

# Lift and shift in Cloud@CNAF il testing environment Kubernetes della DSI

Stefano Bovina - INFN CNAF  
Giuseppe Misurelli - INFN CNAF

# Argomenti trattati

- Governance dei cluster Kubernetes DSI
- Upgrade di Kubernetes
- La suite di test
- Approccio IaC
- Intro a Terraform
- Sviluppi futuri

# Governance dei cluster Kubernetes DSI

## Isolamento

Namespace per multitenancy (app, infra)

Network policy per controllare il traffico da e verso i pod

OS firewall

## Osservabilità

Monitoraggio e allarmistica via Prometheus (app, infra)

Log centralizzati in ELK (operatività, auditing)

## Deployment

Guidati da pattern GitOps via ArgoCD

Policy as code (OPA/OPA gatekeeper): enforce di policy DSI e best practices kubernetes (es: ContainerMustRunAsNonRoot)

## Sicurezza

ArgoCD unica entità abilitata ai deployment (insieme agli admin del cluster)

Certificati gestiti automaticamente via cert-manager (ACME)

Segreti applicativi (password, token) centralizzati in Vault

# Processo di release di Kubernetes

Una nuova versione minor ogni 15 settimane

N-2 support policy - fix di sicurezza e bug fino alle 3 versioni minor più recenti

Ogni versione supportata per 14 mesi - 12 di supporto e 2 di aggiornamento

Release	Released	Active Support	Maintenance Support	Latest
1.27	2 months and 1 week ago (11 Apr 2023)	Ends in 10 months (28 Apr 2024)	Ends in 1 year (28 Jun 2024)	<a href="#">1.27.3</a> (14 Jun 2023)
1.26	6 months ago (08 Dec 2022)	Ends in 6 months (28 Dec 2023)	Ends in 8 months (28 Feb 2024)	<a href="#">1.26.6</a> (14 Jun 2023)
1.25	9 months and 4 weeks ago (23 Aug 2022)	Ends in 2 months and 1 week (27 Aug 2023)	Ends in 4 months (27 Oct 2023)	<a href="#">1.25.11</a> (14 Jun 2023)
1.24	1 year and 1 month ago (03 May 2022)	Ended 3 weeks ago (28 May 2023)	Ends in 1 month and 1 week (28 Jul 2023)	<a href="#">1.24.15</a> (14 Jun 2023)
<del>1.23</del>	1 year and 6 months ago (07 Dec 2021)	Ended 5 months and 3 weeks ago (28 Dec 2022)	Ended 3 months and 2 weeks ago (28 Feb 2023)	<del><a href="#">1.23.17</a></del> (22 Feb 2023)

Fonte <https://endoflife.date/kubernetes>

# Aggiornare i cluster Kubernetes della DSI

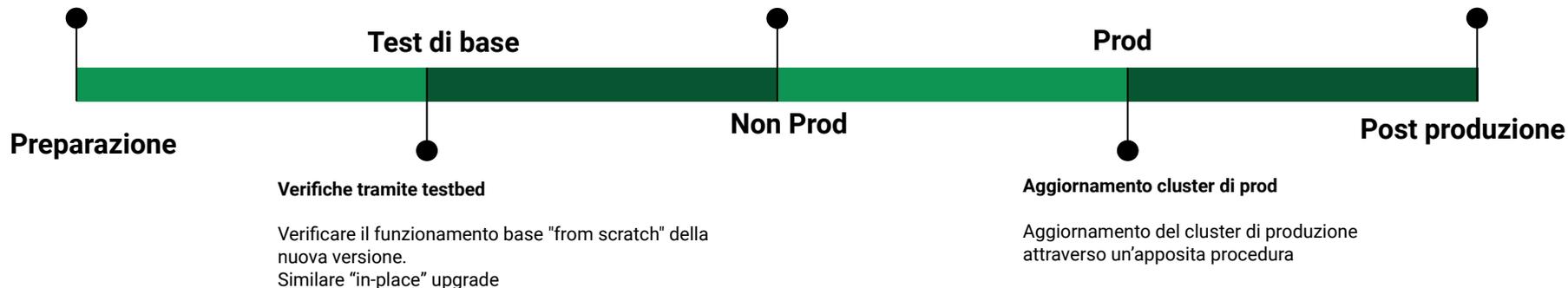
Runbook con procedura semi-automatica (Testbed >> Dev >> Test >> Prod)

