TERRAFORM

POUR PROXMOX ET NETBOX

PIERRE GAMBAROTTO

INSTITUT DE MATHÉMATIQUES
de TOULOUSE

OÙ L'ON VA PARLER DE ...

- Terraform beaucoup
  - notions de base
  - providers
  - comment bâtir une interface au dessus

Pour cela, des exemples en direct d'un labo :

- Proxmox/ceph
- Ipam netbox

Pour finir : retours sur le tout et suites prévues

TERRAFORM

- permet de décrire en json/hcl l'état désiré des resources d'un système
- utiliser des API pour trouver l'état courant du système
- utiliser des API pour changer l'état courant vers l'état désiré

=> machine à états, assure la transition entre état par appel à des API

EXEMPLE

```
# resource "provider" "id"
resource "local_file" "demo" {
# attr     = value
  filename = "servers.json"
  content  = "blah blih bloh"
}
```

Le *provider* `local_file` doit s'assurer que le fichier identifé par «demo» corresponde aux attributs donnés.

```
❯ terraform apply # plan for the plan only

Terraform used the selected providers to generate the following ex
actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # local_file.demo will be created
  + resource "local_file" "demo" {
      + content              = "blah blih bloh"
      + content_base64sha256 = (known after apply)
      + content_base64sha512 = (known after apply)
      + content_md5          = (known after apply)
      + content_sha1         = (known after apply)
      + content_sha256       = (known after apply)
      + content_sha512       = (known after apply)
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "servers.json"
      + id                   = (known after apply)
    }

Plan: 1 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: yes

local_file.demo: Creating...
local_file.demo: Creation complete after 0s
[id=1512f9ea9e6dbeb8fb27b79aba25db2c8480d0f4]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

Remarques :

- l'id est ici le sha1 du fichier.
- le provider rajoute des attributs à la ressource

```
--- a/file.tf     2024-03-14 04:00:21.173652007 +0100
+++ b/file.tf     2024-03-14 04:00:48.004884882 +0100
@@ -1,5 +1,5 @@
 resource "local_file" "demo" {
   filename = "servers.json"
-  content = "blah blih bloh"
+  content = "changed !"
 }
```

```
terraform apply
local_file.demo: Refreshing state... [id=1512f9ea9e6dbeb8fb27b79ab

Terraform will perform the following actions:

  # local_file.demo must be replaced
-/+ resource "local_file" "demo" {
      ~ content              = "blah blih bloh" -> "changed !" # f
      ~ content_sha1         = "1512f9ea9e6dbeb8fb27b79aba25db2c84
        -> (known after apply)
      ~ id                   = "1512f9ea9e6dbeb8fb27b79aba25db2c84
        -> (known after apply)
    }

Plan: 1 to add, 0 to change, 1 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: yes

local_file.demo: Destroying... [id=1512f9ea9e6dbeb8fb27b79aba25db2
local_file.demo: Destruction complete after 0s
local_file.demo: Creating...
local_file.demo: Creation complete after 0s [id=2470a477193a952c4b

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```

Certains providers permettent de modifier sans détruire

Ex : le provider proxmox permet de changer la RAM allouée à une VM sans la détruire

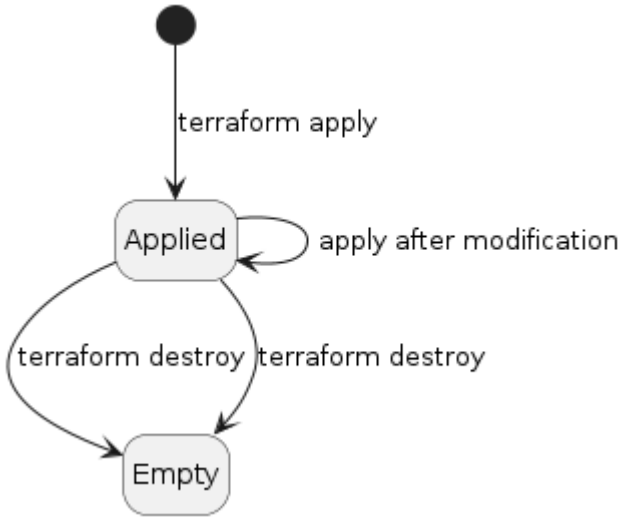
```
❯ terraform destroy
local_file.demo: Refreshing state...
[id=1512f9ea9e6dbeb8fb27b79aba25db2c8480d0f4]

Terraform used the selected providers to generate the following ex
actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

 # local_file.demo will be destroyed
 - resource "local_file" "demo" {
     - content              = "blah blih bloh" -> null
     # some hash content deleted
     - content_sha1         = "1512f9ea9e6dbeb8fb27b79aba25db2c848
         -> null
     - directory_permission = "0777" -> null
     - file_permission      = "0777" -> null
     - filename             = "servers.json" -> null
     - id                   = "1512f9ea9e6dbeb8fb27b79aba25db2c848
         -> null
   }

Plan: 0 to add, 0 to change, 1 to destroy.
```

```
Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

local_file.demo: Destroying...
  [id=1512f9ea9e6dbeb8fb27b79aba25db2c8480d0f4]
local_file.demo: Destruction complete after 0s

Destroy complete! Resources: 1 destroyed.
```

