HashiCorp

Vault

The basics of running Vault as your security foundation

Paralint
Sécurité, code, applications

guillaume@paralint.com

What does it mean to be secure?

Security properties

Confidentiality	Integrity	Availability
• Your data is kept secret (need to know basis)	• Your data is not tampered with	• You can get your data when you need it

Process

Security controls

Authentication	Authorization	Audit
• Proof that the user is who he pretends to be	• Limits what a user can do	• You record what the user did

How Vault solves common security problems

- Vault handles the common security use case
  - Passwords in configuration files
  - Changing secrets (passwords or keys) when someone leaves
  - Granting access to a subset of the security realm
  - Master key management (generation, rotation, sharing)
- Provides a strong toolbox of security controls to support your process
  - Segregation of duties capable
  - Simple “least privilege” security model
- You will still have some passwords in a file when you are done
  - But it will be surrounded by security controls

Vault components (or “back ends”)

Manages secrets and passwords

Secret engines

- **Key/Value**
- AWS
- **PKI**
- SSH
- Databases

Authentication

- LDAP
- AWS
- AppRole
- **Built-in users**

Issues token in exchange for credentials

Vault

Storage of encrypted data

Storage

- AWS
- **Consul**
- Filesystem
- Google Cloud
- Database

Audit

- Syslog
- **Filesystem**

Records everything that happened to anything

Vault paths

- Vault uses path for naming things
  - Paths have a general format `/{API version}/{backend}/{instance}*/{.*}`
  - Vault defaults are reasonable, aligned with Vault Enterprise UI
  - Some planning required for multi-tenancy or other advanced scenarios
  - Version number omitted by command-line tool
- Vault mounts secret and authentication backends to a specific path
  - The Generic secret backend is mounted automatically to `/secret`
  - Other backends are mounted as per your configuration
    - /aws /auth/aws
  - You can mount, unmount and configure secret and auth backends at runtime
- Paths are closely tied to the policies

Running Vault

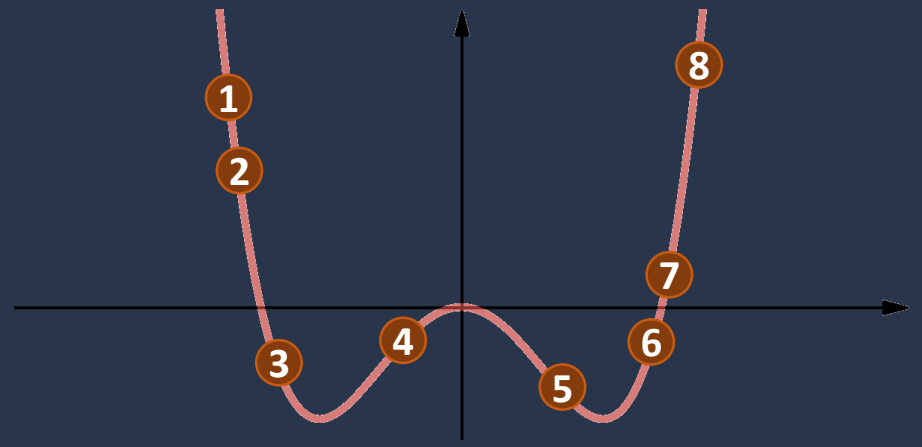
- Dev mode is a useful one-liner for continuous integration
  - Stores secrets in volatile memory
- Choosing a storage back end
  - **First thing you must do**
  - Only a single storage back end can be mounted at a time
  - Not easy to switch storage back end, choose carefully
  - Consul is the only Hashicorp supported HA backend
- A quick "Vault on Consul" setup with room to grow is available on GitHub https://github.com/Paralint/vault-pki-starter

