

# Taming secrets with Vault

Jérémy Courtial - Software Security Architect  
Oodrive

# Secrets & Sensitive data

```
zookeeper:  
  hosts: zk.services.net  
  environment: dev
```

```
datasource:  
  driverClassName: org.postgresql.Driver  
  username: tk_user  
  password: qS5Ji;*bY*pX,94~gepF
```

```
management:  
  enabled: true  
  context-path: /manage  
  health:  
    enabled: true  
  master-key: YXp_lcnR5dW_lvcHFzZGZnaGprbG13eGN2Ym4s0zo9QCMK
```

```
server:  
  port: 8080
```

```
zookeeper:  
  hosts: zk.services.net  
  environment: dev
```

```
datasource:  
  driverClassName: org.postgresql.Driver  
  username: tk_user  
  password: qS5Ji;*bY*pX,94~gepF
```

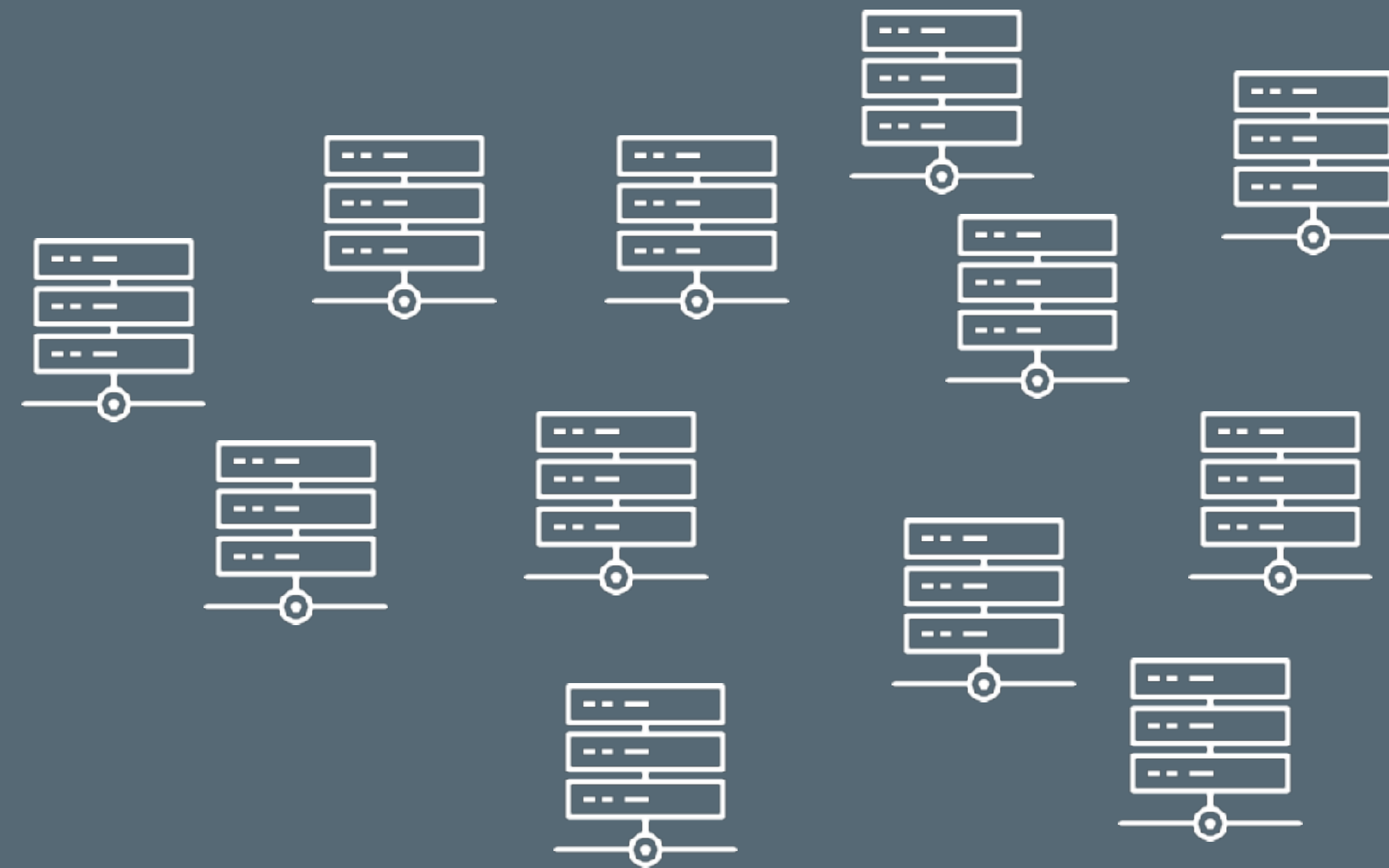
```
management:  
  enabled: true  
  context-path: /manage  
  health:  
    enabled: true
```

```
master-key: YXpLcnR5dWlvcHFzZGZnaGprbG13eGN2Ym4s0zo9QCMK
```