## Attività preliminari

Analisi highlights nuova release kubernetes e RKE2.  
Verifica/adequamento "applicazioni": devono supportare versione corrente e versione target dell'upgrade (es: prometheus).  
Verifica/adequamento "deprecations" ed altre criticità mediante analisi statica del codice e analisi dei log

## Consolidamento

Verifica/adequamento "deprecations" e altre criticità mediante analisi statica del codice e analisi dei log.  
Aggiornamento manifests k8s (es: cambio di api beta --> stable, svecchiamento codice ecc).  
Verifica ticket aperti (es: security issues, bug ecc)



# Requisiti e necessità DSI - cosa testare

Utilizzo di testbed "complesso" simile alla produzione:

- Load balancer (LB), control-plane e worker su nodi separati
- Firewall configurato
- Configurazioni di base simil-produzione
- Servizi di base simil-produzione

Verifica di 2 scenari differenti:

- Setup "from scratch" della nuova versione
- Simulazione upgrade (setup "from scratch" versione corrente ed "in-place" upgrade alla nuova)

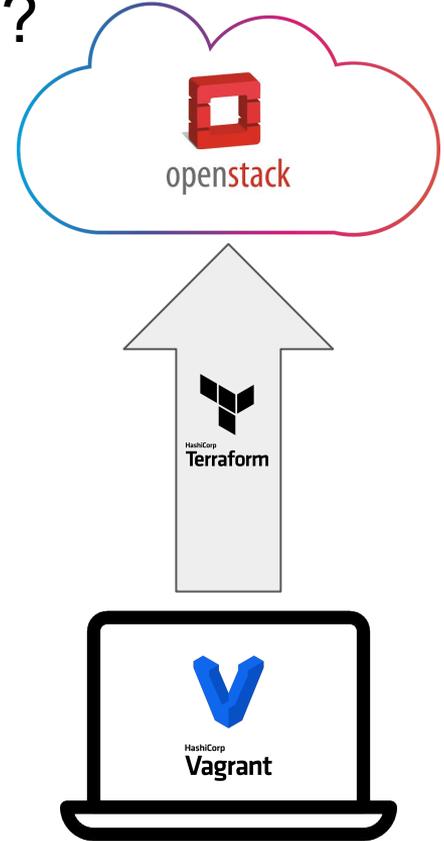


# La suite di test

- Setup automatico dei nodi
- Verifica di base dei servizi RKE2
- Verifica componenti base:
  - stato dei nodi
  - stato API k8s
  - corretta configurazione kube-proxy (IPVS)
  - stato dei deployment, daemonset, ecc
  - verifica stato Cilium ed esecuzione test suite ufficiale
  - verifica corretto funzionamento network policy mediante test suite [bats](#)
- Setup e verifica ingress controller e deployment servizio di test
- Verifica corretta "esposizione" servizio di test tramite ingress (sia da nodi k8s che da LB)

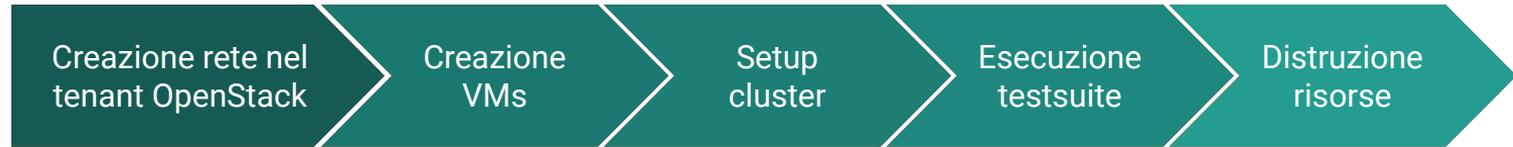
# Perché migrare la testsuite a Cloud@CNAF?

