DevOps Secrets Management
apiVersion: scale/v18
kind: Bio
metadata:
  name: murriel
  labels:
    job: devops
    job: cloud
spec:
  containers:
    - name: orion
      image: russianblue
      command: ["cat"]
  replicas: 3
  hobbies:
    - name: making
    - name: gardening
    - name: community
tell me....

can your systems keep a secret?

share a secret?
what are secrets?
personal and team secrets

- Passwords / Passphrases
- Cloud Provider Logins
- Service Provider (SaaS) Logins
- SSH Keys
- Certificates
- Kubeconfigs
- DB Credentials
- App Dashboards and Logins

not focusing on email passwords, computer logins, etc managed by IT
system* secrets

- API Keys
- Certificates
- DB Credentials
- Encryption Keys
- Tokens
- SSH Keys
- System-to-System Authentication Secrets

Systems like...

- Servers
- Microservices
- Serverless functions
- Web application
- Mobile App
- On Premise App
- IoT Device Firmware
- Other machines
why is this important?
cost of breaches

https://www.informationisbeautiful.net/visualizations/worlds-biggest-data-breaches-hacks/
vectors for compromise

- Credentials in Git
- Inadvertently published secrets
  - Artifacts
  - Machine or Container Images
  - Compiled binaries
- Exposed S3 buckets
- Ex-Employees
- Internal unauthorized access
- Unauthorized password use
  - Weak passwords cracked
  - Shared/reused passwords
- Social engineering
- Network sniffing
  - Tokens/Creds sent unencrypted
- Reverse engineering
let’s dream
I committed our root creds to GitHub

I thought our systems were secure
• Decentralized infrastructure
• Decentralized ownership
• Hybrid Environments
• Inconsistent process and tools
• Path of least resistance
• Legacy code and systems
• Deadlines
• Secret Sprawl

https://www.hashicorp.com/resources/what-is-secret-sprawl-why-is-it-harmful
anti-patterns
hard-coded secrets

changing history is hard

• Secrets in git
  ○ Application Code
  ○ Config files
  ○ Terraform
  ○ Config Management Files
  ○ Kubernetes configs

• Secrets in images
  ○ Machine images
  ○ Docker/container images

• Plaintext secrets
shared/re-used credentials

- Shared SSH keys
  - “Can someone Slack me the devops-prod.pem key?”
- Using master keys or root users
  - Cloud accounts
  - db users
- Creds re-used across environments
  - Dev and Prod share API keys
- Shared kubeconfigs or client certs
- “Default” passwords
- No auth (mongo, elastic, redis...)
insecure sharing methods

- Messaging passwords over chat
- Emailing passwords
- Passwords written publicly
- Passworded Excel sheets
- Passwords in command history
- Passwords saved where they shouldn’t be
ssh key problems

- Shared keys
- Cloud provider generated keys
- Keys embedded in images
- Jump Box that has ALL the keys
- Key Management
  - Managing personal keys
  - Managing system keys
  - Personal keys as machine keys
  - Global access to machine keys
  - No rotation when employees leave
how can* my systems and i keep secrets?

can and not should
process

plus tools
multi

two-factor authentication
we have questions.

where are your secrets?
how to identify secrets?
where are these secrets stored?
when do we need to access them?
how do we update our secrets?
how to revoke secrets?
how is accessing these secrets?
systems for secrets
considerations

- Laws and regulations
- Internal policies and process
- Technical limitations
- Access requirements
- Resources
- Budget
useful considerations

- Human and Infrastructure Resources
  - Team bandwidth and expertise
  - Maintenance overhead
  - Existing tools
  - Buy, build, pay someone to build, or some combination

- (High) Availability and Disaster Recovery
  - Airgapped or segmented networks
  - VPN needed
  - Inter-dependent systems and points of failure
  - Centralization vs distributed
  - Footprint
useful considerations