AUTOMATE À ÉTATS

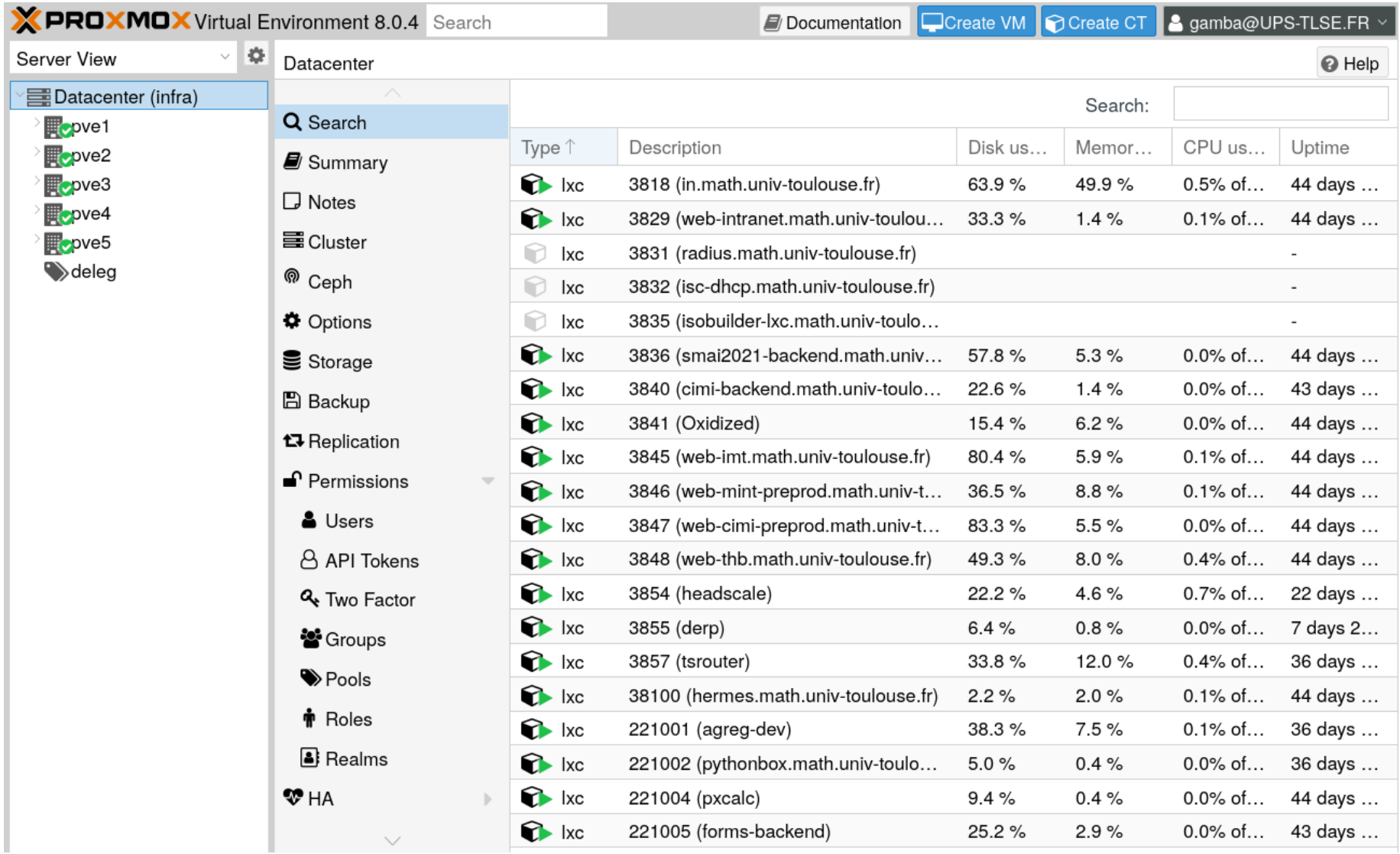
Par défaut : état stocké en local,