Hands-on: Starting Vault

Run a Vault server with a Consul storage backend

Vault's own master key

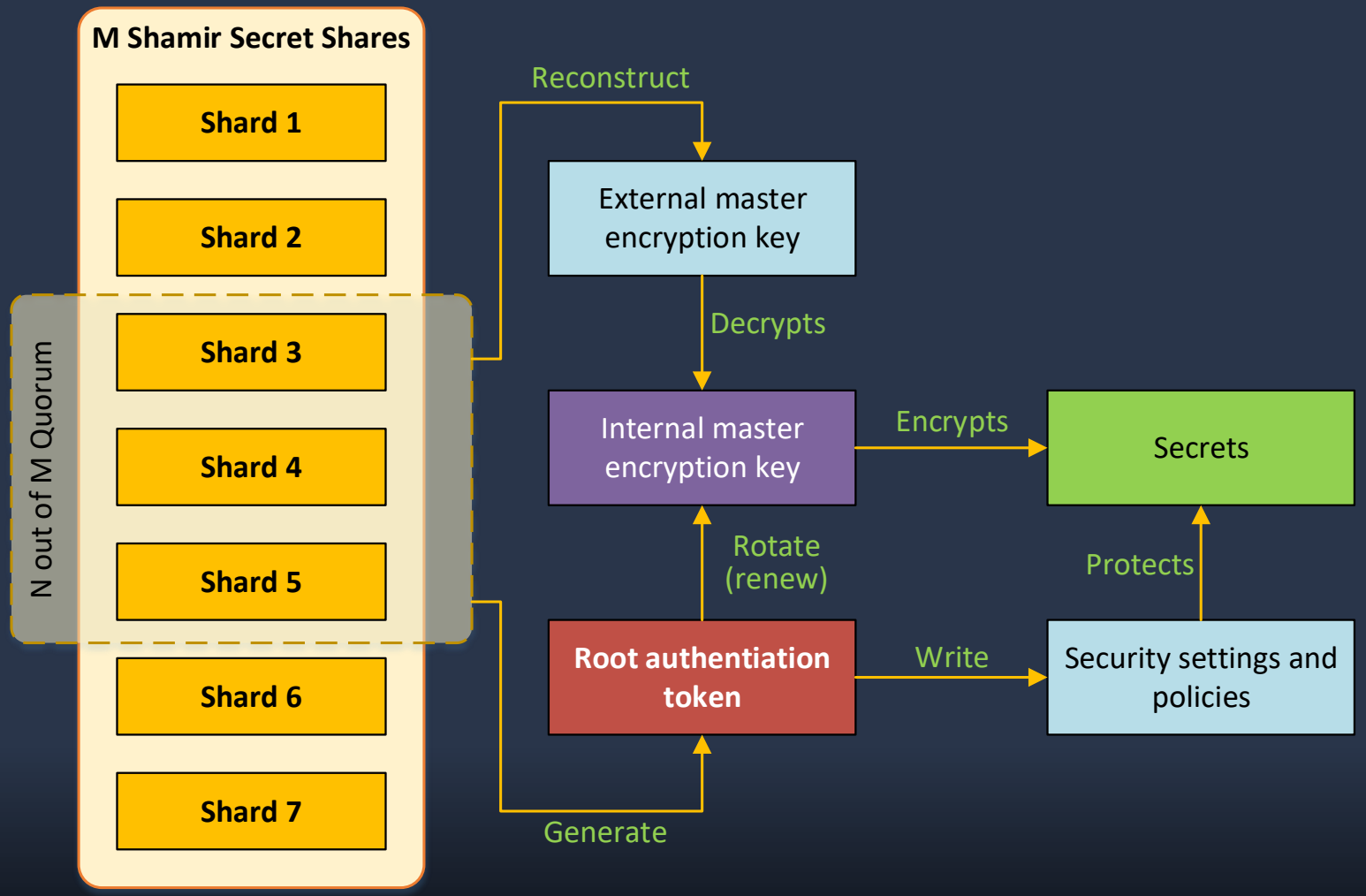
- Vault master key is never persisted to disk
  - Uses Shamir secret sharing scheme
- The master is split amongst N people
  - A piece of the master key is a called a "Shard"
  - You can have as many as you want



$$f(x) = ax^4 + bx^3 + cx^2 + dx + e$$

- A quorum of N shards must be met to reconstruct the master key
  - Vault default is 3 person out of 5 shard holders
  - Those numbers can be changed, but it requires quorum
- Choosing the quorum size and number of shards is Vault Initialization
- When vault starts, it is sealed. Quorum is required to "unseal" it

