Secrets Management for CMSWEB

Aroosha Pervaiz, Muhammad Imran, Valentin Y Kuznetsov, Panos Paparrigopoulos, Spyridon Trigazis, Andreas Pfeiffer.
CMS Offline & Computing, CMS, CERN, Switzerland.

Cyber attacks are inevitable.

- Secret Management enables us to:
  - Centralize the management of our sensitive data, including certificates, database credentials, and API Keys.
  - Protect the integrity of our system.
  - Properly distribute secrets among the organization without compromising confidentiality.
- We explored different secret management strategies, such as:
  - HashiCorp Vault
  - Github Credentials
  - SOPs with Age
- In this poster, we discuss the process by which we investigated these strategies and perform a feasibility analysis between them.
- We chose SOPS with age as a solution as it satisfies our requirements.

Motivation

- Only the operators maintained all CMSWEB services and cluster secrets in a secure place.
  - In case the responsible person is unreachable, we could be locked out from re-deploying our services.
    - The potential of this issue was highlighted by the recent incident at CERN IT when a few k8s clusters were deleted by the cleanup tool accidentally.
  - This incident prompted us to urgently improve our procedures for secrets management.

Feasibility Analysis

- Increasing the security robustness increases the complexity.

HashiCorp Vault

- ✓ Extensive features
- ✓ CERN SSO integration
- ✗ Complex configuration Requirements
- ✗ Added dependency
- ✗ Requires changes in services manifest files

Git Credentials

- ✓ Secure authentication mechanism
- ✗ More focus on git credentials
- ✗ Authentication managed by GitHub
- ✗ Difficult to distribute credentials around the organization

Mozilla SOPS/age

- ✓ Multiple libraries with *age* being simplest
- ✓ Separate key for each group
- ✓ Keys can be distributed through OpenStack or Git
- ✓ Allows us to encrypt secrets and store them directly on git
- ✓ Integrated in Helm
- ✓ Easy to integrate with bash scripts
- ✗ Does not allow for dynamic secret management

Illustration

- We developed bash scripts to incorporate *SOPS* with *age* to encrypt and decrypt secrets.