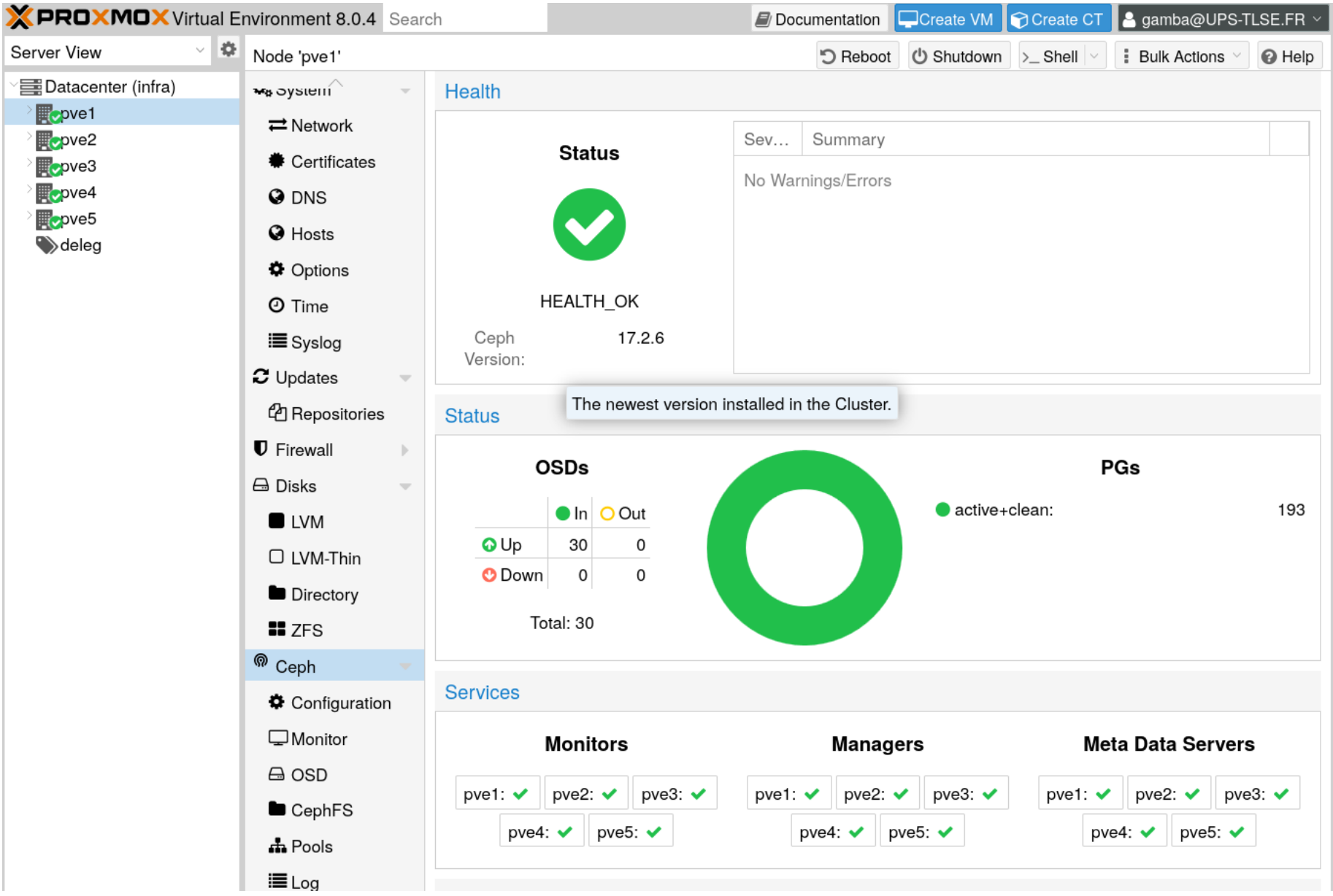
collaborer : partager l'état ! => stockage gitlab

