Scaling Security in the Cloud With Open Source

Cloud Village @ DEF CON 27

James Strassburg
Technical Fellow / Chief Software Architect
Direct Supply

DIRECT
SUPPLY®

DIRECT
SUPPLY®
aptura®

DIRECT
SUPPLY®
DSSI™

DIRECT
SUPPLY®
Equipment & Furnishings

DIRECT
SUPPLY®
TELS®

aws

Our Cloud Vision and Strategy

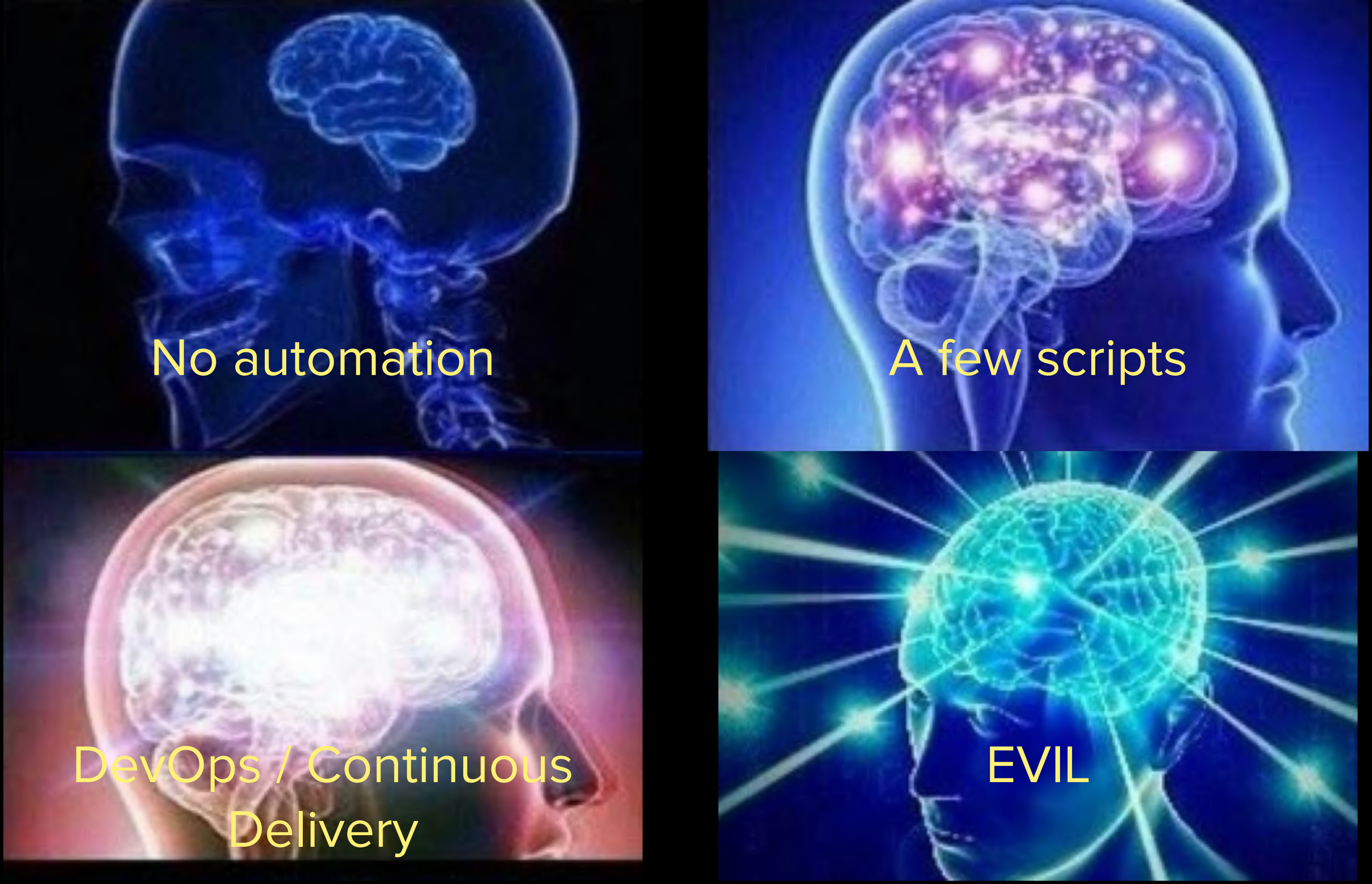
Immutable Servers

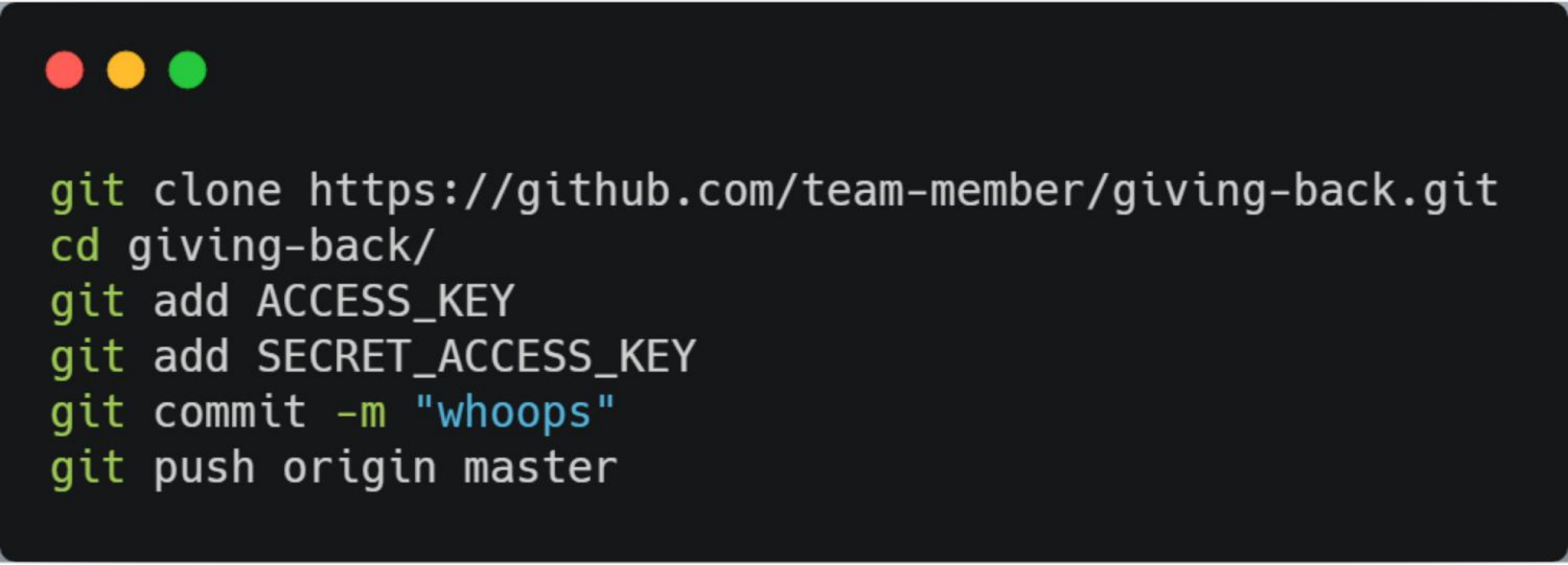
Infrastructure as Code

Automated Deployments

Secure by Default

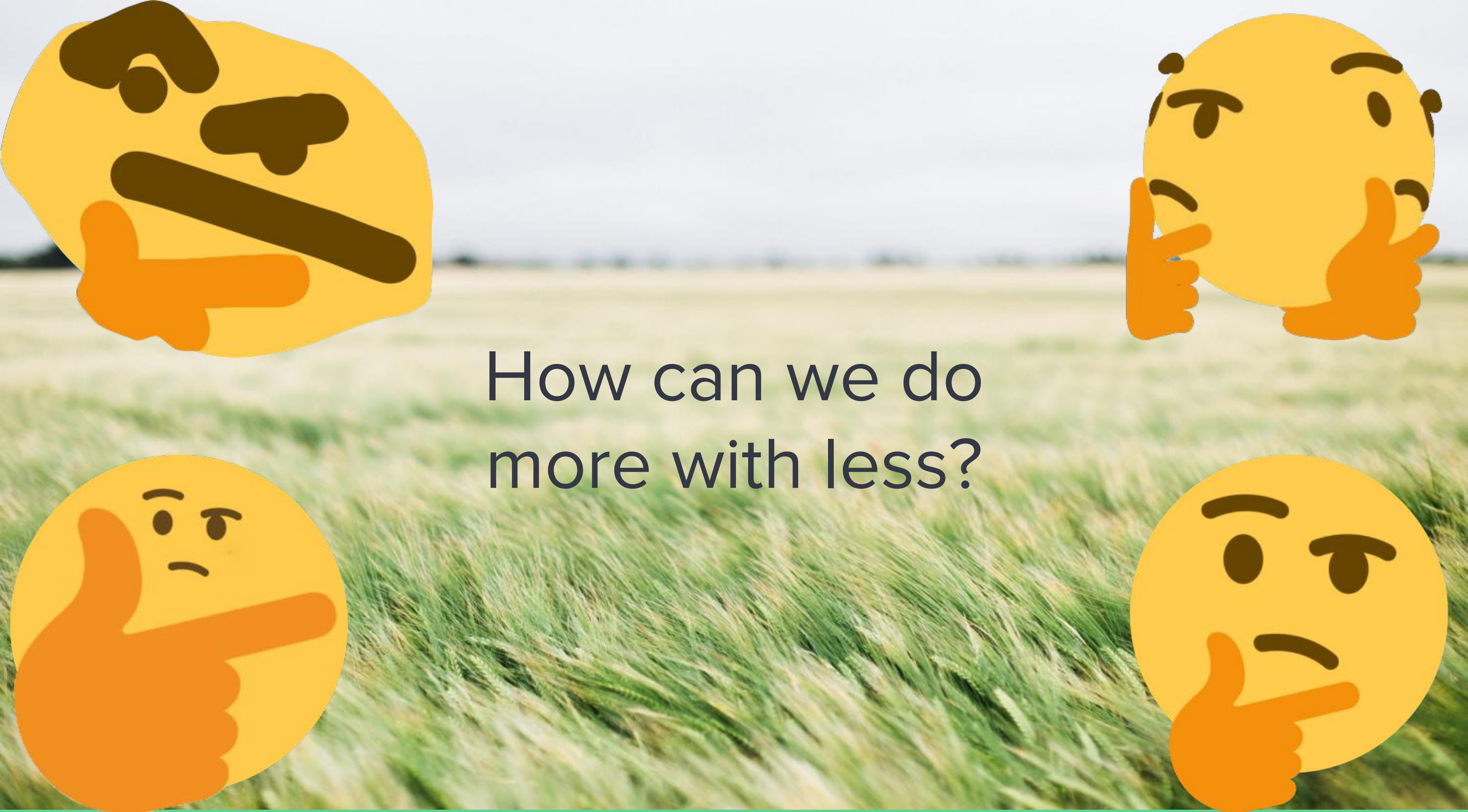
Platform as a Service


No automation
A few scripts
DevOps / Continuous Delivery
EVIL



```
git clone https://github.com/team-member/giving-back.git
cd giving-back/
git add ACCESS_KEY
git add SECRET_ACCESS_KEY
git commit -m "whoops"
git push origin master
```

What happened?



How can we do more with less?

HashiCorp

Vault

“Secure, store and tightly control access to tokens, passwords, certificates, encryption keys for protecting secrets and other sensitive data using a UI, CLI, or HTTP API.”

