Implementation Guide

Landing Zone Accelerator on AWS

Landing Zone Accelerator on AWS: Implementation Guide

Table of Contents

Deploy a cloud foundation to support highly-regulated workloads and complex compliance requirements

The Landing Zone Accelerator on AWS (LZA) is architected to align with AWS best practices and in conformance with multiple, global compliance frameworks. We recommend customers deploy AWS Control Tower as the foundational landing zone and enhance their landing zone capabilities with Landing Zone Accelerator. These complementary capabilities provide a comprehensive no-code solution across 35+ AWS services to manage and govern a multi-account environment built to support customers with highly-regulated workloads and complex compliance requirements. AWS Control Tower and Landing Zone Accelerator help you establish platform readiness with security, compliance, and operational capabilities.

We provide this solution as an open-source project that we built using the AWS Cloud Development Kit (AWS CDK). You can install it directly into your environment, giving you full access to the infrastructure as code (IaC) solution. Through a simplified set of configuration files, you can:

- Configure additional functionality, controls, and security services such as AWS Config Managed Rules and AWS Security Hub.
- Manage your foundational networking topology such as Amazon Virtual Private Cloud (Amazon VPC), AWS Transit Gateway, and AWS Network Firewall.
- Generate additional workload accounts using the AWS Control Tower Account Factory.

There are no additional charges or upfront commitments required to use Landing Zone Accelerator on AWS. You pay only for AWS services turned on to set up your platform and operate your controls. This solution can also support non-standard AWS partitions, including the AWS GovCloud (US), AWS Secret, and AWS Top Secret Regions.

This implementation guide describes architectural considerations and configuration steps for deploying the Landing Zone Accelerator on AWS. It includes links to an AWS CloudFormation template synthesized from AWS CDK that launches and configures the AWS services required to deploy this solution using AWS best practices for security and availability.

Use this navigation table to quickly find answers to these questions:

If you want to...	Read...
Know the cost for running this solution. The estimated cost for running this solution using AWS sample configuration with AWS Control Tower in the US East (N. Virginia) Region within a non-critical sandbox environment with no activity or workloads is approximately **$430.22 (USD) per month**.	Cost
Understand the security considerations for this solution.	Security
Know how to plan for quotas for this solution.	Quotas
Know which AWS Regions are supported for this solution.	Supported AWS Regions
View or download the AWS CloudForm ation template included in this solution to automatically deploy the infrastructure resources (the "stack") for this solution.	AWS CloudFormation template
Deploy this solution in a configuration that supports a specific Region or industry.	Landing Zone Accelerator on AWS solution page
Know how to troubleshoot common deployment errors.	Troubleshooting
Use AWS Support to help you deploy, use, or troubleshoot the solution.	AWS Support
Access the source code and optionally use the AWS Cloud Development Kit (AWS CDK) to deploy the solution.	GitHub repository

This guide is intended for solution architects, business decision makers, DevOps engineers, data scientists, and cloud professionals who want to implement the Landing Zone Accelerator on AWS solution in their environment.

Important

This solution will not, by itself, make you compliant. It provides the foundational infrastructure from which additional complementary solutions can be integrated. The information contained in this solution implementation guide is not exhaustive. You must review, evaluate, assess, and approve the solution in compliance with your organization's particular security features, tools, and configurations. It is the sole responsibility of you and your organization to determine which regulatory requirements are applicable and to ensure that you comply with all requirements. Although this solution discusses both the technical and administrative requirements, this solution does not help you comply with the non-technical administrative requirements.

Use cases

You can use configurations of this solution to support alignment with the following Regions and industries (see the Landing Zone Accelerator on AWS page for more information):

- AWS opt-in Regions
- Country guidelines:
  - Canadian Centre for Cyber Security (CCCS) Cloud Medium
  - United Kingdom (UK) National Cyber Security Centre (NCSC)
  - United States (US) Federal and Department of Defense (DoD)
- Industries:
  - Education
  - Elections
  - Finance (tax)
  - Healthcare
  - National Security, Defense, and National Law Enforcement
  - US aerospace
  - US state and local government Central IT

Concepts and definitions

This section describes key concepts and defines terminology specific to this solution.

AWSAccelerator and **aws-accelerator**

As of version 1.4.0, this solution allows for a user-defined resource name prefix in the Installer stack parameters. This guide uses the default prefix values `AWSAccelerator` and `aws-accelerator` for the named resource it describes. If you input a custom prefix, your solution-deployed CloudFormation stacks and Amazon S3 buckets use your custom prefix value.

landing zone

A cloud environment that offers a recommended starting point—including default accounts, account structure, core networking infrastructure, and security configurations. Using a landing zone as a foundation, you can deploy your mission-critical application workloads and solutions across a centrally-governed multi-account environment.

Installer pipeline (`AWSAccelerator-Installer`)

Deploys an installer that, in turn, deploys the solution's core features. Because this installer functions separately from the Core pipeline, you can update to future versions of the solution with a single parameter through the AWS CloudFormation console.

Core pipeline (`AWSAccelerator-Pipeline`)

Deploys the solution's core features.

> **Note**
>
> For a general reference of AWS terms, see the AWS Glossary.

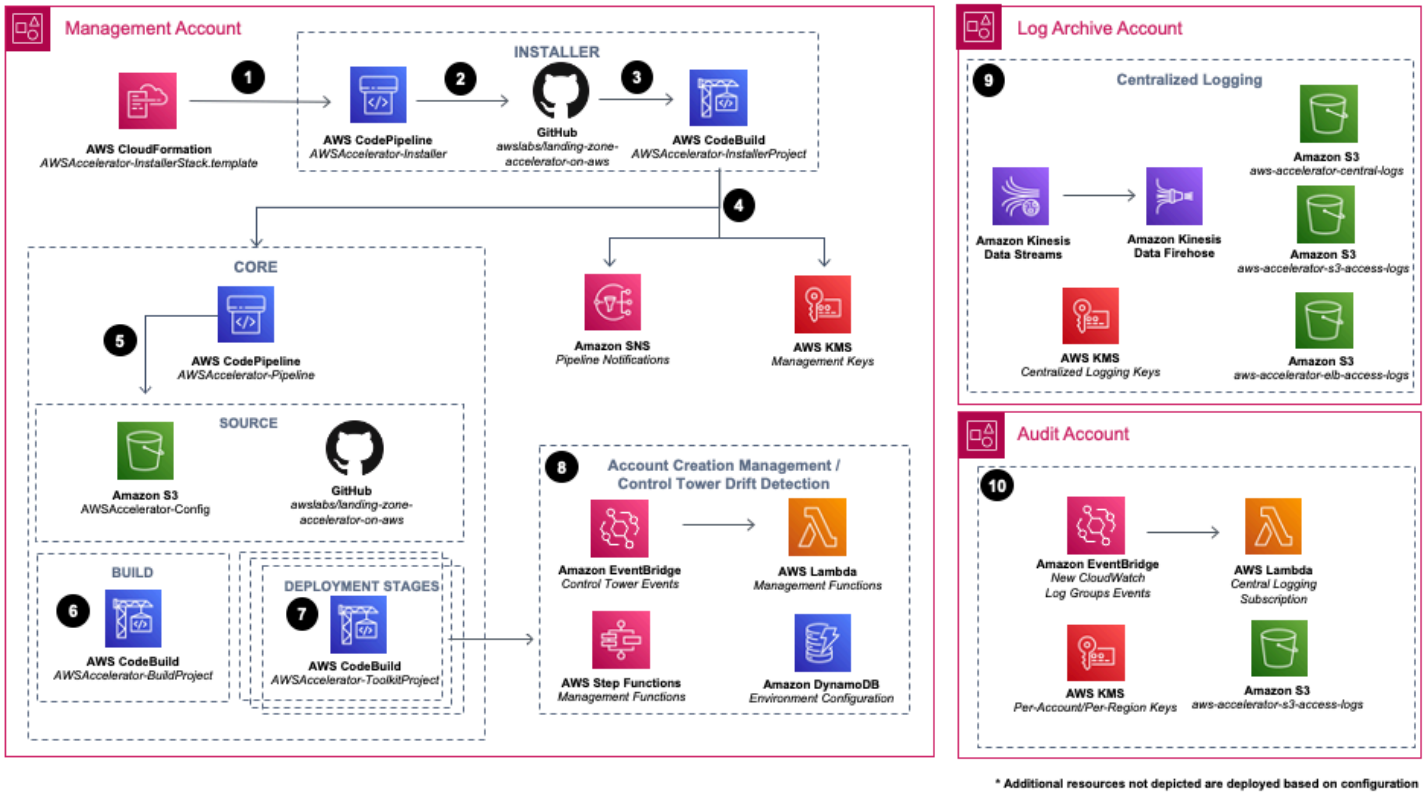
Architecture overview

This section provides a reference implementation architecture diagram for the components deployed with this solution.

Architecture diagram

Deploying this solution with the default parameters deploys the following components in your AWS account.

AWS architecture diagram showing Management, Log Archive, and Audit accounts with various services and their interactions.



1. You use AWS CloudFormation to install the solution into your environment. Your environment must meet prerequisites before deploying the solution. The provided CloudFormation template deploys an AWS CodePipeline that contains the Landing Zone Accelerator on AWS installation engine.
2. The **Installer** pipeline (`AWSAccelerator-InstallerStack`) functions separately from the **Core** pipeline. This way, you can update to future versions of the solution with a single parameter through the AWS CloudFormation console.

3. An AWS CodeBuild project functions as an orchestration engine to build and run the solution's AWS CDK application that deploys the Core pipeline (`AWSAccelerator-PipelineStack`) and its associated dependencies.
4. The solution deploys Amazon Simple Notification Service (Amazon SNS) topics that you can subscribe to for alerts on Core pipeline events, which can increase observability of your Core pipeline operations. Additionally, the solution deploys two AWS Key Management Service (AWS KMS) customer-managed keys to manage encryption at rest of Installer and Core pipeline dependencies.
5. The Core pipeline validates and synthesizes inputs and deploys additional CloudFormation stacks with AWS CDK. An Amazon Simple Storage Service (Amazon S3) bucket (`aws-accelerator-config`) stores the configuration files that the solution uses. These configuration files are the primary mechanism for configuring and managing the solution.
6. An AWS CodeBuild project compiles and validates the solution's AWS CDK application configuration.
7. Multiple AWS CodeBuild deployment stages deploy the resources that were defined in the solution configuration files to your multi-account environment. An optional manual review stage can be included, allowing you to view all the changes that these stages will apply.
8. The solution deploys resources that monitor AWS Control Tower lifecycle events to detect potential drift against a known good state (in other words, when the actual configuration of an infrastructure resource differs from its expected configuration). The solution also deploys resources that can automate the enrollment of new AWS accounts into your multi-account environment. When using AWS Control Tower with this solution, ensure that accounts and organizational units (OUs) within your AWS Control Tower environment are properly enrolled. You can manage this through the AWS Control Tower console.

   > **Note**
   >
   > We provide guidance in For AWS Organizations based installation (without AWS Control Tower) later in this document if you wish not to use AWS Control Tower.

9. The solution deploys centralized logging resources in the **Log Archive** account in your multi-account environment. This includes Amazon Kinesis resources to stream and ingest logs, AWS KMS keys to facilitate encryption at rest, and Amazon Simple Storage Service (Amazon S3) buckets as log storage destinations.
10. The solution provisions the audit account with resources to Amazon CloudWatch log groups to the centralized logging infrastructure in the LogArchive account.

> **Note**
>
> Initial deployment includes, at a minimum, account creation, drift detection, key management, and centralized logging infrastructure. These mandatory components are part of the core feature set of the solution and are described further in Architecture details. Remaining infrastructure that the solution deploys depends on the content of the user-defined configuration files.

Sample configurations

Landing Zone Accelerator on AWS includes example sample configurations that allow you to quickly deploy accounts, infrastructure, and security guardrails across your multi-account environment. The repository includes sample configurations and README.md files that provide guidance for configuring and deploying each of the six mandatory YAML files across both standard and AWS GovCloud (US) Regions. When used with this solution, the sample configurations deploy a baseline security and network architecture. Additional customization of the baselines will likely be required to suit the compliance needs of your business.

We built the sample configurations based on the authorized patterns and guidelines provided in the AWS Prescriptive Guidance Security Reference Architecture (SRA). This solution is a fully automated implementation of the AWS SRA and additionally provides you flexibility to customize your landing zone to suit your organizational security, networking, and compliance requirements.

Deployment pipelines

The AWS CloudFormation template deploys two CodePipeline pipelines, an installer and the core deployment pipeline, along with associated dependencies. This solution uses CodeBuild to build and deploy a series of CDK-based CloudFormation stacks that are responsible for deploying supported resources in the multi-account, multi-Region environment.

> **Note**
>
> AWS CloudFormation resources are created from AWS CDK constructs.

Installer (AWSAccelerator-InstallerStack)

This CloudFormation template deploys the following resources:

- A CodePipeline (AWSAccelerator-Installer) that's used to orchestrate the build and deployment of the AWSAccelerator-PipelineStack AWS CloudFormation template.
- A CodeBuild project is used as an orchestration engine within the pipeline to build the Landing Zone Accelerator on AWS source code and then synthesize and deploy the AWSAccelerator-PipelineStack CloudFormation template.
- An Amazon S3 bucket that's used for pipeline artifact storage.
- An AWS KMS key that's used to activate encryption at-rest for applicable resources deployed in AWSAccelerator-InstallerStack and AWSAccelerator-PipelineStack.
- Supporting AWS Identity and Access Management (IAM) roles for CodePipeline and CodeBuild to perform their actions.

Core (AWSAccelerator-PipelineStack)

This AWS CloudFormation stack is deployed by the AWS CDK with the following resources:

- A CodePipeline (AWSAccelerator-Pipeline) that's used for input validation, synthesis, and deployment of additional CloudFormation stacks by using the AWS CDK. The pipeline contains several stages that are discussed in Architecture details.
- Two CodeBuild projects. The projects are used in the pipeline stages to:
  - Build the Landing Zone Accelerator on AWS source code.
  - Run AWS CDK toolkit commands across the pipeline stages.
- An S3 bucket (awsaccelerator-config) that's used to store the configuration files that are used by the AWSAccelerator-Pipeline. These configuration files are your primary mechanism for configuration and management of the entire Landing Zone Accelerator on AWS solution.
- Two Amazon SNS topics are created and can be optionally subscribed to for AWS CodePipeline run notifications. No topic subscriptions are created by default. One Amazon SNS will notifies for all pipeline run events. The other notifies only on pipeline failure events.
- An optional third SNS topic is created if the **EnableApprovalStage** is set to Yes in **AWSAccelerator-InstallerStack**. Email address(es) listed in the **ApprovalStageNotifyEmailList** will be automatically subscribed to this topic.

- An AWS IAM service-linked role is created to allow AWS CodeStar notifications to publish CodePipeline pipeline run events to the Amazon SNS topics.
- A CloudWatch alarm is created to alarm on pipeline processing failures.
- An Amazon S3 bucket that's used for pipeline artifact storage.

Architecture details

This section describes the components and AWS services that make up this solution and the architecture details on how these components work together.

AWS services in this solution

For more information about services and features, see the Included services features and configuration references.

Core AWS services	Supporting AWS services
AWS CloudFormation	AWS Application Load Balancer
Amazon CloudWatch	AWS Autoscaling
AWS CodeBuild	AWS Backup
AWS CodeCommit	AWS Budgets
AWS CodePipeline	AWS CloudTrail
Amazon DynamoDB	AWS Config
Amazon EventBridge	AWS Control Tower
AWS IAM	AWS Cost and Usage Report
Amazon Kinesis	AWS EC2
AWS KMS	Amazon Elastic Block Store (Amazon EBS)
AWS Lambda	Amazon GuardDuty
Amazon S3	AWS Lambda
Amazon SNS	Amazon Macie
AWS Step Functions	AWS Network Firewall

Core AWS services	Supporting AWS services
	AWS Network Load Balancer
	AWS Organizations
	AWS Resource Access Manager (RAM)
	Amazon Route 53
	AWS Secrets Manager
	AWS Security Hub
	Amazon Virtual Private Cloud (Amazon VPC)

Installer pipeline

This pipeline runs the following stages:

1. **Source** - The Landing Zone Accelerator on AWS source code from the AWS Solutions GitHub repository
2. **Install** - A CodeBuild project is used to run the Landing Zone Accelerator on AWS pipeline CDK project, resulting in the deployment of the `AWSAccelerator-PipelineStack`

> **Note**
>
> The Landing Zone Accelerator on AWS Installer and Core pipelines are separate by design. The functionality of the `AWSAccelerator-InstallerStack` has been minimized to purely support deployment of the Core pipeline, `AWSAccelerator-Pipeline`. This will allow you to update your version of the Landing Zone Accelerator on AWS by updating a single parameter through the AWS CloudFormation update stack console. See Update the solution for more information.

Core pipeline

The solution uses CodeBuild as an orchestration engine for each action completed after the **Source** stage in this pipeline. These actions run a CDK application, which deploys CloudFormation stacks across each of the Landing Zone Accelerator on AWS solution-managed AWS accounts and Regions, unless otherwise specified:

1. **Source** - There are two source actions in this stage:
   - **Source** - The Landing Zone Accelerator on AWS source code from the AWS Solutions GitHub repository.
   - **Configuration** - The Landing Zone Accelerator on AWS configuration repository, named `aws-accelerator-config`.
2. **Build** - In this stage, the Landing Zone Accelerator on AWS source code is transpiled, including input and type validation for the configuration files.
3. **Prepare** - Any AWS accounts that are defined in the configuration are created and/or validated as necessary. If using AWS Control Tower, new AWS accounts are generated using the Control Tower Account Factory and enrolled into the proper AWS Organizations Organizational Unit (OU). We highly recommend that you use AWS Control Tower to generate and enroll new OUs. However, if you're deploying the solution in an AWS Region that isn't yet supported by AWS Control Tower, any OUs that are defined in the configuration are created and/or validated as necessary.
4. **Accounts** - Additional account validation occurs across the environment. All accounts in the configuration are checked to verify if they're part of the AWS Organization. Any configured AWS Organization Service Control Policies (SCPs) are also created and attached to configuration-specified deployment targets in this stage.
5. **Bootstrap** - AWS CDK bootstrap is run; this initializes the environment for CDK. A solution-specific CDK toolkit CloudFormation template (`AWS Accelerator-CDKToolkit`) is deployed to any AWS accounts and Regions that haven't been previously bootstrapped. If you want to deploy additional CDK applications, we recommend that you deploy your own CDK bootstrap template to avoid collisions with the Landing Zone Accelerator on AWS usage of CDK.
6. **Review (optional)** - An optional stage that can be turned on and off using the **EnableApprovalStage** configuration parameter on the `AWSAccelerator-InstallerStack` CloudFormation template. Turning on this option adds this stage to the pipeline, which includes the following actions:

- **Diff** - AWS CDK diff is run on the synthesized CloudFormation templates against each target account and Region. The result of the diff can be reviewed in the build logs of the CodeBuild project.
- **Approve** - A manual approval action. This is meant as a gate to review and approve/deny the changes represented in the **Diff** action. This action publishes to an SNS topic to notify configured email list(s) of the pending approval.