● AAA: Authentication, Authorization and Accounting
  ○ Role-Based Access Control (RBAC)
  ○ Access management
  ○ Auditing
  ○ Principle of least privilege

● Updates and Versioning

● Open Source Tools: Maintenance and Updates

● Backups

● Encryption: in rest and in transit

● Ease of use and ease of deployment

● Automation
managing secrets

- Password Managers
- Secrets Storage
  - Jenkins Credentials, Rundeck Secrets, Kubernetes Secrets, Docker Secrets
  - Config Management Secrets
- Secrets Management Tools
- Cloud Provider Secrets
password managers

- KeePass (open source)
  - KeePass, KeePassXC, KeePassX
- LastPass
- 1Password
- Dashlane
- Keeper
- Password Safe
- Roboform
secret storage
• Secrets API object
• Separate secrets from configmaps, pod definitions, image
• Base 64 **encoded** not encrypted
• Encrypted in etcd
• Accessible with cluster access*
• Role-based control
• [https://github.com/bitnami-labs/sealed-secrets](https://github.com/bitnami-labs/sealed-secrets)
• Helm Secrets
• Kamus

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: mysecret
type: Opaque
stringData:
  config.yaml: |
    apiUrl: "https://my.api.com/api/v1"
    username: {{username}}
    password: {{password}}
```

[https://kubernetes.io/docs/concepts/configuration/secret/#risks](https://kubernetes.io/docs/concepts/configuration/secret/#risks)
pipeline

Jenkins Secrets
- Credentials plugin
- https://github.com/jenkinsci/hashicorp-vault-pipeline-plugin

Rundeck Secrets

Travis CI Secrets
encrypted git

encrypt secrets before committing

- git-crypt
- git-secret
- keybase
- StackExchange BlackBox
- LockGit
github secrets

Secrets

Secrets are environment variables that are **encrypted** and only exposed to selected actions. Anyone with **collaborator** access to this repository can use these secrets in a workflow.

Secrets are not passed to workflows that are triggered by a pull request from a fork. [Learn more.](https://github.com/apps/secret-audit)

**Add a new secret**

Name

YOUR_SECRET_NAME

Value

**ALSO:** [https://github.com/apps/secret-audit](https://github.com/apps/secret-audit)
config management

Ansible Vault
Saltstack Pillars
Chef Vault
Puppet - hiera-yaml and hiera-gpg
secret management
stackshare

**Vault**
- 12 Secure
- 8 Very easy to set up and use
- 8 Dynamic secret generation

**AWS Secrets Manager**
- 0 Managed Service

**Docker Secrets**
- 2 Secure
- 2 Multi-Host aware

**Keywhiz**
- 1 Fuse FS

**Torus CLI**
- Secrets Management
- Stacks 8
- Learn more
generic secrets management pipeline

- Substitute secret with a parameter
- Secret is injected
  - at build time
  - at deploy time
  - dynamically at run-time
- Encryption at rest and in transit
  - Depending on when they are injected they may live unencrypted somewhere
- Determine threat model and access requirements
- Varies depending on type of application
- Sufficient encryption/obfuscation required for secrets embedded in shipped software
open source

- Hashicorp Vault
  - Open Source
  - Enterprise
- SecretsHub
- Mozilla SOPS
- Torus
- CyberArk Conjur
- Square Keywhiz
- Lyft Confidant
- Pinterest Knox
Closed Source / SaaS

- BeyondTrust
  - Password Safe
  - Cloud Vault
  - DevOps Secrets Safe
- Thycotic
  - Secret Server
  - DevOps Secrets Vault
- CryptoMove Key Vault
hashicorp vault

- robust
- rotation and expiration
- integrations
- community support
- documentation
- professional services

- management complexity
- production requires consul and clustering and approles and policies and integrations and configuration and...
Notes on Vault Implementations

● AppRoles and Policies + Rotation and Expiration considerations

● Kubernetes Vault operators
  ○ https://github.com/coreos/vault-operator
  ○ https://github.com/banzaicloud/bank-vaults

● Jenkins plugins

● API-driven

● Many integrations with config management or other systems
