JON BONSO AND KENNETH SAMONTE

AWS CERTIFIED DEVOPS ENGINEER PROFESSIONAL

Tutorials Dojo
Study Guide and Cheat Sheets

AWS CodeBuild Configuration Best Practices

Configuration files are essential in your Continuous Integration / Continuous Delivery (CI/CD) pipeline. It contains vital information on how to build your application and the required permissions, password or other credentials to enable it to connect to its external dependencies such as a database or other AWS services. The sensitive information in these configuration files could pose a security risk. Part of configuration management is to ensure that these files comply with your organization's security standards and policies.

A build specification file (buildspec.yaml) contains the commands and related settings to instruct AWS CodeBuild on how to build your source code. Adding IAM access keys and database passwords in plaintext in your specification file is discouraged as these could be easily seen by unauthorized personnel. A better approach is to create an IAM Role and attach a permissions policy that grants permissions on your resources. You can store passwords and other sensitive credentials in AWS Systems Manager Parameter Store or AWS Secrets Manager. You can also leverage on using AWS Systems Manager Run Command instead of using scp and ssh commands that could potentially expose the SSH keys, IP addresses and root access of your production servers.

Below is an example of a poorly designed `buildspec.yaml` file. What do you notice?

Secrets Manager vs. Systems Manager Parameter Store

AWS Secrets Manager and Systems Manger Parameter Store offer similar functionalities that allow you to centrally manage and secure your secret information that can then be retrieved by your applications and resources in AWS.

Both services offer similar web interfaces on which you can declare key-value pairs for your parameters and secrets. So it is important to know the similarities and differences they have in order to choose the right service for a given situation in the exam.

Here's a summary of features of SSM Parameter Store and AWS Secrets Manager:

	SSM Parameter Store	AWS Secrets Manager
Store values up to 4096 Characters	Yes	Yes
Values can be encrypted with KMS	Yes	Yes
Can be referenced in CloudFormation	Yes	Yes
Built-in password generator		Yes
Automated secret rotation		Yes
Cross-account access		Yes
Additional Cost	Free	Yes

DevOps Exam Notes:

In the exam, the usual situation is when you need to store a database password, you should choose SSM Parameter Store if you don't have to automatically rotate the secret regularly, plus it doesn't cost anything.

As an additional note, Parameter Store is now integrated with Secrets Manager so that you can retrieve Secrets Manager secrets when using other AWS services that already support references to Parameter Store parameters. This is helpful if your application is configured to use Parameter Store APIs, but you want your secrets to be stored in Secrets Manager.

Sources:
https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-parameter-store.html
https://aws.amazon.com/about-aws/whats-new/2018/07/aws-systems-manager-parameter-store-integrates-with-aws-secrets-manager-and-adds-parameter-version-labeling/

AWS Secrets Manager

- A secret management service that enables you to easily rotate, manage, and retrieve database credentials, API keys, and other secrets throughout their lifecycle.
- **Features**
  - AWS Secrets Manager encrypts secrets at rest using encryption keys that you own and store in AWS Key Management Service [customer managed keys]. When you retrieve a secret, Secrets Manager decrypts the secret and transmits it securely over TLS to your local environment.
  - 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 rotating credentials for databases hosted on Amazon RDS and Amazon DocumentDB and clusters hosted on Amazon Redshift.
  - You can extend Secrets Manager to rotate other secrets, such as credentials for Oracle databases hosted on EC2 or OAuth refresh tokens, by using custom AWS Lambda functions.
- A secret consists of a set of credentials (username and password), and the connection details used to access a secured service.
- A secret can contain versions:
  - Although you typically only have one version of the secret active at a time, multiple versions can exist while you rotate a secret on the database or service. Whenever you change the secret, Secrets Manager creates a new version.
  - Each version holds a copy of the encrypted secret value.
  - Each version can have one or more staging labels attached identifying the stage of the secret rotation cycle.
- Supported Secrets
  - Database credentials, on-premises resource credentials, SaaS application credentials, third-party API keys, and SSH keys.
  - You can also store JSON documents.
- To retrieve secrets, you simply replace secrets in plain text in your applications with code to pull in those secrets programmatically using the Secrets Manager APIs.
- Secrets can be cached on the client side, and updated only during a secret rotation.
- During the secret rotation process, Secrets Manager tracks the older credentials, as well as the new credentials you want to start using, until the rotation completes. It tracks these different versions by using staging labels.