Secrets Management

Yulia Tenincheva 12 December 2019

whoami

Yulia Tenincheva

Senior Cloud Engineer @ CCoE

~5 years of Development background with Node.js

Open Source Contributor & Supporter

Agenda

- Configuration vs Secrets
- Environment Variables
- No secrets = No problem (IAM Roles, SSO)
- Amazon Certificate Manager (ACM)
- Encryption & Encrypted Storage (KMS, S3)
- AWS Parameter Store
- AWS Secrets Manager
- Best Practices

Is that a secret?

Logging level
Environment
SSL Certificates
Locale
Dependency Versions
Boolean Flags
AWS Account ID
SSH Key
SQS Queue Name
Database Port
AWS_ACCESS_KEY_ID
Database connection string
AWS Region
JWT Secret
S3 Bucket Name

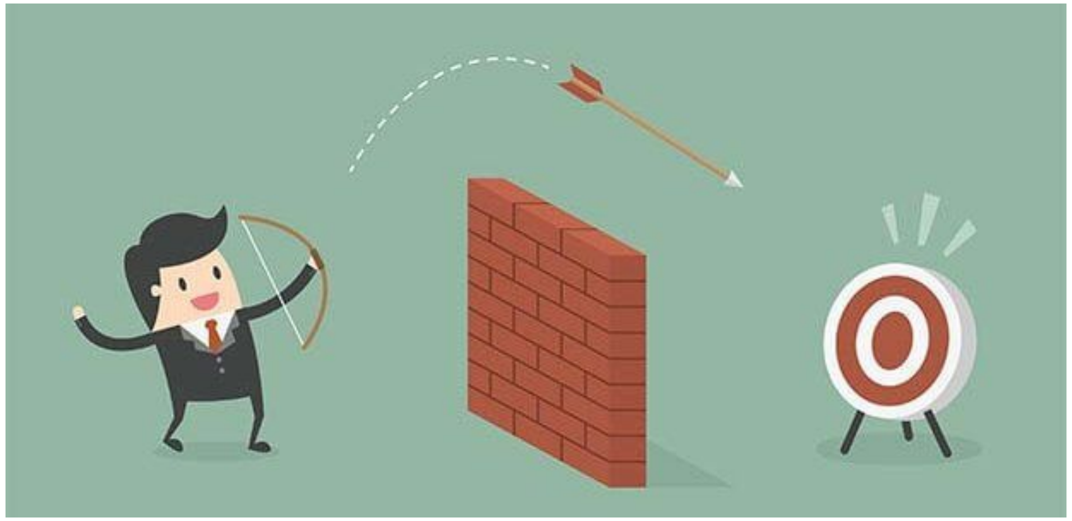
SSL Certificates	Logging level
AWS Account ID	Environment
SSH Key	Locale
AWS_ACCESS_KEY_ID	Dependency Versions
Database connection string	Boolean Flags
JWT Secret	Database Port
	SQS Queue Name
	AWS Region
	S3 Bucket Name

Types of secrets

- Database connection strings
- User credentials
- Cryptographic keys
- API keys
- Access tokens
- Cloud service access credentials

