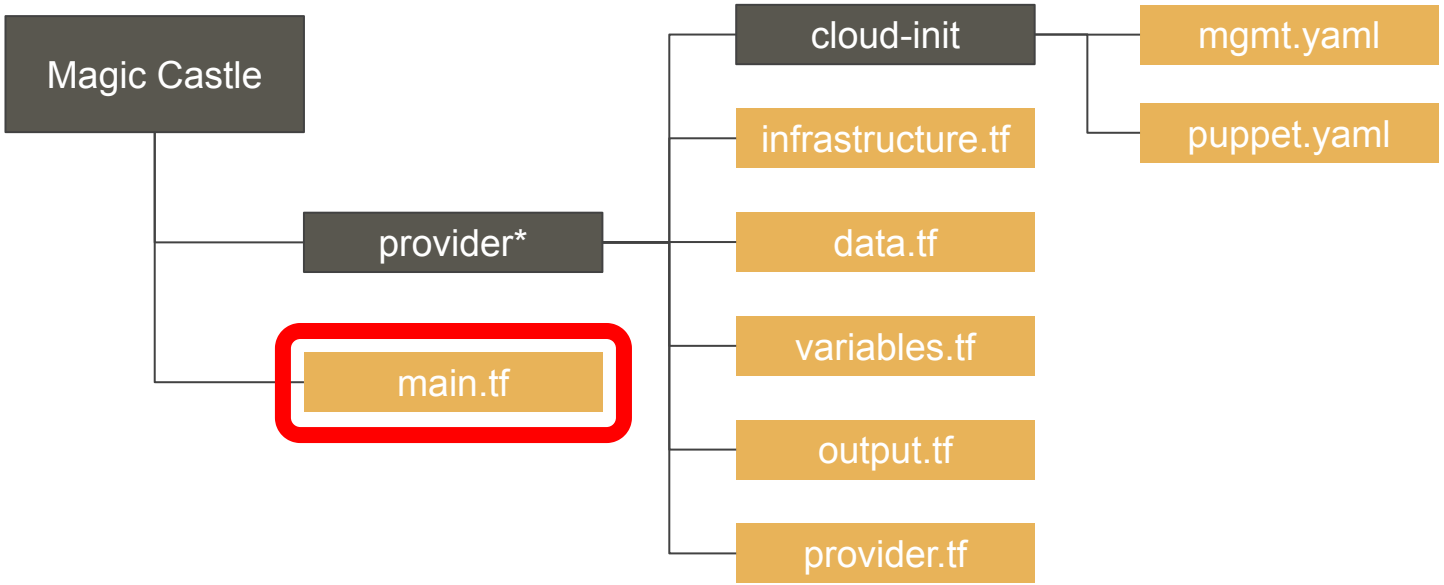


Architecture - compute nodes



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 = "fosdem"
domain       = "computecanada.dev"
image        = "CentOS-7-x64-2019-07"
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

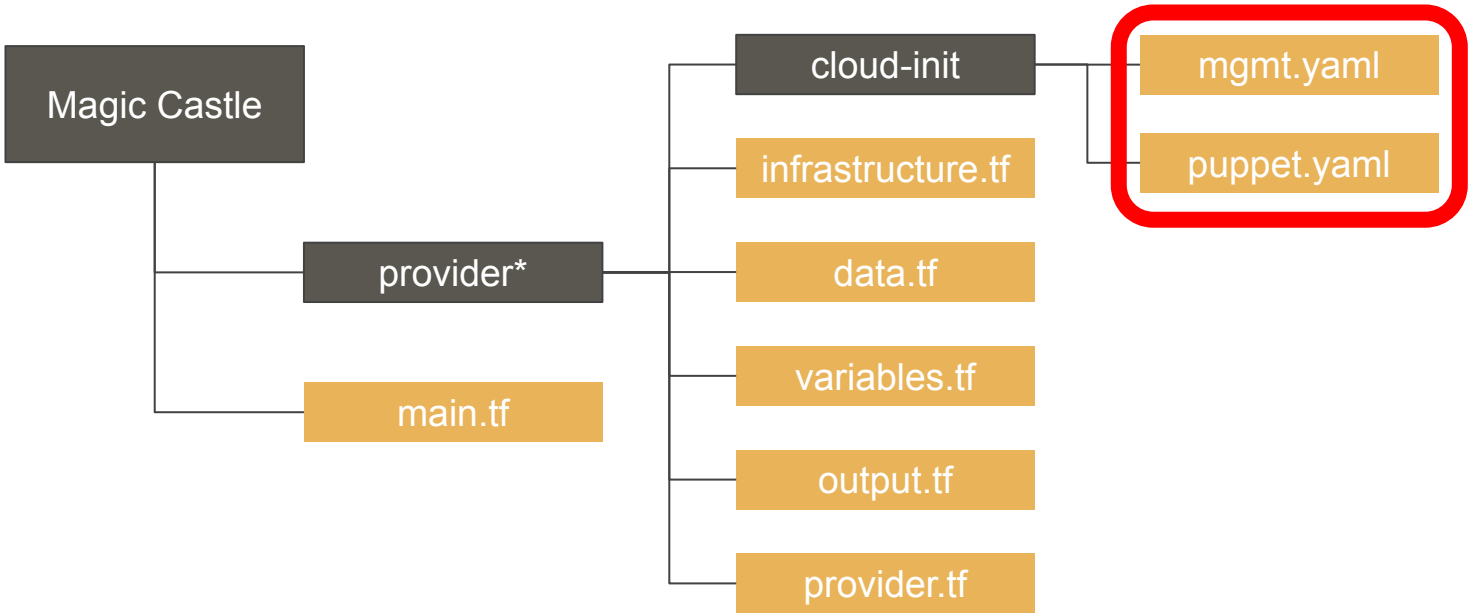
```
$ terraform apply

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

Outputs:

admin_username = centos
guest_passwd = **redacted**
guest_usernames = user[01-10]
hostnames = [pirate.calculquebec.cloud, pirate1.calculquebec.cloud]
public_ip = [206.12.90.97]
```