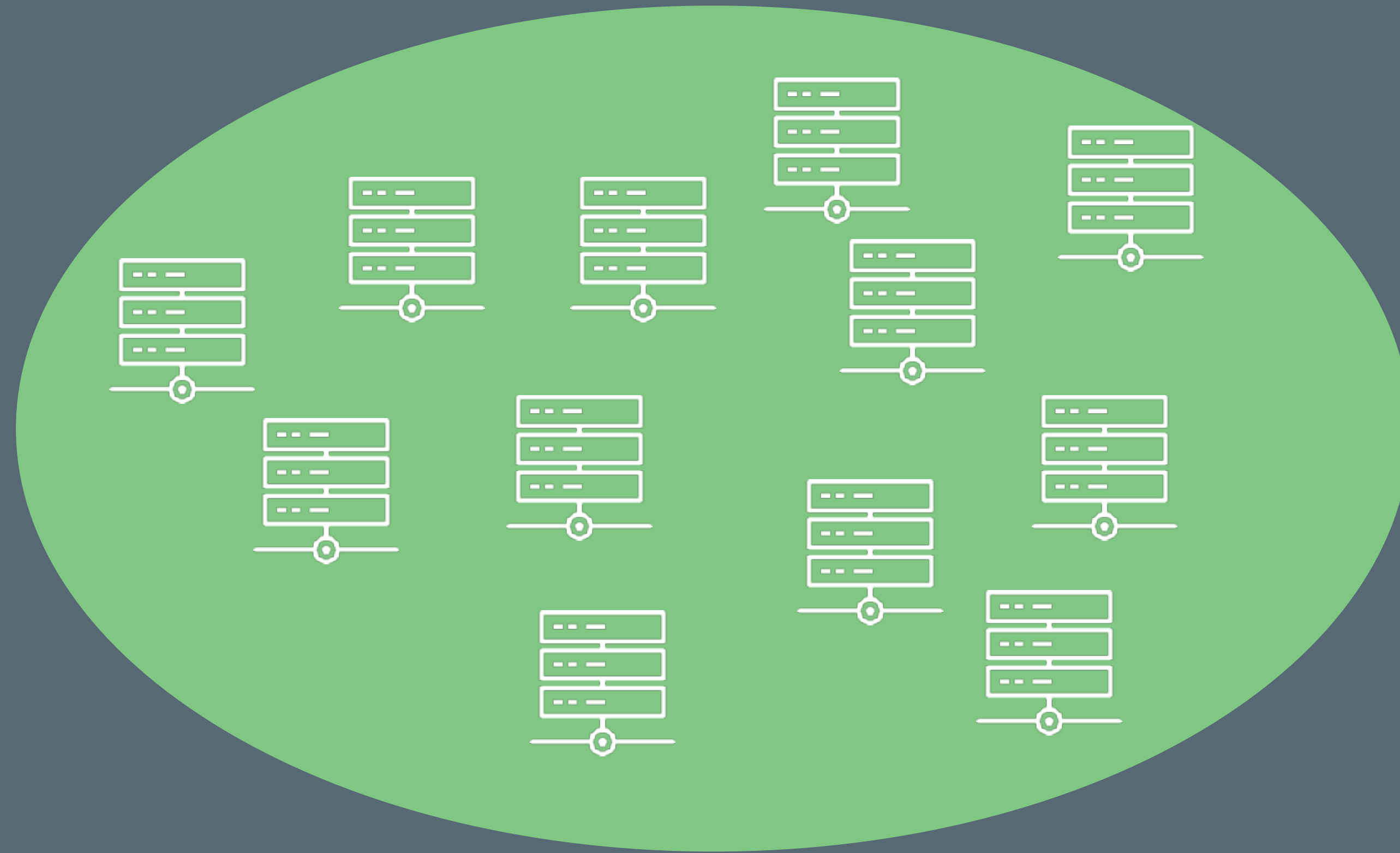
```
server:  
  port: 8080
```

Why bother?

