Automation

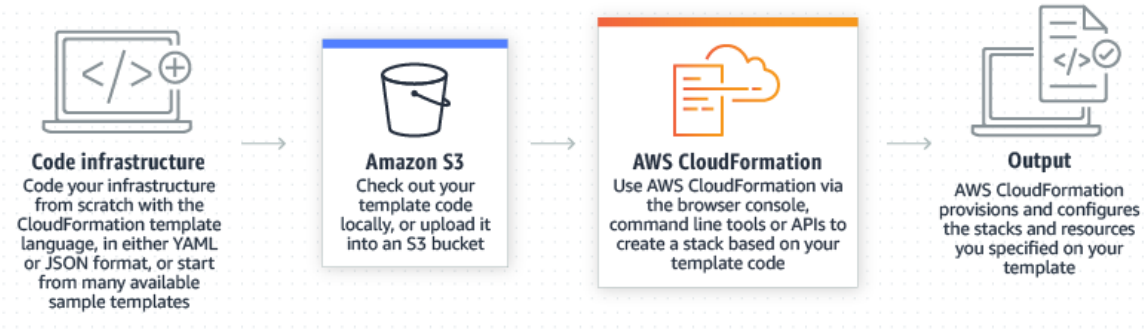
>Why do we automate?

Automating will help with user error with entering things in manually. Think about it… Throughout labs, there are multiple steps to get what is needed and errors will occur.

This will save A LOT of time to automate things, that way you can have more time to focus on more important things. This will help with preventing security incidents and even vulnerabilities. This makes it more consistency once you give it a specific set of instructions.

>CloudFormation

- Cloud formation
  - AWS CloudFormation lets you model, provision, and manage AWS and third-party resources by treating infrastructure as code.
    - Storage, database, analytics, machine learning and more.
  - Create templates for the resources you want to provision
  - Works with most AWS services
  - Infrastructure as a code tool used to define a wide variety of AWS resources
  - Automate the infrastructure-provisioning process for EC2 servers



CloudFormation CodePipeline can automatically deploy your AWS infrastructure using CloudFormation. For example, developers can create their own template that builds a complete test environment. Whenever they update the template, CodePipeline can automatically deploy it to CloudFormation. This approach allows developers to create their own development infrastructure.

- Step 1 – Write Code
  - Supports either JSON or YAML formatting
- Step 2 – Deploy your template
  - Once uploaded, CloudFormation will go through the process of making the needed AWS API calls on your behalf

>Elastic Beanstalk

- AWS Elastic Beanstalk
  - Elastic Beanstalk is even simpler than Lightsail. All that's expected from you is to define the application platform and then upload your code.

    - Elastic Beanstalk manages the underlying infrastructure for your application and automatically scales according to demand.
    - AWS Elastic Beanstalk is an easy-to-use service for deploying and scaling web applications and services developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker on familiar servers such as Apache, Nginx, Passenger, and IIS.
        - Once the code is built out, it takes that information and builds out the environment for you. It also makes it easy to save it to make it easy to deploy it again.
    - This makes you focus on the business application not the infrastructure
- PaaS
    - Platform as a service
    - Single-stop application deployment model that will provision, deploy, manage and monitor all the architecture.

We can use beanstalk to do everything for us, so this falls under that automation category.

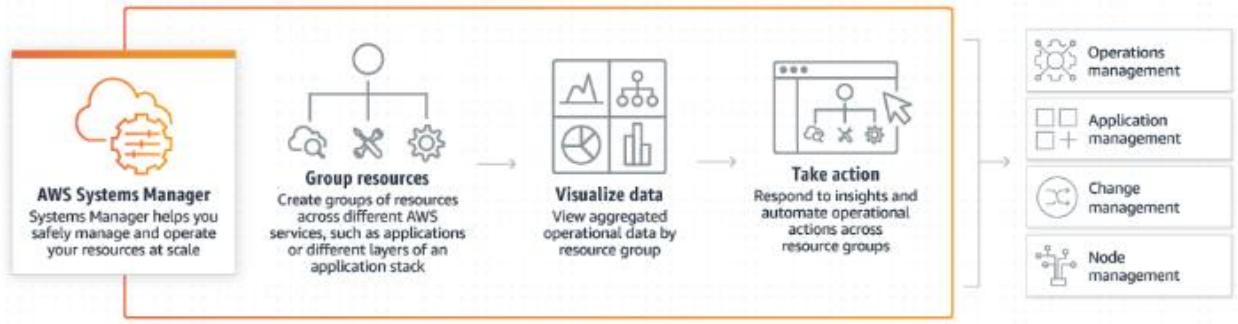
It automates it > builds out EC2 instances > builds out the load balancers, the auto scaling groups, the CloudWatch alarms, and deploys your code into that. So, the cool thing about this, it handles the deployments for us, so we don't have to sign into the host and manually download something or configure, or even use user data to try to bootstrap the architecture.