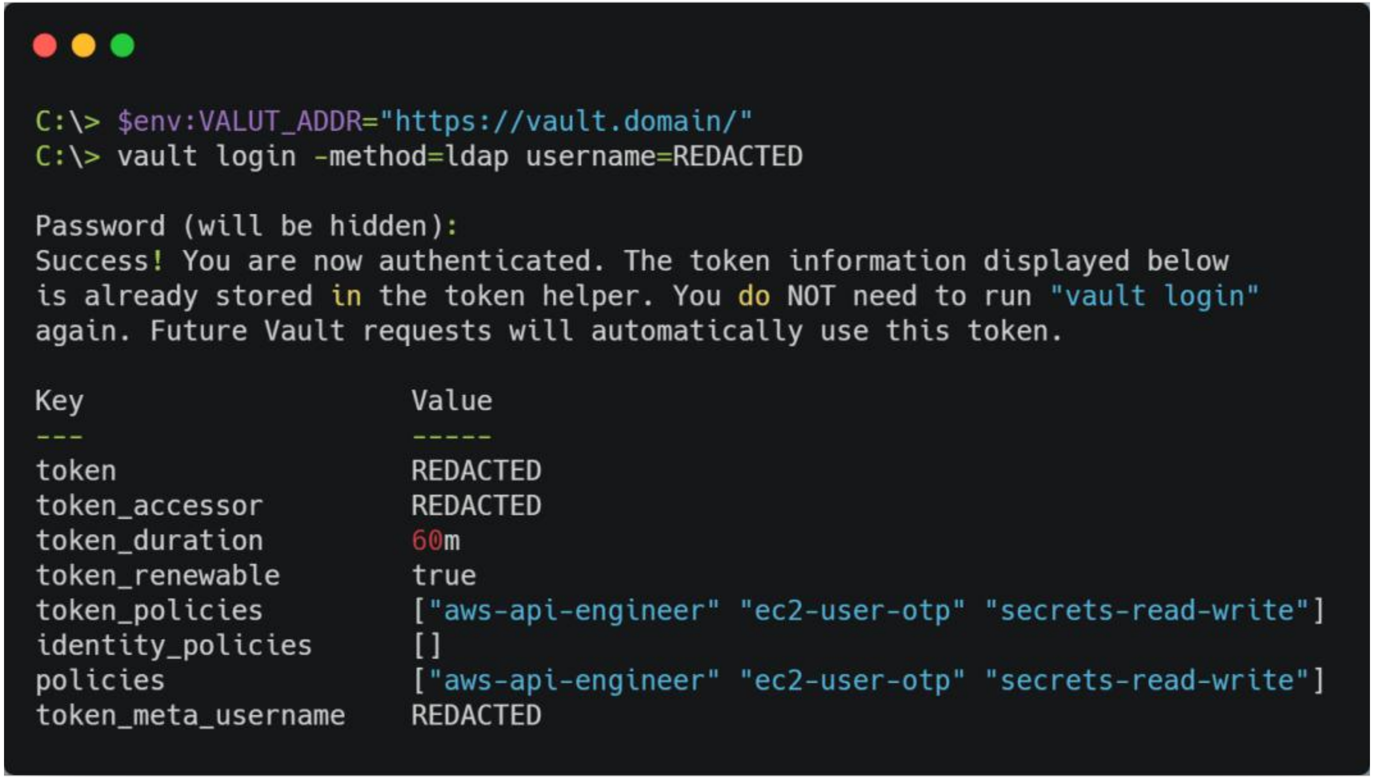
HashiCorp Vault

Our use cases...

- Static Secrets - Key/Value Secrets Engine
- Dynamic Secrets
  - Database Credentials - PostgreSQL Database Secrets Engine
  - AWS Access Keys - AWS Secrets Engine
- SSH One Time Password Secrets Engine

HashiCorp Vault

Authentication



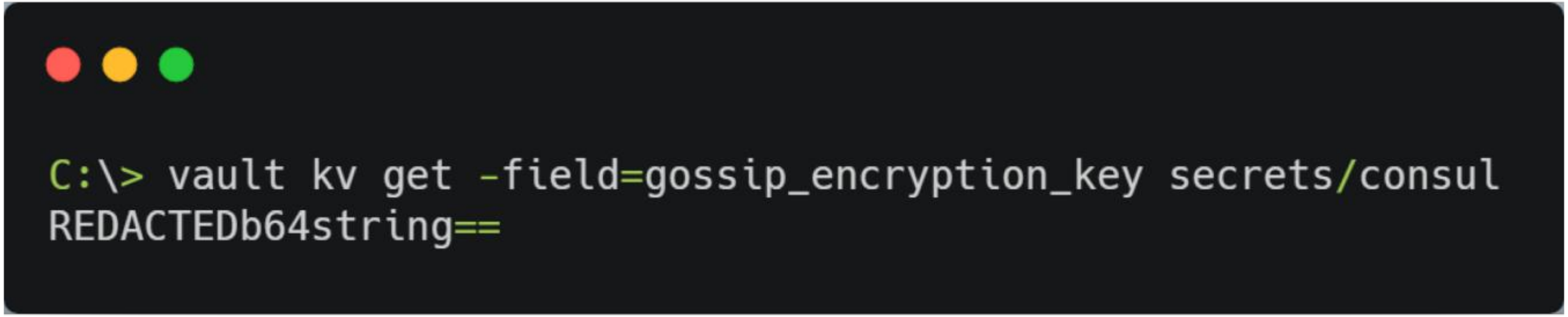
```
C:\> $env:VALUT_ADDR="https://vault.domain/"
C:\> vault login -method=ldap username=REDACTED

Password (will be hidden):
Success! You are now authenticated. The token information displayed below
is already stored in the token helper. You do NOT need to run "vault login"
again. Future Vault requests will automatically use this token.

Key                    Value
---                    -----
token                  REDACTED
token_accessor         REDACTED
token_duration         60m
token_renewable        true
token_policies         ["aws-api-engineer" "ec2-user-otp" "secrets-read-write"]
identity_policies      []
policies               ["aws-api-engineer" "ec2-user-otp" "secrets-read-write"]
token_meta_username    REDACTED
```

HashiCorp
Vault

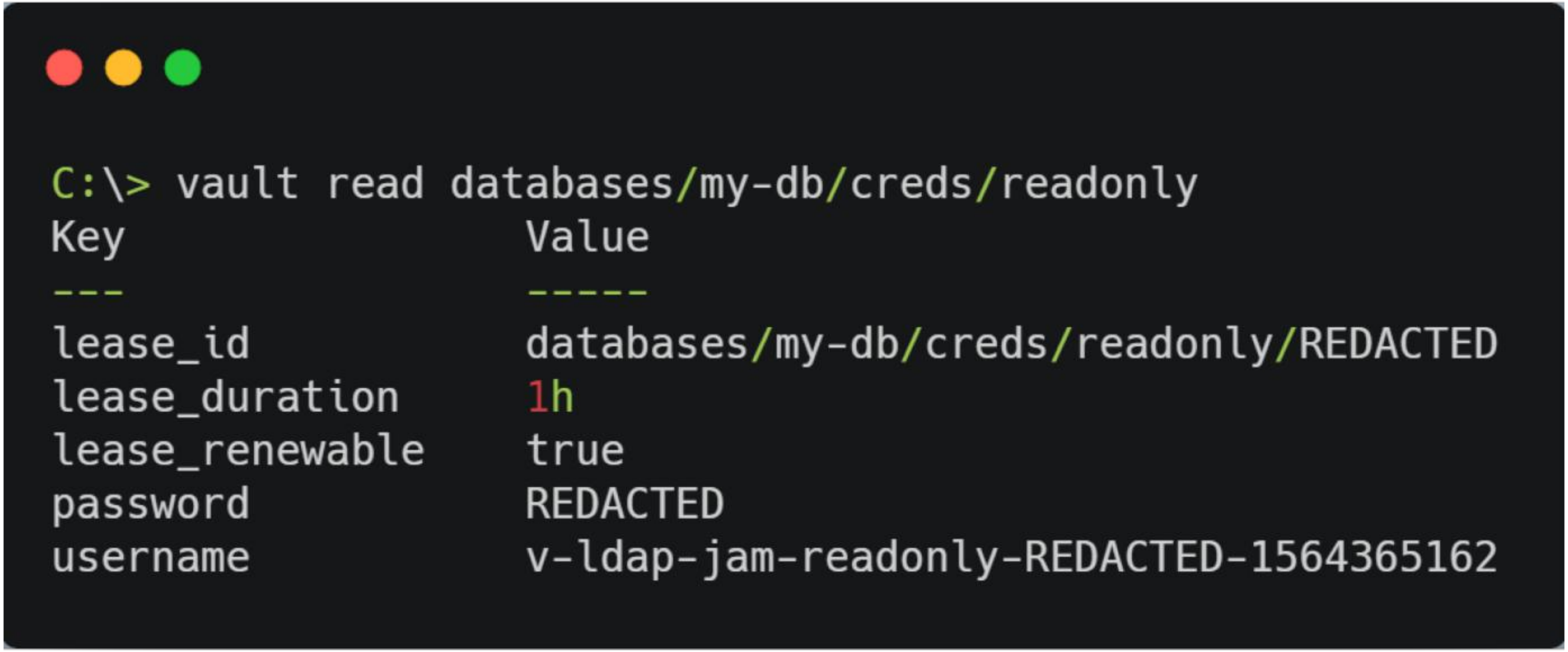
Key/Value Secrets Engine



```
C:\> vault kv get -field=gossip_encryption_key secrets/consul
REDACTEDb64string==
```