7. **Logging** - There are two actions in this stage:

- **Key** - The solution deploys two stacks during this stage:
  - **KeyStack** - Deploys a centralized AWS KMS key to the AWS account designated as the audit account in the configuration. This key is used in subsequent deployments to activate encryption at-rest for applicable resources. The solution also deploys Systems Manager Parameter Store parameters containing the value of the key Amazon Resource Names (ARNs) along with an IAM role that allows cross-account read access for the parameters.
  - **DependenciesStack** - Deploys resources that are required by the solution in subsequent pipeline stages, such as IAM roles for custom resources.
- **Logging** - This solution deploys a centralized logging Amazon S3 bucket, an Amazon Kinesis Data Stream, and Amazon Data Firehose in the AWS account designated as `LogArchive` in the configuration. The solution uses the Kinesis Data Stream as a destination for CloudWatch Logs groups in member accounts so that logs can be streamed to the central logs bucket via Firehose. Optionally, you can specify a dynamic partitioning configuration to map specific CloudWatch Log groups to specific Amazon S3 bucket prefixes.

  The solution creates Amazon S3 buckets for Amazon S3 server access logging in each AWS account and Region activated in the configuration. Optionally, you can activate the Amazon S3 Block Public Access feature at the account level and activate Systems Manager Session Manager logging for each configured account and Region.

The solution also deploys AWS KMS keys for Amazon S3, AWS Lambda, and CloudWatch Logs. These keys deploy in each AWS account and Region activated in the configuration. A solution-deployed Systems Manager automation document named `Accelerator-Put-S3-Encryption` uses the AWS KMS key for Amazon S3 to encrypt any Amazon S3 buckets that were created without encryption. The solution uses the AWS KMS key for Lambda to invoke Lambda environment variable encryption, and it uses the AWS KMS key for CloudWatch Logs to encrypt solution-created CloudWatch Logs groups. . **Organization** - Deployment of AWS Organization-wide resources. These resources are deployed in the Region designated as the organization's home

Region in the organization's management account. This includes actions such as activating trusted services, creating AWS Organizations tagging and backup policies, creating report definitions for AWS Cost and Usage Report, and AWS Budgets. . **Security_Audit** - Deployment of resource dependencies for centralized security services in the AWS account designated as the audit account in the configuration. This includes S3 buckets and/or configurations for Amazon Macie, Amazon GuardDuty, AWS Security Hub, and Systems Manager automation documents. . **Deploy** - The following actions are completed in this stage to deploy the remaining architecture as defined in the configuration files. Refer to our sample configuration as a reference to get started:

+ **Network_Prepare - Network resources that subsequent networking stacks must reference are created in this action. This includes AWS Transit Gateway and AWS Resource Access Manager (AWS RAM) shares, if configured. Security** - Member account security services are configured. **Operations - Users, groups, and roles are deployed. IAM Security Assertion Markup Language (SAML) identity provider configuration is also deployed, if configured. Network_VPCs** - Three stacks are deployed during this stage, each related to VPC networking:

+ * **NetworkVpcStack - VPCs, subnets, route tables, security groups and other associated resources are deployed. AWS Transit Gateway attachments are created, if configured.** * **NetworkVpcEndpointsStack** - VPC endpoints, including Route 53 resolver endpoints and AWS Network Firewall endpoints are deployed. * **NetworkVpcDnsStack - Route 53 private hosted zones and resolver rules are deployed. Security_Resources** - Additional member account security services such as AWS Config, CloudWatch metrics, and alarms are deployed. * *Network_Associations** - The solution deploys two stacks during this stage, each related to network associations that depend on resources created in the **Network_VPCs** stage:

+ * **NetworkAssociationsStack - Network associations that depend on Amazon VPC resources to be created, such as AWS Transit Gateway VPC associations, are deployed.** * **NetworkAssociationsGwlbStack** - Network associations that depend on Gateway Load Balancers to be created, such as Gateway Load Balancer VPC endpoints, are deployed. **Customizations (optional) - The solution deploys custom applications, CloudFormation stacks, and CloudFormation stacksets that are configured in the `customizations-config.yaml` file. Finalize** - If using the account quarantine feature for new account creation, the quarantine SCP is removed during this action.

Pipeline artifact Amazon S3 buckets

Two Amazon S3 buckets are created with the solution by default. These buckets are used to host artifacts for the CodePipeline pipelines. If desired, you can delete artifacts after the pipeline

invocations have completed. However, don't delete the buckets themselves because this breaks the functionality of the pipelines. For more information, refer to Input and output artifacts in the *AWS CodePipeline User Guide*.

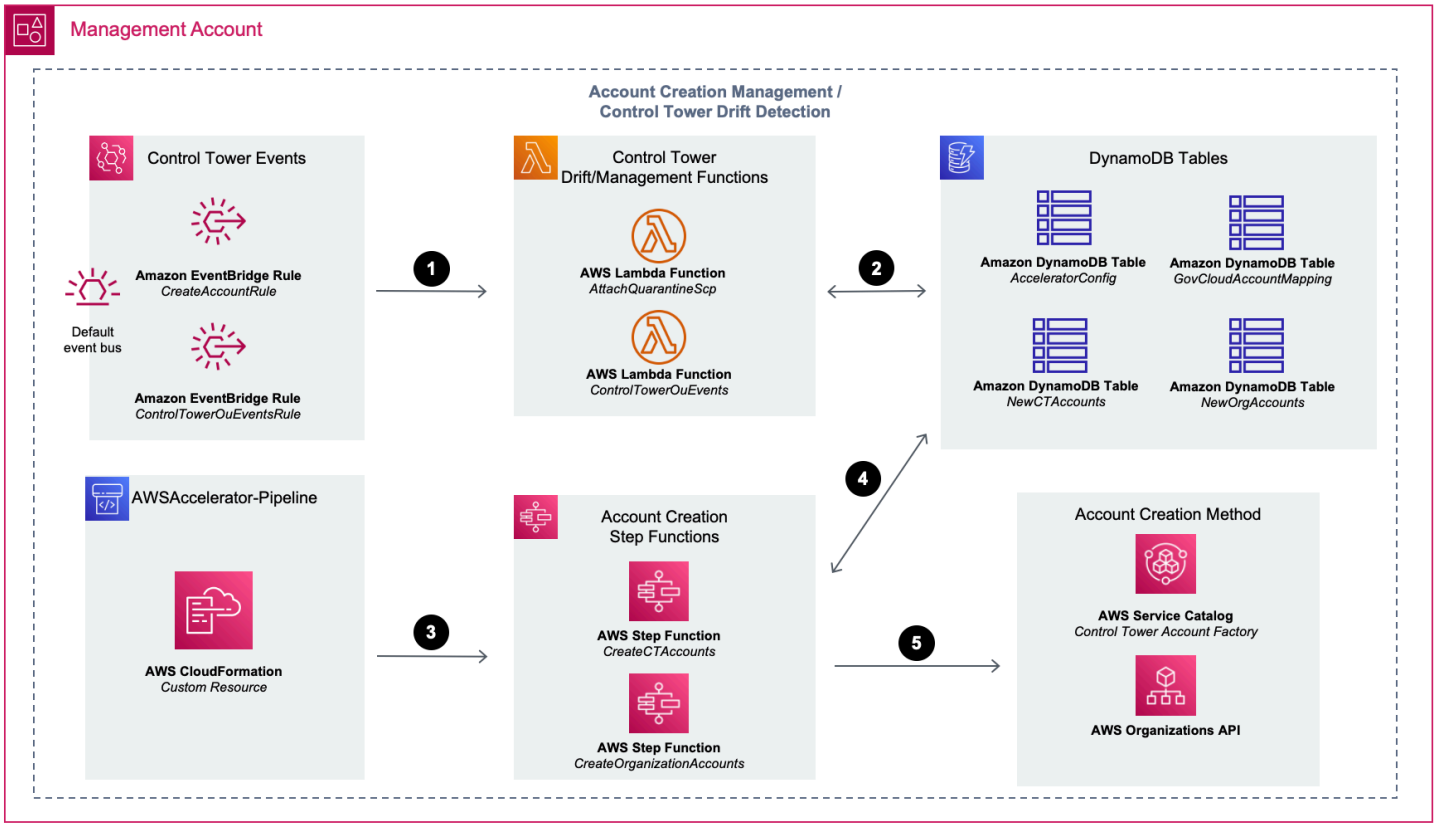
Amazon SNS topics

Two Amazon SNS topics are created with the solution by default. One topic is to notify on all `AWSAccelerator-Pipeline` pipeline events. The second notifies only on `AWSAccelerator-Pipeline` pipeline failures. You can choose to subscribe to these topics to increase the observability of your pipeline operations. For more information, refer to Subscribing to an Amazon SNS topic in the *Amazon SNS Developer Guide*.

An optional third Amazon SNS topic is created if the **EnableApprovalStage** parameter is set to `Yes` in the **AWSAccelerator-InstallerStack**. You can provide a comma-delimited list of email addresses in the **ApprovalStageNotifyEmailList** parameter to automatically subscribe to this Amazon SNS topic.

Account creation and drift detection

AWS account creation and management workflow with EventBridge, Lambda, DynamoDB, and other services.



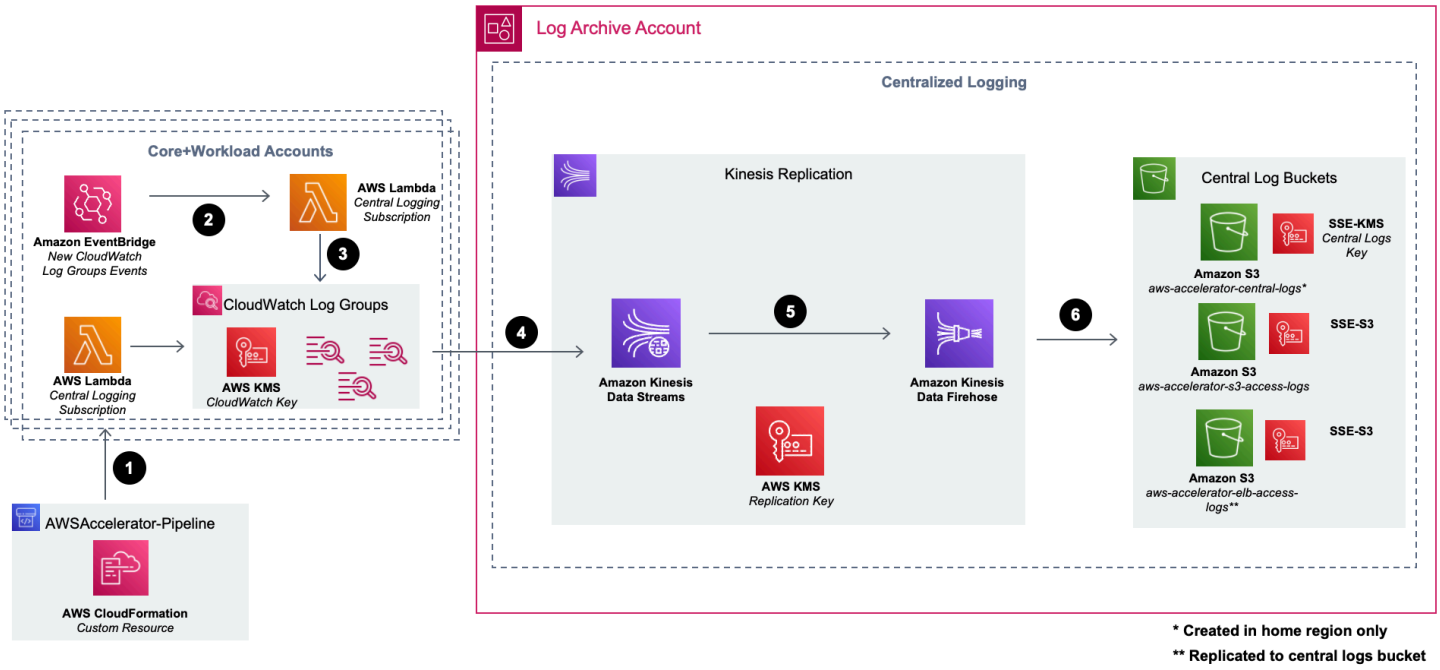
1. The solution deploys Amazon EventBridge rules that monitor for AWS Control Tower lifecycle events. These rules invoke AWS Lambda functions that perform different actions based on the lifecycle event. The solution uses the `AttachQuarantineScp` function to attach an AWS Organizations SCP to newly-enrolled accounts, if configured. The solution uses the `ControlTowerOuEvents` function to detect changes made to OUs in the multi-account environment.
2. The Lambda functions have access to Amazon DynamoDB tables that contain stateful information about the multi-account environment. The functions use this data to validate changes made to the environment against a known good state.
3. The account creation workflow is invoked by the **Prepare** stage of the `AWSAccelerator-Pipeline` when a new account is added to the `accounts-config.yaml` file. Two AWS Step Functions state machines handle this workflow: one for AWS Control Tower-based landing zones and the other for AWS Organizations-based landing zones.
4. The state machines have access to DynamoDB tables that contain stateful information about the multi-account environment. This allows the underlying Lambda functions to validate the environment and store the environment's state in the DynamoDB tables.

5. The state machines initiate the account creation process if a new account is added to the solution configuration. The account creation workflow is dependent on the type of landing zone that the solution has been deployed to. For AWS Control Tower-based landing zones, the solution leverages the Control Tower Account Factory Service Catalog portfolio to provision a new account. For AWS Organizations-based landing zones, the Organizations API invokes account creation. We provide configuration toggles to differentiate the type of landing zone in the `global-config.yaml` file.

> **Note**
>
> Account creation is an asynchronous process, so the state machine workflow is used to periodically check the status of the Account Factory or Organizations-based account creation. As such, the state machine pauses the pipeline stage progression until the account creation succeeds or fails.

Centralized logging

AWS log archiving architecture with EventBridge, Lambda, Kinesis, and S3 components.



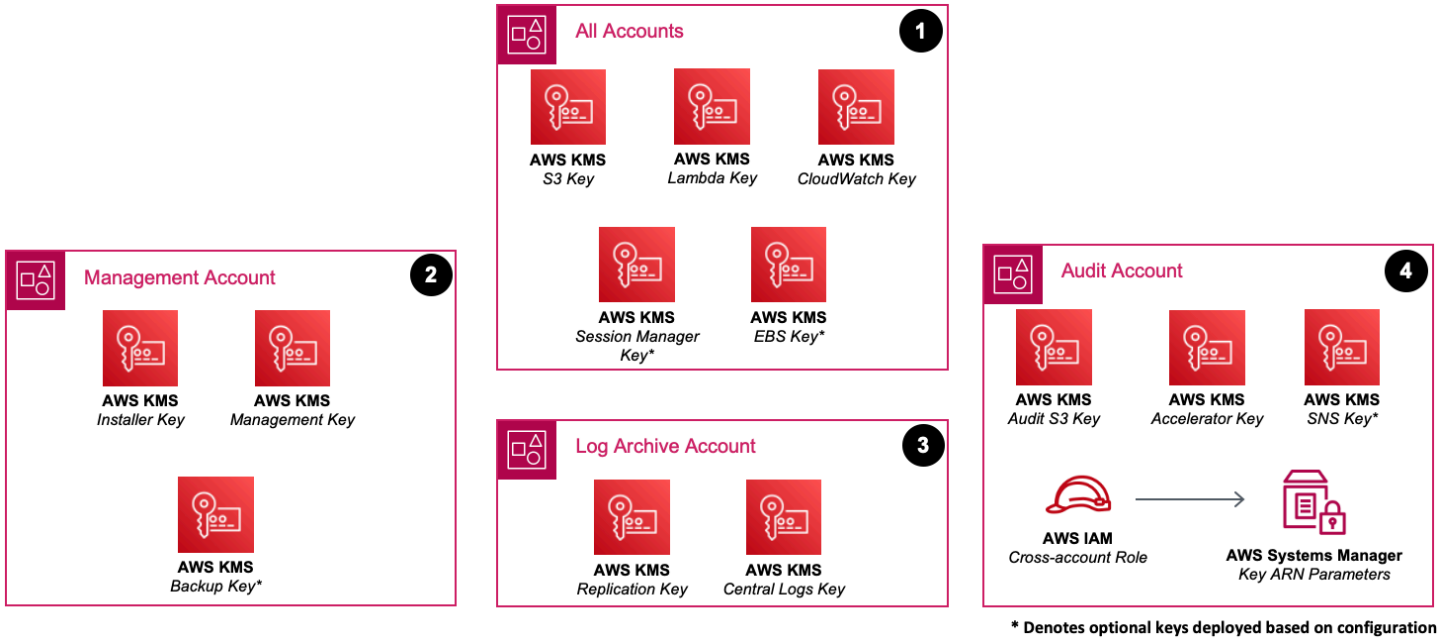
1. A CloudWatch log group update workflow runs during the **Logging** stage of the pipeline. A CloudFormation custom resource invokes a Lambda function that updates existing log groups

to the increase log retention if it's less than the solution log retention period, CloudWatch AWS KMS key, and subscription filter. The destination for the subscription filter is an Amazon Kinesis Data Stream deployed to the **Log Archive** account. For example, before solution is installed if there are existing log groups `LogGroupA` with 5 years retention and `LogGroupB` with 1 week retention. The solution is deployed with 1 year retention in global-config.yaml under `cloudwatchLogRetentionInDays`. Then `LogGroupA` will be unaffected with the update since 5 years is greater than 1 year but `LogGroupB` retention will change to 1 year. If in a subsequent update or initial update, solution is deployed with 10 years retention in global-config.yaml under `cloudwatchLogRetentionInDays`, then both log groups will change retention to 10 years.

