Comparison of Secrets Managers for Amazon Web Services (AWS)

October 2019 | Eric Evans

Table of Contents

Overview

Secrets Management refers to the tools and processes for managing sensitive information which is required to use throughout the application development and operations lifecycle. *Secrets* are any potentially sensitive information which typically grants access to additional data. Examples of secrets include private keys, database passwords, and API keys.

Secrets eventually need to be exposed in plaintext during the course of normal operations to provide access to the required systems. Therefore, the job of a secrets management solution is to manage the lifecycle of the secret, ensuring the secret is encrypted at rest, access to secrets follows least-privilege, secrets are rotated frequently, and access to secrets is audited.

There are many different solutions, both open-source and enterprise, for secrets management. This paper examines a few of the industry standard secrets management solutions for use in AWS. This is not an exhaustive list and one should perform an analysis based on their use cases before selecting a secrets management tool. In addition, please note that examples used in this document are for demonstration purposes and should be refined for production use.

Features

Both of the native AWS services examined specialize exclusively in secrets management. Hashicorp Vault has many capabilities beyond secrets management, with many new capabilities being developed daily as an open-source project[1].

[1] https://github.com/hashicorp/vault

AWS Secrets Manager

AWS Secrets Manager is a robust way to store secrets natively in AWS. It uses AWS Identity and Access Management (IAM) to produce policies to govern both access and management of secrets, and it utilizes AWS Key Management Service (KMS) to encrypt secrets at rest. By using IAM resource-based policies, trust relationships can be established to allow cross-account access of secrets from a centralized account[2]. A distinctive feature of secrets manager is the ability to rotate secrets. In order to achieve this, a separate AWS Lambda function must be created and appropriate roles must be granted to the function to execute rotation of the secret(s). In addition, Secrets Manager has a built-in password generator that is used during the rotation of secrets and can be invoked on the CLI using the `get-random-password` command. Finally, AWS Secrets manager has first class support for AWS CloudFormation templates[3].

AWS Systems Manager Parameter Store

AWS Systems Manager Parameter store is a simple AWS native solution that allows for the storage of two types of secrets, called parameters: standard and advanced. Standard parameters is the default tier that holds secrets up to 4 KB in size and have no additional charge associated with them. Advanced parameters increased the maximum size to 8 KB, and allows for parameter access and management policies expressed in IAM. Standard parameters can be changed to advanced, but not vice-versa. Like AWS Secrets Manager, all secrets are encrypted using AWS KMS. Unlike AWS Secrets manager, there are no auto-rotation or password generation capabilities.

Hashicorp Vault

Hashicorp Vault is described as the "swiss army knife" for cloud security,[4] with capabilities well beyond secrets management. Vault is cloud agnostic and has been deployed at scale in multiple public clouds with success. Vault secrets are encrypted at rest using an open-source implementation of Golang crypto and x/crypto libraries. Both access and management of secrets is governed using policies written in either Hashicorp Configuration Language (HCL) or JSON (JavaScript Object Notation). Vault supports password auto-rotation and dynamic credential capabilities with wide compatibility using a variety of secret backends. In addition to secrets management,

[2] https://aws.amazon.com/blogs/security/how-to-access-secrets-across-aws-accounts-by-attaching-resource-based-policies/
[3] https://aws.amazon.com/blogs/security/how-to-create-and-retrieve-secrets-managed-in-aws-secrets-manager-using-aws-cloudformation-template/
[4] https://www.hashicorp.com/resources/manage-ssh-with-hashicorp-vault

Vault can be used for other tasks such as brokering SSH access, encryption on demand, and password generation.

Security

The security of each solution varies. The two native AWS solutions have the ability to leverage IAM to provide security for managing and accessing secrets. Hashicorp Vault has secrets governance capabilities using policies. All solutions examined have encryption at rest capabilities enabled by default. A table summary is given below:

	Secrets Manager	Parameter Store - Standard	Parameter Store - Advanced	Hashicorp Vault
Access & Management Governance	IAM Policies	Limited	IAM Policies	Vault Policies
Encryption at Rest	AWS KMS	AWS KMS	AWS KMS	AES-256

AWS Secrets Manager

Access to AWS Secrets Manager requires AWS credentials. The credentials must have permissions to access secrets. IAM policies can be used to control both access and management of secrets. One example of leveraging the power of granular IAM policies is tagging by secret, thus creating an ability to restrict a specific IAM user or role access to a specific subset or version of secrets.

AWS Secrets Manager store uses AWS KMS to encrypt parameter values. A custom AWS KMS key can be used to encrypt your secrets instead of the default Secrets Manager customer managed key (CMK) for your account.

Access Policy Example For AWS Secrets Manager

The first statement of the following policy grants the user read access to the metadata about all of the secrets in the account. The second statement allows the user to perform any actions on only a single, specified secret by name—or on any secret that begins with the string secret2- followed by exactly 6 characters. Note the `??????` syntax used for the 6 characters:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:List*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "secretsmanager:*",
            "Resource": [
"arn:aws:secretsmanager:<region>:<account-id>:secret:secret-a1b2c3",
"arn:aws:secretsmanager:<region>:<account-id>:secret:secret2-??????"
            ]
        }
    ]
}
```

AWS Systems Manager Parameter Store

Access to AWS Systems Manager Parameter Store requires AWS credentials. The credentials must have permissions to access secrets. IAM policies can be used for advanced parameters to control Get, Describe, Put, and Delete actions invoked by the API. Note that *standard parameters cannot use access policies*, *therefore IAM policies only apply to advanced parameters*[5]. One example of leveraging the power of granular IAM policies is tagging by secret, thus creating an ability to restrict a specific IAM user or role access to a specific subset or version of secrets.

AWS Systems Manager Parameter store uses AWS KMS to encrypt parameter values. A custom AWS KMS key can be used to encrypt your secrets instead of the default Systems Manager CMK for your account.

Access Policy Example For AWS Systems Manager Parameter Store

The following IAM policy allows instances to get an *advanced* parameter value only for those that begin with `dev-`. If the parameter is a secure string, then it is decrypted using AWS KMS.

[5] https://docs.aws.amazon.com/systems-manager/latest/userguide/parameter-store-advanced-parameters.html

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "ssm:GetParameters"
         ],
         "Resource":[
            "arn:aws:ssm:<region>:<account-id>:parameter/dev-*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "kms:Decrypt"
         ],
         "Resource":[
            "arn:aws:kms:<region>:<account-id>:key/<key-name>"
         ]
      }
   ]
}
```