HashiCorp Vault

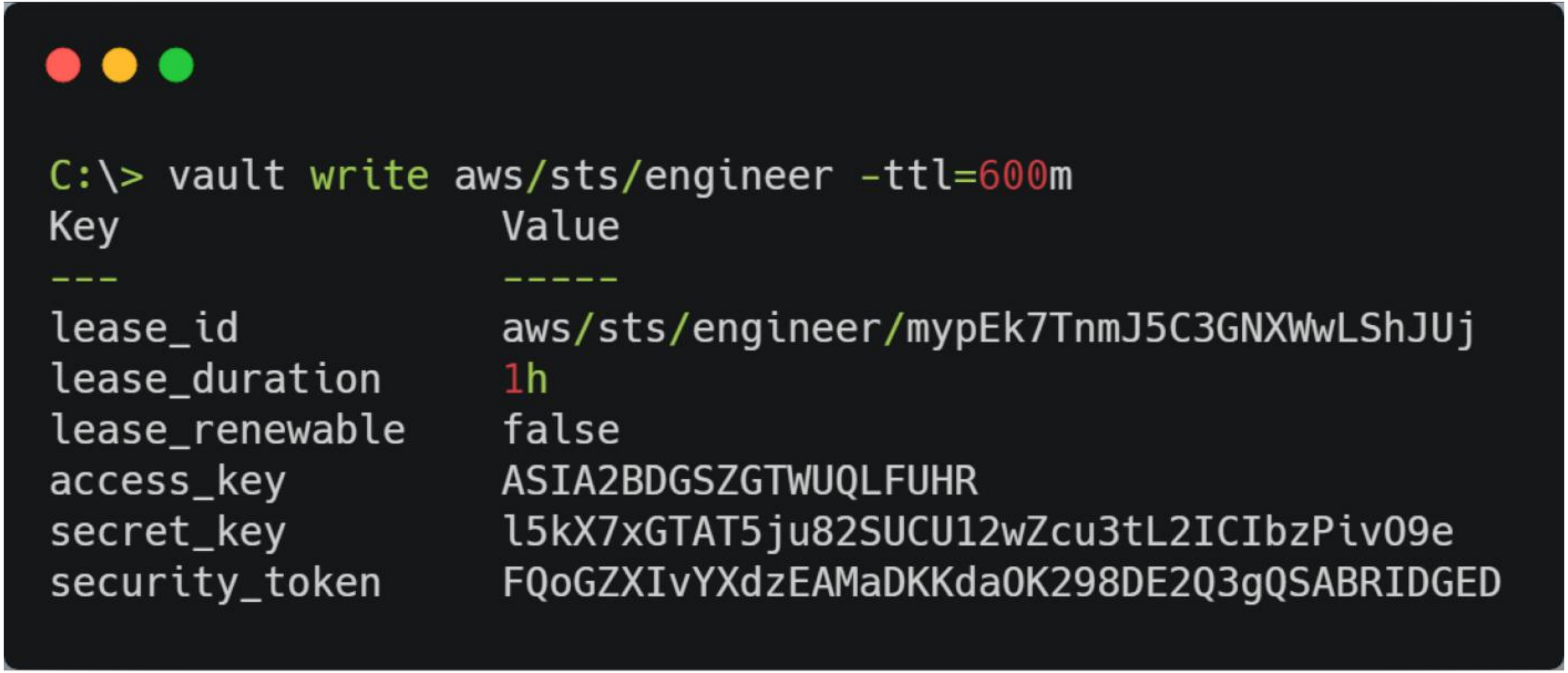
PostgreSQL Database Secrets Engine



```
C:\> vault read databases/my-db/creds/readonly
Key                  Value
---                  -----
lease_id             databases/my-db/creds/readonly/REDACTED
lease_duration       1h
lease_renewable      true
password             REDACTED
username             v-ldap-jam-readonly-REDACTED-1564365162
```

HashiCorp Vault

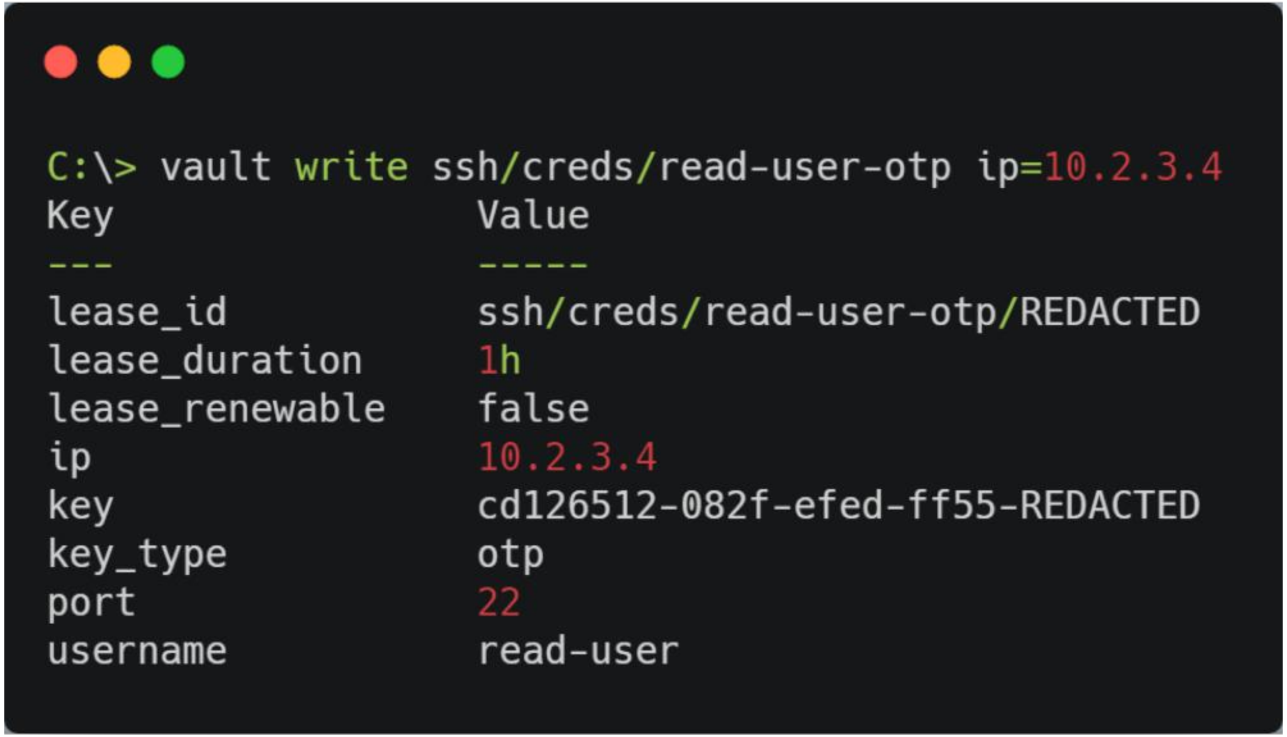
AWS Secrets Engine



```
C:\> vault write aws/sts/engineer -ttl=600m
Key                Value
---                -----
lease_id           aws/sts/engineer/mypEk7TnmJ5C3GNXWwLShJUj
lease_duration     1h
lease_renewable    false
access_key         ASIA2BDGSZGTWUQLFUHR
secret_key         l5kX7xGTAT5ju82SUCU12wZcu3tL2ICIbzPiv09e
security_token     FQoGZXIvYXdzEAMaDKKdaOK298DE2Q3gQSABRIDGED
```