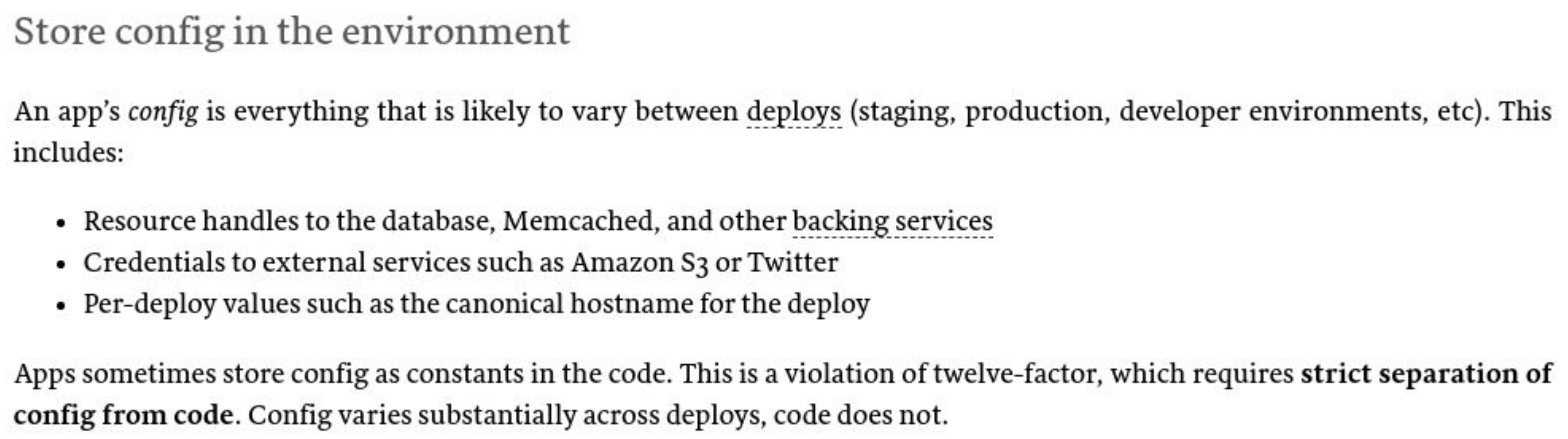
Mistakes and challenges

- Incomplete visibility and awareness
- Hardcoded/embedded credentials
- Privileged credentials
- DevOps Tools
- 3d party vendor accounts
- Manual secrets management processes



Configuration & Secrets

- The Twelve-Factor App Manifesto - III. Config



Store config in the environment

An app's *config* is everything that is likely to vary between deploys (staging, production, developer environments, etc). This includes:

- Resource handles to the database, Memcached, and other backing services
- Credentials to external services such as Amazon S3 or Twitter
- Per-deploy values such as the canonical hostname for the deploy

Apps sometimes store config as constants in the code. This is a violation of twelve-factor, which requires **strict separation of config from code**. Config varies substantially across deploys, code does not.

Why you should NOT use ENV vars for secret data

- It's hard, if not **impossible**, **to track access** and how the contents get exposed.
- It's common to have applications grab the whole environment and **print it out for debugging**.
- Environment variables are **passed down to child processes**, which allows for **unintended access**.
- When applications crash, it's common for them to **store the environment variables in log-files** for later debugging. This means **plain-text secrets on disk**.
- Putting secrets in ENV variables quickly turns into **tribal knowledge**. New engineers who are not aware of the sensitive nature of specific environment variables will not handle them appropriately/with care.

What are we looking for?

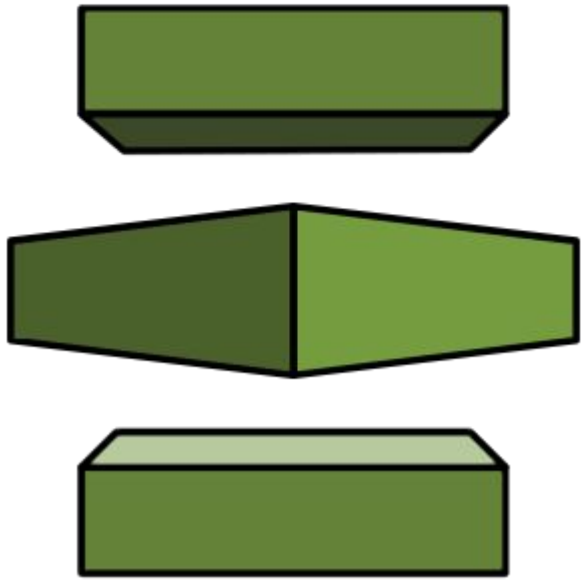
- Manage access with fine-grained policies
- Transparency of use
- Encryption at rest & transit
- Encryption key export
- Rotate secrets safely. Automatically
- Revocation of encryption keys
- Pay as you go

IAM

- Instance Profiles & IAM Roles
- IAM DB Authentication - tutorial
  - For MySQL, PostgreSQL & Aurora
- SSO Solutions (AD, LDAP)
  - AD for EC2 & SQL Server on RDS