Vault internal keys



Hands-on: Initialize Vault

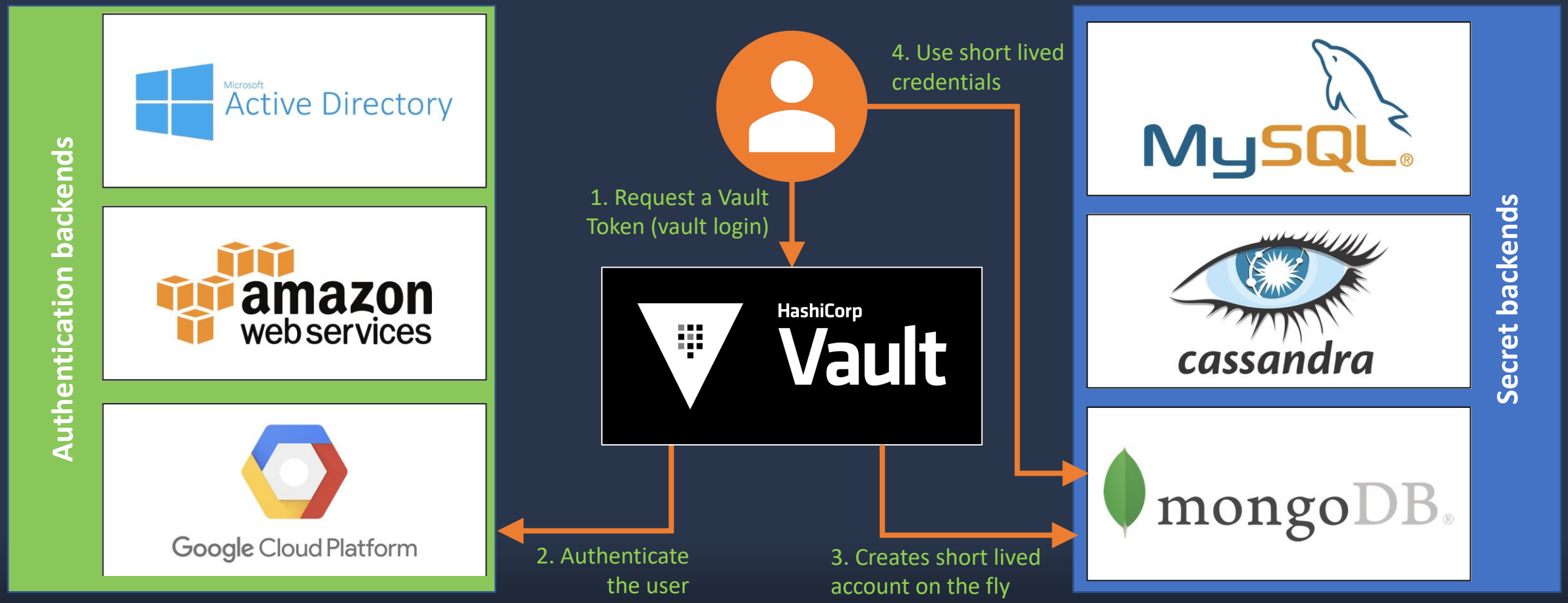
Initialize Vault with a 2 out of 7 quorum

Authentication

- Vault can authenticate users in many ways
  - Any combination of any number of instances of any protocol
  - Always mounted under Vault path `/auth` (`/auth/ldap`, `/auth/approle`, etc.)
  - Complete list and details at https://www.vaultproject.io/docs/auth/
- There are no default Authentication mount
- Every request to Vault requires a Vault token
  - A token is tied to policies
  - You should not use or even keep the root token
- Authentication backends exchange user's credential for a Vault token

```
vault login –method=ldap username=guillaume
```

Brokering cloud identity



https://www.hashicorp.com/blog/brokering-cloud-identity

Do not run as Vault root

- The root token is all powerful and it can read and write anything
- You should revoke the root token after initial configuration
- If you need to change the configuration, generate a new root token
  - Requires quorum, which is a security control
- Mount as many authentication backend as required
  - You can mix and match authentication backends and policies
- Requires thoughts in getting your process right
- If you are running with the root token, you've not getting the whole security you deserve after this hard work.

Authorization

- Authorization are specified as policies
  - A policy ties one or more Vault path with one or more rights
- There are two default policy
  - Root that can do anything
  - Default that can’t do much
- You will usually give rights to the secret backends

Key/Value (Generic) Secret backend

- Stores anything that can be represented in JSON
  - Simplistic Key-Value storage, not a general purpose K/V
  - Does not have search or aggregation capabilities
  - Does not support update, you must read and overwrite
- Useful to store unmanaged passwords or other secrets
  - Password to a database not connected to Vault as a secret backend
  - The combination of a safe
  - An SVG file showing locations of hidden physical keys

Hands-on: Authentication

Authenticate and authorize users

Thanks!