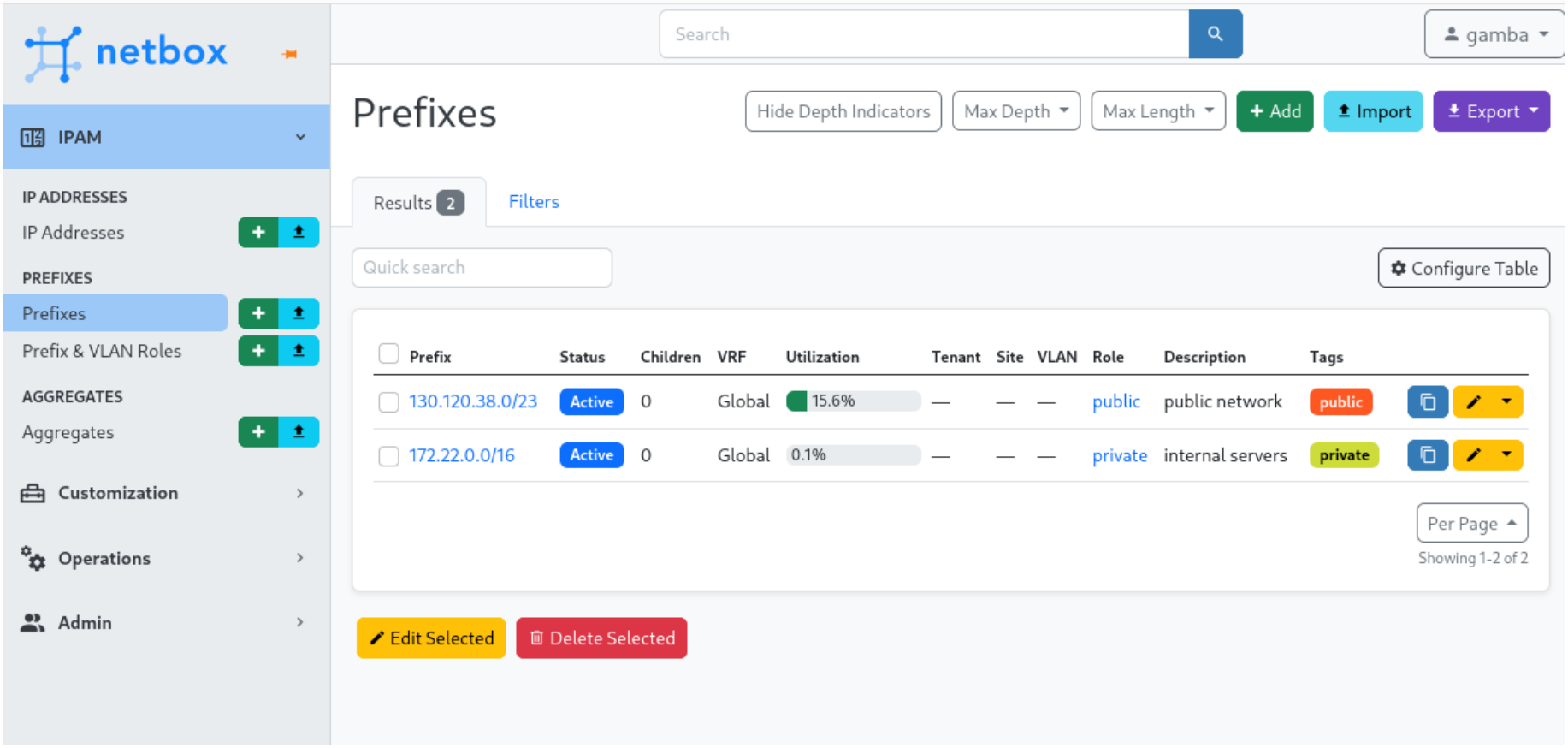


IMT

- cluster proxmox/ceph de 5 nœuds/60To
- 1 réseau IP public, 1 privé, géré avec netbox