>Systems Manager

Allows you to view, control, and automate both AWS architecture and on-prem resources. defines the actions that Systems Manager performs on your managed instances and other AWS resources when an automation runs.

- Systems Manager
    - Automates common operational tasks such as patching instances and backing up Elastic Block Store (EBS) volumes
    - Gives you visibility and control over your AWS resources
    - Systems Manager uses the imperative approach to get your instances and AWS environment into the state that you want
    - In addition to providing configuration management for instances, Systems Manager lets you perform many administrative AWS operations

Systems manager requires an agent to be installed onto either your EC2 instances or your on-prem architecture because this supports both.



- Services System manager offers
    - Automation Documents
        - Can be used to control your instances or AWS resources
        - Configure the inside of your EC2 instances or configure AWS resources
            - Such as setting an S3 bucket policy, or something inside your account
        - Automation documents is now called Runbooks
    - Run Command
        - Uses System Manager Agent. This is installed on your EC2 instanced to do things.
        - Run scripts, calls inside the OS and get a report back to see if it was successful or not
    - Patch Manager
        - Schedules a time on what patches do you want to apply, which systems manager can handle that for me.
    - Parameter Store
        - Where you keep all secrets, more like your usernames and passwords
        - Things you should not be hardcoding into the code itself
    - Session Manager
        - Remotely connect and interact with your architecture
        - So you don't have to manage all those SSH keys yourself

>SecureString parameters

AWS recommends SecureString parameters if you want to use data/parameters across AWS services without exposing the values as plaintext in commands, functions, agent logs, or CloudTrail logs.

A SecureString parameter is any sensitive data that needs to be stored and referenced in a secure manner. If you have data that you don't want users to alter or reference in plaintext, such as passwords or license keys, create those parameters using the SecureString data type.

Exam Tips

1. Automate
    a. Whenever possible, select an answer that doesn't include manual steps.
    b. Replace manual steps with automated tools
    c. More reliable and faster
    d. Can easily be destroyed and easily build the architecture back up
2. CloudFormation
    a. Creating immutable architecture
        i. Immutable infrastructure takes a different approach – where servers once deployed cannot be modified.
        ii. a model in which no updates, security patches, or configuration changes happen "in-place" on production systems. If any change is needed, a new version of the architecture is built and deployed into production.
    b. That is where templates come in play with having the immutable architecture
        i. This makes it easy to create and destroy the entire architecture
    c. Cross-Region
        i. Hard-coded values and resource IDs can be the reason templates fail to create
    d. If it finds an error, CloudFormation rolls back to the last known good state
    e. CloudFormation makes the same API calls you make manually
3. Elastic Beanstalk
    a. For simple one-stop solutions, think Elastic Beanstalk
    b. PaaS
        i. Handles it all, there is very little you need to do since it is automated
    c. Platform Types
        i. Containers, Windows and Linux applications. Don't need to memorize them all
    d. Not serverless, beanstalk is creating and managing standard EC2 architecture
4. Systems Manager
    a. Many will not say systems manager, instead, they will mention the names of the features that will be used.
        i. Like using automation documents to fix S3 bucket permissions or using Session Manager to connect to an instance.
    b. Can be managed/supported both on prem and cloud architecture
        i. Have to have systems manager agent
    c. Free free free as long as it is inside of AWS
        i. Small fee for managing on-prem, so technically not free
    d.