

# HashiCorp Taller de Vault

- ponente: Pedro Coca (gerente regional)

## introducción modelo operativo cloud

- herramientas
  - vagrant
  - packer
  - terraform
  - vault
  - consul
  - nomad
- elementos infraestructura
  - conectividad
  - desarrollo
  - seguridad
  - operaciones
- transición a un mundo multi-cloud
  - sistemas de petición → auto gestión
  - llamadas API a servicios



- soluciones Hashicorp:
  - Nomad = desarrollo
  - consul = conectar
  - vault = securizar
  - terraform = infraestructura

## terraform

- núcleo + proveedores (ofrecen API)
  - no solo nube pública
  - proveedores secundarios: +1000
- reutilización, control versiones, automatización
- marco de control (compliant) → terraform enterprise
  - mejores prácticas
  - políticas (as Code)

## vault

- dispersión de secretos, falta de control
  - postit
  - archivo de texto
  - control de versiones
  - logs
- modelo:
  - autenticación
  - cliente
  - sistema

- centralización de secretos
  - más rotación, periodos más cortos
- encriptación como servicio
  - encriptar todo el tráfico de aplicación sin modificar la aplicación
- protección avanzada de datos

## consul

- conectar aplicaciones
- migración de monolitos a microservicios
- middleware:
  - load-balancers
  - firewalls
- registro de aplicaciones
  - ip-address → name
  - descubrimiento de servicios
- confianza cero
- service mesh
- consul intentions: definición control de acceso para servicios (multicloud)

## nomad

- orquestador cargas de trabajo
- contenedores o tradicionales

## taller vault

### info

- <https://hashicorp.github.io/field-workshops-vault/slides/multi-cloud/vault-oss/#1>
- <https://play.instruqt.com/hashicorp/tracks/vault-basics>

### overview

- API
- agnóstico plataforma
- manejo de secretos centralizado
  - credenciales dinámicas de corto plazo ++
  - encriptación on-the-fly
- «castle and moat» : castillo+fosa → protección capas/perímetros
  - firewalls, reglas por IP
  - al final las credenciales se almacenaban en el código o en sitios estáticos
  - problemas modelo tradicional
    - restricciones por IP (con cada vez más IPs en danza)
    - revocación contraseña
    - rotaciones, cambios, acceso(log)
    - usuarios/aplicaciones
  - concepto de identidad
    - usuarios: AD/LDAP
    - token
    - trasciende los perímetros de seguridad

- dinámico, credenciales de corto plazo, rotadas frecuentemente
- credenciales y entidades pueden ser invalidados fácilmente
- motores de secretos Vault
  - <https://www.vaultproject.io/docs/secrets>
  - <https://www.vaultproject.io/docs/internals/architecture>
- alta disponibilidad
  - 3 nodos
    - en versión free, 1 activo, 2 en espera
    - no más de 8ms de diferencia entre nodos
  - vault enterprise replication
  - cluster disaster recovery

## chap2: interactuando con Vault

- interacción
  - CLI (wrapper)
  - UI (wrapper)
  - API

### cli

- aplicación GO
- multiplataforma

### instruqt

- modo DEV (para desarrollo, todo en memoria, para trastear)
- `vault server -dev -dev-listen-address=0.0.0.0:8200 -dev-root-token-id=root`
- `vault kv put secret/my-first-secret age=48`
- `curl http://localhost:8200/v1/sys/health | jq`
- `curl -header «X-Vault-Token: root» http://localhost:8200/v1/secret/data/my-first-secret | jq`

### production vault server

- configuración parámetros
  - HCL o JSON
  - listener:
  - storage: donde se guardan los datos <https://www.vaultproject.io/docs/configuration/storage>
    - Hashicorp solo mantiene 4: consul, integrado (raft)
  - seal:
  - log\_level
  - ui
  - api\_addr
  - cluster\_addr
- keys
  - llave almacenamiento
  - llave maestra
    - key-holders en su momento: dividir la llave en varias personas, todas necesarias para volver a levantar el servicio
    - proveedores externo para almacenar llaves
  - llave sello

- evita key-holders (5 por defecto, mínimo 3)
- `vault operator init` → ceremonia de inicialización
  - `-key-shares`
  - `-key-threshold`
  - llaves gpg/pgp enviadas al cluster
    - retorna root-token, por encima de políticas y auditorias
    - retorna llaves recuperación, una por cada key-holder, cifrada con su clave GPG
  - se establece, a través del root-token:
    - `auth`
    - `basic policy`
    - auditoria
  - una vez hecho, se revoca la root-token y los key-holders ya pueden desaparecer (ya no hay manera de no hacer nada sin ellos)
  - `unsealing vault server`
    - `vault operator unseal`
  - guía securización: <https://learn.hashicorp.com/vault/operations/production-hardening>
  - [https://en.wikipedia.org/wiki/Secure\\_multi-party\\_computation](https://en.wikipedia.org/wiki/Secure_multi-party_computation)
  - [https://en.wikipedia.org/wiki/Shamir%27s\\_Secret\\_Sharing](https://en.wikipedia.org/wiki/Shamir%27s_Secret_Sharing)

## chap4:vault secret engines

- muy extensible
  - Key/Value (KV)
  - PKI
  - SSH
  - TOTP
  - proveedores públicos
  - ...
- `vault secrets enable`
- `vault secrets enable -path=aws-east aws`
- KV 2 versiones
  - v1 no soporta versionado
- en modo de producción no se habilita por defecto
- `vault secrets enable -version=2 kv`
- `vault kv put kv/a-secret value=1234`
- `vault kv put kv/a-secret value=4321`
- `vault kv get -version=2 kv/a-secret`

## chap5: vault authentication method

- usuarios
  - ...
- aplicaciones
  - ...
- `vault auth enable`

## otros

- <http://nicolas.corrarello.com/general/vault/security/ci/2017/04/23/Reading-Vault-Secrets-in-your-Jenkins-pipeline.html>
- <https://www.vaultproject.io/docs/auth/ldap>
- <https://www.hashicorp.com/resources/managing-vault-with-terraform/>

- <https://www.vaultproject.io/docs/secrets/transform>
- <https://www.vaultproject.io/>
- <https://hashicorp.github.io/field-workshops-vault/slides/#1>

certificación vault

From:

<https://matebcn.eu/> - **miguel angel torres egea**

Permanent link:

<https://matebcn.eu/info:cursos:hashicorp:vault>

Last update: **26/05/2020 17:12**