2. An EventBridge rule monitors for new CloudWatch log groups created in core and workload accounts.
3. When new log groups are created, the EventBridge rule invokes a Lambda function that updates the log group with the configured log retention period, CloudWatch AWS KMS key, and subscription filter. The destination for the subscription filter is the Kinesis Data Stream deployed to the **Log Archive** account. Since log replication to s3 is active, any CreateLogGroup API call will get the retention specified in global-config.yaml under `cloudwatchLogRetentionInDays`. So if `cloudwatchLogRetentionInDays` is set to 1 week and new log group is created with 5 year retention then it will change to 1 week. The solution ensures that entire organization's CloudWatch retention for any new log group is compliant under the value specified in global-config.yaml under `cloudwatchLogRetentionInDays`.
4. Log groups stream their logs to the Kinesis Data Stream. The data stream is encrypted at rest with the replication AWS KMS key.
5. A delivery stream is configured with the Kinesis Data Stream and Firehose, allowing the logs to be transformed and replicated to Amazon S3.
6. The destination of the Firehose delivery stream is the `aws-accelerator-central-logs` Amazon S3 bucket. This bucket is encrypted at rest with the central logging AWS KMS key. In addition, the `aws-accelerator-s3-access-logs` and `aws-accelerator-elb-access-logs` buckets are encrypted at rest with Amazon S3-managed server-side encryption (SSE-S3) because these services don't support customer-managed AWS KMS keys. Logs delivered to the `aws-accelerator-elb-access-logs` bucket replicate to the central logs bucket with Amazon S3 replication.

Key management

Architecture diagram showing key management for accounts.



The solution uses AWS KMS keys to provide encryption at rest capabilities for resources deployed by the solution. Some AWS KMS keys are deployed to every account and Region managed by the solution, while others are centralized in a single core account.

All accounts

- **Amazon CloudWatch key** - used to encrypt CloudWatch Logs groups created by the solution
- **Amazon S3 key** - used to encrypt Amazon S3 buckets created by the solution
- **AWS Lambda key** - used to encrypt environment variables for Lambda functions created by the solution
- **AWS Systems Manager Session Manager key (optional)** - used to encrypt Session Manager sessions if Session Manager logging is activated in the `global-config.yaml` file
- **Amazon Elastic Block Store (Amazon EBS) key (optional)** - used for default encryption of Amazon EBS volumes if activated in the `security-config.yaml` file

Management account

- **Installer key** - created by `AWSAccelerator-InstallerStack` to activate encryption at rest for Installer pipeline dependencies
- **Management key** - created by `AWSAccelerator-PipelineStack` to activate encryption at rest for Core pipeline dependencies

- **AWS Backup key (optional)** - used to activate encryption at rest for AWS Backup vault if configured in the `organization-config.yaml` file

LogArchive account

- **Central logs key** - used to encrypt the `aws-accelerator-central-logs` Amazon S3 bucket

> **Note**
>
> This key is distinct from the per-account/Region key because additional services such as Config, CloudTrail, and log delivery require access. Macie, GuardDuty, and Audit Manager might also require access, if activated.

- **Log replication key** - used to encrypt a Kinesis Data Stream used as a destination for log replication from CloudWatch Logs to Amazon S3

Audit account

- **Accelerator KMS key** - used by the entire organization to decrypt AWS Systems Manager parameters (SSM parameters) stored centrally in the **Audit** account
- **Audit S3 key** - used to encrypt authorize-created CloudTrail Amazon S3 buckets and Audit Manager publishing bucket, if configured
- **Amazon SNS key (optional)** - used to encrypt Amazon SNS topics created to alert on security events, if configured

Plan your deployment

This section describes the Region, cost, security, quota, and other considerations for planning your deployment.

Supported AWS Regions

This solution uses the AWS Control Tower and AWS Organizations services, which aren't currently available in all AWS Regions. We recommend using AWS Control Tower and AWS Organizations when launching this solution in an AWS Region where these services are available. For the most current availability of AWS services by Region, refer to the AWS Regional Services List.

To deploy this solution to AWS GovCloud(US) Regions, see Deploy to AWS GovCloud(US) Regions.

To deploy this solution to a Region that is deactivated by default, see Opt-in Regions.

Cost

You are responsible for the cost of the AWS services used while running this solution. As of this revision, the cost for running this solution using the Landing Zone Accelerator on AWS sample configuration with AWS Control Tower in the US East (N. Virginia) Region within a non-critical sandbox environment with no activity or workloads is approximately **$430.22 (USD)** each month.

We recommend creating a budget through AWS Cost Explorer to help manage costs. Prices are subject to change. For full details, refer to the pricing webpage for each AWS service used in this solution.

Sample cost table

The following table provides a sample cost breakdown for deploying this solution with the default parameters in the US East (N. Virginia) Region, with no activity, for one month.

AWS service	Dimensions	Monthly cost [USD]
AWS CloudTrail	• 4 million read/write management events • 4.7 million paid events	$99.00

AWS service	Dimensions	Monthly cost [USD]
	• Insights turned off (sample configuration)	
AWS Config	• 2,000 AWS Config items • 17,000 AWS Config rule evaluations	$23.00
AWS KMS	• 43 customer managed keys (CMKs) • 521,288 symmetric requests	$44.56
Amazon Kinesis	• 4.5 million PUT payload units • 744 `Shard` hours	$11.22
Amazon Data Firehose	33,735 records x 5 KB	$4.66
Amazon S3	• 7 GB standard storage • 505,000 PUT, COPY, POST, or LIST requests • 265,000 GET, SELECT, and other requests	$2.79
Amazon VPC	• 2 Transit Gateway attachments with 3 GB each month ($73.12) • 14 endpoints and 1 Availability Zone at 3 GB each month ($102.23)	$175.35
Amazon CloudWatch	• 12 metrics • 12 GB standard logs • 2 GB logs delivered with 1-month retention	$15.71

AWS service	Dimensions	Monthly cost [USD]
AWS Security Hub	• 1 account • 30,000 security checks	$30.00
Amazon GuardDuty	• 2.4 million management event analysis • 2.4 million Amazon S3 data event analysis	$11.52
Amazon Route 53	6 hosted zones	$3.00
Amazon Macie	16 Amazon S3 buckets	$1.60
AWS Secrets Manager	2 secrets for 30 days	$0.81
AWS CodePipeline	2 pipelines each month	$1.00
AWS CodeBuild	60 builds a month x 5 minutes	$6.00
Total monthly cost		**$430.22**

Note

Data transfer, AWS CodeArtifact, Amazon Detective, Amazon DynamoDB, AWS Lambda, AWS Service Catalog, Amazon Simple Notification Service (Amazon SNS), Amazon Simple Queue Service (Amazon SQS), AWS Step Functions, and AWS Systems Manager are priced at the Free Tier or less than $0.01 each month.

Security

When you build systems on AWS infrastructure, security responsibilities are shared between you and AWS. This shared model reduces your operational burden because AWS operates, manages, and controls the components including the host operating system, the virtualization layer, and the physical security of the facilities in which the services operate. For more information about AWS security, visit AWS Cloud Security.

IAM roles

AWS Identity and Access Management (IAM) roles allow customers to assign granular access policies and permissions to services and users on the AWS Cloud. This solution creates IAM roles that grant the solution's CodePipeline pipelines read/write access to their respective artifact S3 buckets, source code repositories, and run CodeBuild projects. Additional IAM roles are created that grant CodeBuild projects to write to Amazon CloudWatch Logs log groups and create Regional resources.

AWS KMS keys

AWS KMS helps you create and manage cryptographic keys and control their use across a wide range of AWS services and in your applications. This solution uses AWS KMS keys to turn on encryption at rest for the applicable services it deploys. In a default installation, these keys will rotate automatically once per year. More information about the key management infrastructure for this solution is outlined in Architecture details.

Quotas

Service quotas, also referred to as limits, are the maximum number of service resources or operations for your AWS account. Make sure you have sufficient quota for each of the services implemented in this solution. For more information, see AWS service quotas. To view the service quotas for all AWS services in the documentation without switching pages, view the information in the Service endpoints and quotas page in the PDF instead.

Deployment options

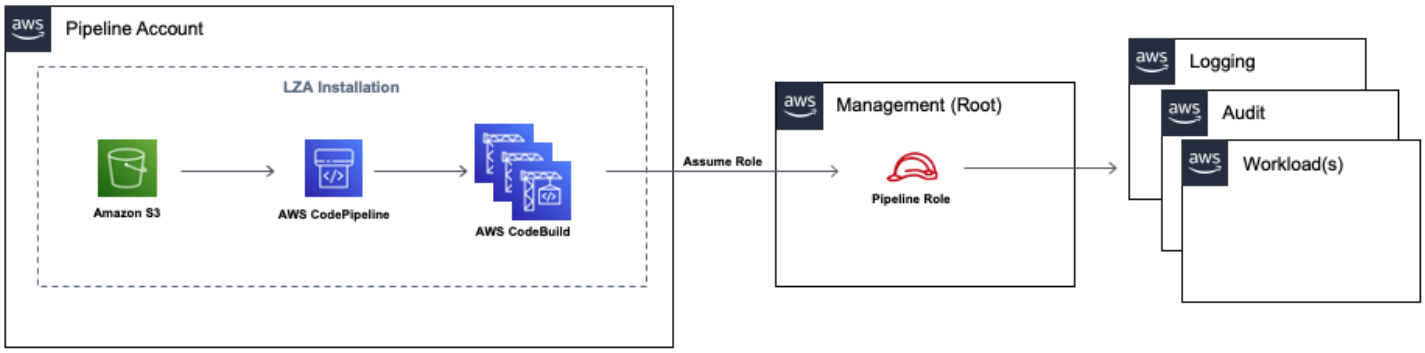
Before deploying the Landing Zone Accelerator on AWS, you need to choose a method to centralize the management of resources provisioned by this solution. You can use either AWS Control Tower or AWS Organizations for the management capabilities. We strongly recommend AWS Control Tower if you're deploying in a Region where it's supported, as it automatically provisions best practice security configurations and guardrails across your multi-account environment.

> **Note**
>
> If you want to deploy the solution in an existing multi-account environment, refer to Prerequisites and Working with existing landing zones before deploying the solution.

External pipeline deployment

In a default Landing Zone Accelerator on AWS installation, the CodePipeline and S3 bucket deploys into the AWS Organizations management account. You may want to deploy and operate these components in a member AWS account to limit access to the management account. This solution supports this model with an optional pipeline deployment account.

External pipeline deployment



Follow these instructions to implement this pattern:

1. Select an AWS account for the pipeline deployment account. We recommend having the account as a member of the AWS Organizations environment.
2. Create a new IAM role in the AWS Organizations management account that allows access from the pipeline deployment account. `AcceleratorPipelineDeploymentRole` is the preferred name for this role.
3. Update the trust policy of the `AcceleratorPipelineDeploymentRole` to allow access from the pipeline deployment account:

```
{
     "Version": "2012-10-17",
     "Statement": [
         {
             "Effect": "Allow",
             "Principal": {
"arn:aws:iam::<PIPELINE_DEPLOYMENT_ACCOUNT_ID>:root"
     },
    "Action": "sts:AssumeRole"
    "Condition": {
 "StringLike": {
```

```
  "aws:PrincipalArn": "arn:aws:iam::<PIPELINE_DEPLOYMENT_ACCOUNT_ID>:role/
<AcceleratorQualifier-*>"
     }
      }

       }
    ]
  }
```

1. Attach the `AdministratorAccess` AWS managed IAM policy to the role.

> **Note**
>
> By default, AWS IAM roles with prefix AcceleratorQualifier in the pipeline account are used by AWS CodeBuild to assume role in the management account and deploy resources. To protect these roles, you should implement additional security measures, such as Service control policies (SCPs).

After you create the IAM role in the management account, synthesize the Landing Zone Accelerator on AWS installer template configured for external deployments by following these instructions:

1. Clone or download the latest release of the Landing Zone Accelerator on AWS source code.
2. Navigate to the `source` folder:

```
cd landing-zone-accelerator-on-aws/source
```

3. Install dependencies and build the source code:

```
yarn install && yarn build
```

4. Navigate to the installer folder:

```
cd packages/\@aws-accelerator/installer/
```

5. Synthesize the installer template by running:

```
cdk synth --context use-external-pipeline-account=true
```

6. Retrieve the synthesize template named `AWSAccelerator-InstallerStack.template.json` from the `cdk.out` directory.
7. Use this template to create the `AWSAccelerator-Installer` CloudFormation stack in the external deployment account.
8. The deployment now follows the same process as the standard deployment process with the addition of the following parameters:
   a. **AcceleratorQualifier** - Names the resources in the external deployment account. This must be unique for each Landing Zone Accelerator on AWS pipeline created in a single external deployment account, for example "env2" or "app1." Do not use "aws-accelerator" or a similar value that could be confused with the prefix.
   b. **ManagementAccountId** - This is the AWS account ID of the AWS Organizations management account.
   c. **ManagementAccountRoleName** - This is the name of the IAM role used to access the management account from the external deployment account.

Source code location

In a default Landing Zone Accelerator on AWS deployment, CodePipeline retrieves the source code from the solution's GitHub repository. You may want to instead store the source code in Amazon S3 to use only Amazon-provided products. This solution supports this operating model by uploading the LZA source code to an existing S3 bucket before deploying the solution.

Follow these instructions to implement this pattern:

1. Create an S3 bucket with versioning enabled. This bucket should be created in the same AWS account and region you plan to deploy the Landing Zone Accelerator on AWS solution.
2. Clone or download the latest release of the Landing Zone Accelerator on AWS source code.
3. Navigate to the source folder:

```
cd landing-zone-accelerator-on-aws/source
```

4. Compress the contents of the `source` folder into a new zip archive file.
5. Upload the zip archive to the S3 bucket created in Step 1.
6. Install dependencies and build the source code:

```
yarn install && yarn build
```

7. Navigate to the installer folder:

```
cd packages/\@aws-accelerator/installer/
```

8. Synthesize the installer template by running:

```
cdk synth -context use-s3-source=true
```

> **Note**
>
> If your S3 bucket is encrypted with KMS (S3-KMS), you must pass the KMS key ID when synthesizing the template:
>
> ```
> cdk synth -context use-s3-source=true --context
> s3-source-kms-key-arn=arn:aws:kms:us-east-1:000000000000:key/aaaaaaaa-1111-
> bbbb-2222-cccccc333333
> ```

1. Retrieve the synthesize template named `AWSAccelerator-InstallerStack.template.json` from the cdk.out directory.
2. Use this template to create the `AWSAccelerator-Installer` CloudFormation stack in the account and region the S3 bucket was created in.
3. The deployment now follows the same process as the standard deployment process with the addition of the following parameters:
   - **RepositoryBucketName** - The name of the S3 bucket used to contain the source code.
   - **RepositoryBucketObject** - The S3 object key of the source code uploaded in Step 5.
   - **RepositoryBucketKmsKeyArn** - (OPTIONAL) The ARN of the KMS key used to encrypt the S3 bucket.

Mandatory accounts

The Landing Zone Accelerator on AWS builds on top of an existing AWS Control Tower or AWS Organizations multi-account structure. If using AWS Control Tower, this solution uses the same

initial accounts that are generated by deploying the Control Tower Landing Zone. If using AWS Organizations only in a Region without AWS Control Tower, the following mandatory accounts must be created:

- **Management account** - This account is designated when first creating an AWS Organization. It's a privileged account where all AWS Organizations global configuration management and billing consolidation occurs.
- **LogArchive account** - This account is used for centralized logging of AWS service logs and AWS CloudTrail trails.
- **Audit account** - This account is used to centralize all security operations and management activities. This account is typically used as a delegated administrator of centralized security services such as Amazon Macie, Amazon GuardDuty, and AWS Security Hub.

Administrative role

Landing Zone Accelerator on AWS uses an IAM role with administrative privileges to manage the orchestration of resources across the environment. We recommend you activate AWS Control Tower and use the `AWSControlTowerExecution` role. You can also leverage other existing cross-account access roles such as `OrganizationAccountAccessRole`, which is the default cross-account role that's utilized by AWS Organizations.