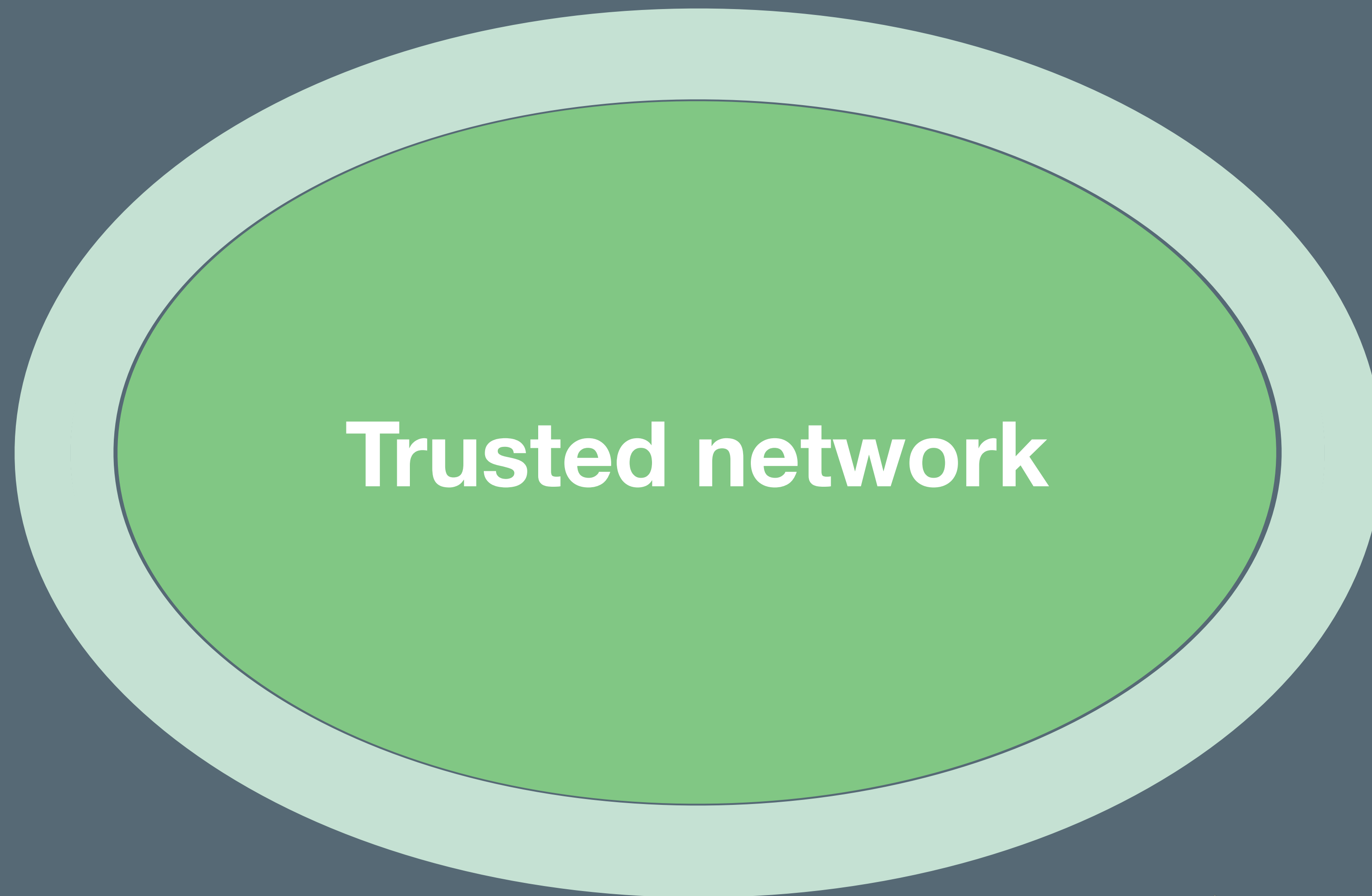
# Traditional security



# Traditional security



# Traditional security

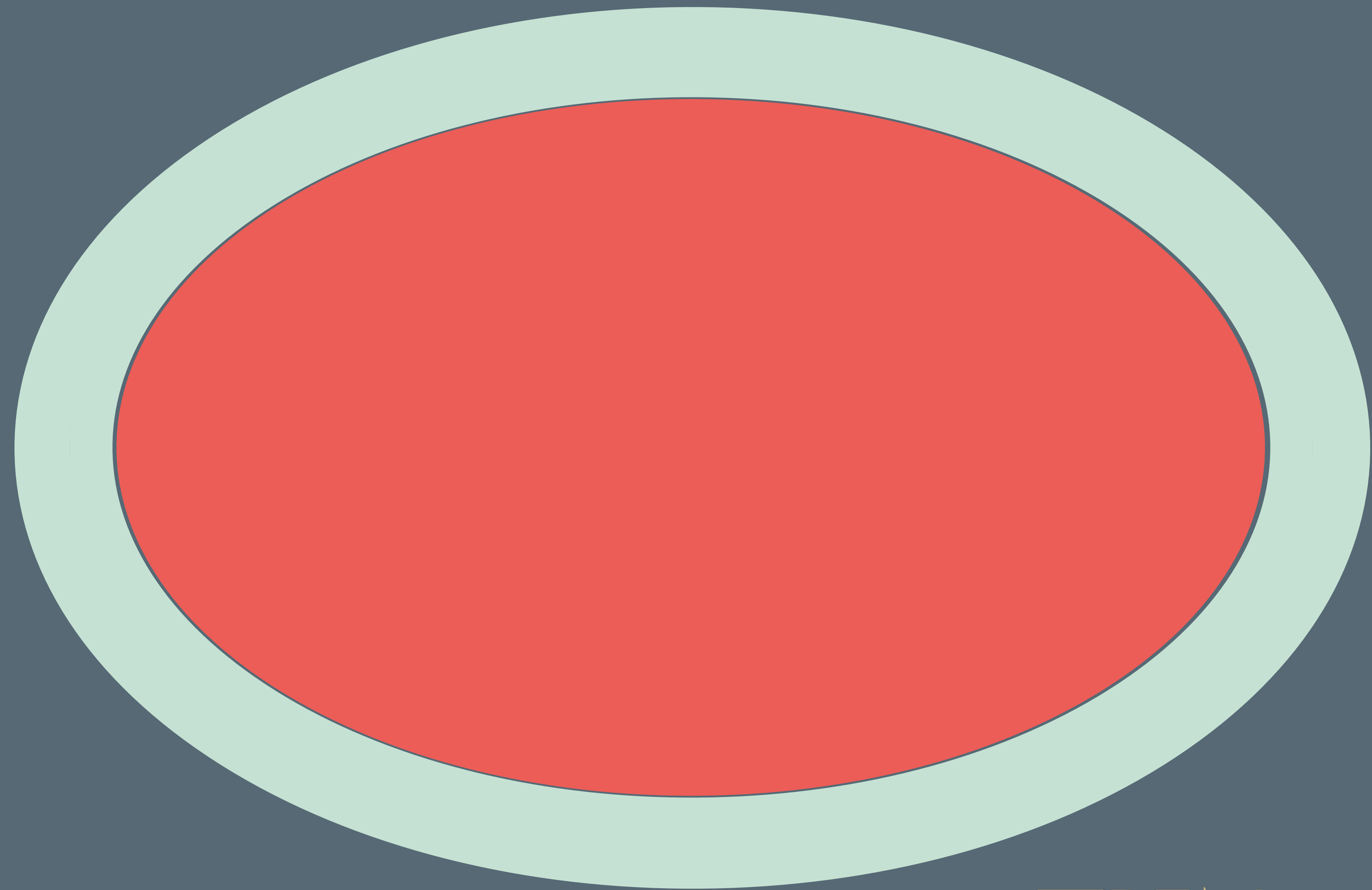




# Traditional security

```
def this_is_fine
  eval(someStr)
end
```

# Traditional security

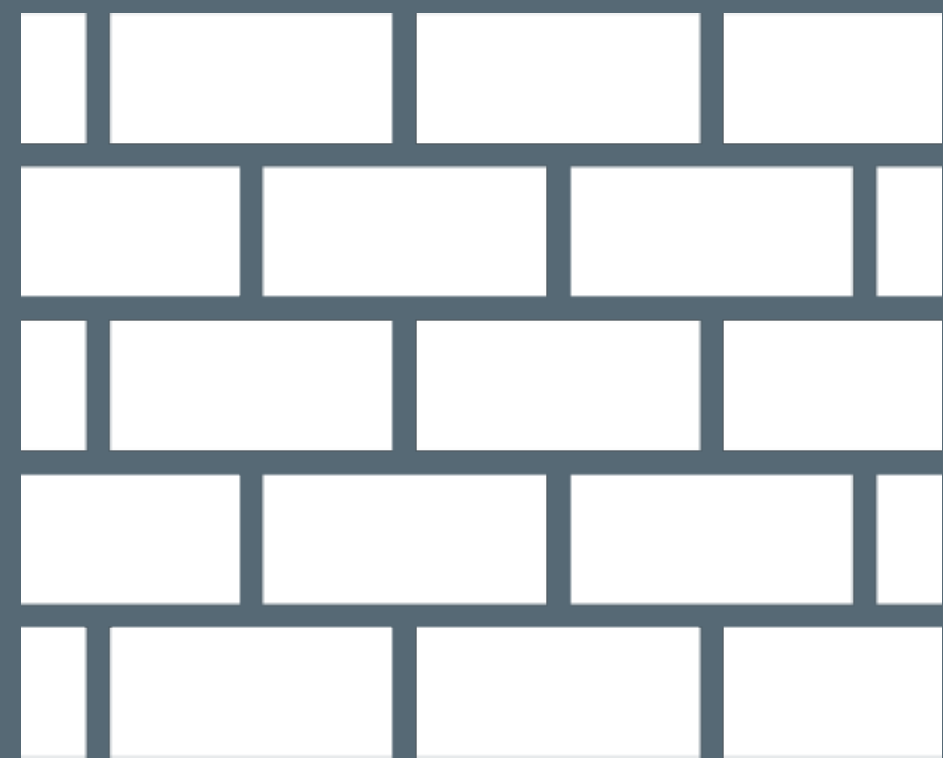


*"I'm simply saying that hackers,  
uh... finds a way."*

Dr. Ian  
Malcolm



# Defense in depth



Don't rely on a single line of defense

# Defense in depth



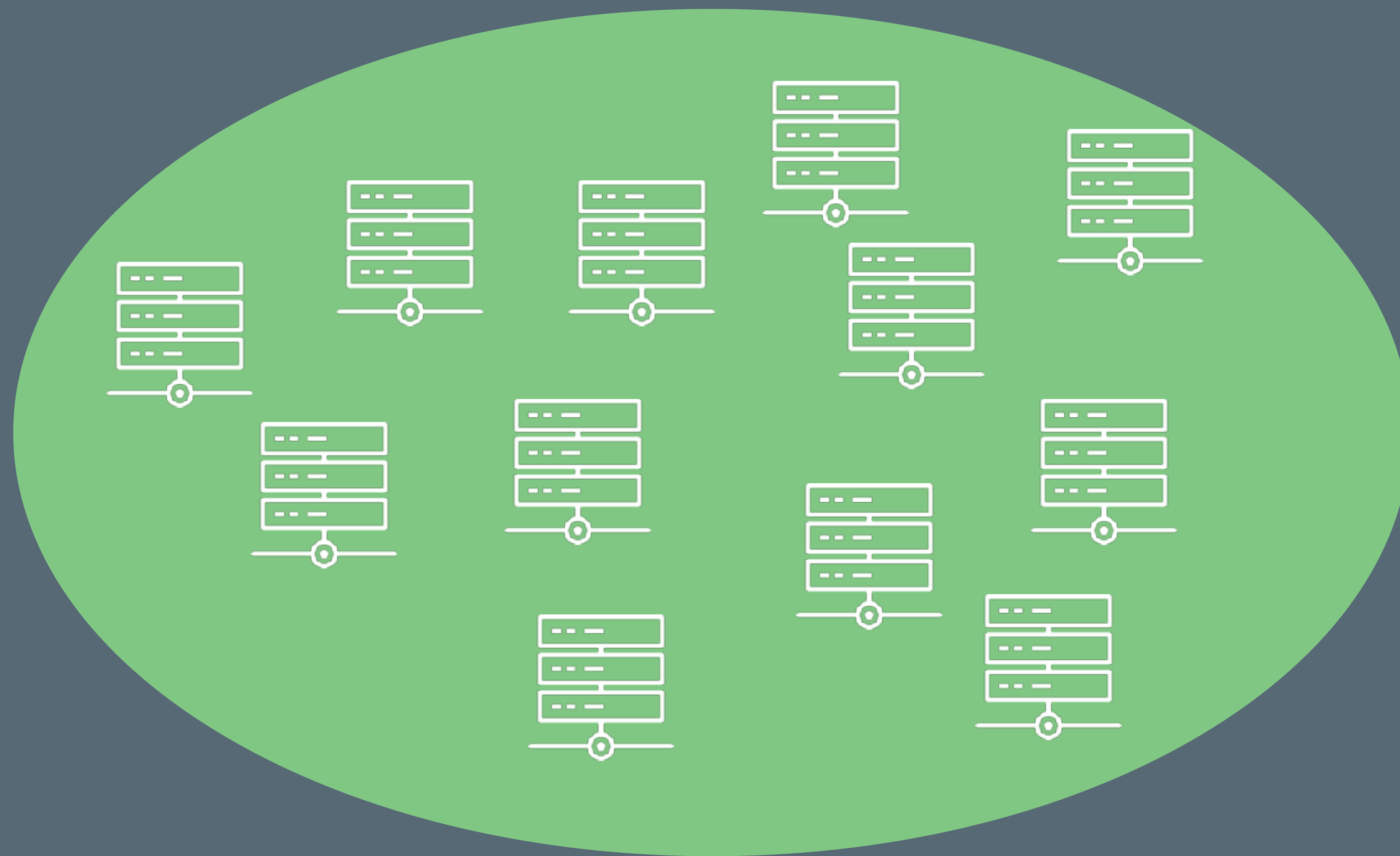
Don't assume threats  
stop at the gateway