Amazon Certificate Manager

- Free public certificates for ACM-integrated services (ELB, API Gateway, CloudFront)
- Managed certificate renewal
- Help meet compliance requirements



Encrypted Storage

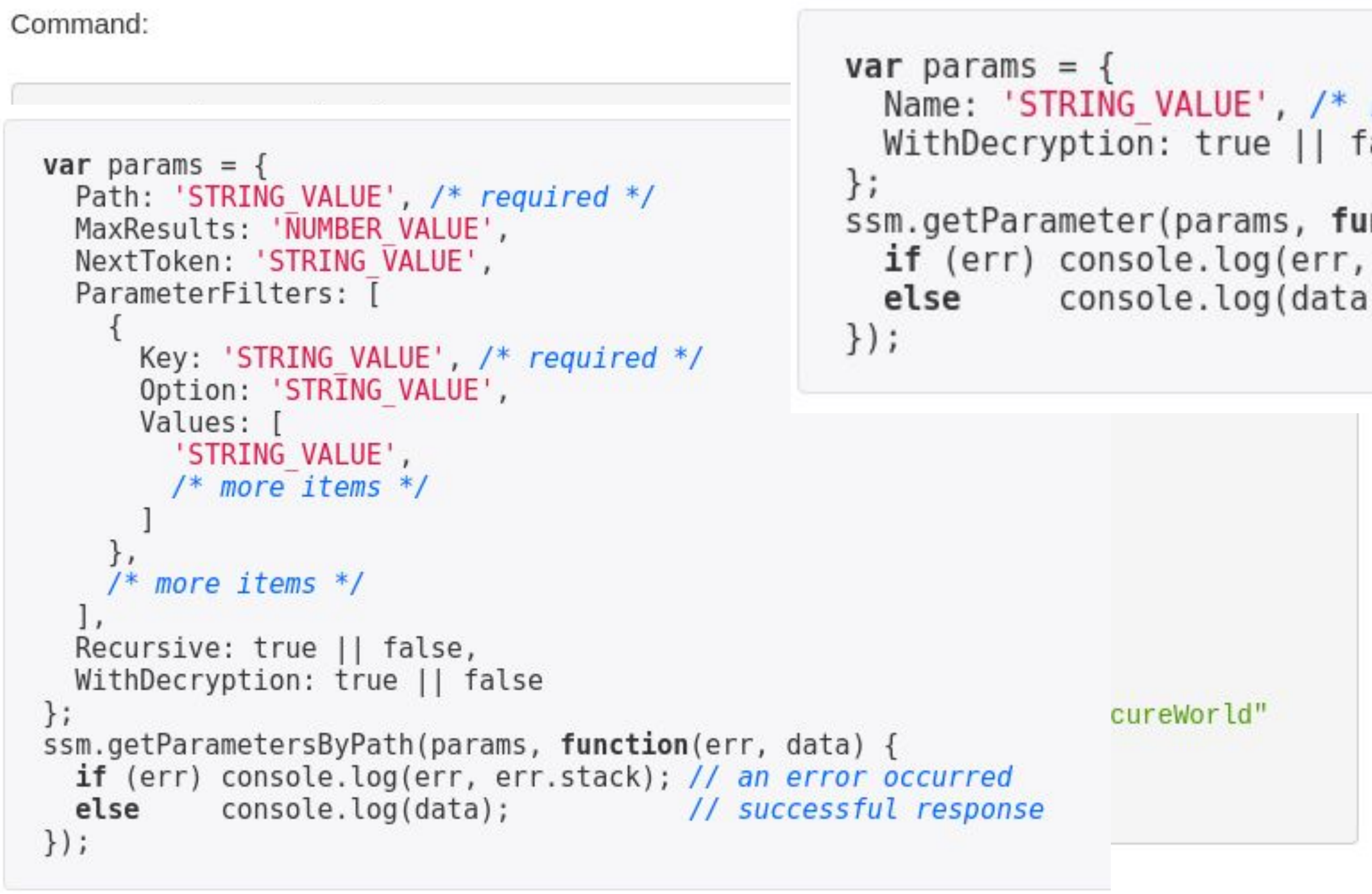
- Encrypted S3 with IAM Policies
- KMS API
- Is that secure?
- Is that convenient?