If you prefer using custom roles, a role with administrative privileges must be deployed in each member account managed by the Landing Zone Accelerator on AWS. These roles must have a trust relationship defined that grants the `sts:AssumeRole` permission to the IAM service role for the Landing Zone Accelerator on AWS CodeBuild projects. The following demonstrates the ARN changes based on the partition of the resource:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:$PARTITION:iam::$MANAGEMENT_ACCOUNT_ID:root"
            },
            "Action": "sts:AssumeRole"
        }
    ]
```

```
}
```

Customizing the solution

This solution deploys an S3 bucket with six customizable YAML configuration files contained in a single ZIP archive.. The YAML files are pre-populated with a minimal configuration for the solution. You can create an optional seventh configuration file (`customizations-config.yaml`) to define customizations to the core solution. You can customize the YAML configuration files to deploy additional resources and infrastructure to the solution environment. Refer to Using configuration files for more information, and our sample configuration for an example of sample implementation.

Support for specific regions and industries

You can use this solution to support alignment with specific regional and industry guidelines. See the Landing Zone Accelerator on AWS solution page for more information.

Deploy the solution

This solution uses AWS CloudFormation templates and stacks to automate its deployment. The CloudFormation template specifies the AWS resources included in this solution and their properties. The CloudFormation stack provisions the resources described in the template.

Deployment process overview

Before you launch the solution, review the cost, architecture, network security, and other considerations discussed earlier in this guide.

Time to deploy: Approximately eight minutes for the `AWSAccelerator-Installer` CloudFormation stack and 45 minutes for the initial run of the `AWSAccelerator-Pipeline` pipeline.

> **Note**
>
> If you have previously deployed this solution, refer to Update the solution for update instructions.

Use the following steps to deploy this solution on AWS. For detailed instructions, follow the links for each step.

Step 1. Launch the stack

- Launch the AWS CloudFormation template into your AWS account.
- Review the templates parameters and enter or adjust the default values as needed.

Step 2. Await initial environment deployment

- Await successful completion of `AWSAccelerator-Pipeline` pipeline.

Step 3. Update the configuration files

- Navigate to the stored Landing Zone Accelerator on AWS configuration files.

- Update the configuration files to match the desired state of your environment.
- Release a change manually to the `AWSAccelerator-Pipeline` pipeline.

> **Important**
>
> This solution includes an option to send anonymized operational metrics to AWS. We use this data to better understand how customers use this solution and related services and products. AWS owns the data gathered though this survey. Data collection is subject to the AWS Privacy Notice.
>
> To opt out of this feature, download the template, modify the AWS CloudFormation mapping section, and then use the AWS CloudFormation console to upload your updated template and deploy the solution. For more information, refer to the Anonymized data collection section of this guide.

Prerequisites

You must meet the following prerequisites before launching the stack.

Activate a multi-account management solution

Landing Zone Accelerator on AWS solution can create, update, or reset an AWS Control Tower Landing Zone. When enabled, the solution will deploy AWS Control Tower in the home Region.

For AWS Control Tower based installation

Auto-deploy AWS Control Tower by the solution (recommended)

Using the Landing Zone Accelerator on AWS solution, you can create, update, or reset an AWS Control Tower Landing Zone. It is possible to maintain the AWS Control Tower Landing Zone using the Landing Zone Accelerator solution. When the installer stack of the solution is deployed with the `ControlTowerEnabled` parameter set to `Yes`, then the Landing Zone Accelerator solution will deploy the AWS Control Tower Landing Zone with the most recent version available.

The Landing Zone Accelerator solution can deploy AWS Control Tower Landing Zone when the following prerequisites are met.

- Configured AWS Organizations with all feature enabled in management account.

Create AWS Organization and verify that your own the email address is provided for the management account in the organization. In order to learn more about setting up an AWS organization, you may refer to this Creating an organization in the *AWS Organizations__User Guide*.

> **Note**
>
> In the event that AWS Organizations has been configured, but not all features have been enabled, the solution will enable all features for your organization.

- There are no AWS services enabled for AWS Organizations.
- There are no organization units created in the AWS Organizations.
- The only AWS account in the AWS Organization is the management account.
- The management account does not have AWS IAM Identity Center configured.
- The following AWS Control Tower service roles are not preset in the management account.
  - AWSControlTowerAdmin
  - AWSControlTowerCloudTrailRole
  - AWSControlTowerStackSetRole
  - AWSControlTowerConfigAggregatorRoleForOrganizations

Landing Zone Accelerator performs the following prerequisites before deploying AWS Control Tower Landing Zone. This document provides more information about AWS Control Tower prerequisites. The solution will not perform any of the prerequisites if there is an existing AWS Control Tower Landing Zone.

- Deploy following AWS Control Tower service roles in the management account:
  - AWSControlTowerAdmin
  - AWSControlTowerCloudTrailRole
  - AWSControlTowerStackSetRole
  - AWSControlTowerConfigAggregatorRoleForOrganizations
- Deploy AWS KMS CMK with alias `alias/aws-controltower/key` in the management account home Region.

- Create shared accounts (`LogArchive` and `Audit`) and invite to AWS Organizations.
- Deploy AWS Control Tower Landing Zone in the management account home Region.

> **Note**
>
> Landing Zone Accelerator on AWS uses the AWS Control Tower API to create and manage the AWS Control Tower Landing Zone.

> **Important**
>
> The AWS Console should be used to enable or disable the Region deny property for your AWS Control Tower Landing Zone. Currently, the Landing Zone Accelerator solution does not support the modification of the Region deny feature. Due to the fact that the Landing Zone Accelerator may deploy certain global AWS services, such as AWS IAM and AWS Organizations, the solution will add the global Region to the list of governed Regions in the AWS Control Tower if the home Region of the Landing Zone Accelerator is not the same as the global Region.

Manually deploy AWS Control Tower

To set up AWS Control Tower, refer to Getting started with AWS Control Tower in the *AWS Control Tower User Guide*.

> **Note**
>
> If you're using AWS Control Tower, we strongly recommended creating an AWS KMS customer managed key before deploying your landing zone. This AWS KMS key is used by services that AWS Control Tower manages to apply encryption at rest to sensitive log files. For more information on activating encryption for AWS Control Tower, see Configure your shared accounts and encryption.
> If you're deploying a new AWS Control Tower landing zone, you can add the prerequisite **Infrastructure** OU during the initial setup wizard. By default, the landing zone deploys with an additional **Sandbox** OU. You can rename this OU to **Infrastructure** if desired. Alternatively, you can create the **InfrastructureOU** after the landing zone is provisioned.

For more information about customizing the additional OU created during Control Tower setup, see Step 2b. Configure your organizational units (OUs) in the *Control Tower User Guide*.

For AWS Organizations based installation (without AWS Control Tower)

To set up AWS Organizations, refer to Getting started with AWS Organizations in the *AWS Organization User Guide*.

Ensure the Mandatory accounts are created. The Landing Zone Accelerator on AWS requires these three accounts at minimum to successfully deploy to your environment.

For more information on managing accounts in an AWS Organization, refer to Managing the AWS accounts in your organization in the *AWS Organization User Guide*.

Update AWS CodeBuild concurrency quota

Follow this procedure to check your current CodeBuild concurrency quota.

1. Navigate to the Service Quotas console in the account and Region for which you will deploy the Landing Zone Accelerator on AWS solution.
2. In the navigation pane, choose **AWS services**.
3. Search for then select AWS CodeBuild.
4. Select **Concurrently running builds for Linux/Large environment**.
5. If the value under **Applied quota value** is less than 3, select the quota link. Otherwise, skip the remaining steps.
6. Choose **Request increase at account-level**. In the **Increase quota value** box, enter 3 or more as the new quota value.
7. Choose **Request**. Ensure this quota increase request has been approved prior to deploying the solution. You can view your request status by choosing **Quota request history** in the navigation sidebar.

Ensure your global Region is accessible

Some AWS services and features apply configurations to your accounts at a global level rather than a regional level. In addition to the Regions that you enable in the solution configuration files; this

solution requires access to the Region where global service API endpoints are hosted. The global Region depends on the AWS partition you will be deploying the solution to.

AWS partitions and their corresponding global Region

AWS Partition	Global Region
Standard (aws)	`us-east-1`
GovCloud US (aws-us-gov)	`us-gov-west-1`
China (aws-cn)	`cn-northwest-1`

Important

Ensure that you don't have any existing AWS Organizations service control policies and/or Control Tower Region deny settings configured in your environment that would block access to the global Region listed above. You might experience Core pipeline failures if you do not allow access to this Region.

Create a GitHub personal access token and store in Secrets Manager

You require a GitHub access token to access the Landing Zone Accelerator on AWS code repository. Instructions on how to create a personal access token are located on GitHub Docs.

Note

The GitHub access token must have `public_repo` permissions.

Store the personal access token in Secrets Manager as plain text in the home Region. Name the secret `accelerator/github-token` (case sensitive).

With the AWS Management Console in the home Region:

1. Store a new secret, and select **Other type of secrets**, **Plaintext**.

2. Paste your secret with no formatting, leading, or trailing spaces (completely remove the example text).
3. Select an encryption key.
4. Set the secret name to `accelerator/github-token` (case sensitive).
5. Select **Disable rotation**.

AWS CloudFormation template

You can download the CloudFormation template for this solution before deploying it.

View template

AWSAccelerator-InstallerStack.template - Use this template to launch the solution and all associated components. The default configuration deploys the core and supporting solutions found in the Architecture overview. Manual changes to the template are strongly discouraged.

Before you launch the solution, review the cost, architecture, network security, and other considerations discussed earlier in this guide.

> **Note**
>
> - AWS CloudFormation resources are created from AWS CDK constructs.
> - If you have previously deployed this solution, refer to Update the solution for update instructions.

Step 1. Launch the stack

This automated AWS CloudFormation template deploys the Landing Zone Accelerator on AWS in the AWS Cloud. You must complete the applicable steps in Prerequisites before launching the stack.

1. Sign into AWS Management Console and select the button to launch `AWSAccelerator-InstallerStack` CloudFormation template.

   Launch solution

2. The template launches in the US East (N. Virginia) Region by default. To launch the solution in a different AWS Region, use the Region selector in the console navigation bar.

> **Note**
>
> This solution recommends using the AWS Control Tower service, which isn't currently available in all AWS Regions. We recommend launching this solution in an AWS Region where AWS Control Tower is available. For the most current availability by Region, refer to the AWS Regional Services List.

3. On the **Create stack** page, verify that the correct template URL is in the **Amazon S3 URL** text box and choose **Next**.
4. On the **Specify stack details** page, assign a name to your solution stack. We recommend you name your stack `AWSAccelerator-InstallerStack` to match the naming convention used by additional stacks that will be created by the Landing Zone Accelerator on AWS. For information about naming character limitations, refer to IAM and STS quotas in the *AWS Identity and Access Management User Guide*.
5. Under **Parameters**, review the parameters for this solution template and modify them as necessary. This solution uses the following default values.

Parameter	Default	Description
Source	`github`	Specify the git host.
Repository Owner	`awslabs`	The owner of the git repository hosting the solution code.
Repository Name	`landing-zone-accelerator-on-aws`	The name of the git repository hosting the solution code.
Branch Name	`<requires input>`	The name of the git branch to use for installation. NOTE: The Branch Name parameter defaults to the latest release branch name. To determine the branch

Parameter	Default	Description
		name, navigate to the Landing Zone Accelerator on AWS GitHub branches page and choose the release branch you want to deploy. Release branch names align with the semantic versionin g of our GitHub releases. New release branches will be available as the open-source project is updated with new features.
Enable Approval Stage	`Yes`	Select `Yes` to add a manual approval stage to accelerator pipeline.
Manual Approval Stage notification email list	*optional input*	Provide comma separated list of email IDs to receive manual approval stage notification email.
Management Account Email	`<requires input>`	The management (primary) account email. NOTE: Use a unique email address.
LogArchive Account Email	`<requires input>`	The log archive account email. NOTE: Use a unique email address.
Audit Account Email	`<requires input>`	The security audit account (also referred to as the audit account). NOTE: Use a unique email address.

Parameter	Default	Description
Control Tower Environment	`Yes`	Select `Yes` if you want to deploy to an AWS Control Tower environment. Select `No` if you're not using AWS Control Tower.
Accelerator Resource name prefix	`AWSAccelerator`	The prefix value for accelerator-deployed resources. Leave the default value if you're using the solution-defined resource name prefix. IMPORTANT: Updating this value after initial installation will cause stack failure. Non-default value cannot start with keywords `aws` or `ssm` (case insensitive). Trailing dashes (for example, `input-`) in a non-default value will be ignored.
Use Existing Config Repository	`No`	Select `Yes` to deploy the solution with an existing configuration repository. Leave the default value if you're using the solution-deployed repository. NOTE: Updating this value after initial installation may cause adverse effects such as unexpected failures and resource replacements.

Parameter	Default	Description
Existing Config Repository Name	*optional input*	The name of an existing CodeCommit repository hosting the solution configuration. When the *Use Existing Config Repository* parameter is set to `Yes`, the value for this parameter must be a valid name of an existing CodeCommit repository that holds the solution configuration. NOTE: When the **Use Existing Config Repository** parameter is set to `Yes` and this parameter is empty, then the Installer stack validation will fail, which will cause stack deployment failure.

Parameter	Default	Description
Existing Config Repository Branch Name	*optional input*	The name of an existing CodeCommit repository to pull the solution configuration from. When the *Use Existing Config Repository* parameter is set to `Yes`, the value for this parameter must be a valid name of an existing CodeCommit repository that holds the solution configuration. NOTE: When the **Use Existing Config Repository** parameter is set to `Yes` and this parameter is empty, then the Installer stack validation will fail, which will cause stack deployment failure.
Enable Diagnostics Pack	`Yes`	Select `Yes` to deploy the diagnostics pack tool. For more information about the diagnostics pack tool deployed by the solution, refer to Diagnostics pack in the Troubleshooting section.

Parameter	Default	Description
Configuration Repository Location	`<requires input>`	Determines where to store the LZA configuration files used to customize your landing zone. In previous versions of the solution, files were stored in CodeCommi t by default.IMPORTANT: Updating this value after initial installation will cause stack failures.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next.**
8. On the **Review and create** page, review and confirm the settings. Select the box acknowledging that the template might create IAM resources.
9. Choose **Submit** to deploy the stack.

   You can view the status of the stack in the AWS CloudFormation console in the **Status** column. You should receive a `CREATE_COMPLETE` status in approximately eight minutes.

Step 2. Await initial environment deployment

Use the following procedure to ensure the Landing Zone Accelerator on AWS deploys a minimum configuration to your environment.

1. Sign in to the AWS Management Console and navigate to the **AWS CodePipeline** console. The `AWSAccelerator-Installer` pipeline should show a status of either `In Progress` or `Complete`. If `In Progress`, wait for the pipeline to complete.
2. When the `AWSAccelerator-Installer` pipeline has completed, a new `AWSAccelerator-Pipeline` pipeline is created that's now `In Progress`. Refresh the AWS CodePipeline console if the new pipeline isn't visible.
3. The `AWSAccelerator-Pipeline` pipeline takes approximately 45 minutes to complete. This initial deployment prepares your environment for Landing Zone Accelerator on AWS and deploy a minimal configuration. Resources deployed include AWS CloudFormation custom resources,

CloudWatch Logs log groups for the custom resources, AWS KMS keys for encryption at rest, and Amazon S3 buckets for AWS service logging.

4. After completion of the preceding steps, your environment is ready to customize.

Step 3. Update the configuration files

Use the following procedure to customize Landing Zone Accelerator on AWS to fit your environment's needs. These files are stored in either a CodeCommit repository, S3 bucket, or a custom repository using AWS CodeConnections depending on parameters selected during deployment. If you aren't sure, check the `Configuration Repository Location` parameter of your `AWSAccelerator-Installer` stack.

Using CodeCommit

1. Sign in to the AWS Management Console and navigate to the **CodeCommit** console. Navigate to the repository named **aws-accelerator-configuration**. The repository shows the Landing Zone Accelerator on AWS configuration files.
2. Each configuration file is named based on its purpose in Landing Zone Accelerator on AWS. A sample configuration is available on our GitHub repository. Customize each configuration file to deploy the additional AWS services and infrastructure required. You can use the CodeCommit console or a compatible Git client to manipulate these files. For more information, refer to Edit the contents of a file in a CodeCommit repository in the *AWS CodeCommit User Guide*.
3. When finished editing the configuration files, navigate to the AWS CodePipeline console. Select **AWSAccelerator-Pipeline**, then **Release change**. This initiates a new pipeline instantiation and deploy the configuration changes to your environment.
4. Await successful completion of the pipeline. If any failures occur, the CodePipeline console displays the failure stage and action in red. To troubleshoot any errors, choose **Details** on the CodeBuild action to navigate to the failed action. In the CodeBuild console, you can view the **Build logs**, which indicates the error encountered during deployment. For more information, refer to Troubleshooting.

Using Amazon S3

1. Sign in to the Amazon S3 console.
2. Navigate to the bucket named `aws-accelerator-config-<ACCOUNT_ID> - <REGION> .`

3. Download the S3 object `zipped/aws-accelerator-config.zip` and extract the contents to view your Landing Zone Accelerator on AWS configuration files.
4. Each configuration file is named based on its purpose in Landing Zone Accelerator on AWS. A sample configuration is available on our GitHub repository. Customize each configuration file to deploy the additional AWS services and infrastructure required. Make desired changes to these files locally, then save your changes.
5. When you're finished editing the configuration files, compress the files into a new zip archive file named `aws-accelerator-config.zip`. Upload this file to the same S3 object path `zipped/aws-accelerator-config.zip` used in Step 1.

> **Note**
>
> The `aws-accelerator-config.zip` file contains all of the files at the root of the zip archive file. The following is an example of using the `tree` command to list the contents:
>
> ```
> > tree -a .
> .
> ### accounts-config.yaml
> ### global-config.yaml
> ### iam-config.yaml
> ### network-config.yaml
> ### organization-config.yaml
> ### security-config.yaml
> ```

6. Sign in to the AWS CodePipeline console. Select **AWSAccelerator-Pipeline**, then **Release** change. This initiates a new pipeline instantiation and deploys the configuration changes to your environment.
7. Await successful completion of the pipeline. If failures occur, the CodePipeline console displays the failure stage and action in red. To troubleshoot errors, choose **Details** on the CodeBuild action to navigate to the failed action. In the CodeBuild console, you can view the **Build logs**, which indicate the error encountered during deployment. For more information, refer to Troubleshooting.

Using AWS CodeConnections

1. Sign in to the Amazon Developer Tools console.
2. From the left-hand sidebar, select the **Settings** drop down and select **Connections**.

3. On the **Connections** page, select the **Create Connection** button.
4. To create a connection, follow the Create a connection user guide in the *Developer Tools console*.

> **Note**
>
> When creating a connection, select **Install a new app**, otherwise it is possible the source stage in your pipeline may fail while attempting to connect to your configuration repository

5. After creating the Code Connection successfully, make sure to note the Code Connection ARN.
6. Once you have the Code Connection ARN, you can fill out the following Parameters in the LZA Installer Stack:
   - UseExistingConfigRepo: **Yes**
   - ExistingConfigRepositoryName: **aws-accelerator-config**
   - ExistingConfigRepositoryOwner: **awslabs**

     > **Note**
     >
     > This needs to be your 3rd party "owner" or namespace

   - ExistingConfigRepositoryBranchName: **main**

     > **Note**
     >
     > This needs to match your branch name in the 3rd party repo

   - ConfigurationRepositoryLocation: **codeconnection**

Opt-in Regions

We built the opt-in Region configuration to help customers use the Landing Zone Accelerator on AWS solution in opt-in Regions.

> **Note**
>
> Not all AWS services are available in all Regions, including the AWS opt-in Regions. We update our AWS Regional Services list daily with which services are available in which Regions.

You must initially launch Landing Zone Accelerator on AWS in a Region where CodeCommit, AWS CodeBuild, and AWS CodePipeline are available. This will deploy the default resources that are depicted in the Architecture overview.

The following installation instructions leverage opt-in AWS Regions. Following these instructions deploys the default resources into the management account for items 1-8 of the architecture diagram. Items 9-10 of the architecture diagram, centralized logging and workload accounts, deploy in the opt-in (target) AWS Region.

> **Note**
>
> While the Landing Zone Accelerator on AWS solution can help you align with frameworks and best practices, customers are responsible for their own security and compliance practices.

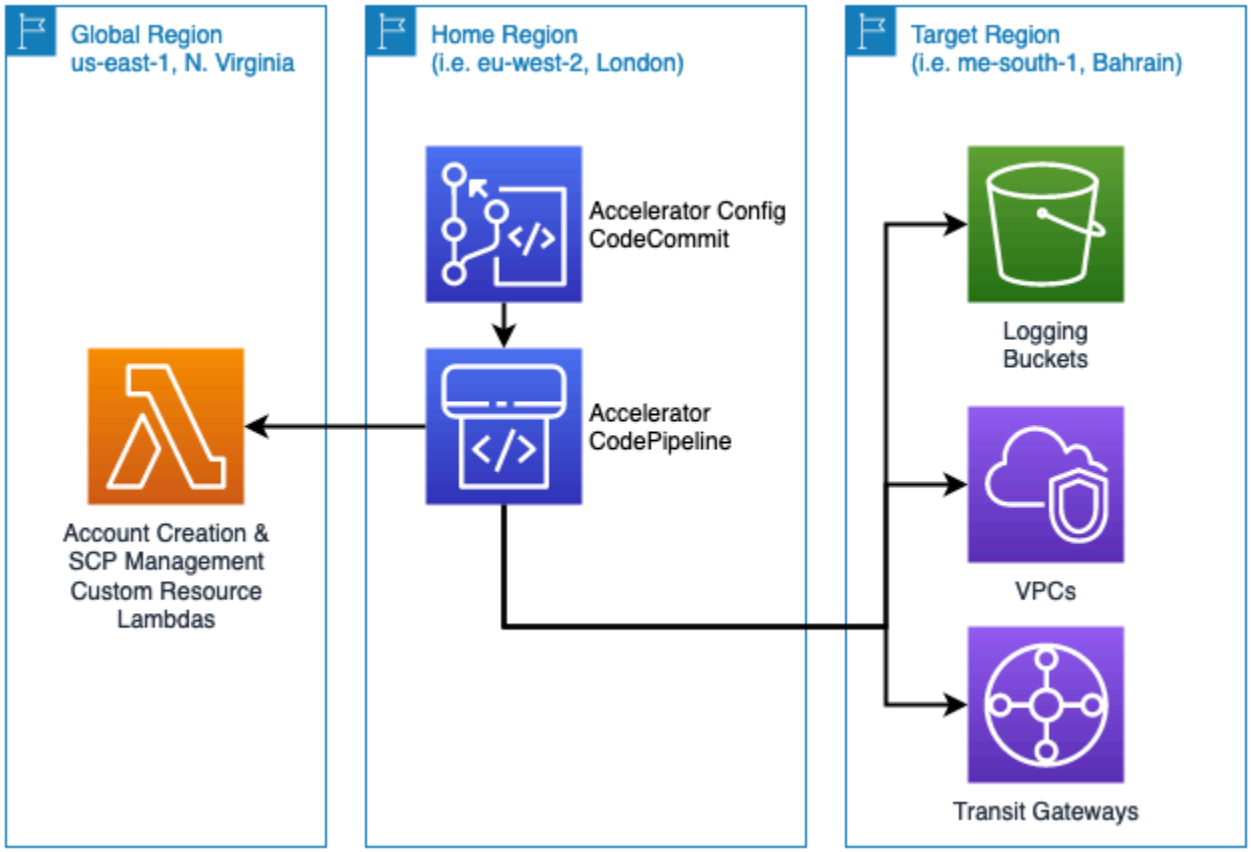
Prerequisites

To launch the Landing Zone Accelerator on AWS solution into opt-in AWS Regions, verify that the user who launches the solution can:

- Allow opt-in AWS Regions
- Perform IAM administration tasks

Architecture

Architecture diagram depicting Landing Zone Accelerator on AWS architecture in opt-in (Target) Regions.



Deployment

Using an opt-in Region as the target Region

Deploying this solution with the default parameters builds the environment depicted in the previous figure. The default parameters use the **Home Region** for the Landing Zone Accelerator on AWS Core pipeline and the **Target Region** for centralized logging.

Step 1. Deploy the solution in your AWS Management account

1. Identify the **Home Region** that you want to use. This Region must have Amazon S3, CodeBuild, and CodePipeline availability.

> **Note**
>
> Two main factors contribute to which Region to select as your **Home Region**: latency and cost. Choosing an AWS Region with close proximity to your user base location can

achieve lower network latency. AWS services are priced differently from one Region to another.

2. Prepare for an AWS Organizations based installation (without AWS Control Tower). Use the following notes to guide you:
   - For a new environment, set up AWS Organizations.
   - Create a **LogArchive** account and an **Audit/Security Tooling** account.
   - Create a **Security** OU and **Infrastructure** OU.
3. Set up Landing Zone Accelerator on AWS in your AWS standard account.

Step 2. Allow your desired opt-in AWS Regions for all accounts

1. Sign in to your management account.
2. Allow the Regions you want to use.

> **Note**
>
> When you allow a Region, AWS prepares your account in that Region, such as by distributing your IAM resources to the Region. This process takes a few minutes for most accounts, but it can take several hours. You can't use the Region until this process is complete.

3. Log in to the **LogArchive** and **Audit/Security Tooling** accounts to repeat the actions to allow the opt-in Regions that you want to use.

Step 3. Update the configuration file in your AWS Management account

1. Using your management account, update the `global-config.yaml` file to list the new Region under the `enabledRegions` option, as shown in the following sample. In the sample, Europe (London) (`eu-west-2`) is the home Region and Middle East (Bahrain) (`me-south-1`) is the opt-in (target) Region:

