NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Secret Management

Cloud Infrastructure Engineering

Nanyang Technological University & Skills Union - 2022/2023

Course Content

- Understanding the importance of securely storing sensitive information and the risks associated with poor secret management practices.
- Acquiring knowledge of common techniques used for secret management, such as encryption, access controls, and secure storage practices.
- Learning how to identify and classify secrets based on their level of sensitivity and the threats they face.

Q1: What is secret management and why is it important?

Q2: List some examples of secrets in AWS

Q3: What is AWS KMS and what are its key features?

Q4: What is AWS Systems Manager Parameter Store and what are its use cases?

Hands-On Activity: Create parameter in parameter store and retrieve it using AWS CLI

Activity

Instructor

- Ask to use AWS use single region for all learner for easier monitoring

What is Secret Management?

- Secret Management for cloud refers to the **practice of securely storing and managing sensitive information**, such as passwords, API keys, and other credentials, that are used to access and authenticate to cloud resources and services.
- Cloud providers offer various secret management solutions that enable users to securely store, manage, and access secrets. These solutions often **use encryption and access controls** to protect secrets **from unauthorized access or disclosure**.

What is Secret Management?

Common types of secrets

- Auto-generated passwords
- User passwords
- System-to-system passwords
- Database passwords
- Private encryption keys
- Authorization tokens
- Application keys and APIs
- SSH keys
- Private certificates (TLS, SSL, etc.)
- One-time passwords

Popular Secret Management Tools?

Some of the popular secret management solutions available in the cloud include:

- AWS Secrets Manager,
- Azure Key Vault,
- Google Cloud Secrets Manager, and
- HashiCorp Vault

These solutions provide centralized management of secrets, automate their rotation, and provide granular access control to them.

AWS Secrets Manager

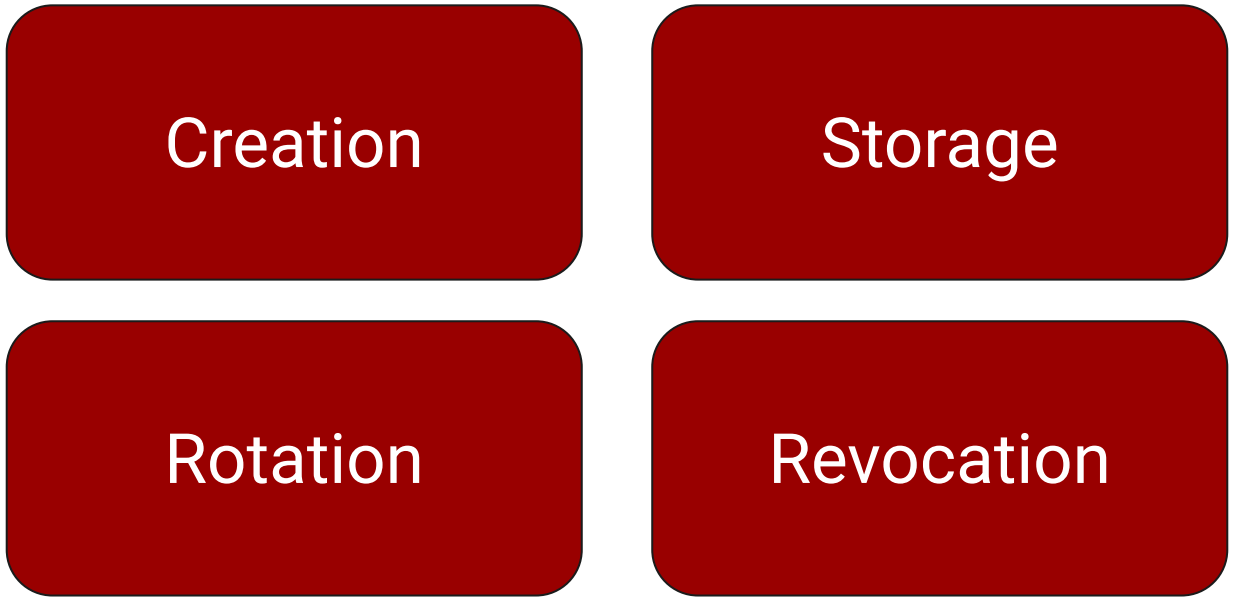
The need for Secret Management

Secrets management helps to ensure security at three levels:

- **Infrastructure security** – Protect user and application accounts, devices, and other network elements from intrusions.
- **Cloud service security** – Limit and manage access to cloud accounts and important cloud-based services.
- **Data security** – Protect critical systems, storages, databases, and other resources from data compromise.

Secrets’ Lifecycle

Secrets’ Lifecycle



Secret Lifecycle

Password or other secrets can undergo four main stages:

- **Creation** - Secrets can be manually created by a user (such as a personal account password) or automatically generated (such as an encryption key for a protected database).
- **Storage** - Secrets can be stored either centrally or separately, using dedicated solutions like a PAM-based secrets management tool or password manager, or common approaches like storing them in a text file or on a shared disk.
- **Rotation** - Secrets can be changed or reset on a regular schedule to enhance the overall security of an organization's infrastructure. Many regulations and standards, such as NIST and PCI DSS, require secrets rotation as a critical measure.
- **Revocation** - In the event of a cybersecurity incident, secrets can be revoked to prevent or mitigate the negative consequences of an attack. This measure ensures that attackers cannot use compromised credentials to gain access to an organization's critical resources, systems, endpoints, or applications.

Secrets’ Lifecycle

Note:

At each of these phases, **secrets should be protected** from unauthorized access, intervention, and manipulation.

However, many organizations struggle to build an efficient password management system.

Challenges with Secrets

Lack of visibility of all secrets

Lack of secrets' management policy

Manual handling of secrets

Secrets' Anti-Patterns

Using weak (default, hard-coded) passwords

Storing secrets in plain text

Sharing passwords

Not revoking secrets in time

Never rotating secrets

Reusing secrets

Anti-Pattern 1: Weak Passwords

Users tend to use weak passwords, including default account passwords, embedded and hard-coded application secrets, and easily guessable passwords.

This is one of the biggest password management sins.

The simpler it is to remember a password, the easier it is to crack it.

Common passwords like "password," "admin," and "123123" are especially vulnerable.

Embedded and hard-coded passwords are easily discovered by hackers through scanning tools, guessing, or brute-force attacks.

Anti-Pattern 2: Plain-text Secrets

Some teams and departments use **shared text files or emails** to **store and transmit passwords** for critical resources.

This practice creates multiple security risks since it is easier for attackers to obtain the file or message and gain access to the system.

Anti-Pattern 3: Sharing Secrets

Some organizations use shared accounts, making it difficult to identify who performed specific actions during a security incident.

Employees may also share personal account credentials with colleagues or outsiders, which can lead to malicious insiders gaining access to sensitive data.

Anti-Pattern 4: Not Revoking Unused Secrets

Revoking user credentials should be a standard response to an employee's resignation, the expiration of an agreement with a third-party vendor, or failed authorization attempts.

However, not all organizations follow this procedure in their secrets management routines.

Anti-Pattern 5: Not Rotating Secrets

Many security standards require regular password changes to reduce the risk of compromise.

Not all organizations rotate secrets on a regular basis, increasing the risk of their exposure.

Anti-Pattern 6: Reusing Secrets

Employees may use the same secret for different accounts, services, or applications to save time.

However, if a reused secret is compromised, all accounts and resources it was used for will be at risk.

Best Practices for Managing Secrets

Build a policy

Automate processes

Manage privileges

Best Practices 1: Building a Policy

To ensure the security of your organization's secrets, **formulate a strategy for managing them**, incorporating the list of poor practices from the previous section, and build a basic secrets management policy. Include the following key elements in your policy:

- Limit the use of hard-coded secrets and default passwords.
- Enforce strict password format requirements.
- Specify certain cases where mandatory revocation of secrets is required.
- Set a fixed timeframe for mandatory secrets rotation.