Challenges: Infrastructure as Code

- Designing the main user interface that would limit the references to a provider specific implementation / API.
- Terraform configuration language tends to favor repetition over re-use of code.
- Regrouping every components that are common amongst providers

Provisioning

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

Provisioning with Puppet and Consul

Challenges: Provisioning

- Every steps of the provisioning need to work without human intervention.
- Once provisioned, the cluster need to stay healthy on itself - users are not necessarily sys admins.
- Provisioning both master and slave services without proper syncing mechanism.

Software

Batteries Included

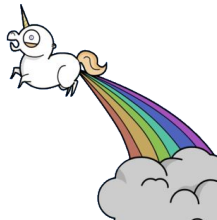
- FreeIPA
  - Kerberos
  - BIND
  - 389 DS LDAP
- NFS
- Slurm
- Globus Endpoint
- JupyterHub with BatchSpawner
- Compute Canada CVMFS
- LMOD

Compute Canada Software Stack - CVMFS

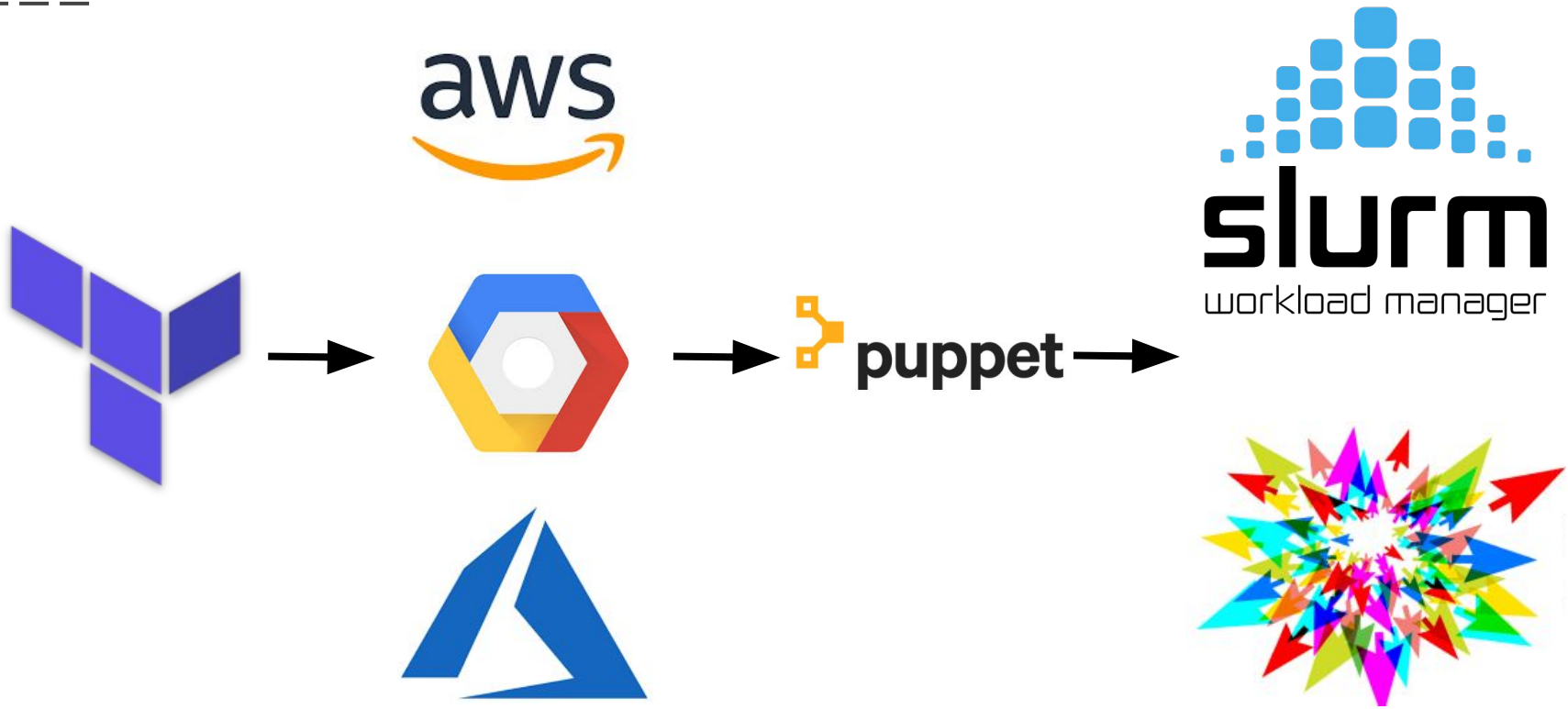
- CernVM File System (CVMFS) provides a scalable, reliable and low-maintenance software distribution service;
- Compute Canada CVMFS repo:
  - 600+ scientific applications
  - 4,000+ permutations of version/arch/toolchain
  - All compiled with EasyBuild
- Available from anywhere
- PEARC19 paper

Key Takeaways

1. Terraform can be used to build complex things and modules simplify that complexity.
2. Magic Castle is a teaching and development meta-platform for HPC.



Magic Castle Replicates a Compute Canada Cluster in **20 min.**



Questions ?