# Defense in depth



Try smaller trust boundaries

# From..



# To...





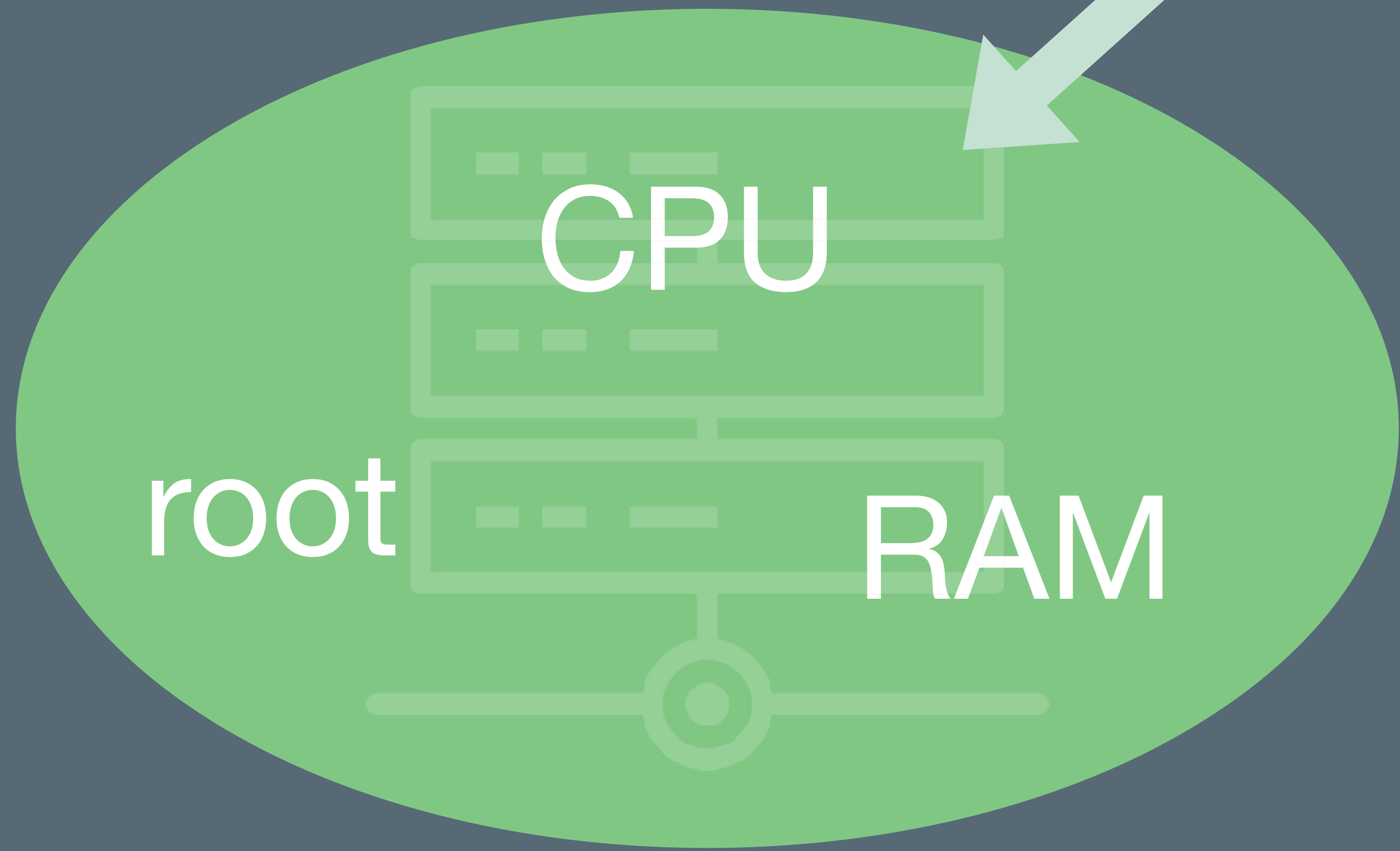
# To...



To...

Secrets go here

persistent storage



network?

users

# Traceability & Management



Where are our secrets?

The background of the slide is a dark blue color with a repeating pattern of light blue server rack icons. Each icon consists of three stacked rectangular units representing server racks, with a small circle at the bottom center of each stack. The icons are arranged in a grid-like pattern, slightly offset from each other.

Who can access them?

The background of the slide is a dark blue color with a repeating pattern of light blue server rack icons. Each icon consists of three stacked rectangular units representing server racks, with a small circle at the bottom center of the stack. The icons are scattered across the entire page, creating a subtle, textured effect.

When are they accessed?

The background of the slide is a dark blue color with a repeating pattern of light blue server rack icons. Each icon consists of three stacked rectangular units representing server racks, with a small circle at the bottom representing a power button or indicator. The icons are scattered across the entire slide, creating a textured, grid-like effect.

How do we rotate them?

We want those answers  
**BEFORE** we need them



# The right tool

SCM?

Configuration management?

Secrets management tool

# Lots of secrets management tools nowadays

Keywhiz (Square)

Confidant (Lyft)

KMS (Amazon)

Docker Secret

Encryption-as-a-Service

Multiple storage backends

Standalone microservice

Multiple types of secrets



**Vault**

HashiCorp

Audit friendly

Open Source

High availability

Authentication & Authorization

API & CLI

# Demo

```
$ vault write secret/srvA/credentials pwd=azerty  
Success! Data written to: secret/srvA/credentials
```

```
$ vault read secret/srvA/credentials
```

Key	Value
refresh_interval	768h0m0s
pwd	azerty

```
$ vault read /sys/policy/my-service-A  
path "secret/srvA/*" {  
  capabilities = ["read"]  
}
```

```
$ curl https://vault01/v1/secret/srvA/credentials  
-H 'X-Vault-Token:...'
```

```
{  
  "request_id": "f9e9b545-c624-ebbb-1dbd-  
d35e1074a478",  
  "lease_id": "",  
  "renewable": false,  
  "lease_duration": 2764800,  
  "data": {  
    "pwd": "azerty"  
  }  
  ...  
}
```