  ● https://github.com/bruj0/vault_jenkins
  ● https://www.hashicorp.com/resources/how-to-share-secrets-pipeline
cloud providers
amazon web services

- Secrets Manager
- Key Management System (KMS)
- Amazon Certificate Manager
- Security Token Service (STS)
Google Cloud

- Secret Manager
  - New! (Beta Release)
- Cloud Key Management System
- Berglas
microsoft azure

Azure Key Vault

- Secrets Management
- Certificate Management
- Key Management
Let's Encrypt

Automatically enable HTTPS on your website.

https://letsencrypt.org/
https://certbot.eff.org/
addressing anti-patterns
fixing* hard-coded secrets

Fixing Committed Secrets
- https://securitytrails.com/blog/github-dorks
- https://github.com/awslabs/git-secrets
- https://github.com/dxa4481/truffleHog
- https://github.com/awslabs/git-secrets

Clean Up Repos
- git-filter-branch
- BFG Repo Cleaner
  https://rtyley.github.io/bfg-repo-cleaner/

Rotate Published Creds
Testing
Refactor
fixing shared credentials

Implement RBAC where possible

Databases:
- tiered creds
- root, read only, read-write

Evaluate scope
- some secrets should be “shared team” secrets
- user license limitations

Refactor
fixing insecure communications

- define securer* communications channels
- enforce first-time reset if possible
- use your password manager
- encrypt and send
- secure shared location
ssh key solutions

- methods for rotation
- store / manage public keys
- config management
- audit images
- audit auto-generated keys
- use a secret manager/vault
- remove keys when not needed
  - employee leaves
  - access no longer needed
- limit shared keys, or access to shared keys
personal and team strategies

● audit local workstation or shared systems/filesystems

● limit password re-use or password sharing

● use a password manager

● use temporary / time-limited tokens*

● avoid root users/accounts

● USE MULTI-FACTOR
personal and team policies

- awareness of secrets (visibility)
- secure sharing methods
- limit access
- password policies
  - expiration (does not necessarily work)
  - complexity
- onboarding and offboarding
  - Remove access and secrets when people leave
- password storage policies
- secret storage
closing thoughts

- knowable process
- team buy-in and education
- reduce barriers to usage
- auditing, rotation, encryption
- manage access to master keys
- plaintext is bad
- parameters and dynamic secrets
- roles and least privilege
- iterate, monitor, review
designing secure systems

- NIST 800-63B - Digital Identity Guidelines
  

- Google Cloud - User Account Management
  
  https://cloud.google.com/blog/products/gcp/12-best-practices-for-user-account

- Kubernetes Security/Secrets
  
  https://github.com/kubernetes/community/blob/master/contributors/design-proposals/auth/secrets.md

- 12-factor patterns
  
  https://12factor.net/
resources & references

Secrets Management Articles

- https://techbeacon.com/security/top-resources-cloud-native-secrets-management
- https://www.beyondtrust.com/resources/glossary/secrets-management
- https://techbeacon.com/security/top-resources-cloud-native-secrets-management
- https://www.hashicorp.com/resources/how-to-share-secrets-pipeline
- https://docs.cloudposse.com/secrets-management/anti-patterns/
resources & references

Cloud Providers

- https://cloud.google.com/blog/products/identity-security/introducing-google-clouds-secret-manager
- https://github.com/GoogleCloudPlatform/berglas
- https://azure.microsoft.com/en-us/services/key-vault

Config Management

- https://docs.ansible.com/ansible/latest/user_guide/vault.html
- https://github.com/voxpupuli/hiera-eyaml
- https://docs.chef.io/chef_vault

Password Management

- https://hackernoon.com/we-reverse-engineered-16k-apps-heres-what-we-found-51bdf3b456bb#.io6e11q6n
share your secrets
management

murriel@murrielgrace.com

xmurriel