1. Limiti mostrati dall'approccio Localdev basato su Vagrant (es. scarso supporto ad architetture ARM)
2. Necessità di VM (gestite alla Localdev) senza stravolgere la testsuite
3. Ambiente di test "disposable" da attivare alla bisogna e distruggere appena fatto



# Approccio tenuto nella migrazione

*Lift and shift (aka rehosting) is a way to migrate an application to a cloud provider without rewriting the app or workflow to be optimized for that modern cloud environment.*



# Principi guida

## Infrastructure as Code

Utilizzare il più possibile codice dichiarativo per l'infrastruttura e per orchestrare i cambiamenti successivi

Evitare errori umani e scarsa scalabilità dell'interazione "pointing-and-clicking" nella console cloud

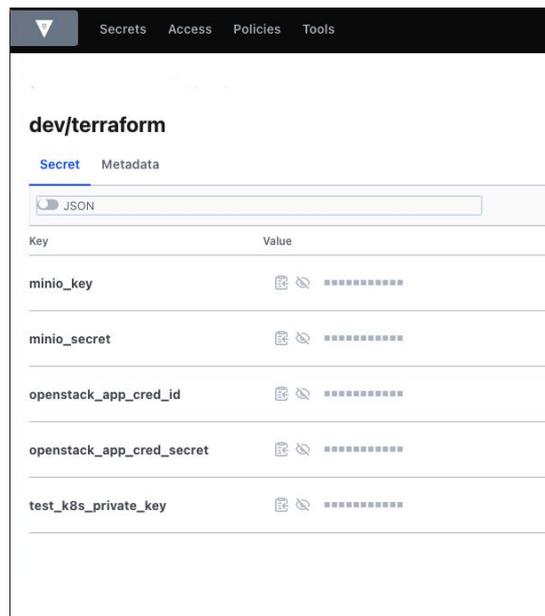
## Sicurezza

Mantenere il giusto livello di sicurezza per le credenziali usate (es. OpenStack Application Credentials)

L'automatismo deve avere accesso alle credenziali solo per il tempo necessario al ciclo creazione risorse, esecuzione test, distruzione risorse