Encrypted S3 Storage is perfect for storing sensitive data, but it's not a solution for secrets management.

AWS SSM Parameter Store

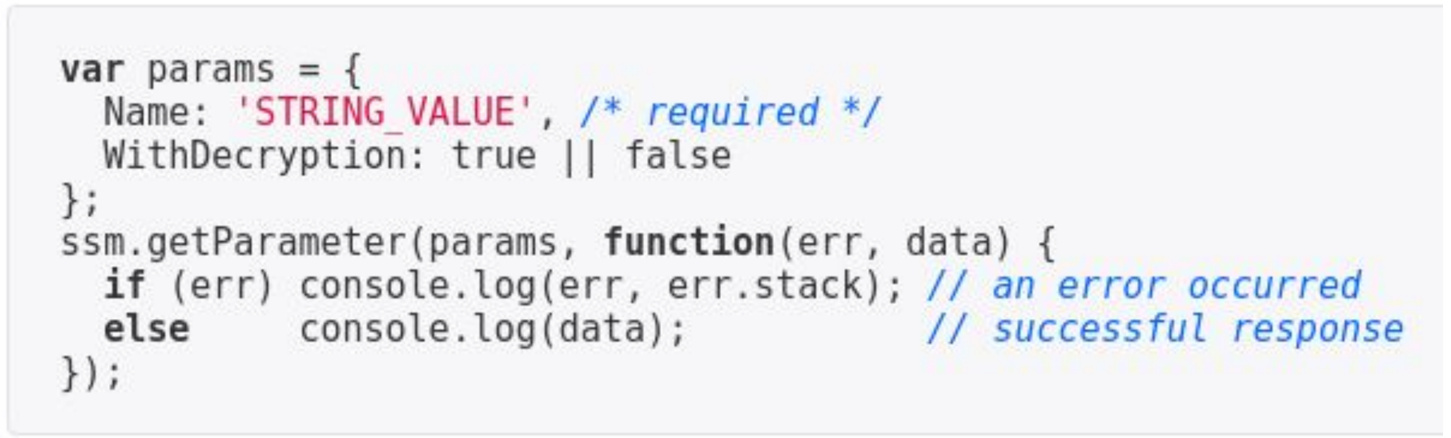
- Key-Value
- Data types
  - String
  - String list
  - SecureString (Encryption via KMS)
- Access controlled via IAM Policy
- Auditable with CloudTrail
  - KMS operations also logged
- Considerations
  - Region-based (key too)
  - Limits (Standard vs Advanced)
- Pricing

Examples


Command:

```
var params = {
  Path: 'STRING_VALUE', /* required */
  MaxResults: 'NUMBER_VALUE',
  NextToken: 'STRING_VALUE',
  ParameterFilters: [
    {
      Key: 'STRING_VALUE', /* required */
      Option: 'STRING_VALUE',
      Values: [
        'STRING_VALUE',
        /* more items */
      ]
    },
    /* more items */
  ],
  Recursive: true || false,
  WithDecryption: true || false
};
ssm.getParametersByPath(params, function(err, data) {
  if (err) console.log(err, err.stack); // an error occurred
  else     console.log(data);           // successful response
});
```

cureWorld"



```
var params = {
  Name: 'STRING_VALUE', /* required */
  WithDecryption: true || false
};
ssm.getParameter(params, function(err, data) {
  if (err) console.log(err, err.stack); // an error occurred
  else     console.log(data);           // successful response
});
```

[Get parameter value with AWS CLI - Example]()