HashiCorp Vault

SSH One Time Password Secrets Engine

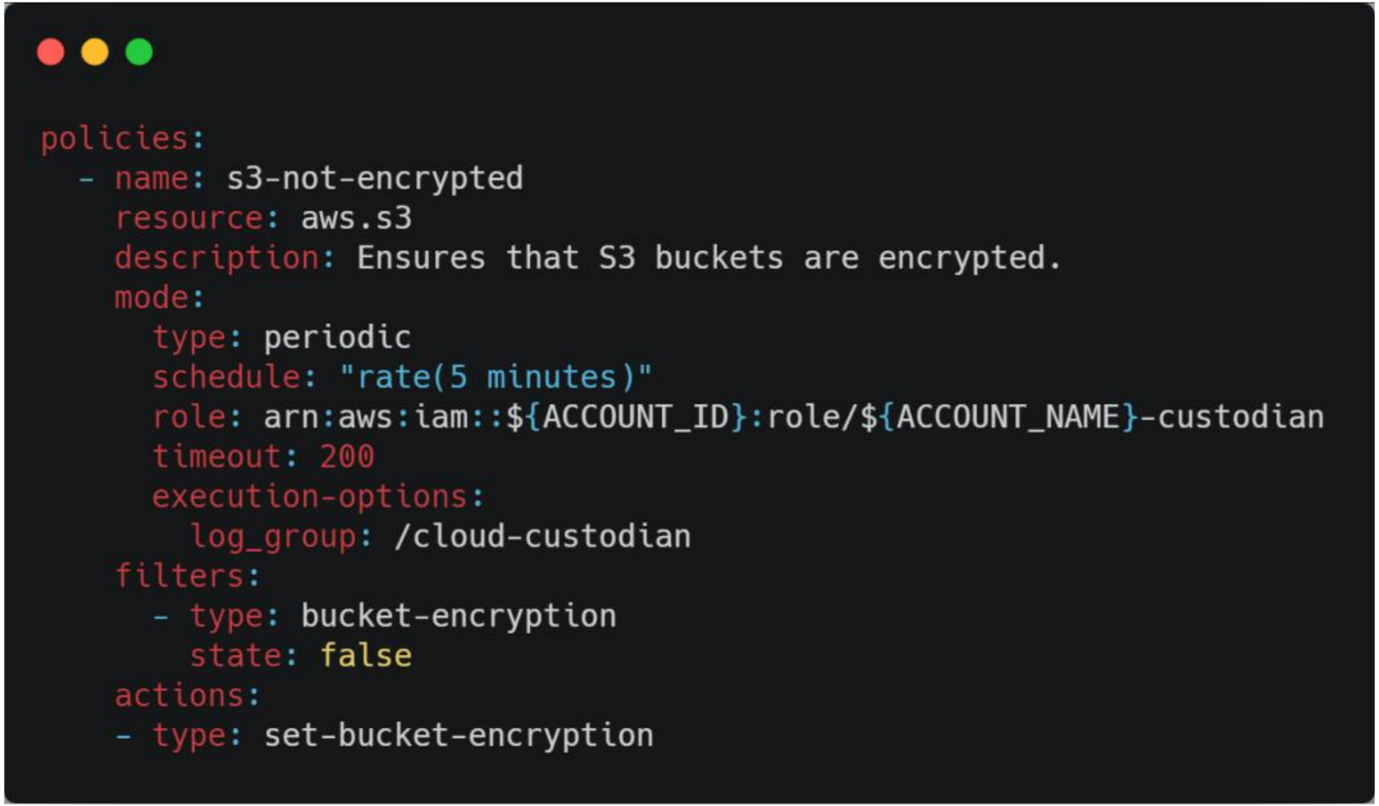


```
C:\> vault write ssh/creds/read-user-otp ip=10.2.3.4
Key                  Value
---                  -----
lease_id             ssh/creds/read-user-otp/REDACTED
lease_duration       1h
lease_renewable      false
ip                   10.2.3.4
key                  cd126512-082f-efed-ff55-REDACTED
key_type             otp
port                 22
username             read-user
```