```
homeRegion: eu-west-2
enabledRegions:
  - eu-west-2
  - me-south-1
```

2. Using your management account, update the `global-config.yaml` file to list the opt-in Region under the `centralizedLoggingRegion` option, as shown in the following sample:

```
logging:
  account: LogArchive
  centralizedLoggingRegion: me-south-1
  cloudtrail:
    enable: true
    organizationTrail: true
    organizationTrailSettings:
      multiRegionTrail: true
      globalServiceEvents: true
      managementEvents: true
      s3DataEvents: true
      lambdaDataEvents: true
      sendToCloudWatchLogs: true
      apiErrorRateInsight: false
      apiCallRateInsight: false
    accountTrails: []
    lifecycleRules: []
  sessionManager:
    sendToCloudWatchLogs: false
    sendToS3: false
    excludeRegions: []
    excludeAccounts: []
    lifecycleRules: []
    attachPolicyToIamRoles: []
```

3. After the commit, confirm that the pipeline runs successfully.

Deploy to AWS GovCloud (US) Regions

We architected this solution to follow the Defense Information Systems Agency (DISA) Cloud Computing Security Requirements Guide (CC SRG) for hosting Impact Level (IL)4 and IL5 workloads in the cloud when deployed in AWS GovCloud (US) Regions. Using this solution, you can deploy an architecture baseline that accommodates US Federal and DoD requirements to rapidly achieve Authority to Operate (ATO).

> **Note**
>
> While the Landing Zone Accelerator on AWS solution can help you align with frameworks and best practices, customers are responsible for their own ATO readiness.

An installation into AWS GovCloud (US) Regions is treated as an independent installation of the Landing Zone Accelerator on AWS solution. You can use this solution to manage your corresponding standard AWS environment, resulting in two concurrent Landing Zone Accelerator on AWS-based environments that you can manage in a unified way.

> **Note**
>
> Not all AWS services are available in all Regions, including the AWS GovCloud (US) Regions. We update our AWS Regional Services list daily with which services are available in which Regions.

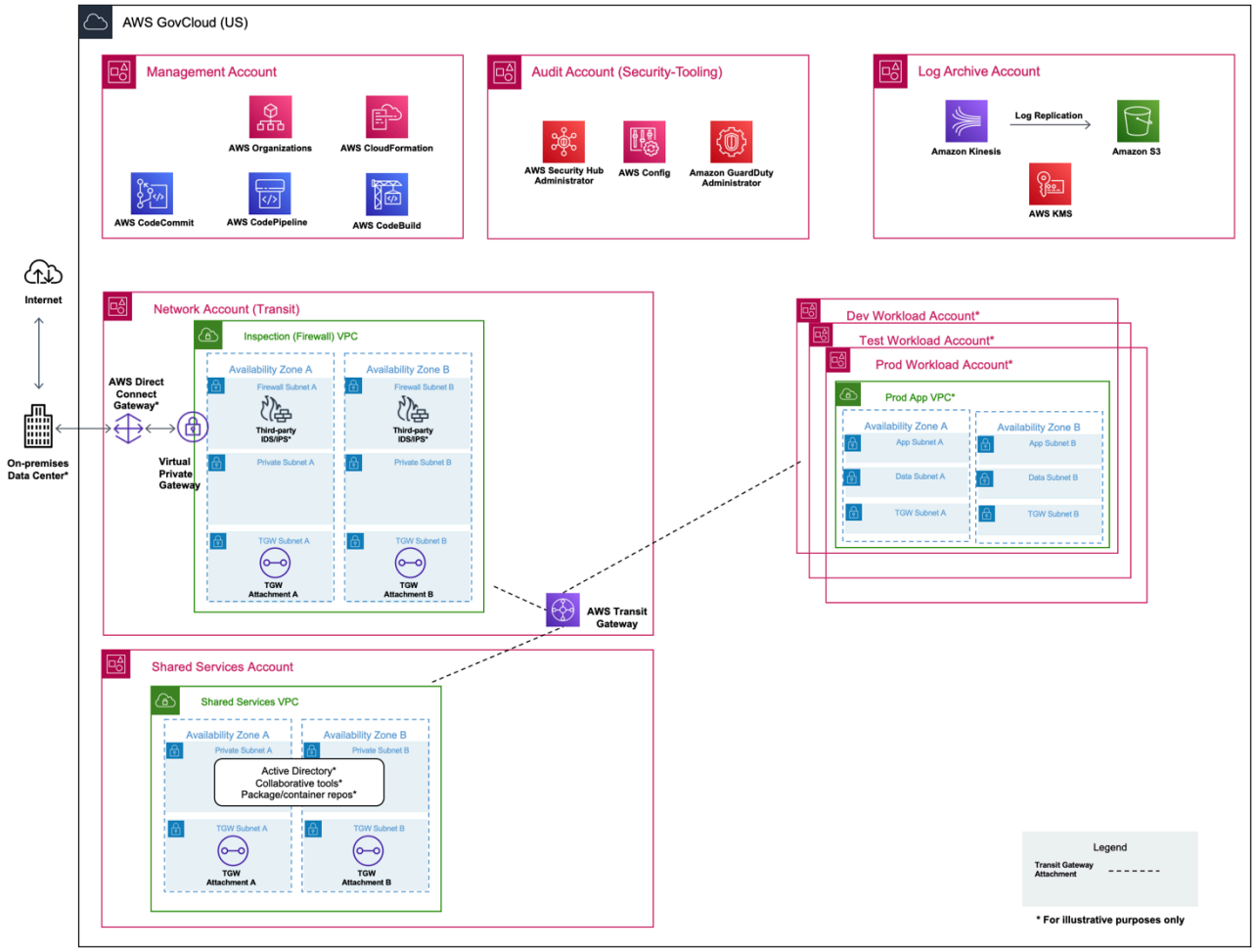
Prerequisites

To launch the Landing Zone Accelerator on AWS solution, verify the following:

- The account used to launch the solution is allowed to access AWS GovCloud (US) Regions.
- You're authorized to create accounts in the AWS GovCloud (US) Regions. For more information on the AWS GovCloud (US) Regions, refer to the AWS GovCloud (US) User Guide.
- You have an account in an AWS GovCloud (US) Region that's paired with a management account of an organization in a standard AWS Region.

Architecture

AWS GovCloud architecture diagram showing account types, services, and network connections.



Deployment options for AWS GovCloud (US) workloads

We base the following options on amount of access type of workloads:

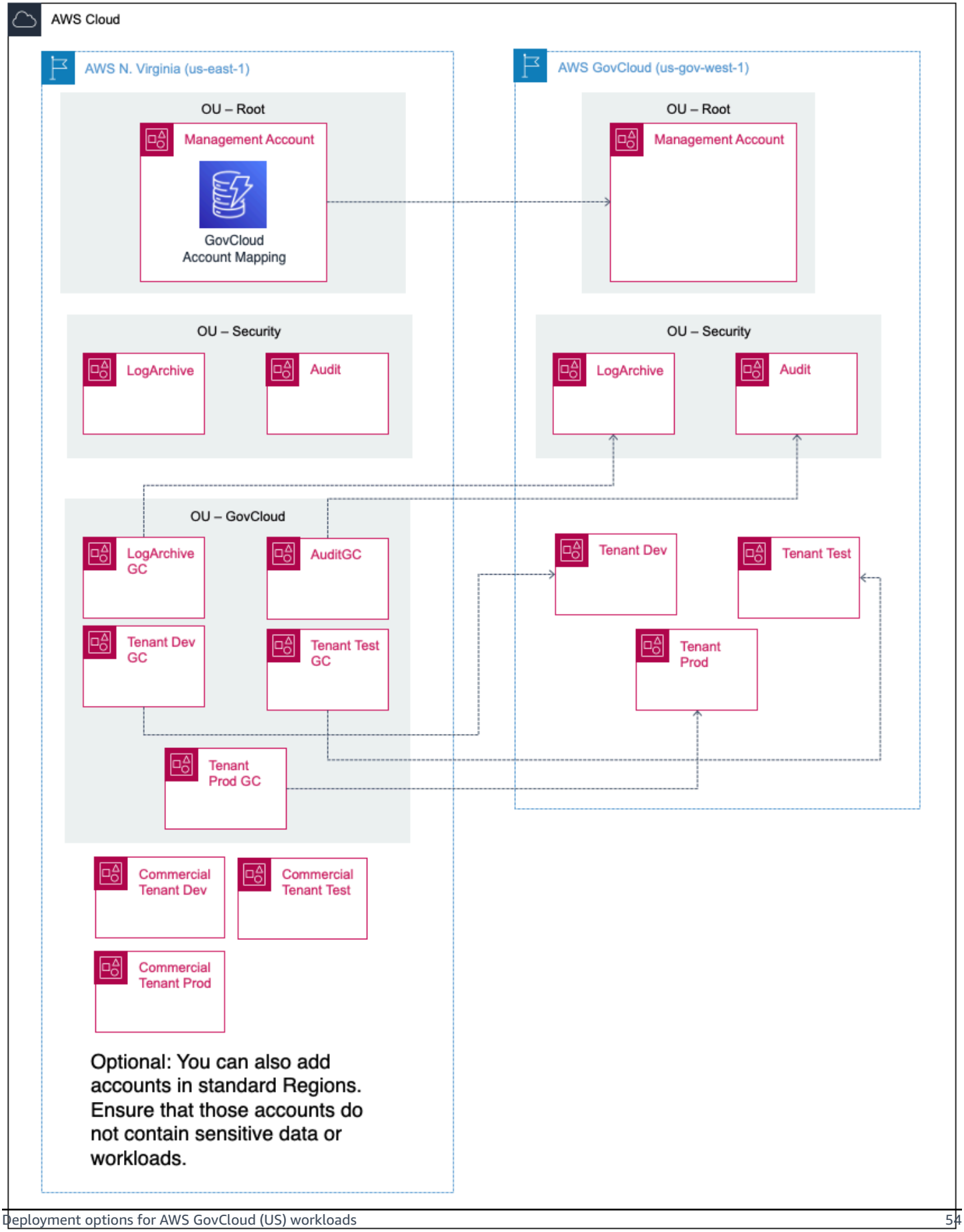
- Option 1 - Deploy to new standard and AWS GovCloud (US) accounts. This is recommended for customers who are planning to host workloads in both standard and AWS GovCloud (US) Regions. Both Region types will have a Landing Zone Accelerator on AWS.
- Option 2 - Deploy on new AWS GovCloud (US) accounts. This environment has access to both standard and AWS GovCloud (US) Regions. To create new AWS GovCloud (US) accounts, you can use the `CreateGovCloudAccount` API with Service Catalog to create new accounts in the standard Region and add these new accounts into the solution in the AWS GovCloud (US) Region. You only use the standard Region to vend new accounts; no workloads are present in the standard Region.
- Option 3 - Deploy on existing AWS GovCloud (US) accounts. In this option, users have access to AWS GovCloud (US) only and can't create their own AWS GovCloud (US) accounts. In this

situation, AWS GovCloud (US) accounts are provided by third-party providers such as partners or resellers. If AWS Organizations is activated in the management account with administrative permissions, then you can deploy the solution.

Option 1: Deploy to new standard and AWS GovCloud (US) accounts

Deploying this solution with the default parameters builds the following environment in the AWS GovCloud (US) Region(s).

Architecture diagram depicting AWS GovCloud (US) deployment.


AWS Cloud
AWS N. Virginia (us-east-1)
OU – Root
Management Account
GovCloud
Account Mapping
OU – Security
LogArchive
Audit
OU – GovCloud
LogArchive
GC
AuditGC
Tenant Dev
GC
Tenant Test
GC
Tenant
Prod GC
Commercial
Tenant Dev
Commercial
Tenant Test
Commercial
Tenant Prod
Optional: You can also add accounts in standard Regions. Ensure that those accounts do not contain sensitive data or workloads.
AWS GovCloud (us-gov-west-1)
OU – Root
Management Account
OU – Security
LogArchive
Audit
Tenant Dev
Tenant Test
Tenant
Prod

The AWS CloudFormation template includes a set of configuration files that have been specifically customized for AWS GovCloud (US) Regions. By following these instructions, you can deploy an environment that includes:

1. Use of AWS Control Tower to manage and govern your AWS standard accounts.

> **Note**
>
> In this implementation guide, the terms "AWS standard account" and "AWS standard Region" mean "AWS account that isn't in an AWS GovCloud (US) Region" and "AWS Region that isn't an AWS GovCloud (US) Region."

2. A deployment of the solution in your **AWS standard accounts** (refer to the left side of the previous figure), allowing you to activate additional security features and guardrails into your AWS standard accounts and providing you the ability to generate AWS GovCloud (US) accounts.
3. A deployment of the solution in your **AWS GovCloud (US) accounts** (refer to the right side of the previous figure) with the AWS best practices configuration of security services and an AWS best practices-recommended network topology. This configuration is architected to follow the US Department of Defense (DoD) Cloud Computing Security Requirements Guide (CC SRG) for hosting Impact Level (IL)4 and IL5 workloads in the cloud. Using this configuration, you can quickly deploy an architecture baseline that accommodates US federal and DoD requirements to rapidly achieve Authority to Operate (ATO). In addition, this solution is architected to support and accelerate DoD Cybersecurity Maturity Model Certification (CMMC) readiness.

> **Important**
>
> Don't use the AWS standard account paired to AWS GovCloud (US) accounts to host any workloads.

Step 1. Deploy the solution in your AWS standard Management account and create AWS GovCloud (US) accounts

1. Create an AWS standard account that is allowed to access AWS GovCloud (US) Region(s) and is the AWS Organizations Management account.
2. Set up and verify AWS Organizations through email. (This step is optional but saves time in AWS Control Tower setup [Step 1.3].)

3. Set up Landing Zone Accelerator on AWS in your AWS standard account.
4. After successfully setting up Landing Zone Accelerator on AWS in your AWS standard account, update the `organization-config.yaml` file in the `aws-accelerator-config` CodeCommit repository to make the new OU visible to Landing Zone Accelerator on AWS. Run the Landing Zone Accelerator on AWS pipeline with this change.