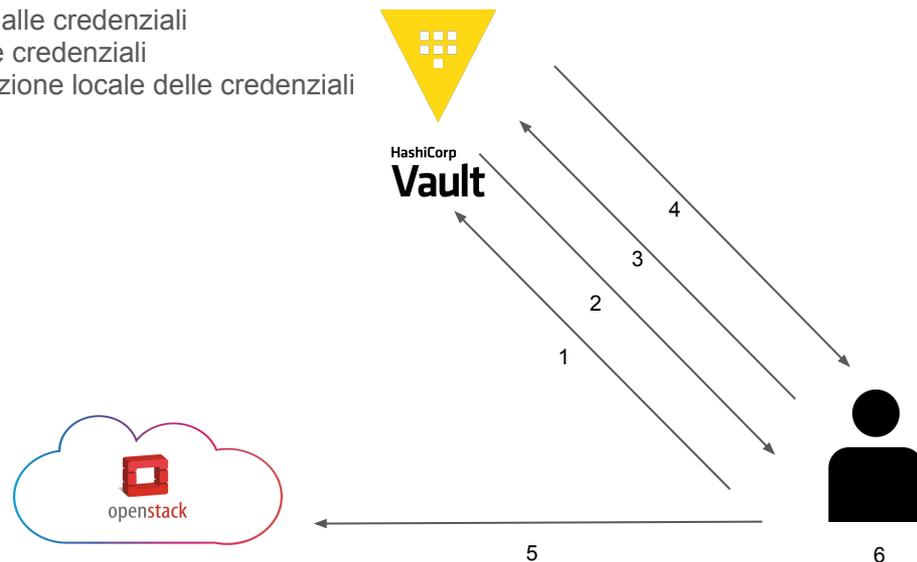
# Sicurezza con Vault

## Centralizzazione delle credenziali in gioco



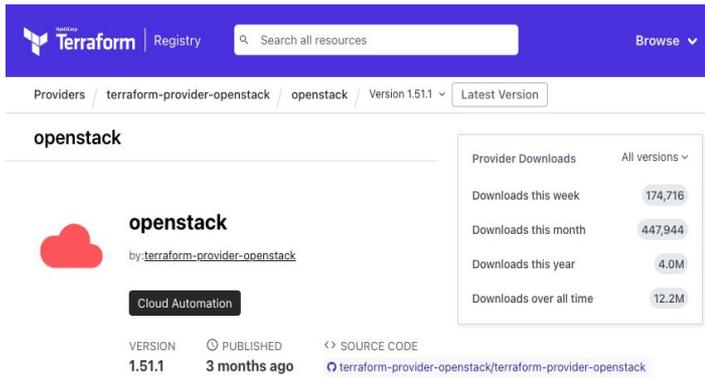
## Uso delle credenziali

1. Login in Vault
2. Vault token
3. Creazione short lived token e revoca token ricevuto in 2
4. Accesso alle credenziali
5. Uso delle credenziali
6. Cancellazione locale delle credenziali



# Infrastructure as Code con Terraform

## Provider OpenStack supportato



The screenshot shows the Terraform Registry page for the 'openstack' provider. The page header includes the Terraform logo, 'Registry', a search bar, and a 'Browse' dropdown. Below the header, there are navigation links for 'Providers', 'terraform-provider-openstack', 'openstack', and 'Version 1.51.1' with a 'Latest Version' button. The main content area features the 'openstack' logo, the text 'by terraform-provider-openstack', and a 'Cloud Automation' badge. A table shows download statistics: Provider Downloads (All versions), Downloads this week (174,716), Downloads this month (447,944), Downloads this year (4.0M), and Downloads over all time (12.2M). At the bottom, it lists the version '1.51.1' published '3 months ago' and provides a link to the source code.

Downloads	All versions
Downloads this week	174,716
Downloads this month	447,944
Downloads this year	4.0M
Downloads over all time	12.2M

Risorse e moduli per coprire l'interno ciclo creazione risorse >> esecuzione test >> distruzione risorse

```
module "test-k8s" {
  source = "tf-openstack-modules/networks/openstack"
  version = "0.1.0"

  network = {
    name = "net-test-k8s"
    subnet_name = "sub-test-k8s"
    cidr = "192.168.20.0/24"
  }

  router_id = resource.openstack_networking_router_v2.sysinfo.id
  dns_ip = []
}

resource "openstack_networking_floatingip_v2" "rk2lb" {
  pool = "public"
}

resource "openstack_networking_floatingip_v2" "rk2server" {
  pool = "public"
}

resource "openstack_networking_floatingip_v2" "rk2client" {
  pool = "public"
}

resource "openstack_compute_instance_v2" "node" {
  for_each = var.nodes
  name = each.key
  image_id = local.image_id
  flavor_name = each.value.flavor_name
  key_pair = local.keypair
  security_groups = [openstack_compute_secgroup_v2.sg-test-k8s.name]

  metadata = {
    "rke2role" = each.value.role
  }

  network {
    name = local.network_name
    fixed_ip_v4 = each.value.priv_ipv4
  }

  resource "openstack_compute_floatingip_associate_v2" "node-foip" {
    for_each = var.nodes
    floating_ip = each.value.pub_ipv4
    instance_id = openstack_compute_instance_v2.node[each.key].id
  }

  resource "null_resource" "cp-setup-node-folder" {
    for_each = var.nodes
    provisioner "file" {
      source = local.local_setup_folder
      destination = local.remote_setup_folder
    }

    connection {
      type = "ssh"
      user = "almalinux"
      private_key = file(local.private_key_path)
      host = each.value.pub_ipv4
    }
  }
}
```

# Terraform - cos'è

Tool di Infrastructure as Code di Hashicorp - dal 2014 uno dei più usati

Si basa sul paradigma di codice dichiarativo (HCL) per la formalizzazione delle risorse da creare/modificare/distruggere

Si avvale di una serie di providers con i quali interagisce con le API di cloud providers e qualsiasi altro servizio