PROXMOX Virtual Environment 8.0.4
Search
Documentation
Create VM
Create CT
gamba@UPS-TLSE.FR
Server View
Datacenter
Help
Datacenter (infra)
pve1
pve2
pve3
pve4
pve5
deleg
Search
Summary
Notes
Cluster
Ceph
Options
Storage
Backup
Replication
Permissions
Users
API Tokens
Two Factor
Groups
Pools
Roles
Realms
HA
Search:
Type
Description
Disk us…
Memor…
CPU us…
Uptime
lxc 3818 (in.math.univ-toulouse.fr) 63.9 % 49.9 % 0.5% of… 44 days …
lxc 3829 (web-intranet.math.univ-toulou… 33.3 % 1.4 % 0.1% of… 44 days …
lxc 3831 (radius.math.univ-toulouse.fr) -
lxc 3832 (isc-dhcp.math.univ-toulouse.fr) -
lxc 3835 (isobuilder-lxc.math.univ-toulo… -
lxc 3836 (smai2021-backend.math.univ… 57.8 % 5.3 % 0.0% of… 44 days …
lxc 3840 (cimi-backend.math.univ-toulo… 22.6 % 1.4 % 0.0% of… 43 days …
lxc 3841 (Oxidized) 15.4 % 6.2 % 0.0% of… 44 days …
lxc 3845 (web-imt.math.univ-toulouse.fr) 80.4 % 5.9 % 0.1% of… 44 days …
lxc 3846 (web-mint-preprod.math.univ-t… 36.5 % 8.8 % 0.1% of… 44 days …
lxc 3847 (web-cimi-preprod.math.univ-t… 83.3 % 5.5 % 0.0% of… 44 days …
lxc 3848 (web-thb.math.univ-toulouse.fr) 49.3 % 8.0 % 0.4% of… 44 days …
lxc 3854 (headscale) 22.2 % 4.6 % 0.7% of… 22 days …
lxc 3855 (derp) 6.4 % 0.8 % 0.0% of… 7 days 2…
lxc 3857 (tsrouter) 33.8 % 12.0 % 0.4% of… 36 days …
lxc 38100 (hermes.math.univ-toulouse.fr) 2.2 % 2.0 % 0.1% of… 44 days …
lxc 221001 (agreg-dev) 38.3 % 7.5 % 0.1% of… 36 days …
lxc 221002 (pythonbox.math.univ-toulo… 5.0 % 0.4 % 0.0% of… 36 days …
lxc 221004 (pxcalc) 9.4 % 0.4 % 0.0% of… 44 days …
lxc 221005 (forms-backend) 25.2 % 2.9 % 0.0% of… 43 days …


PROXMOX Virtual Environment 8.0.4
Search
Documentation
Create VM
Create CT
gamba@UPS-TLSE.FR
Server View
Node 'pve1'
Reboot
Shutdown
Shell
Bulk Actions
Help
Datacenter (infra)
pve1
pve2
pve3
pve4
pve5
deleg
System
Network
Certificates
DNS
Hosts
Options
Time
Syslog
Updates
Repositories
Firewall
Disks
LVM
LVM-Thin
Directory
ZFS
Ceph
Configuration
Monitor
OSD
CephFS
Pools
Log
Health
Status
HEALTH_OK
Ceph Version:
17.2.6
Sev...
Summary
No Warnings/Errors
The newest version installed in the Cluster.
Status
OSDs
In
Out
Up
30
0
Down
0
0
Total: 30
PGs
active+clean:
193
Services
Monitors
Managers
Meta Data Servers
pve1:
pve2:
pve3:
pve4:
pve5:


netbox
Search
gamba
IPAM
IP ADDRESSES
IP Addresses
PREFIXES
Prefixes
Prefix & VLAN Roles
AGGREGATES
Aggregates
Customization
Operations
Admin
Prefixes
Hide Depth Indicators
Max Depth
Max Length
+ Add
Import
Export
Results 2
Filters
Quick search
Configure Table
Prefix
Status
Children
VRF
Utilization
Tenant
Site
VLAN
Role
Description
Tags
130.120.38.0/23
Active
0
Global
15.6%
public
public network
public
172.22.0.0/16
Active
0
Global
0.1%
private
internal servers
private
Per Page
Showing 1-2 of 2
Edit Selected
Delete Selected

MATHRICE + CALCUL

ANF Mathrice 2022, ANF UST4HPC 2023 :
convaincu par terraform

- terraform pour décrire les vm/ct créés ?