# Protecting secrets

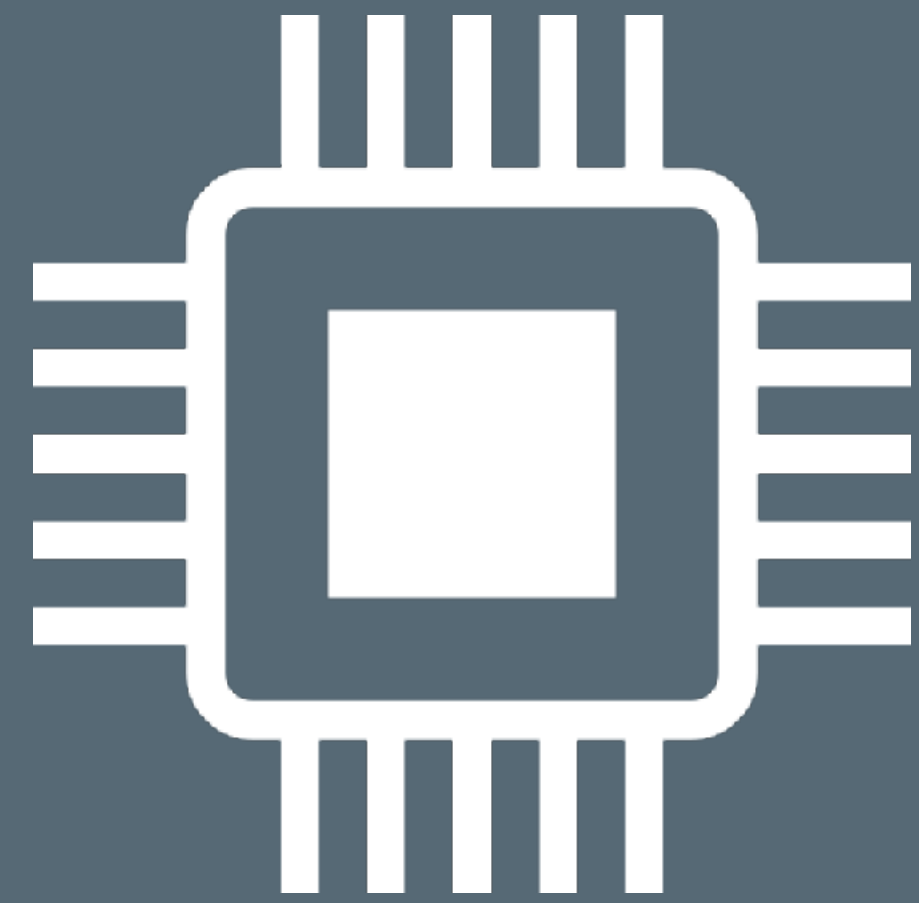
## Turtles all the way down

# How does Vault protect secrets?



Secrets are  
encrypted at rest

# How does Vault protect secrets?



Encryption keys only  
live in memory

# How does Vault protect secrets?



## How to obtain the Master Key?

# Shamir secret sharing

Master key split in **N** shards

**K** shards required

Vault sealed until a quorum is reached

# How do apps access secrets?

Ask Vault at runtime

Need an authentication token

How to pass the token to the app?

# Secure introduction

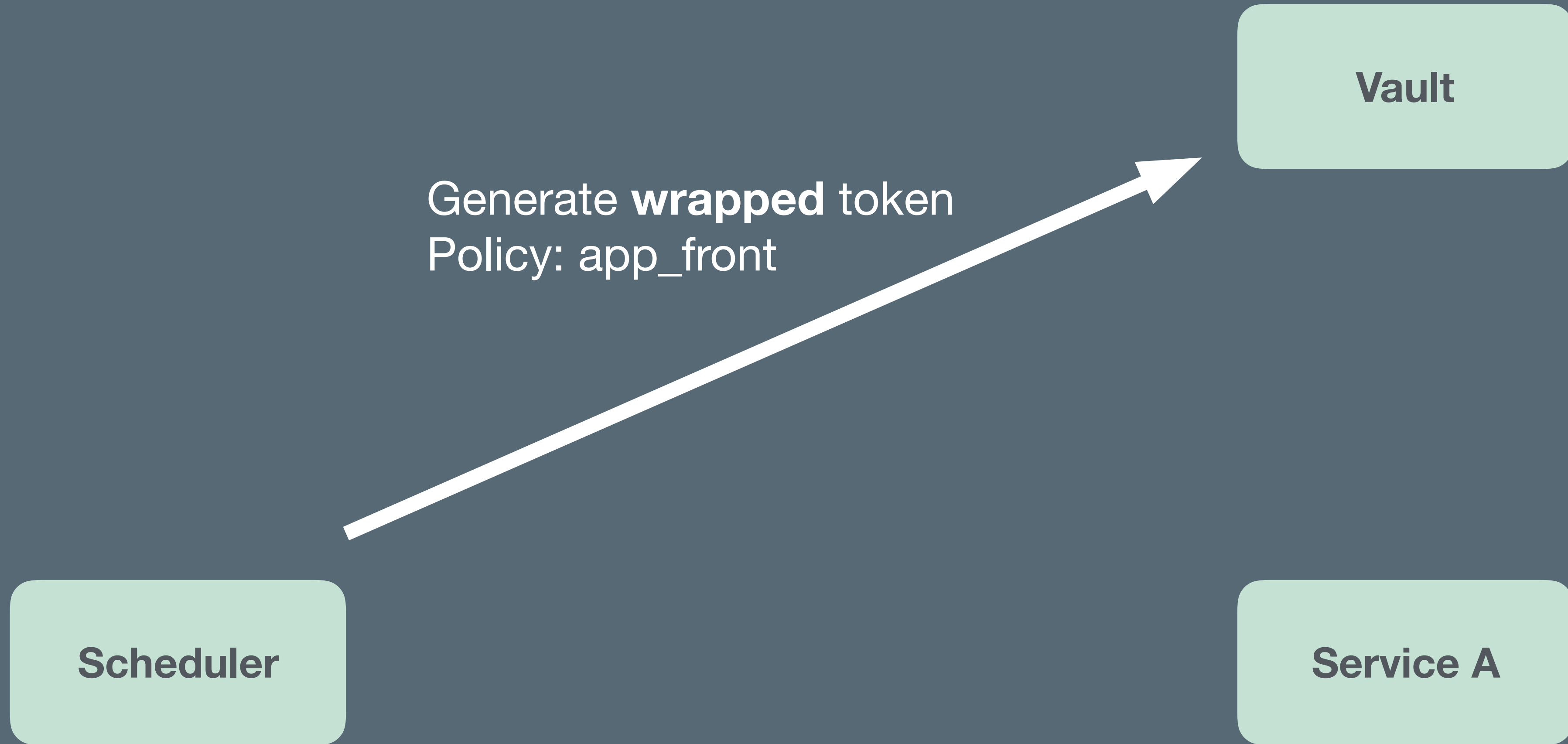
- Secure distribution
- Short token lifetime
- Access detection