```
enable: true
organizationalUnits:
  - name: Security
  - name: Infrastructure
  - name: GovCloud
serviceControlPolicies: []
taggingPolicies: []
backupPolicies: []
```

1. After the Landing Zone Accelerator on AWS pipeline completes, create new AWS GovCloud (US) accounts using the `enableGovCloud` field in the `workloadAccounts` definition. These are AWS GovCloud accounts paired to your AWS standard account. You must specify these under `workloadAccounts:`. The following is a sample account configuration.

```
# commercial accounts-config.yaml
mandatoryAccounts:
  - name: Management
    description: >-
      The management (primary) account. Do not change the name field for this mandatory
 account.
    email: < landing-zone-management-email@example.com> <----- UPDATE EMAIL ADDRESS
    organizationalUnit: Root
  - name: LogArchive
    description: >-
      The log archive account. Do not change the name field for this mandatory account.
    email: <commercial-log-archive-email@example.com> <----- UPDATE EMAIL ADDRESS
    organizationalUnit: Security
  - name: Audit
    description: >-
      The security audit account (also referred to as the audit account). Do not change
 the name field for this mandatory account.
    email: <commercial-audit-email@example.com> <----- UPDATE EMAIL ADDRESS
    organizationalUnit: Security
```

```
workloadAccounts:
  - name: LogArchiveGC # referred to as LogArchive in the GovCloud account-config.yaml
    description: The log archive account for GovCloud.
    email: <govCloud-log-archive-email@example.com> <----- UPDATE EMAIL ADDRESS
    # this OU has all GovCloud accounts.
    # OU was created from Control Tower
    # in organization-config.yaml this OU was added.
    organizationalUnit: GovCloud
    # enableGovCloud is a one-time non-reversible option
    # which only works with creation of new accounts
    enableGovCloud: true
  - name: AuditGC # referred to as LogArchive in the GovCloud account-config.yaml
    description: The security audit account (also referred to as the audit account) for
 GovCloud.
    email: <govCloud-audit-email@example.com> <----- UPDATE EMAIL ADDRESS
    organizationalUnit: GovCloud
    enableGovCloud: true
```

1. The solution creates paired accounts which are joined in AWS Organizations in the AWS standard Region. These accounts will have a cross-account assume role in the AWS GovCloud (US) Region(s) but will not be a part of the AWS GovCloud (US) Organization.
2. Add new AWS GovCloud (US) accounts to the `accounts-config.yaml` file in the AWS standard Region and run the solution pipeline.

   > **Note**
   >
   > We highly recommend that you vend all AWS GovCloud (US) accounts from the Landing Zone Accelerator on AWS solution.

3. After the pipeline completes, navigate to AWS Organizations console page to retrieve the commercial account IDs of the newly created accounts.
4. Navigate to the AWS GovCloud (US) account mapping table in Amazon DynamoDB. Find the table name from AWS Systems Manager parameter (SSM parameter) `/accelerator/prepare-stack/govCloudAccountMappingTableName`. In that table, look up rows with commercial account IDs from the previous step. The AWS GovCloud (US) account IDs are shown under the **govCloudAccountId** column. You need these AWS GovCloud (US) account IDs to onboard AWS GovCloud (US) accounts.

Step 2. Deploy the solution in your AWS GovCloud (US) Management account

1. Log in to the AWS GovCloud (US) Management account.
2. Set up and verify AWS Organizations through email.
3. Invite AWS GovCloud (US) LogArchive and Audit account to your organization.
4. Accept the invite by using switch to the role for the member account.

> **Note**
>
> The role is defined as `managementAccountAccessRole` in the global-config.yaml configuration file for the AWS standard Management account.

1. Deploy the solution in the AWS GovCloud (US) Management account. The input into the installer stack for LogArchive and Audit accounts will be the AWS GovCloud (US) accounts vended from the linked AWS standard account. (This implementation guide uses <govCloud-audit- email@example.com> (<email@example.com>) and <govCloud-log-archive-email@example.com> (<govCloud-log-archive-email@example.com>) as example accounts.)

Step 3. Update the configuration file in your AWS standard account to create new AWS GovCloud (US) accounts

1. Using the AWS standard account, update the `accounts-config.yaml` file to have two new accounts with the `enableGovCloud` option, as shown in the following sample.

```
# commercial accounts-config.yaml
mandatoryAccounts:
  - name: Management
    description: >-
      The management (primary) account. Do not change the name field for this
 mandatory account.
    email: <landing-zone-management-email@example.com> <----- UPDATE EMAIL ADDRESS
    organizationalUnit: Root
  - name: LogArchive
    description: >-
      The log archive account. Do not change the name field for this mandatory
 account.
    email: <commercial-log-archive-email@example.com> <----- UPDATE EMAIL ADDRESS
```

```
    organizationalUnit: Security
  - name: Audit
    description: >-
      The security audit account (also referred to as the audit account). Do not
 change the name field for this mandatory account.
    email: <commercial-audit-email@example.com> <----- UPDATE EMAIL ADDRESS
    organizationalUnit: Security
workloadAccounts:
  - name: LogArchiveGC # referred to as LogArchive in the GovCloud account-
config.yaml
    description: The log archive account for GovCloud.
    email: <govCloud-log-archive-email@example.com> <----- UPDATE EMAIL ADDRESS
    # this OU has all GovCloud accounts.
    # OU was created from Control Tower
    # in organization-config.yaml this OU was added.
    organizationalUnit: GovCloud
    # enableGovCloud is a one-time non-reversible option
    # which only works with creation of new accounts
    enableGovCloud: true
  - name: AuditGC # referred to as Audit in the GovCloud account-config.yaml
    description: The security audit account (also referred to as the audit account)
 for GovCloud.
    email: <govCloud-audit-email@example.com> <----- UPDATE EMAIL ADDRESS
    organizationalUnit: GovCloud
    enableGovCloud: true
  - name: SharedServicesGC # referred to as SharedServices in the GovCloud account-
config.yaml
    description: Shared services account for GovCloud.
    email: <govCloud-shared-services-email@example.com> <----- UPDATE EMAIL ADDRESS
    organizationalUnit: GovCloud
    enableGovCloud: true
  - name: NetworkGC # referred to as Network in the GovCloud account-config.yaml
    description: Network account for GovCloud.
    email: <govCloud-network-email@example.com> <----- UPDATE EMAIL ADDRESS
    organizationalUnit: GovCloud
    enableGovCloud: true
```

2. After the commit, confirm that the pipeline runs successfully.
3. From the AWS GovCloud (US) mapping table, retrieve the AWS GovCloud (US) account ID for the **SharedServicesGC** and **NetworkGC** accounts.

Step 4. Configure solution in AWS GovCloud (US) Region(s) to manage new accounts

1. Log in to the AWS GovCloud (US) Management account.
2. Add the **SharedServices** and **Network** accounts as shown in the following sample.

```
# govCloud accounts-config.yaml
mandatoryAccounts:
  - name: Management
    description: >-
      The management (primary) account. Do not change the name field for this
 mandatory account.
    email: <landing-zone-management-email@example.com> <----- UPDATE EMAIL ADDRESS
    organizationalUnit: Root
  - name: LogArchive
    description: >-
      The log archive account. Do not change the name field for this mandatory
 account.
    email: <govCloud-log-archive-email@example.com> <----- UPDATE EMAIL ADDRESS
    organizationalUnit: Security
  - name: Audit
    description: >-
      The security audit account (also referred to as the audit account). Do not
 change the name field for this mandatory account.
    email: <govCloud-audit-email@example.com> <----- UPDATE EMAIL ADDRESS
    organizationalUnit: Security
workloadAccounts:
  - name: SharedServices
    description: Shared services account for GovCloud.
    email: <govCloud-shared-services-email@example.com> <----- UPDATE EMAIL ADDRESS
    organizationalUnit: Infrastructure
  - name: Network
    description: Network account for GovCloud.
    email: <govCloud-network-email@example.com> <----- UPDATE EMAIL ADDRESS
    organizationalUnit: Infrastructure

accountIds:
  - email: <landing-zone-management-email@example.com> <----- UPDATE EMAIL ADDRESS
    accountId: '000000000000'  <----- UPDATE GOVCLOUD ACCOUNT ID from Commercial
 GovCloud mapping table
  - email: <govCloud-log-archive-email@example.com> <----- UPDATE EMAIL ADDRESS
    accountId: '111111111111'  <----- UPDATE GOVCLOUD ACCOUNT ID from Commercial
 GovCloud mapping table
  - email: <govCloud-audit-email@example.com> <----- UPDATE EMAIL ADDRESS
```

```
    accountId: '222222222222'  <----- UPDATE GOVCLOUD ACCOUNT ID from Commercial
GovCloud mapping table
  - email: <govCloud-shared-services-email@example.com> <----- UPDATE EMAIL ADDRESS
    accountId: '333333333333'
  - email: <govCloud-network-email@example.com> <----- UPDATE EMAIL ADDRESS
    accountId: '444444444444'
```

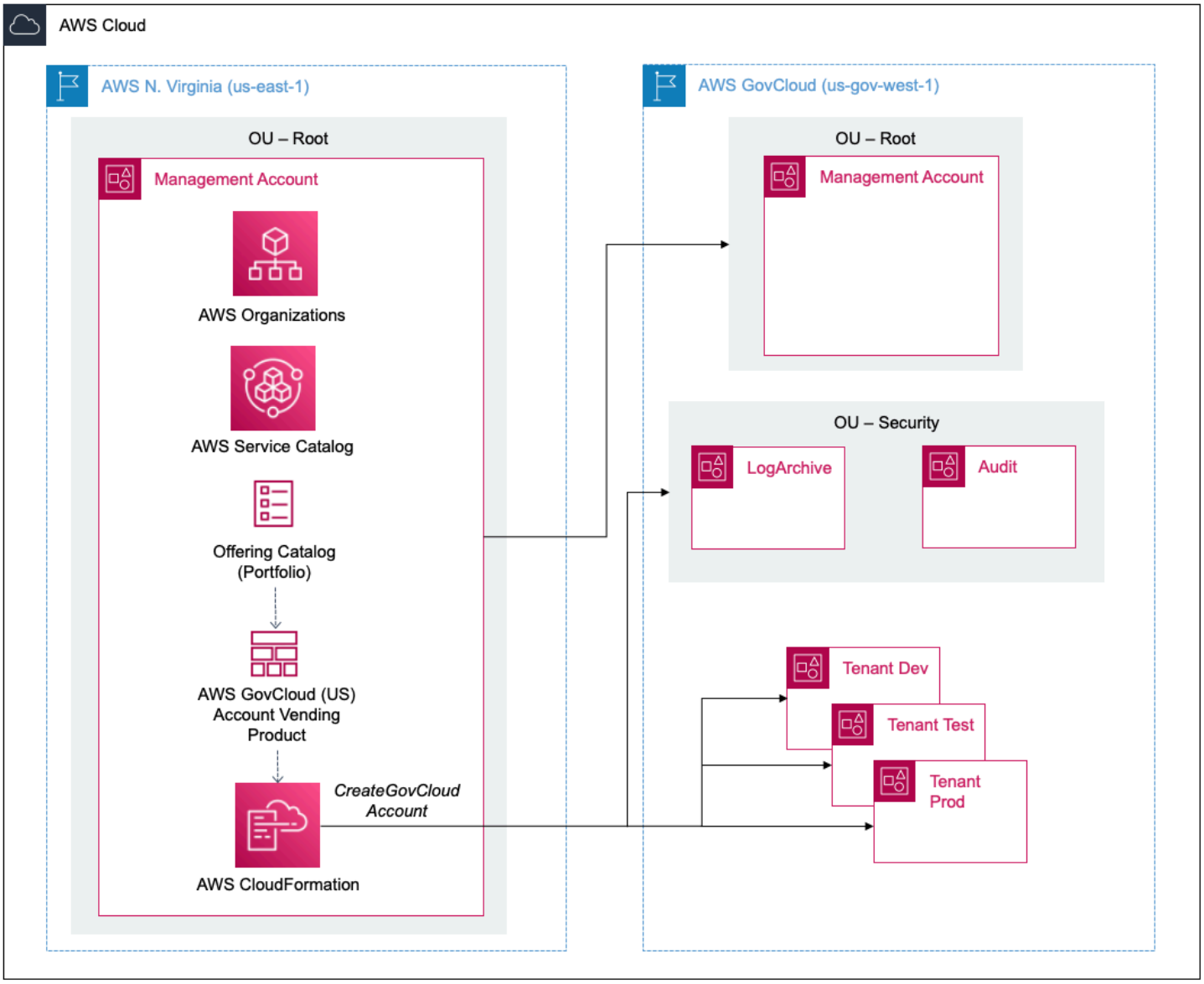
3. After the commit, confirm that the pipeline runs successfully.

Option 2: Deploy on new AWS GovCloud (US) accounts

Deploying the solution in this pattern allows users to have workloads in AWS GovCloud (US) Regions only. The standard Region on the left is used to create AWS GovCloud (US) using Service Catalog.

> **Note**
>
> This deployment assumes that you want to limit your use of standard AWS Regions, and it includes steps to incorporate AWS Organizations SCPs that limit what the AWS standard accounts can do. If you also want to use standard AWS Regions (such as a US DoD customer that wants to run IL2 workloads in AWS US East/West Regions and IL4/IL5 workloads in AWS GovCloud [US] Regions through a shared AWS standard Management billing account), AWS recommends that you create new AWS standard accounts specifically for AWS standard Region usage.

Architecture diagram depicting AWS GovCloud (US) account deployment.



Step 1. Launch the stack

1. Ensure that all prerequisites are complete. Ensure that you've set up AWS Organizations and that the account where the stack is launched can run the `CreateGovCloudAccount` API. See For AWS Organizations based installation (without AWS Control Tower) for more information.
2. Sign in to the AWS Management Console of your organization's management account and select the following button to launch the `AWSAccelerator-GovCloudAccountVending` AWS CloudFormation template.

View template

AWSAccelerator-GovCloudAccountVending.template - Use this template to launch the AWS GovCloud (US) account vending component.

3. On the **Create stack** page, verify that the correct template URL is in the **Amazon S3 URL** text box and choose **Next**.
4. On the **Specify stack details** page, assign a name to your solution stack. We recommend you name your stack `AWSAccelerator-GovCloudAccountVending` to match the naming convention used for additional stacks that the solution creates. For information about naming character limitations, refer to IAM and STS quotas in the *AWS Identity and Access Management User Guide*.
5. Choose **Next**.
6. On the Configure stack options page, choose Next.
7. On the **Review** page, review and confirm the settings. Select the box acknowledging that the template will create IAM resources.
8. Choose **Create stack** to deploy the stack.

Step 2. Use Service Catalog to launch the product

1. In the AWS Management Console upper left section, select **Services** and then select **Service Catalog**.
2. Ensure that the in-use IAM resource that has permissions to access the portfolio **Landing Zone Accelerator on AWS**. Refer to Grant Access to Users in Service Catalog Administrator Guide.
3. In the left-hand navigation menu, under **Provisioning**, choose **Products**.
4. In **Products**, choose a Landing Zone Accelerator on AWS - GovCloud Account Vending product and then **Launch product**.
5. In **Provisioned product name**, enter or generate a name (for example, `Landing_Zone_Accelerator_GovCloud_Account_LogArchive`).
6. In **Product versions**, choose a version of the product (for example, v1.0.0).
7. In **Parameters**, specify the following parameters:
   - **Account Name** - Name of account (for example, `Accelerator Log Archive` Account)
   - **Account Email** - Valid email address (for example, `example+log-archive@amazon.com`)

- **Organization Role Name** - Name of the IAM role that AWS Organizations automatically pre-configures in the new member accounts in both the AWS GovCloud (US) Regions and in the standard Region (for example, `OrganizationAccountAccessRole`)

8. Choose **Launch product**.
9. On the **Review** page, review the configuration information, and select **LAUNCH**. This creates a CloudFormation stack. The initial status of the product is shown as **Under change**. Wait for about ten minutes, and then refresh the screen until the status changes to **AVAILABLE**.

Step 3. Get account IDs

1. In the AWS Management Console upper left section, select **Services** and then select **Service Catalog**.
2. In the left-hand navigation menu, under **Provisioning**, choose **Provisioned products**.
3. In **Provisioned Products**, choose the product that you created in step 3.8.
4. Choose **Events**.
5. Under the **Provisioned products** output, get the `GovCloudAccountId` and `AccountId`, which correspond to the AWS GovCloud (US) account ID and standard account ID, respectively.

Step 4. Deploy the solution in your AWS GovCloud (US) Management account

> **Important**
>
> Ensure that the prerequisites have been completed.

1. Log in to the AWS GovCloud (US) Management account.
2. Deploy the solution by following Step 2 of Option 1.
3. To add more accounts:
   a. Follow Step 2 and Step 3 of Option 2.
   b. Follow Step 4 of Option 1.

Option 3: Deploy on existing AWS GovCloud (US) accounts

If you don't have access to a standard Region to create new AWS GovCloud (US) accounts, work with your third party to request them. Then follow the instructions in Deploy the solution.

Update the solution

If you have previously deployed this solution, follow this procedure to update the Landing Zone Accelerator on AWS CloudFormation stack to get the latest version of the solution's framework.

Before updating the solution, run the Core pipeline manually on your current version. Troubleshoot any existing issues so that your current version runs smoothly. Performing a dry run of your existing version before proceeding with the following instructions ensures that there is no drift and the environment is stable.

1. Sign in to the AWS CloudFormation console, select your existing Landing Zone Accelerator on AWS CloudFormation stack, and select **Update**.
2. Select **Replace current template**.
3. Under **Specify template**:
   a. Select **Amazon S3 URL**.
   b. Copy the link of the latest template.
   c. Paste the link in the **Amazon S3 URL** box.
   d. Verify that the correct template URL shows in the **Amazon S3 URL** text box, and choose **Next**. Choose **Next** again.
4. Under **Parameters**, review the parameters for the template and modify them as necessary. At minimum, modify the **Branch Name** parameter to the release branch of the version you are updating to. Refer to Step 1. Launch the stack for details about the parameters.
5. Choose **Next**.
6. On the **Configure stack options** page, choose **Next**.
7. On the **Review** page, review and confirm the settings. Check the box acknowledging that the template might create IAM resources.
8. Choose **View change set** and verify the changes.
9. Choose **Update stack** to deploy the stack.

Updating the solution automatically invokes the Core pipeline (`AWSAccelerator-PipelineStack*`). You can view the status of the stack in the AWS CloudFormation console in the **Status** column. You should receive a `UPDATE_COMPLETE` status in approximately 10 minutes.

Troubleshooting

This section provides instructions for the Diagnostics pack and troubleshooting instructions for deploying and using the solution.

Known issue resolution provides instructions to mitigate known errors. If these instructions don't address your issue, see the Contact AWS Support section for instructions on opening an AWS Support case for this solution.

Diagnostics pack

The diagnostics pack tool is deployed when the installer stack of the solution is deployed with `Yes` as the value for the **Enable Diagnostics Pack** parameter. To create diagnostic reports and other configuration files, the diagnostics pack tool creates the `${ACCELERATOR_PREFIX}-Diagnostic` AWS Lambda function and the `${ACCELERATOR_PREFIX}-DiagnosticProject` AWS CodeBuild project.

To collect diagnostic reports, you can run the CodeBuild project manually. A diagnostic report will be stored in the installer's S3 bucket under the prefix `$[ACCELERATOR_PREFIX]-Diagnostics-Pack`. You can also collect configuration files for Landing Zone Accelerator on AWS using the diagnostics tool. The diagnostic report along with configuration files will assist AWS Support in troubleshooting the failure. The diagnostic report can be reviewed to determine the root cause of the problem. As part of the CodeBuild project, the `DAYS_PIPELINE_IN_FAILED_STATUS` environment variable determines how long back logs will be retrieved by the tool.

Run the diagnostics tool AWS CodeBuild project by running the following AWS CLI command:

```
aws codebuild start-build --project-name $[ACCELERATOR_PREFIX]-DiagnosticProject
```

This command will override the default value for the `DAYS_PIPELINE_IN_FAILED_STATUS `variable for the run only, and will not update the CodeBuild project definition.

```
aws codebuild start-build --project-name $[ACCELERATOR_PREFIX]-DiagnosticProject --
environment-variables-override name=DAYS_PIPELINE_IN_FAILED_STATUS,value=2
```

> **Note**
>
> We recommend that you review the diagnostic report and configuration files before providing them to AWS Support to ensure that no sensitive information is located within the logs or configuration files.

Known issue resolution

When troubleshooting issues with deployments using Landing Zone Accelerator on AWS, understanding its core architectural components is key. The main interfaces for this solution are the configuration files and the Core pipeline. You can find any issues arising during deployments to your environment in these interfaces.

Problem: Configuration file issue

Resolution

It is critical that the configuration files follow the property conventions defined. For more details, refer to the configuration reference in our GitHub Pages website. Deviations cause an error during the **Build** stage of the pipeline. During this stage, type validation of the configuration files occurs, and variances cause the pipeline to fail.

Problem: Configuration file not found issue

Resolution

When S3 buckets are used for configuration files, you might receive the following error:

```
error | accelerator | ENOENT: no such file or directory, open '/codebuild/
output/src437/src/s3/01/accounts-config.yaml'
```

Ensure that the configuration zip file uploaded to the S3 config bucket does not contain a top-level directory. Once the zip file has been unzipped, it should contain the solution configuration `yaml` files and other resource policy related folders at the root. Refer to the Update the configuration files section for more information about the proper structure of the zip archive configuration files.

Problem: Core pipeline failure

Resolution

To determine the cause of a deployment failure, use the following steps:

1. Sign in to the AWS Management Console and navigate to the **AWS CodePipeline** console. Select **AWSAccelerator-Pipeline** and find the pipeline stage that failed.
2. The pipeline stage has a CodeBuild project as an action provider. Select the **Details** link under the failed status indicator for the action, then choose **Link to execution details**. This opens the failed processing of the CodeBuild project.
3. Select the **Build logs** tab. This shows the output of the CodeBuild project run. Scrolling to the bottom of this output, you will see an error message. Some common examples are:
   - Misconfiguration or missing properties in the Landing Zone Accelerator on AWS configuration files. This causes the Core pipeline to fail in the **Build** stage. The configuration validator provides a specific error message indicating what caused the property validation to fail.
   - CloudFormation deployment error. This can occur in any stack. CloudFormation provides a specific error message indicating what caused the deployment failure.

> **Note**
>
> Regardless of the failure that occurs, build logs will show the following error message at the end:
> `[Container] Phase context status code: COMMAND_EXECUTION_ERROR Message: Error while executing command: yarn run ts-node --transpile-only cdk.ts --require-approval never $CDK_OPTIONS --config-dir $CODEBUILD_SRC_DIR_Config --partition aws --app cdk.out. Reason: exit status 1`
> This is a generic error message that CodeBuild outputs when the CDK application fails to complete successfully. When troubleshooting deployment errors, the text before this error message indicates which resource(s) failed to deploy.

Problem: Account enrollment and environment validation failures

When you enroll new or existing accounts in the solution, you can encounter Core pipeline errors during the **Prepare** stage of the pipeline. Failures during this stage typically indicate an issue with enrolling the account into AWS Organizations or AWS Control Tower.

The following are potential errors you might see in **Prepare** stage build logs when enrolling accounts:

General account enrollment failure

You might receive the following Core pipeline error message when experiencing a general account enrollment failure:

```
AWSAccelerator-PrepareStack | UPDATE_FAILED |
Custom::CreateControlTowerAccounts | CreateCTAccounts/Resource/Default
(CreateCTAccounts) Received response status [FAILED] from custom resource.
Message returned: Account creation failed. Error: Accounts failed to enroll
in Control Tower. Check Service Catalog Console
```

Resolution

Complete the following steps when this error occurs:

1. Ensure that the prerequisites listed in Adding an existing account are complete.
2. Sign in to the Service Catalog console from your Management account.
3. Select **Provisioned products** from the left-hand navigation pane.
4. Choose **Account** in the **Access Filter** drop-down menu.
5. The screen lists the reason provisioning failed. Select the Control Tower Account Factory product that failed provisioning. From the drop-down menu, select **Terminate**.
6. Sign in to the AWS CloudFormation console.
7. Select the **Prepare** stack, which will be in the `ROLLBACK_FAILED` or `UPDATE_ROLLBACK_FAILED` state after the account enrollment failure.
8. Select **Continue update rollback** from the **Stack actions** dropdown menu. Choose **Advanced troubleshooting**. Select the resource with prefix `CreateCTAccounts*`, then choose **Continue update rollback**.
9. Await rollback completion.

10Retry the **Prepare** stage of **AWSAccelerator-Pipeline**.

Environment validation error

You might receive a Core pipeline error message when experiencing an environment validation error. For example:

`AWSAccelerator-PrepareStack` `| UPDATE_FAILED | Custom::ValidateEnvironmentConfig |` **`ValidateEnvironmentConfig`**`/Resource/Default (ValidateEnvironmentConfig) Received response status [FAILED] from custom resource.` *`Message returned: Error: AWS Control Tower has detected that the managed account <account_ID> has been removed from organization <organization_ID>.`*

> **Note**
>
> This error message might differ depending on the type of drift detected.

If you have made any changes to your account(s), OU(s), or managed SCPs outside of the AWS Control Tower console, the solution's drift detection functionality likely caught these changes and caused this error. You can't run the pipeline until you undo these changes or enroll the changed account(s) or OU(s) in AWS Control Tower.

Resolution

Complete the following steps when this error occurs:

1. Ensure that all account(s), OU(s), and AWS Control Tower-managed SCPs are properly enrolled in Control Tower. For more information, see Detect and resolve drift in AWS Control Tower in the *AWS Control Tower User Guide*.
2. Sign in to the Systems Manager Parameter Store console from your Management account.
3. Search for the parameter named `/accelerator/controlTower/driftDetected`.
4. If the value of this parameter is true, select **Edit** and change the parameter value to false.
5. Sign in to the AWS CloudFormation console.
6. Select the **Prepare** stack, which will be in the `ROLLBACK_FAILED` or `UPDATE_ROLLBACK_FAILED` state after the environment validation failure.

7. Select the **Stack actions** dropdown menu, then choose **Continue update rollback**. Select **Advanced troubleshooting**. Select the resource with prefix `ValidateEnvironmentConfig*`, then choose **Continue update rollback**.
8. Await rollback completion.
9. Retry the **Prepare** stage of **AWSAccelerator-Pipeline**.

Problem: Suspended account causing enrollment or environment validation failure

After you suspend accounts from your AWS Organization, the solution environment validation feature still attempts to enroll and validate these suspended accounts in the **Prepare** stage unless you contain them within an ignored OU. You will receive a Core pipeline error until you complete the following resolution steps.

Resolution

Follow the steps in Closing an account. The solution then ignores the suspended account.

For AWS Control Tower-based environments

If you run the Core pipeline before ignoring the account, the account might have a tainted Account Factory product associated with it. Use the following procedure to remove that resource:

1. Follow the steps in Closing an account.
2. Sign in to the Service Catalog console from your Management account.
3. Select **Provisioned products** from the navigation menu.
4. Choose **Account** in the **Access Filter** drop-down menu.
5. Select the Control Tower Account Factory product that failed provisioning. From the drop-down menu, select **Terminate**.

Problem: "S3 bucket name already exists" error

This solution creates Amazon S3 buckets during deployment. Some of these buckets (such as those deployed along with the Centralized logging infrastructure) are mandatory. Others (such as the report destination buckets created for Cost and Usage Reports and AWS Audit Manager) deploy based on your defined configuration.

> **Note**
>
> By default, Amazon S3 buckets deployed by CloudFormation have a deletion policy that's set to retain the resources. Landing Zone Accelerator on AWS uses this default policy so that you can deactivate a service that the solution previously managed and still preserve your data stored in Amazon S3.

Scenarios that can cause this error include:

1. If you deactivate a solution-managed service and then reactivate it later.
2. If you uninstall the solution and then reinstall it later into the same environment.

These errors result from a standard naming convention for Amazon S3 buckets that this solution deploys. Because Amazon S3 bucket names must be globally unique, you receive an error message if the previous Amazon S3 buckets were not deleted. The following is an example, with `aws-accelerator-<SERVICE>-<ACCOUNT_ID>-<REGION>` representing the bucket name:

`AWSAccelerator-<STACK_NAME>- <ACCOUNT_ID>-<REGION>` failed: Error: The stack named AWSAccelerator- `<STACK_NAME>- <ACCOUNT_ID>-<REGION>` failed creation, it may need to be manually deleted from the AWS console: ROLLBACK_COMPLETE: aws-accelerator- `<SERVICE>-<ACCOUNT_ID>-<REGION>` already exists.

Resolution

Complete the following steps when this error occurs:

1. If you want to retain the data, make a local copy or copy the data to another Amazon S3 bucket in your account.
2. Delete the solution-created Amazon S3 bucket that's causing the conflict.
3. Retry the failing **AWSAccelerator-Pipeline** stage.

Problem: "ValidationError: Stack <stack-name> cannot be deleted while TerminationProtection is enabled" error

Depending on your deployment, you might choose to remove an existing solution-provisioned CloudFormation stack. Solution-provisioned stacks have termination protection activated by

default. If you attempt to delete a stack with termination protection activated, the deletion fails. The stack and its status remain unchanged. You might receive a Core pipeline error message like the following, with `AWSAccelerator-<STACK_NAME>- <ACCOUNT_ID>-<REGION>` representing the stack name:

```
AWSAccelerator-<STACK_NAME>- <ACCOUNT_ID>-<REGION> failed: Error
[ValidationError]: Stack <STACK_NAME>- <ACCOUNT_ID>-<REGION>] cannot be
deleted while TerminationProtection is enabled
```

Resolution

Option 1: Use the AWS Management Console

1. Deactivate termination protection on the stack. For more information, see Protecting a stack from being Deleted in the *AWS CloudFormation User Guide*.
2. Attempt deletion again.

Option 2: Use the AWS Command Line Interface

1. Deactivate termination protection on the stack by running the update-termination-protection command with the CLI:

```
$ aws cloudformation update-termination-protection --stack-name <stack-name> --no-
enable-termination-protection
```

2. Attempt deletion again.

Problem: GitHub personal access token expired

The solution uses a GitHub personal access token to access the Landing Zone Accelerator on AWS code repository. If you have set an expiration date on the access token, the token's privilege is revoked when it expires. You will see an error when trying to run the **Source** stage and action of the Installer or Core pipeline. For example:

```
Could not access the GitHub repository: "landing-zone-accelerator-on-aws".
The access token might be invalid or has been revoked. Edit the pipeline to
reconnect with GitHub.
```

Resolution

Option 1 (releases as of version 1.3.1): Use the Landing Zone Accelerator automated GitHub token update functionality

1. Create a new GitHub personal access token and update the secret value in AWS Secrets Manager.

> **Note**
>
> Saving the updated value in Secrets Manager will invoke the `UpdatePipelineGithubToken` Lambda function, which automates the process of updating the GitHub Personal Access Token in CodePipeline.

2. Retry the failed **Source** stage of the affected pipeline.

Option 2 : (releases before version 1.3.1): Update access token in AWS CodePipeline

1. Create a new GitHub personal access token and update the pipeline structure with the new token. For step-by-step instructions, see Configure authentication (Github version 1 source actions) in the *AWS CodePipeline User Guide*.
2. Retry the failed **Source** stage of the affected pipeline.

Problem: Couldn't find or create service linked role

We updated this solution to make service linked role creation idempotent. When you create a new resource, the solution checks for existing service linked role. If no service linked role exists, the solution creates one. During cleanup, the `AWS::IAM::ServiceLinkedRole` resource might have been removed successfully, which can cause issues.

Example event from CodeBuild:

```
AWSAccelerator-OrganizationsStack-<account>-<region> | …
 | DELETE_IN_PROGRESS | AWS::IAM::ServiceLinkedRole |
FirewallManagerServiceLinkedRole