Cloud Custodian

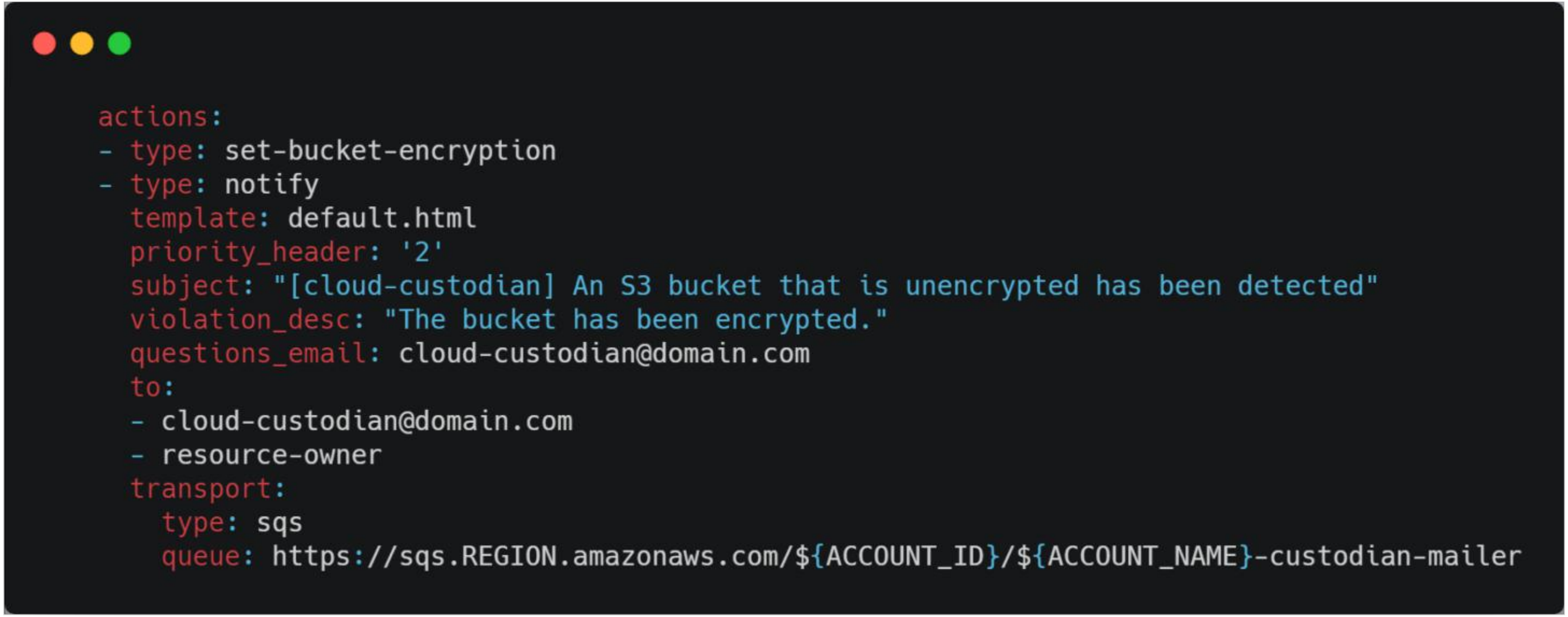
- Cloud configuration governance
- Automated checking and enforcement
- Policy as code
- Notifications

Cloud Custodian - Example Policy



```yaml
policies:
  - name: s3-not-encrypted
    resource: aws.s3
    description: Ensures that S3 buckets are encrypted.
    mode:
      type: periodic
      schedule: "rate(5 minutes)"
      role: arn:aws:iam::${ACCOUNT_ID}:role/${ACCOUNT_NAME}-custodian
      timeout: 200
      execution-options:
        log_group: /cloud-custodian
    filters:
      - type: bucket-encryption
        state: false
    actions:
    - type: set-bucket-encryption
```

Cloud Custodian - Notifications

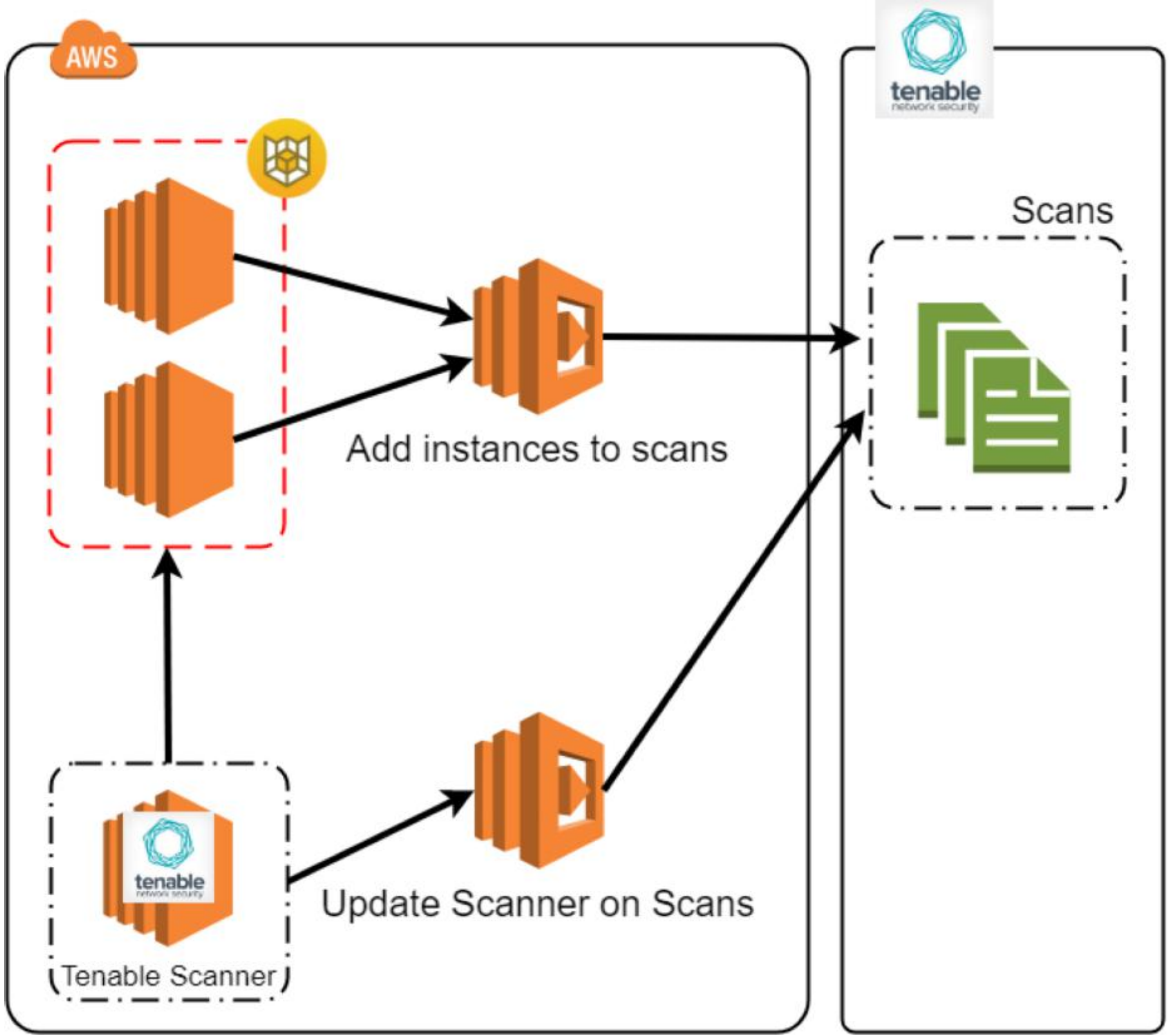


```yaml
actions:
- type: set-bucket-encryption
- type: notify
  template: default.html
  priority_header: '2'
  subject: "[cloud-custodian] An S3 bucket that is unencrypted has been detected"
  violation_desc: "The bucket has been encrypted."
  questions_email: cloud-custodian@domain.com
  to:
  - cloud-custodian@domain.com
  - resource-owner
  transport:
    type: sqs
    queue: https://sqs.REGION.amazonaws.com/${ACCOUNT_ID}/${ACCOUNT_NAME}-custodian-mailer
```