Fa uso di uno “state file” per tenere traccia dello stato delle risorse gestite

Fonte: [What is Infrastructure as Code with Terraform?](#)

```
# Define required providers
terraform {
  required_version = ">= 0.14.0"
  required_providers {
    openstack = {
      source = "terraform-provider-openstack/openstack"
      version = "~> 1.48.0"
    }
  }
}

# Configure the OpenStack Provider
provider "openstack" {
  tenant_name = "SYSINFO"
  region      = "sdds"
}

# Remote state file in S4
terraform {
  backend "s3" {
    bucket = "tf-remote-state-testbed"
    key    = "test-k8s"

    endpoint = "https://_____."

    region = "main"
    skip_credentials_validation = true # skip credentials validation via the AWS STS API
    skip_metadata_api_check    = true # skip the AWS Metadata API check
    skip_region_validation     = true # skip validating the AWS region
    force_path_style           = true # enable path-style AWS S3 URLs
  }
}
```

# Terraform - funzionamento

## Ambito

Identificazione del contesto in cui si andranno a creare le risorse (provider, risorse, API)

## Configuration files

Uso delle risorse messe a disposizione dal provider per dichiarare lo stato in cui deve trovarsi l'infrastruttura e le sue risorse

## Inizializzazione

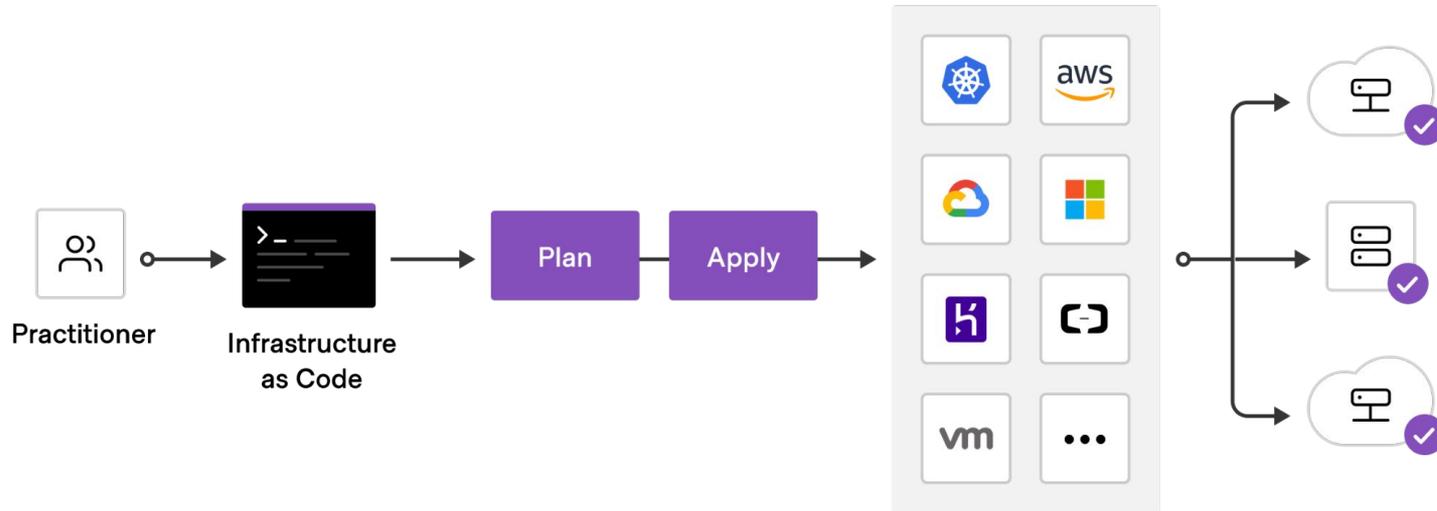
Installazione dei plugins necessari al provider