PROVIDER E-BREUNINGER/NETBOX

```
locals {
  nets = {
    public  = {
      ipRangeMsg = "130.120.38.0/23"
    }
    private = {
      ipRangeMsg = "172.22.0.0/16"
    }
  }
}
# data : read only access
data "netbox_prefixes" "nets" {
  for_each = {for k,v in local.nets: k => v}
  filter {
    name = "prefix"
    value = each.value.ipRangeMsg
  }
}

resource "netbox_available_ip_address" "want-an-ip" {
  prefix_id = netbox_prefixes.nets["private"]
}

resource "netbox_ip_address" "ips-static" {
  ip_address = "172.22.1.2/32"
}
```

PROVIDER TELMATE/PROXMOX

```
resource "proxmox_lxc" "basic" {
  target_node  = "pve3"
  hostname     = "myct"
  ostemplate   = "cephfs:vztmpl/ubuntu-20.04-standard_20.04-1_amd64.tar.gz"
  ssh_public_keys     = "ssh-ed255#19 …"
  unprivileged = true

  // Terraform will crash without rootfs defined
  rootfs {
    storage = "vms"
    size    = "8G"
  }

  network {
    name   = "eth0"
    bridge = "vmbr0"
    ip     = "172.22.1.2/32"
  }
}
```

La combinaison des deux devient vite complexe :

- ct ou vm ?
- choix du prefixe
- ip fixe ou automatique ?
- erreurs en cas de collision d’IP ?

DÉFINITION DE L'INTERFACE VOULUE

À partir d'un projet template sur plmlab, un·e ASR va

- plmlab : fork
- local : `git clone`
- local : fichier de config avec les token API
- local : modifier un fichier : spécifier les vm/ct à créer
- local : `terraform apply`

=> vm et/ou ct créés, avec ip, joignable(s) par ssh

La phase de provisionning est alors terminée.

Le fichier `servers.json` est généré, et contient les noms/ip des machines construites.

- local : modifier fichiers de configuration os (ansible, nix ...)
- local : `configure` : agit sur l'os par ssh

`=>` os configuré

- après test, `git add/commit/` et `git push`

On obtient un dépôt décrivant entièrement la machine et sa configuration.

PROJET INITIAL À FORKER

Basé sur devenv, présenté lors des JM Bordeaux 2023

- fournit la procédure à suivre dans le README
- installe les outils nécessaires
- définit des scripts pour gérer la configuration terraform
  - providers proxmox et netbox
  - vérifie la validité des token des API
  - gère l'état terraform : `tf-gitlab-init`

DÉMO

TERRAFORM : RETOURS D'EXPÉRIENCE

- `terraform apply` multiples parfois nécessaires : info provenant de la réalisation des ressources

Solution : décider de l'orde de création des ressources par des gestions de dépendances.

Ex : allouer une IP fixe ET une IP dynamique

```
# static ip
resource "netbox_ip_address" "ips-static" {
  for_each = { for s in module.servers: s.server.name => s if s.server.netIsIP}
  ip_address = "${element(split("/",each.value.server.ip),0)}/${module.net[each.value.server.net].net.mask}"
  status = "active"
  lifecycle {
    precondition {
      condition = can(regex("${module.net[each.value.server.net].net.ipRangeRegex}",
        "${element(split("/",each.value.server.ip),0)}" ))
      error_message = "${element(split("/",each.value.server.ip),0)} must be in
         ${module.net[each.value.server.net].net.ipRangeMsg}"
    }
  }
}

# dynamic ip
resource "netbox_available_ip_address" "dynamic_ips" {
  for_each = { for s in module.servers: s.server.name => s if !s.server.netIsIP}
  prefix_id = local.ipPrefix2Id[each.value.server.net]
  depends_on = [ netbox_ip_address.ips-static ] # force AFTER static ips
  status = "active"
}
```

HCL ~ MOTEUR DE TEMPLATE AU DESSUS DE JSON, PAS UN VRAI LANGAGE.