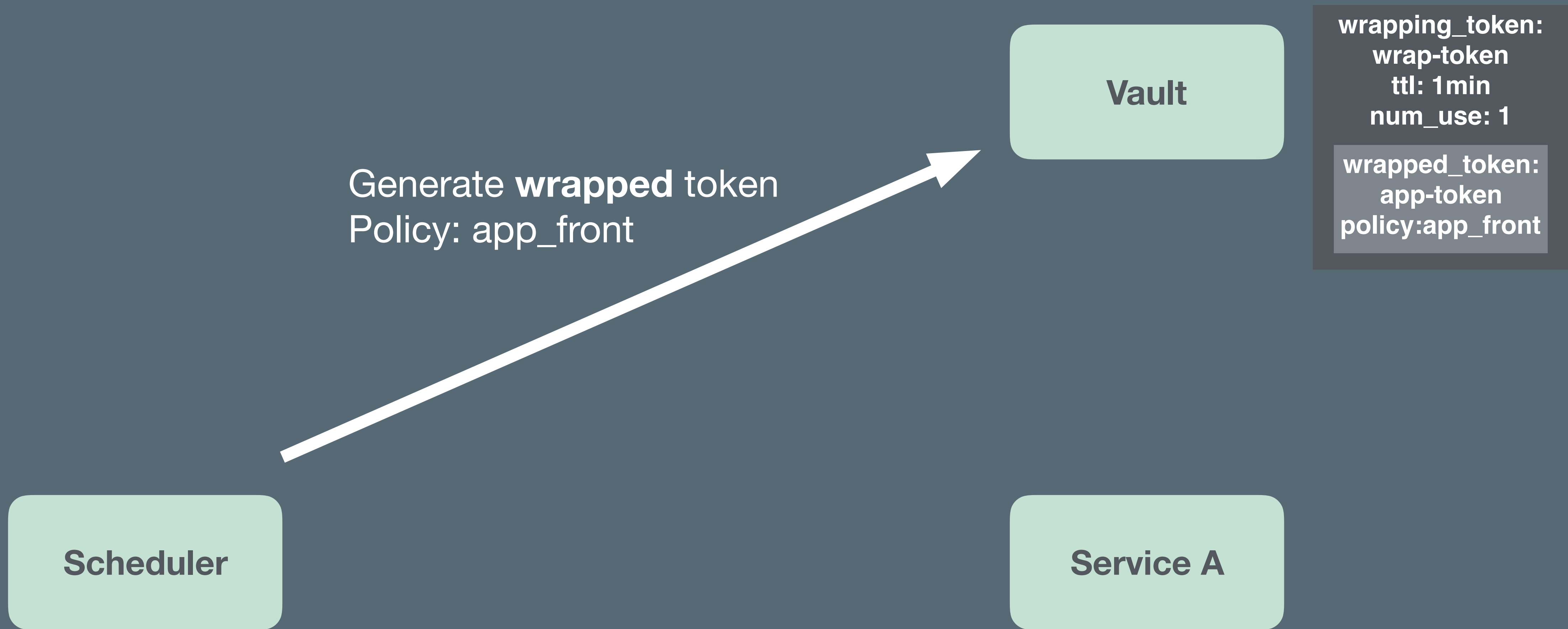
**Vault**

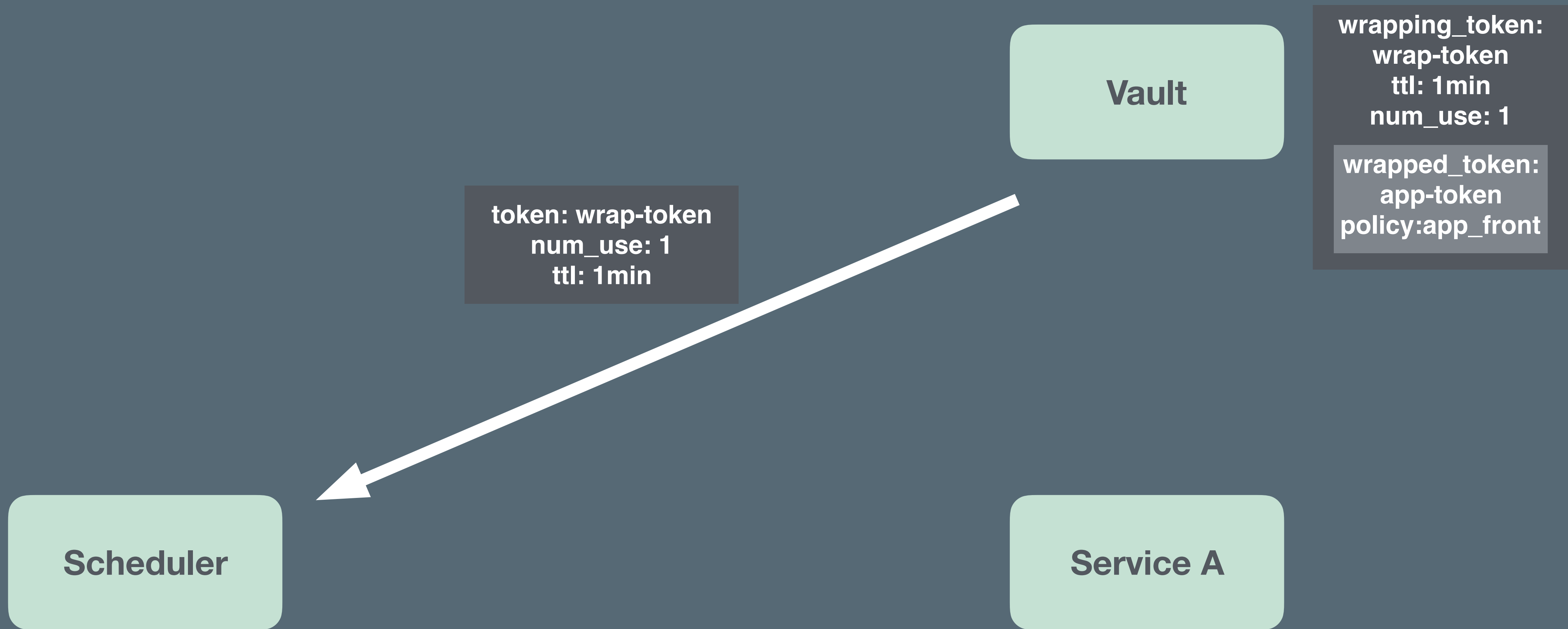
**Scheduler**

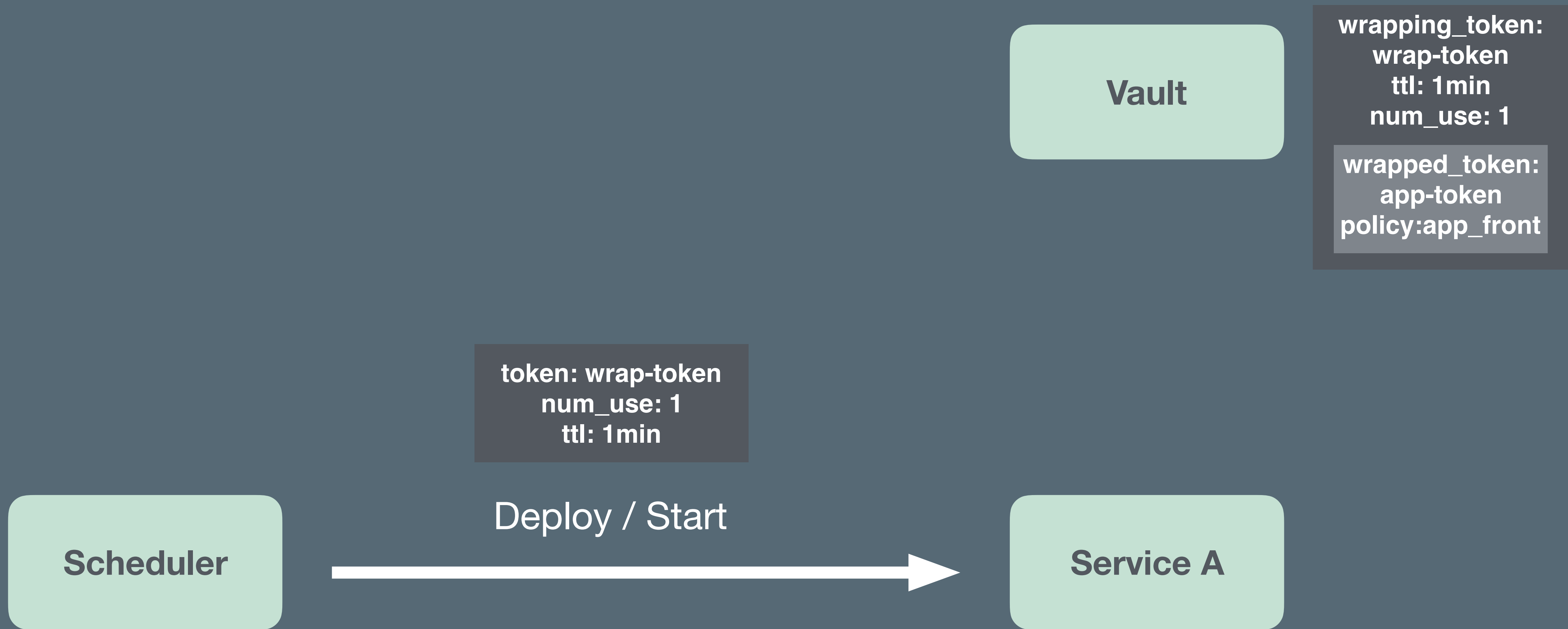
**Service A**



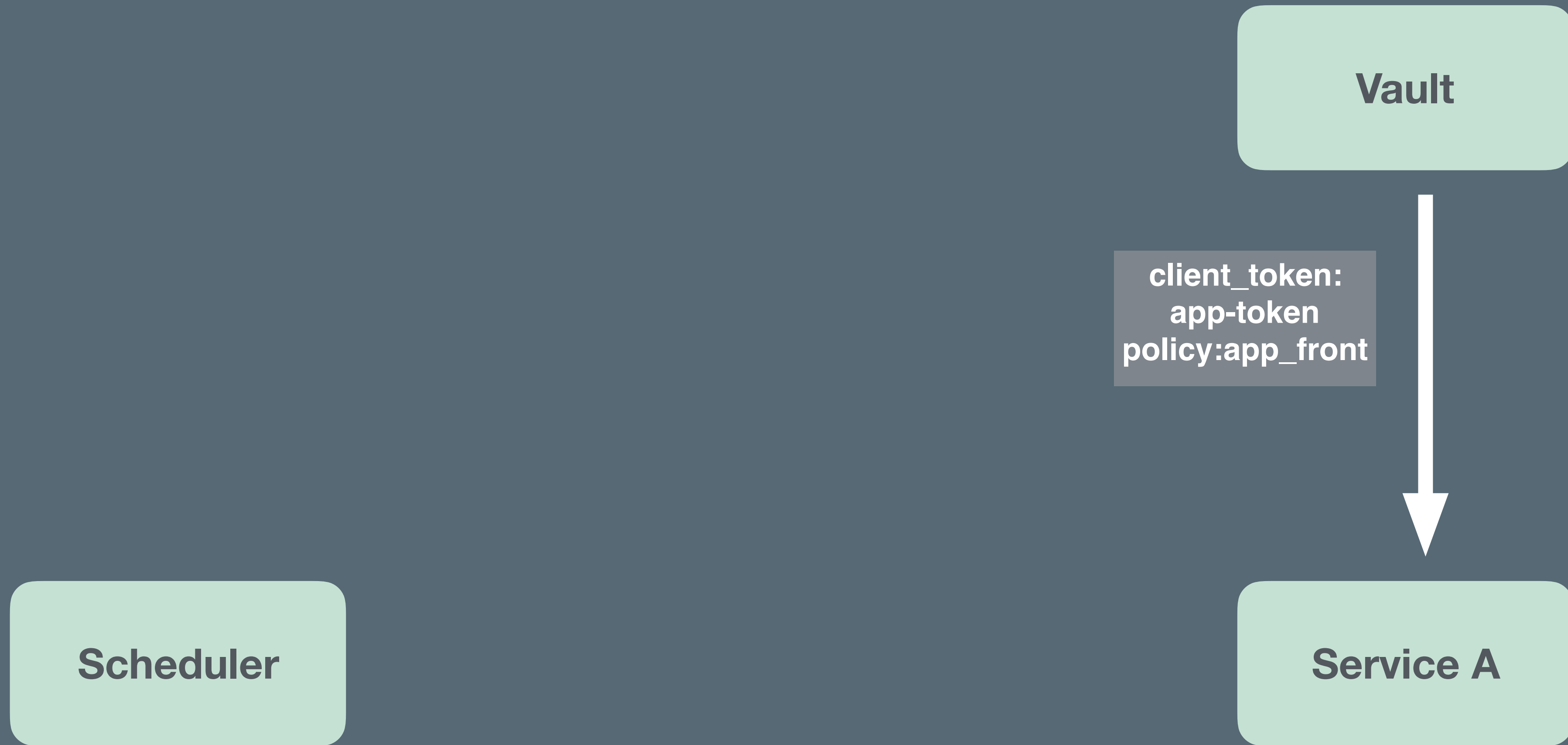