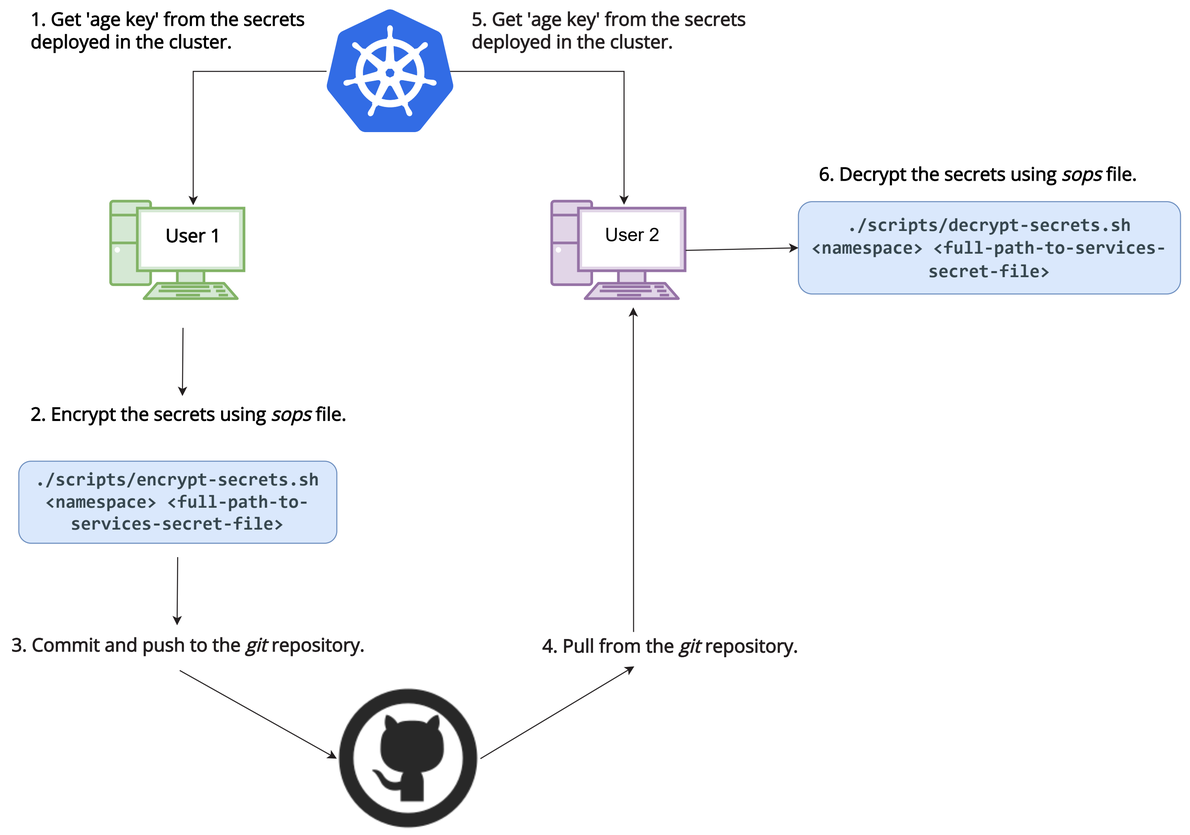

Figure 1: This flow explains how the encryption and decryption of secrets is facilitated by *SOPS/age*.

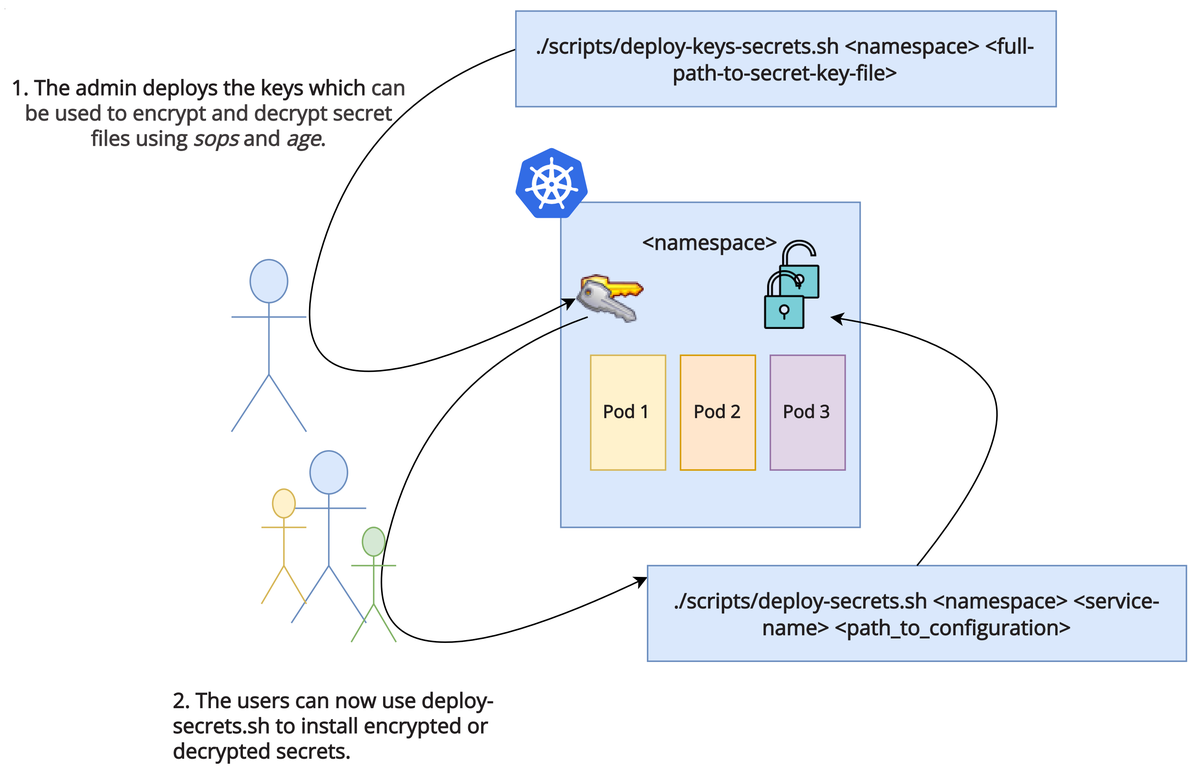

Figure 2: In order to deploy secrets, which are encrypted, we need to first deploy the *key secrets*. These keys are required to decrypt the original encrypted secret.

Conclusion and Future Work

- CMSWeb has adopted the use of *SOPS* with *age*, and we also recommend this to all interested CMS groups.
- Since the deployment of the secret is directly incorporated in the services deployment script, it makes the whole procedure seamless.
- Detailed documentation on how to install and use these tools has already been created.
- We developed bash scripts to automate the deployment of the secrets using encryption/decryption with *SOPS/age*.
- In the future, we will incorporate this methodology in helm charts.