- boucle par attribut `for_each/count`, peu compréhensible
- pas de création de fonction ! contraint de passer par un module
  - validation des entrées
  - transformation ip en vmid

Passé les usages basiques, cela rend le code HCL peu lisible.

```
locals {
  proxmox-vm_servers = { for s in var.servers: s.name => s if s.terraform && s.type == "proxmox-vm" }
}

# proxmox_vm_qemu.vm_servers_${each.key} !
resource "proxmox_vm_qemu" "vm_servers" {
  for_each = local.proxmox-vm_servers
  clone             = each.value.template
  name              = each.key
  target_node       = each.value.pve_node
  vmid = module.ip2vmid[each.key].vmid
  # other attributes removed …

  # Cloud Init Settings
  ipconfig0         = "ip=${module.ip2vmid[each.key].ipWithoutMask}/32,
    gw=${module.net[each.value.net].net.gateway}" # beurk syntax
  sshkeys = each.value.ssh_public_keys
  provisioner "local-exec" {
    command = <<EOF
      ssh ${each.value.pve_node} -- qm set ${module.ip2vmid[each.key].vmid} --hookscript \
        local:snippets/priv_static_route.sh
      ssh ${each.value.pve_node} -- qm start ${module.ip2vmid[each.key].vmid}
    EOF
  }
}
```

```
# exemple de module terraform
# in
variable "ip" {
  description = "ipv4 with mask, e.g. 10.2.3.5/16"
  type = string
}
# out
output "vmid" {
  value = local.vmid
  description = "vmid"
}

output "vmidElems" {
  value = local.vmidElems
}

output "ipWithoutMask" {
  value = local.ipWithoutMask
}
# module code
locals {
  elems = split(".",element(split("/",var.ip),0))
  private = slice(local.elems,0,2) == tolist(["172","22"]) ? true : false
  vmidElems = local.private ? slice(local.elems, 1,4) : slice(local.elems, 2,4)
  vmidFormat = local.private ? "%s%s%.3d" : "%s%s"
  vmid = local.private ? format(local.vmidFormat, element(local.vmidElems,0),element(local.vmidElems,1),elemen
  ipWithoutMask = join(".",local.elems)
}
```

V

PERSPECTIVES

- rajouter des fonctionnalités dans le projet `deployment-template` :
  - activation des backups par Proxmox Backup Server. L'api existe, mais pas de provider terraform.
  - reconfigurer à partir d'un disque contenant des données initiales, e.g. dump base de données
- limite provider proxmox
- rendre le code terraform plus maintenable

UN «VRAI» LANGAGE POUR GÉNÉRER DU TERRAFORM

- pulumi : clone de terraform, moins de providers disponibles. Tous les providers ne sont pas dispo dans tous les langages :-(, communauté restreinte
- terraform CDKTF : code java/python/typescript/c#/go — synth —> conf terraform en json
- terranix : code nix –> terranix –> conf terraform en json

BOUCLE AVEC TERRAFORM

```
locals {
  myfiles = {
    f1 = { name = "first"
      content = "this is a file"
    },
    f2 = {
      name = "second"
      content = "and another one"
    }
  }
}

resource "local_file" "fileloop" {
  for_each = local.myfiles
  filename = each.value.name
  content = each.value.content
}
# resource "provider" "id"
```

Le json équivalent, généré par `hcl2json` :

```
{
    "locals": [
        {
            "myfiles": {
                "f1": {
                    "content": "this is a file",
                    "name": "fist"
                },
                "f2": {
                    "content": "and another one",
                    "name": "second"
                }
            }
        }
    ],
    "resource": {
        "local_file": {
            "fileloop": [
                {
                    "content": "${each.value.content}",
                    "filename": "${each.value.name}",
                    "for_each": "${local.myfiles}"
                }
            ]
        }
    }
}
```

BOUCLE AVEC NIX

```
# files.nix
{lib, ...}:

let myfiles = [
      {
        name = "first";
        content = "this is a file";
      }
      {
        name = "second";
        content = "and another one";
      }
    ];
    mkResourceLocalFile = f: s: (s // {
      "${f.name}" = {
        filename = f.name;
        content = f.content;
      };
    });
in

{
  resource.local_file =  lib.lists.foldr mkResourceLocalFile {} myfiles;
}
```

```
terranix files.nix > files.tf.json
terraform apply
```

```
{
  "resource": {
    "local_file": {
      "first": {
        "content": "this is a file",
        "filename": "first"
      },
      "second": {
        "content": "and another one",
        "filename": "second"
      }
    }
  }
}
```