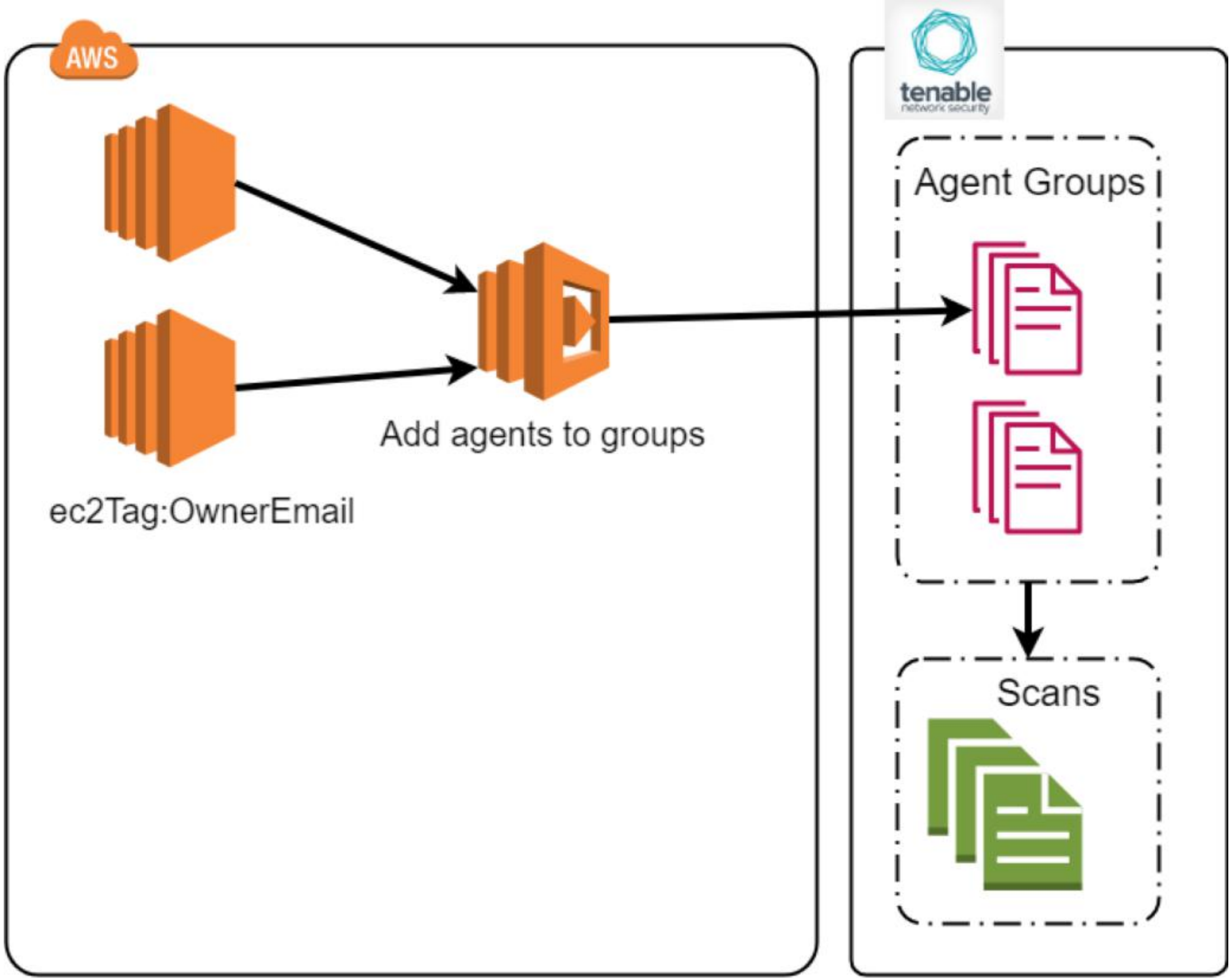
Security Scanning Automation / Dealing With Cattle

- Eliminate manual effort to update scan configurations
- Group instances logically by owner
- Support custom scans based on tags