Best Practices 2: Automate Processes

Automate manual management of secrets that can lead to errors. Using dedicated secrets management software that can automate and centralize secrets management tasks securely can help to overcome this issue. Consider the following features when selecting a software tool:

- Capabilities to discover and account for all secrets in use.
- Options for key generation, rotation, revocation, storage, and transmission.

Best Practices 3: Manage Privileges

Users and applications with elevated privileges **have access to the most critical data, services, and resources**. Consequently, privileged accounts are high-value targets for cyber attackers. Therefore, it's crucial to manage privileges efficiently. To do so, follow these guidelines:

- Users and applications should only have the necessary privileges for performing routine duties.
- Any privilege elevation should have a legitimate reason and a limited duration.
- Keep a close eye on the management and monitoring of such privileges.

AWS Secrets Manager

AWS Secrets Manager

- One of the newer AWS services, meant for storing secrets
- Capability to force rotation of secrets every X hours, days, weeks, months etc.
- Automate generation of secrets on rotation (uses Lambda)
- Integration with Amazon RDS (MySQL, PostgreSQL, Aurora)
- Secrets are encrypted using KMS
- Mostly meant for RDS integration

Secure Secret Storage

AWS Secrets Manager **encrypts secrets at rest** using encryption keys that you own and store in **AWS Key Management Service (AWS KMS)**.

- When you retrieve a secret, Secrets Manager **decrypts the secret** and transmits it securely over TLS to your local environment.
- Secrets Manager integrates with AWS Identity and Access Management (IAM) to control access to the secret using fine-grained IAM policies and resource-based policies.

Automatic Secret Rotation

With AWS Secrets Manager, you can **rotate secrets on a schedule or on demand** by using the Secrets Manager console, AWS SDK, or AWS CLI.

- Secrets Manager natively supports r**otating credentials for databases** hosted on Amazon RDS and Amazon DocumentDB and clusters hosted on Amazon Redshift.
- You can extend Secrets Manager to rotate secrets used with other AWS or 3P services by modifying sample Lambda functions.

Automatic Replication Across Regions

With AWS Secrets Manager, you can **automatically replicate your secrets** to **multiple AWS Regions** to meet your unique disaster recovery and cross-regional redundancy requirements.

Specify the AWS Regions where a secret needs to be replicated and Secrets Manager will securely create regional read replicas, eliminating the need to maintain a complex solution for this functionality.

You can give your multi-Region applications access to replicated secrets in the required Regions and rely on Secrets Manager to keep the replicas in sync with the primary secret.

Programmatic Secrets Retrieval

Build your applications with security of secrets top of mind.

- Secrets Manager provides code samples to **call Secrets Manager APIs** from common programming languages.
- Configure Amazon Virtual Private Cloud (VPC) endpoints to keep traffic between your VPC and Secrets Manager **within the AWS network**.
- You can also use Secrets Manager client-side caching libraries to **improve availability and reduce latency** during secrets retrieval.

Audit & Monitor Secrets Usage

AWS Secrets Manager enables you to audit and monitor secrets through integration with AWS logging, monitoring, and notification services.

For example, after enabling **AWS CloudTrail** for an AWS Region, you can **audit when a secret is created or rotated by viewing AWS CloudTrail logs**.

Similarly, you can **configure Amazon CloudWatch** to receive email messages using Amazon **Simple Notification Service (SNS)** when secrets remain unused for a period, or you can configure Amazon CloudWatch Events to receive push notifications when Secrets Manager rotates your secrets.

Summary of AWS Secrets Manager

Summary of Secrets Management

Activity

Learner:

- Clean up AWS.
- Remove/delete/terminate all service/ resources that you created.

Instructor

- Clean up AWS.
- Remove/delete/terminate all service/ resources that you created.
- Check the AWS account after learner clean up.

What’s Next?