Hashicorp Vault

Access to Hashicorp Vault is managed by policies to govern the behavior of clients and instrument Role-Based Access Control (RBAC) by specifying access privileges (authorization). Policies use path-based matching to test the set of capabilities against a request. A policy path may specify an exact path to match, or it could specify a glob pattern which instructs Vault to use a prefix match.

The storage backends used by Vault are untrusted by design. Vault uses a security barrier for all requests made to the backend. The security barrier automatically encrypts all data leaving Vault using a 256-bit Advanced Encryption Standard (AES) cipher in the Galois Counter Mode (GCM) with 96-bit nonces. The nonce is randomly generated for every encrypted object. When data is read from the security barrier the GCM authentication tag is verified during the decryption process to detect any tampering.

Access Policy Example for Hashicorp Vault

The following vault policy grants all access to the `secret/*` path, with the exception of the secret/super-secret path. The `secret/restricted` path can only contain "foo" (any value) and "bar" (one of "zip" or "zap"). Because policies have an implicit deny, this particular policy grants no other access in Vault.

```
path "secret/*" {
  capabilities = ["create", "read", "update", "delete", "list"]
}

path "secret/super-secret" {
  capabilities = ["deny"]
}

path "secret/restricted" {
  capabilities = ["create"]
  allowed_parameters = {
    "foo" = []
    "bar" = ["zip", "zap"]
  }
}
```

Cost

Both of the AWS native solutions have a per secret cost and cost per API call, but have no operational cost due to the fact that they are maintained by AWS. On the other hand, the open source version of Hashicorp Vault has no per secret or access cost, but is self-hosted and has a cost for enterprise features.

The table below shows a summary of charges associated with each solution:

	Secrets Manager	Parameter Store - Standard	Parameter Store - Advanced	Hashicorp Vault
Per Secret Monthly Charge	$0.40	$0.00	$0.50	$0.00
Per 10,000 API Calls	$0.05	$0.00	$0.05	$0.00
Operational Cost	$0.00	$0.00	$0.00	varies*

** A pricing estimate for Hashicorp Vault's reference architecture is examined below*

AWS Secrets Manager

AWS Secrets Manager is $0.40 per secret per month, for secrets that are stored in less than a month the price is prorated. There is an additional charge of $0.05 per 10,000 API calls[6].

Pricing Example for AWS Secrets Manager

If 1000 secrets are stored using AWS Secrets Manager, with 400,000 API calls there is:

A monthly charge of $400 per month	API calls will cost an additional $2

AWS Systems Manager Parameter Store

AWS Systems Manager Parameter Store consists of standard and advanced parameters. Standard parameters are available at no additional charge. Advanced parameters are charged $0.05 per secret per month. Charges for parameters stored in less than a month are prorated. There is an additional charge of $0.05 per 10,000 API calls[7].

Pricing Example for AWS Parameter Store - Standard Parameters

Using standard parameter secrets are allowed at no additional charge

Pricing Example for AWS Parameter Store - Advanced Parameters

For example, if 1000 advanced parameter secrets are stored with 400,000 API calls using AWS Systems Manager Parameter Store, there is:

A monthly charge of $500 per month	API calls will cost an additional $2

[6] https://aws.amazon.com/secrets-manager/pricing/
[7] https://aws.amazon.com/systems-manager/pricing/#Parameter_Store

Hashicorp Vault

Hashicorp Vault comes in both open-source and two enterprise versions: Enterprise Platform and Enterprise Modules. Both enterprise solutions come at an additional cost, which can be given by contacting Hashicorp[8]. Hashicorp Vault is also a self-hosted solution for both open source and enterprise, therefore both provisioned infrastructure and operational upkeep must be considered when calculating cost. There is no additional cost for number of stored secrets or secrets retrieval.

Pricing Example for Hashicorp Vault

Using Vault's reference architecture[9]: three m5.large instances with 25GB gp2 drives attached to achieve n-2 redundancy (where the loss of 2 objects within the failure domain can be tolerated), we can provide an estimated cost for Hashicorp Vault. The calculation below was performed using the AWS pricing calculator:

```
3 instances x $0.096 x 730 hours in month = $210.24 (monthly, on-demand instances)
```

Conclusion

For low quantities of secrets, AWS Secrets Manager and AWS Systems Manager Parameter store is a cost-effective solution. AWS Systems Manager Standard Parameters is appealing if you have less than 10,000 secrets and no secrets greater than 4 KB since there is no additional cost, but is not recommended due to the lack of security features - namely a lack of parameter policies. It may still be useful for development environments or quick proofs of concepts that do not hold sensitive secrets. If you are running in multiple clouds and/or handling a high quantity of secrets that are accessed frequently, then Hashicorp Vault is the solution of choice.

[8] https://www.hashicorp.com/products/vault/enterprise
[9] https://learn.hashicorp.com/vault/operations/ops-reference-architecture#sizing-for-vault-servers