Scanner Based Scanning



Agent Based Scanning

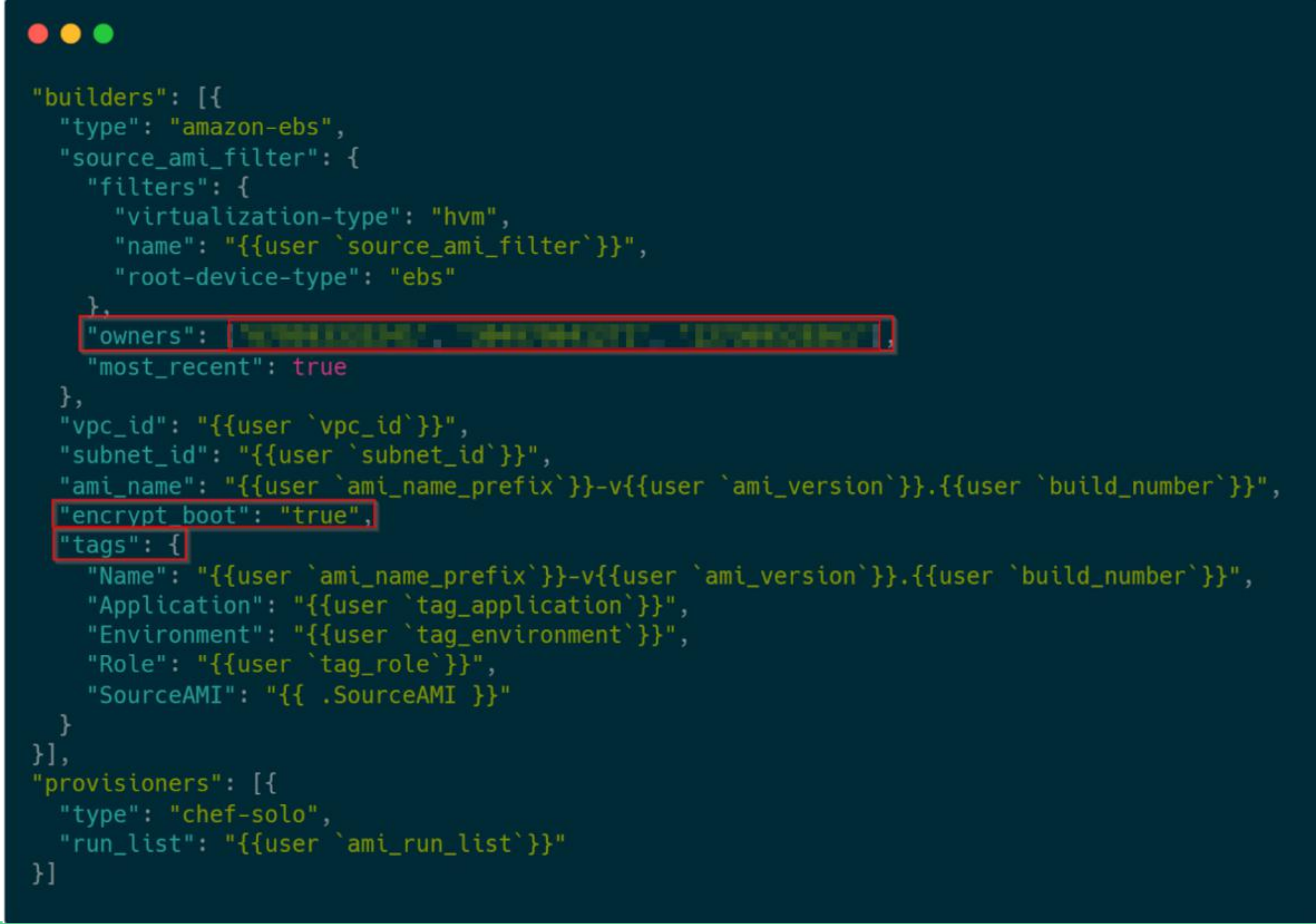


Secure by Default

HashiCorp
Packer

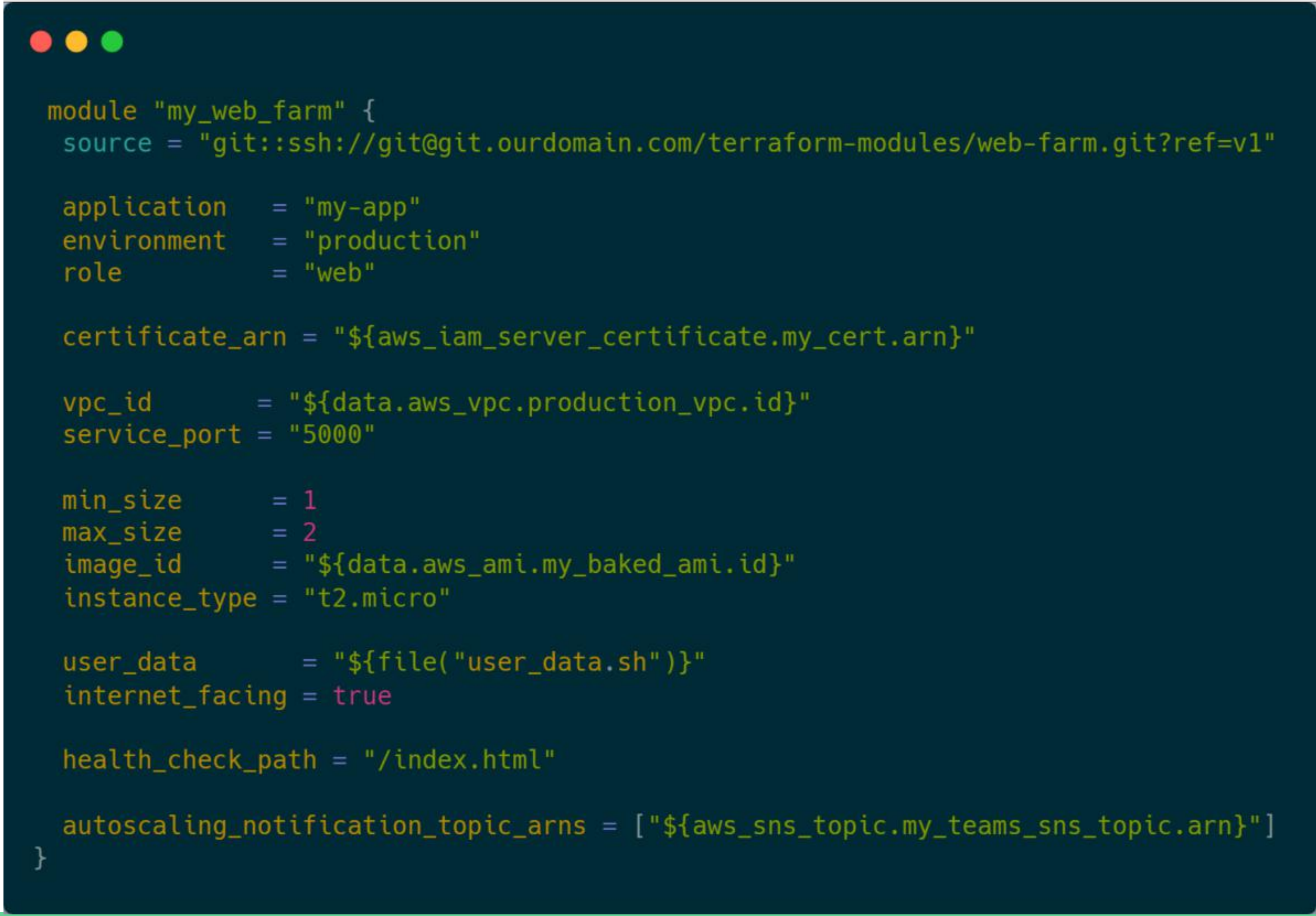
HashiCorp
Terraform

HashiCorp
Packer



```
"builders": [{
  "type": "amazon-ebs",
  "source_ami_filter": {
    "filters": {
      "virtualization-type": "hvm",
      "name": "{{user `source_ami_filter`}}",
      "root-device-type": "ebs"
    },
    "owners": [redacted],
    "most_recent": true
  },
  "vpc_id": "{{user `vpc_id`}}",
  "subnet_id": "{{user `subnet_id`}}",
  "ami_name": "{{user `ami_name_prefix`}}-v{{user `ami_version`}}.{{user `build_number`}}",
  "encrypt_boot": "true",
  "tags": {
    "Name": "{{user `ami_name_prefix`}}-v{{user `ami_version`}}.{{user `build_number`}}",
    "Application": "{{user `tag_application`}}",
    "Environment": "{{user `tag_environment`}}",
    "Role": "{{user `tag_role`}}",
    "SourceAMI": "{{ .SourceAMI }}"
  }
}],
"provisioners": [{
  "type": "chef-solo",
  "run_list": "{{user `ami_run_list`}}"
}]
```

HashiCorp
Terraform



```
 module "my_web_farm" {
  source = "git::ssh://git@git.ourdomain.com/terraform-modules/web-farm.git?ref=v1"

  application   = "my-app"
  environment   = "production"
  role          = "web"

  certificate_arn = "${aws_iam_server_certificate.my_cert.arn}"

  vpc_id       = "${data.aws_vpc.production_vpc.id}"
  service_port = "5000"

  min_size      = 1
  max_size      = 2
  image_id      = "${data.aws_ami.my_baked_ami.id}"
  instance_type = "t2.micro"

  user_data       = "${file("user_data.sh")}"
  internet_facing = true

  health_check_path = "/index.html"

  autoscaling_notification_topic_arns = ["${aws_sns_topic.my_teams_sns_topic.arn}"]
}
```

What went not so well...

- Lots of work before Custodian deployed
- Teams started rolling their own sans modules
- Our integration with a cloud consultancy is preventing us from taking advantage of some newer features

Resources

https://carbon.now.sh/	code formatting
https://unsplash.com/	images
https://www.vaultproject.io/	HashiCorp Vault
https://cloudcustodian.io/	Cloud Custodian

james.strassburg@directsupply.com
@jstrassburg