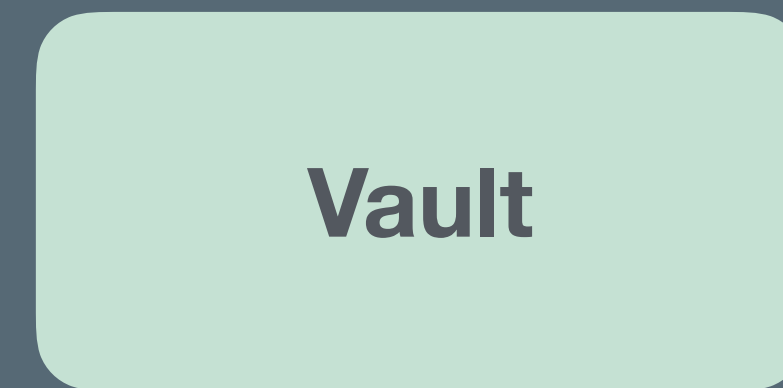
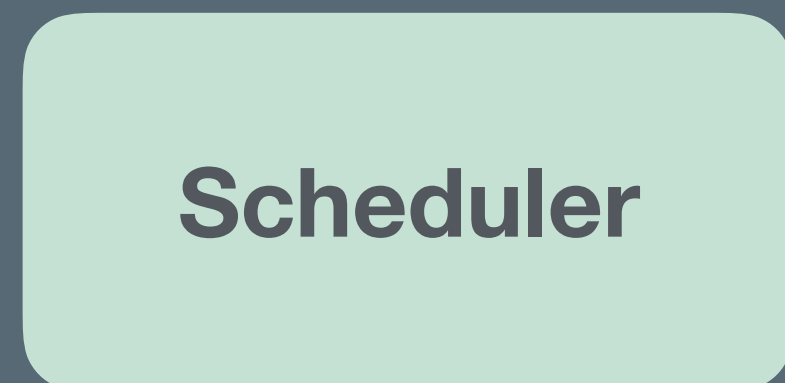












Read secret  
token: app-token

Text describing the action and the token used for the request.