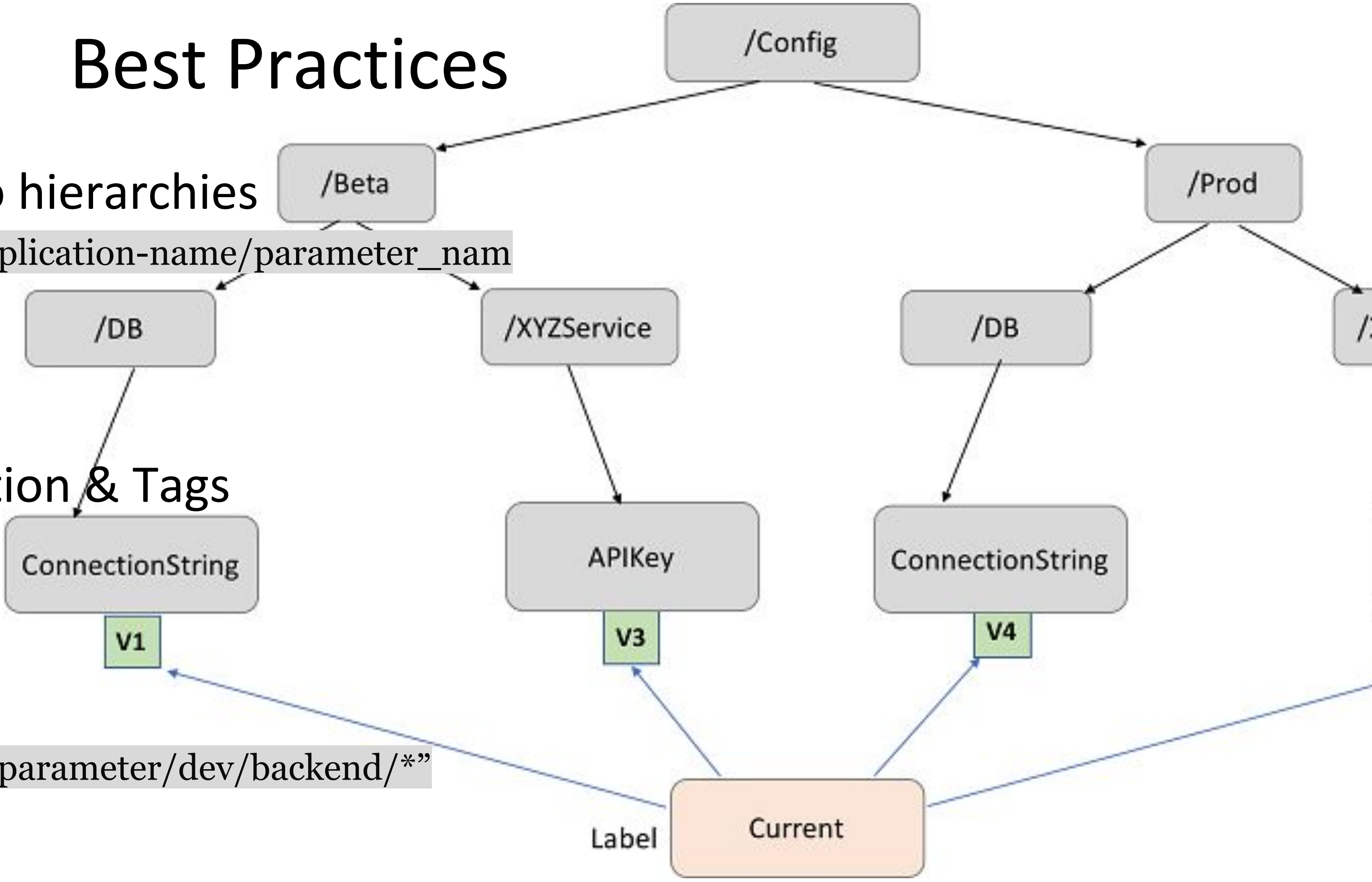
AWS SSM Parameter Store

Best Practices

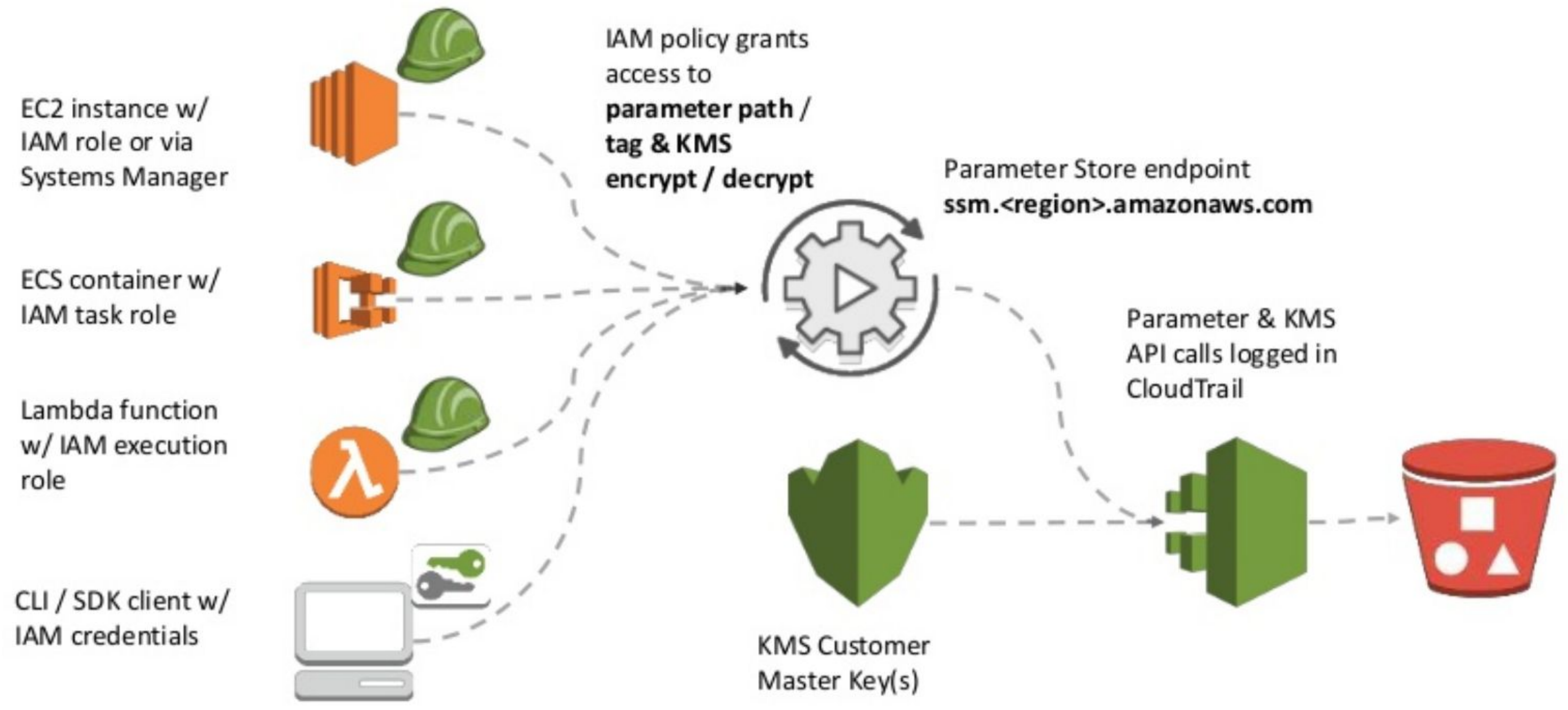
1. Organizing parameters into hierarchies
   /environment/service-name/type/application-name/parameter_nam
   e