## Plan

Riscontro sui cambi che Terraform andrà a fare

## Apply

Applicazione delle cambi dichiarati nel plan



# Terraform - concetti chiave (non esaustivi)

terraform {...}

Blocco contenente i settings  
Terraform come i providers da usare

```
# Define required providers
terraform {
  required_version = ">= 0.14.0"
  required_providers {
    openstack = {
      source = "terraform-provider-openstack/openstack"
      version = "~> 1.48.0"
    }
  }
}
```

resource {...}

Blocco di definizione dei  
componenti dell'infrastruttura

```
resource "openstack_compute_instance_v2" "node" {
  for_each = var.nodes
  name     = each.key
  image_id = local.image_id
  flavor_name = each.value.flavor_name
  key_pair  = local.keypair
  security_groups = [openstack_compute_secgroup_v2.sg-test-k8s.name]

  metadata = {
    "rke2role" = each.value.role
  }

  network {
    name = local.network_name
    fixed_ip_v4 = each.value.priv_ipv4
  }
}
```

backend "remote" {...}

Non necessario ma fortemente  
raccomandato per workflow Terraform  
collaborativi (centralizzazione state file)

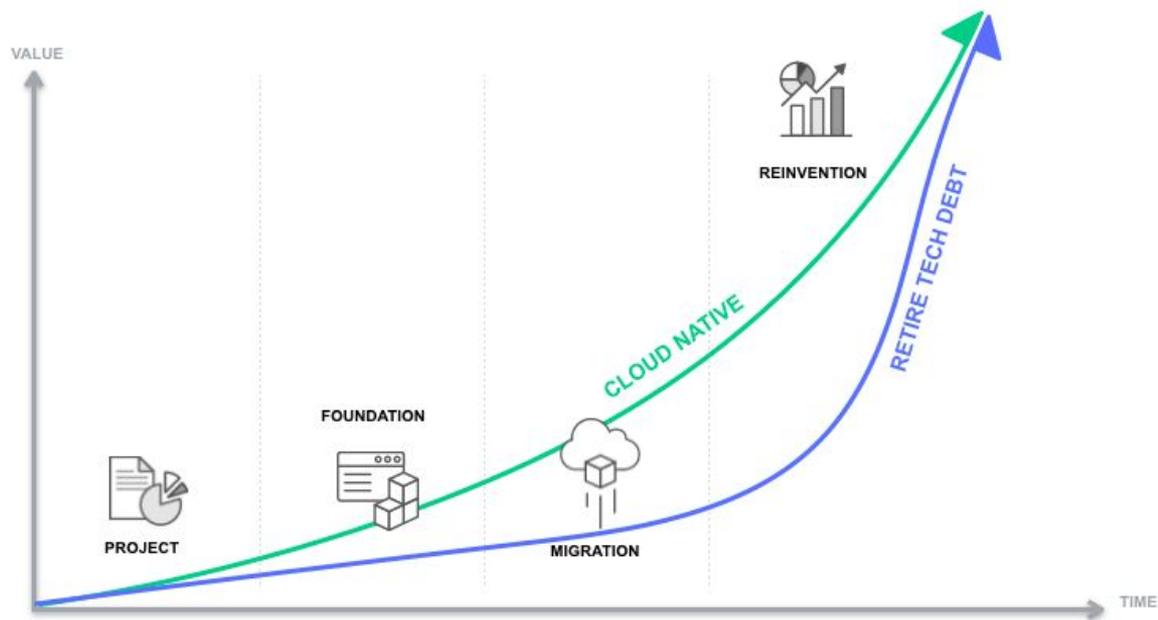
```
backend "s3" {
  bucket = "tf-remote-state-testbed"
  key    = "test-k8s"

  endpoint = "https://_____"
```

Risorse provider OpenStack : [OpenStack Provider](#)

# Sviluppi futuri

Pensando in grande



Fonte: [How the Evolution of Cloud Computing Is Accelerating Builder Velocity](#)

# Verso una soluzione più nativa Cloud

- Autenticazione con credenziali temporanee
  - OpenStack Identity vs Vault plugin?
- Modulo Terraform upstream
  - Ottimizzazione del codice Terraform
  - Flessibilità nelle diverse tipologie di deployment
  - Automatizzare l'upgrade
- Pipeline CI/CD
  - Asincronia nell'esecuzione dei test
  - Ottimizzazione consumo log

# Ringraziamenti

Diego Michelotto e il Cloud@CNAF team per la creazione del tenant e il supporto fin qui avuto