# Secure introduction

- ❌ Secure distribution
- ❌ Short token lifetime
- ❌ Access detection



# Vault @ Oodrive

# Vault @ Oodrive

One service in staging

ZooKeeper as storage backend

"Good enough" introduction

# Challenges @ Oodrive



Ongoing transition to an  
orchestrated architecture

# Challenges @ Oodrive



For now: Puppet.  
No scheduler.

# Challenges @ Oodrive



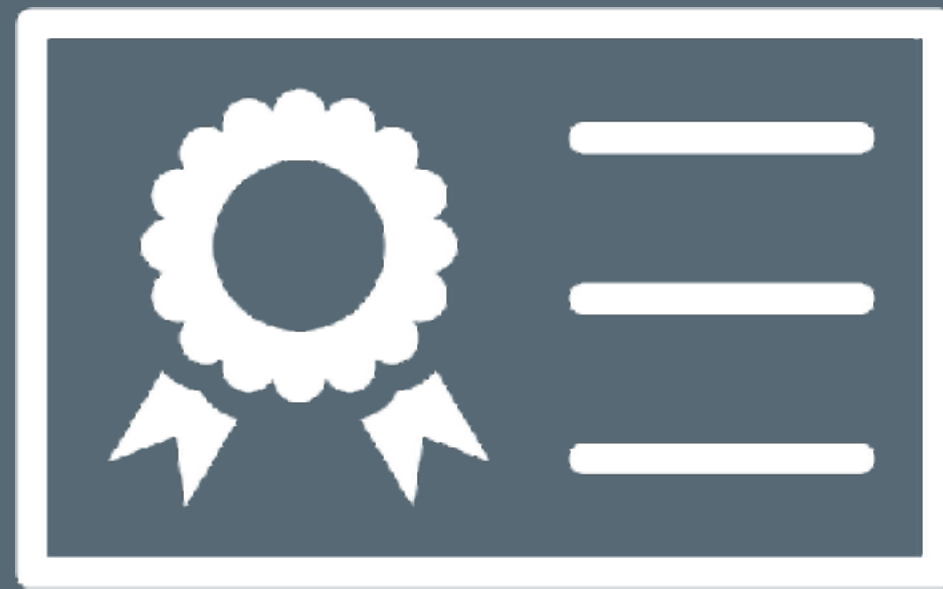
Don't want to break  
everything twice

# Challenges @ Oodrive



Puppet Server and init script as "scheduler"

# Challenges @ Oodrive



Leverage existing Puppet authentication

# Vanilla Java Client

```
VaultClient client = new Builder(vaultUri)
                        .withTokenAuth(token)
                        .build();

Credentials cred = client.readSecret(
    "secret/my-service/db-credentials",
    Credentials.class);
```



# Spring

```
@SecretPath("credentials")  
Secret<Credentials> credentialsSecret;  
  
...  
  
Credentials cred = credentialsSecret.getValue();
```

Hopefully to be  
open sourced

# Best practices



Least Privilege all  
the way!



Vault as single  
secrets repository



Secrets grouped  
under service name



Use roles for easier  
management



# Enable ACL on ZooKeeper



# Next steps

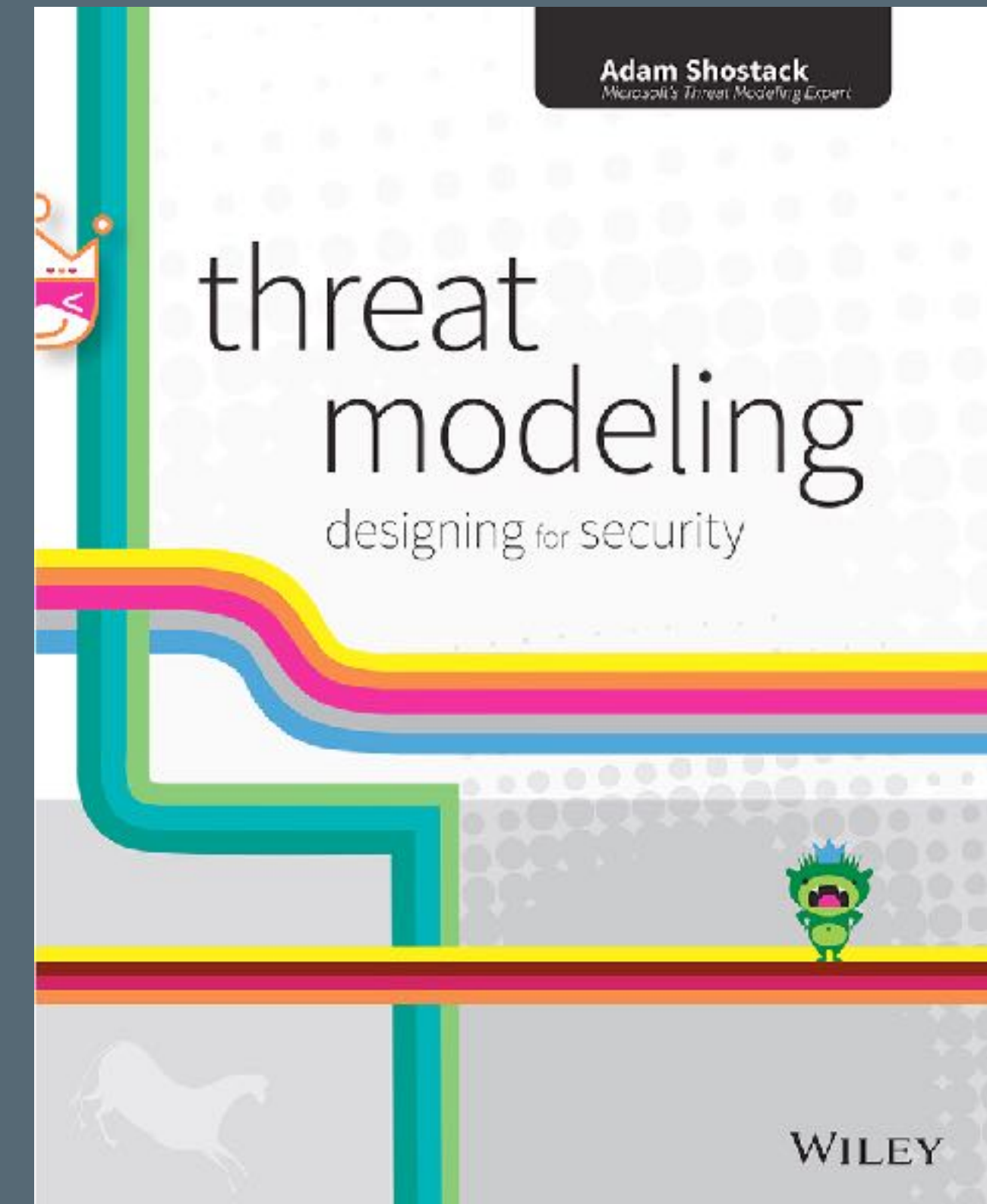
Fully secure introduction

Dynamic secrets (db credentials)

Vault as internal PKI

To win!

Thanks  
Questions?



OPEN R&DAY { } #1