2. Consistent naming convention & Tags

3. Restrict IAM permission
   ”Resource”: “arn:aws:ssm:us-east-2::parameter/dev/backend/*”



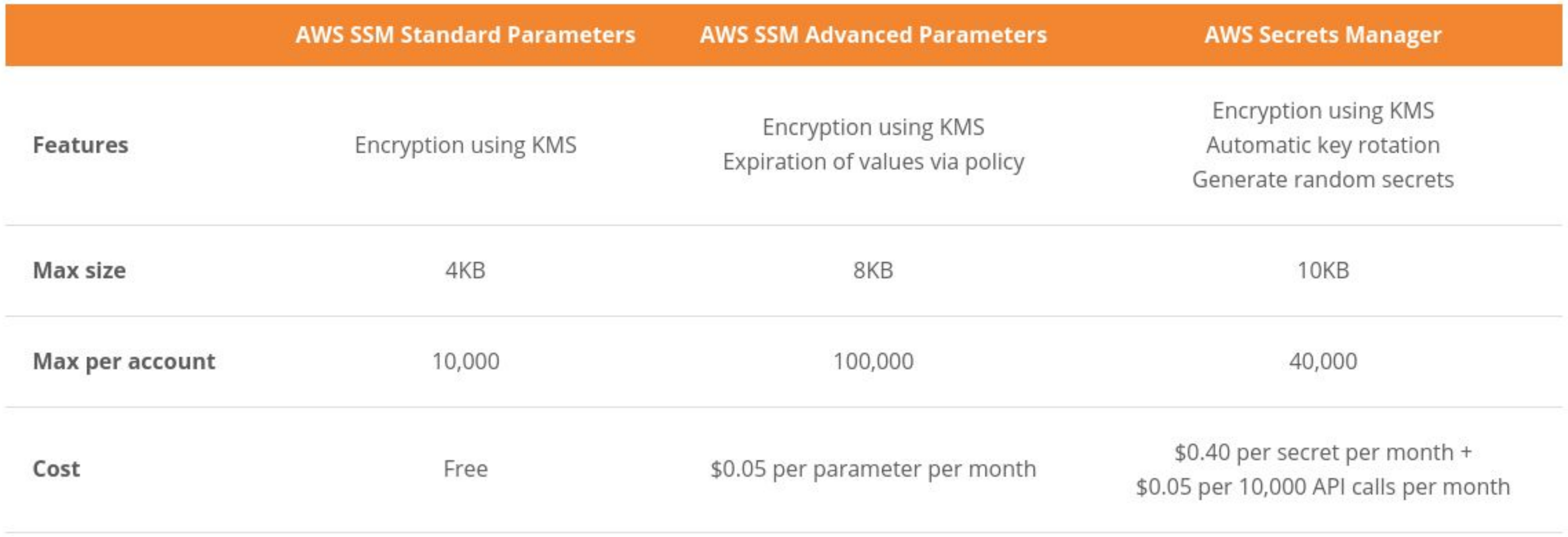
AWS SSM Parameter Store


EC2 instance w/ IAM role or via Systems Manager
ECS container w/ IAM task role
Lambda function w/ IAM execution role
CLI / SDK client w/ IAM credentials
IAM policy grants access to **parameter path / tag & KMS encrypt / decrypt**
Parameter Store endpoint **ssm.<region>.amazonaws.com**
Parameter & KMS API calls logged in CloudTrail
KMS Customer Master Key(s)

AWS Secrets Manager

- Manage the lifecycle for secrets
- Automatic secrets rotation
- Built-in integrations for Amazon RDS
- Extensible via Lambda
- Referencing AWS Secrets Manager Secrets from Parameter Store Parameters

AWS SSM vs Secrets Manager



	AWS SSM Standard Parameters	AWS SSM Advanced Parameters	AWS Secrets Manager
Features	Encryption using KMS	Encryption using KMS Expiration of values via policy	Encryption using KMS Automatic key rotation Generate random secrets
Max size	4KB	8KB	10KB
Max per account	10,000	100,000	40,000
Cost	Free	$0.05 per parameter per month	$0.40 per secret per month + $0.05 per 10,000 API calls per month

Demo

How to share secrets securely

- In-person hand off
- Don’t send sensitive documents over email
    - If you’re adventurous - checkout GnuPG, PGP, Enigmail
- Use an encrypted file-sharing service
- Encrypt transferred files + password-protected archive
- Use a password manager
    - KeePass vault file

Resources

- OWASP Key Management Cheat Sheet
- How to create and retrieve secrets managed in AWS using AWS CloudFormation templates - tutorial
- Parameter Store use cases and best practices - documentation
- Secrets Manager - Rotating a Secret for an AWS Database - tutorial

MENTORMATE ||

See you ...next year

MENTORMATE ||

Thank you!