AWSAccelerator-OrganizationsStack-<account>-<region> | … | DELETE_COMPLETE
| AWS::IAM::ServiceLinkedRole | FirewallManagerServiceLinkedRole
```

Resolution

Manually release the pipeline again. The service linked role will run on every pipeline. If no service linked role exists, the solution creates a new one in the account.

Problem: "The 'link' command was removed" error

We updated this solution to the newest version of `lerna` which has deprecated the "link" command. CodePipeline stages uses the link command as part of the build process for multiple stages in the pipeline.

Resolution

The latest installer template starting with v1.5.0 removed this command. Upgrades after v1.5.0 will require an update to the installer template to capture all new changes. Follow the Update the solution steps to update the installer template and the pipeline will run again resolving the error.

Problem: "AWSCloudFormationStackSetExecutionRole already exists" error

When creating AWS CloudFormation StackSets using Landing Zone Accelerator on AWS, the solution attempts to create IAM roles required for deploying StackSets with self-managed permissions. Specifically, the two required roles are:

- **AWSCloudFormationStackSetAdministrationRole** - This role is deployed to the Management account.
- **AWSCloudFormationStackSetExecutionRole** - This role is deployed to all accounts.

When deploying Landing Zone Accelerator on AWS to an environment where these roles already exist, the pipeline will fail with the `AWSCloudFormationStackSetAdministrationRole already exists` or `AWSCloudFormationStackSetExecutionRole already exists` error.

Resolution

1. Delete the `AWSCloudFormationStackSetAdministrationRole` IAM role from the Management account.
2. Delete the `AWSCloudFormationStackSetExecutionRole` IAM role from all accounts.

3. Retry the failed pipeline stage.

Contact AWS Support

If you have AWS Developer Support, AWS Business Support, or AWS Enterprise Support, you can use the Support Center to get expert assistance with this solution. The following sections provide instructions.

Create case

1. Sign in to Support Center.
2. Choose **Create case**.

How can we help?

1. Choose **Technical**.
2. For **Service**, select **Control Tower**.
3. For **Category**, select **Landing Zone Accelerator**.
4. For **Severity**, select the option that best matches your use case.
5. When you enter the **Service**, **Category**, and **Severity**, the interface populates links to common troubleshooting questions. If you can't resolve your question with these links, choose **Next step: Additional information**.

Additional information

1. For **Subject**, enter text summarizing your question or issue.
2. For **Description**, describe the issue in detail.
3. Choose **Attach files**.
4. Attach a `0zip` file containing the following:
   - Your Landing Zone Accelerator on AWS configuration files, noting modifications if applicable
   - Sanitized code build logs from the **Failed** stage which were obtained after setting the `LOG_LEVEL` to debug in the CodeBuild environment
   - Failed CloudFormation template ARN

Help us resolve your case faster

1. Enter the requested information.
2. Choose **Next step: Solve now or contact us**.

Solve now or contact us

1. Review the **Solve now** solutions.
2. If you can't resolve your issue with these solutions, choose **Contact us**, enter the requested information, and choose **Submit**.

Uninstall the solution

You can uninstall the Landing Zone Accelerator on AWS solution from the AWS Management Console or by using the AWS Command Line Interface. You must manually delete the Amazon S3 buckets and CloudFormation stacks created by this solution. AWS Solutions Implementations don't automatically delete these resources in case you have stored data to retain.

Step 1. Delete the Installer and Core pipelines

Option 1: Use the AWS Management Console

1. Sign in to the AWS CloudFormation console.
2. On the **Stacks** page, select the **AWSAccelerator-InstallerStack** and **AWSAccelerator-PipelineStack** stacks.
3. These stacks will have `TerminationProtection` enabled, which needs to be disabled before deletion. Follow the steps outlined in Problem: "ValidationError: Stack <stack-name> cannot be deleted while TerminationProtection is enabled" error.
4. Choose **Delete** for each stack.

Option 2: Use the AWS Command Line Interface

Determine whether the AWS Command Line Interface (AWS CLI) is available in your environment. For installation instructions, refer to What Is the AWS Command Line Interface in the *AWS CLI User Guide*. After confirming that the AWS CLI is available, run the following commands.

> **Note**
>
> These stacks will have TerminationProtection enabled and need to be disabled prior to deletion. Follow the steps outlined in Problem: "ValidationError: Stack <stack-name> cannot be deleted while TerminationProtection is enabled".

```
$ aws cloudformation delete-stack --stack-name AWSAccelerator-InstallerStack
```

```
$ aws cloudformation delete-stack --stack-name AWSAccelerator-PipelineStack
```

Step 2. Delete the Amazon S3 buckets

This solution is configured to retain the solution-created Amazon S3 buckets if you decide to delete the AWS CloudFormation stack to prevent accidental data loss. After uninstalling the solution, you can manually delete the Amazon S3 buckets if you don't need to retain the data. Follow these steps to delete the Amazon S3 buckets in each account Landing Zone Accelerator on AWS was configured to manage.

1. Sign in to the Amazon S3 console.
2. Choose **Buckets** from the left navigation pane.
3. Locate the `aws-accelerator-*` Amazon S3 buckets.
4. Select each Amazon S3 bucket and choose **Empty**.
5. Select each Amazon S3 bucket and choose **Delete**.

To delete the Amazon S3 buckets using AWS CLI, run the following command for each bucket:

```
$ aws s3 rb s3://<bucket-name> --force
```

Step 3. Delete additional CloudFormation stacks

This solution deploys several CloudFormation stacks to each account and AWS Region that's activated for management by Landing Zone Accelerator on AWS. Each stack deployed by the solution uses the following naming convention:

```
AWSAccelerator-<pipeline action>-<account number>-<region>
```

To successfully delete all stacks without experiencing dependency issues, delete the stacks in the reverse order that they're deployed. See the AWSAccelerator-Pipeline section for a list of actions orchestrated by the pipeline. Complete the following steps to delete each of the stacks:

1. Sign in to the AWS CloudFormation console.
2. On the **Stacks** page, select this solution's stack.
3. Choose **Delete**.

Use the solution

This section provides instructions on how to use the Landing Zone Accelerator on AWS solution after you've deployed it.

Using configuration files

Landing Zone Accelerator on AWS includes seven configuration files that you can use to customize the solution. Six of the files are mandatory. The `customizations-config.yaml` file is for optional extensions of the core solution. The solution orchestrates the creation of resources and configurations based on the input from the configuration files. Resources are generated using AWS CDK constructs defined in the solution's source code.

Having your configuration in a Git-compatible repository introduces the following benefits:

- You can use version control for your configuration like you would for source code. You can introduce feature branching and other commonly-used strategies to ensure changes to the environment meet your standards.
- You can audit the change history of the configuration files.
- The files serve as declarative manifests for your environment's configuration. The `AWSAccelerator-Pipeline` sources changes to the main branch of the repository and orchestrates your defined configuration properties with CodeBuild projects and the AWS CDK toolkit. Users who make edits to these configuration files aren't required to know how to write code.
- Because the repository is hosted in CodeCommit, you can use IAM to define which users and roles can view or make changes to the repository. You can use this strategy as a gate to allow members in your organization to make changes to the environment.

Configuration file descriptions

- **accounts-config.yaml** - Used to manage all of the AWS accounts within the AWS Organization. Adding a new account to this configuration file invokes the account creation process from Landing Zone Accelerator on AWS.
- **customizations-config.yaml (optional)** - Used to manage configuration of custom applications, third-party firewall appliances, and CloudFormation stacks.

- **global-config.yaml** - Used to manage all of the global properties that can be inherited across the AWS Organization.
- **iam-config.yaml** - Used to manage all of the IAM resources across the AWS Organization.
- **network-config.yaml** - Used to manage and implement network resources to establish a WAN/LAN architecture to support cloud operations and application workloads in AWS.
- **organization-config.yaml** - Used to manage all of the organization units in the AWS Organization.
- **replacements-config.yaml (optional)** - Used to manage all of the replacement values across the configuration files, see Parameter Store reference variable for more details.
- **security-config.yaml -** Used to manage configuration of AWS security services.

Using JSON schema

Landing Zone Accelerator on AWS fully supports JSON Schema, empowering you with enhanced configuration validation and auto-completion directly in your IDE.

- **Validation**: Configuration files are validated in real-time as you type, reducing syntax errors and improving your feedback loop. If any of your configuration files contain an error that does not align with the Landing Zone Accelerator on AWS schema, you will know immediately before pushing your config to your git-compatible repository.
- **Auto-Completion**: As you type, you will receive suggestions for configurations straight from the schema, making it easier and faster to edit Landing Zone Accelerator on AWS configuration files.
- **Discoverability**: By exploring each Landing Zone Accelerator on AWS configuration file using the schema, you can discover what options are available to you without ever leaving the IDE. For example, you can highlight any of the Landing Zone Accelerator on AWS configuration entries to view a description of it. You can also, initiate a suggestion (Ctrl/Cmd+Space in VSCode) anywhere in the Landing Zone Accelerator on AWS configuration code to show available options to you.

This feature is designed to enhance the experience of working with the Landing Zone Accelerator on AWS configuration files and is immediately available. To take advantage of this, open the Landing Zone Accelerator on AWS configuration files in an editor that supports JSON schema.

The following are examples of validated IDEs:

- VSCode: requires YAML extension

- IntelliJ

Configuration file API reference

A full reference for this solution's configuration API is available in the Services, Features, and Configuration References section of the solution's GitHub Pages website.

Performing administrator tasks

This solution allows you to automate operational tasks associated with managing a multi-account environment. This section includes common instructions and examples for managing your organizational units, accounts, core networking, and security guardrails.

> **Note**
>
> The following samples aren't an exhaustive list of what you can define in the solution configuration files. For a full configuration reference, refer to the Services, Features, and Configuration References section of the solution's GitHub Pages website.

Adding an Organizational Unit (OU)

The Landing Zone Accelerator supports the registration of AWS Organizations organizational units with the AWS Control Tower using AWS Control Tower Baseline API.

To add news OUs, complete the following steps to add the OU to the solution configuration:

1. Sign in to the AWS CodeCommit console.
2. Select the **aws-accelerator-config** repository.
3. Select the **organization-config.yaml** file.
4. Choose **Edit**.
5. In the `organizationalUnits` configuration block, append additional itemized OU name(s) and path(s). For example, to add `Testing` and `Production` OUs, your configuration would look like the following sample:

```
enable: true
organizationalUnits:
  - name: Security
```

```
  - name: Infrastructure
  - name: Testing
  - name: Production
serviceControlPolicies: []
```

> **Note**
>
> Landing Zone Accelerator on AWS uses the AWS Organizations API to create your Ous, and AWS Control Tower Baseline API to register your Ous with the AWS Control Tower.

Adding a new account

To add an account using this solution, update the `accounts-config.yaml` file with new account definitions. Account names and emails for each account must be unique. You can add new accounts to the solution configuration using the following steps.

1. Sign in to the AWS CodeCommit console.
2. Select the **aws-accelerator-config** repository.
3. Select the **accounts-config.yaml** file.
4. Choose **Edit**.
5. In the `workloadAccounts` configuration block, append additional account definition(s). For example, to add `SharedServices` and `Network` accounts to the `Infrastructure` OU and the `Testing-Workload` account to the `Testing` OU, your configuration would look like the following sample:

```
workloadAccounts:
  - name: SharedServices
    description: The SharedServices account
    email: <shared-services>@example.com  <----- UPDATE EMAIL ADDRESS
    organizationalUnit: Infrastructure
  - name: Network
    description: The Network account
    email: <network>@example.com  <----- UPDATE EMAIL ADDRESS
    organizationalUnit: Infrastructure
  - name: Testing-Workload
    description: The Workload account
    email: <workload>@example.com  <----- UPDATE EMAIL ADDRESS
```

```
    organizationalUnit: Testing
```

> **Note**
>
> **Note:** For account creation steps specific to the AWS GovCloud (US) Regions, refer to Deploy to AWS GovCloud (US) Regions.

Adding an existing account

If your existing account has already been registered with AWS Organizations and AWS Control Tower, you can follow the steps listed in Adding a new account to add the account to the solution configuration.

If the account has not yet been invited to your organization, follow the steps in Inviting an account to join your organization in the *AWS Organizations User Guide*.

For AWS Control Tower-based installations, refer to Prerequisites for enrollment in the *AWS Control Tower User Guide*0** The Landing Zone Accelerator on AWS solution can enroll the account in AWS Control Tower for you after you have completed these prerequisites.

Moving an account between OUs

If you need to move one of your accounts between OUs, complete the following steps.

> **Important**
>
> If the solution previously deployed resources to this account, identify resources in your configuration files that use the `deploymentTargets` property. You can use this property to deploy resources to all accounts contained within an OU. The solution might add or remove resources from the account depending on the targeted OU(s) in this configuration property. For example, if you target only the **Testing** OU and move the account to the **Workloads** OU, the solution removes resources using that target from the account during the next Core pipeline run.

For AWS Organizations-only deployments

For versions 1.3.0 and later of the solution, follow the Update procedure to update the `accounts-config.yaml` file with the new `organizationalUnit` property value. When you release the pipeline, the solution will move the account to the specified OU.

For versions prior to 1.3.0, first move the account from the AWS Organizations console. See Moving accounts to an OU in the *AWS Organizations User Guide* for additional details. Then follow the Update procedure instructions.

For AWS Control Tower deployments

> **Note**
>
> There might be implications of moving an Account Factory account prior to completing the following procedure. See Update and move accounts in the *AWS Control Tower User Guide* for more information.
> This section is intended for accounts actively managed by AWS Control Tower Account Factory. If moving an unmanaged or suspended account, refer to Moving accounts to an OU in the *AWS Organizations User Guide*.

1. Sign in to the AWS Control Tower console from your management account.
2. From the navigation menu, select **Organization**.
3. Select **\+** to expand the OU your account currently belongs to. Choose the account you want to move.
4. Select **Actions**, then select **Update**.
5. From the **Account detail** page, in **Organizational unit**, select the OU.
6. Choose **Update account**.
7. A confirmation box displays. Confirm the details, then select **Update account**.

   It takes approximately 5-10 minutes to process this update.
8. Proceed with the Update procedure.

Update procedure

1. Sign in to the AWS CodeCommit console.

2. Select the **aws-accelerator-config** repository.
3. Select the **accounts-config.yaml** file.
4. Choose **Edit**.
5. In the `workloadAccounts` configuration block, change the `organizationalUnit` property to the name of the new OU. In this example, we have replaced the previous OU value of **Testing** with a new value of **Workloads**:

```
workloadAccounts:
  - name: Testing-Workload
    description: The Workload account
    email: <workload>@example.com  <----- UPDATE EMAIL ADDRESS
    organizationalUnit: Workloads
```

Ignoring an account from resource provisioning

As of v1.3.0 of the solution, you can designate an ignored OU in the `organization-config.yaml` file. If you would like the solution to ignore an account so that no resources are deployed to it, perform the following steps.

1. Create or identify an existing OU that you'd like to use to contain ignored accounts. For this example, our OU is named `Suspended`.
2. Move your account(s) that you'd like to ignore to this OU. Do this manually and not using the solution so that resources are not provisioned in the account. **Do not** follow the additional steps to update the `accounts-config.yaml` file or release changes to the Core pipeline if following the steps from the previous section.
3. Sign in to the AWS CodeCommit console.
4. Select the **aws-accelerator-config** repository.
5. Select the **organization-config.yaml** file.
6. Choose **Edit**.
7. In the `organizationalUnits` configuration block, append or edit the OU you've selected with the `ignore: true` flag:

```
organizationalUnits:
  - name: Security
  - name: Infrastructure
  - name: Testing
```

```
  - name: Production
  - name: Suspended
    ignore: true
```

8. Commit these changes, and then select the `accounts-config.yaml` file.
9. Choose **Edit**.
10. Comment out or remove the configuration(s) for the account(s) that you would like to be ignored. In this example, we are commenting out our **Testing-Workload** account with hash (#) marks so that it is no longer processed by the solution engine:

```
workloadAccounts:
  # - name: Testing-Workload
  #   description: The Workload account
  #   email: <workload>@example.com  <----- UPDATE EMAIL ADDRESS
  #   organizationalUnit: Workloads
```

11. Commit these changes. It is now safe to make additional configuration changes and release the Core pipeline without provisioning resources to the ignored account(s).

> **Note**
>
> Landing Zone Accelerator on AWS will not deploy any AWS CloudFormation stacks into ignored OU accounts, but AWS Organizations features could result in stacks being deployed within accounts in the ignored OUs.

When you use AWS Security Services, such as Macie, Security Hub, and GuardDuty with an AWS Organizations organization, security services are managed for every account in your organization. The solution configures auto-enable for new accounts, therefore when new accounts are added to your AWS Organizations, security services are automatically enabled.

Closing an account

To close an account that is managed by the solution, use one of the following procedures based off of your deployment type:

For Control Tower deployments, use the following procedure:

1. Unmanage the account.
2. Once the account is unmanaged, move it under an ignored OU.

3. Release changes to run the pipeline.
4. Once pipeline run is successful, close the account.

For AWS Organizations deployments, use the following procedure:

1. Move the account under an ignored OU.
2. Release changes to run the pipeline.
3. Once pipeline run is successful, close the account.

For all deployments, follow the steps in Ignoring an account from resource provisioning.

Adding a Service Control Policy (SCP)

We recommend creating a new directory in your **aws-accelerator-config** repository to store SCPs. Follow these steps to add a custom SCP to your multi-account environment.

1. Sign in to the AWS CodeCommit console.
2. Select the **aws-accelerator-config** repository.
3. Choose **Create file**.
4. Copy and paste the following policy into the blank box:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "NoInternet",
      "Effect": "Deny",
      "Action": [
        "ec2:CreateInternetGateway",
        "ec2:CreateEgressOnlyInternetGateway"
      ],
      "Resource": "*",
      "Condition": {
        "ArnNotLike": {
          "aws:PrincipalARN": [
            "arn:${PARTITION}:iam::*:role/${ACCELERATOR_PREFIX}-*",
            "arn:${PARTITION}:iam::*:role/AWSControlTowerExecution",
            "arn:${PARTITION}:iam::*:role/cdk-accel-*"
```

```
          ]
        }
      }
    }
  ]
}
```

5. For **File name**, save the file as `service-control-policies/block-internet.json`.
6. Enter the **Author name** and **Email address**.
7. Choose **Commit changes**.
8. Select the **organization-config.yaml** file.
9. Choose **Edit**.
10. In the `serviceControlPolicies` configuration block, append the SCP definition. For example, to add the `block-internet` SCP to the `Testing` OU, your configuration would look like the following sample:

```
serviceControlPolicies:
  - name: BlockInternetAccess
    description: >
      Blocks creating Internet gateways
    policy: service-control-policies/block-internet.json
    type: customerManaged
    deploymentTargets:
      organizationalUnits:
        - Testing
```

Adding an AWS Config rule

You can add managed and custom AWS Config rules to account(s) and OU(s) within your organization. Use the following steps to add a rule to your solution configuration:

1. Sign in to the AWS CodeCommit console.
2. Select the **aws-accelerator-config** repository.
3. Open the **security-config.yaml** file.
4. Add the following lines to the **awsConfig** configuration block of the **security-config.yaml** file. This example adds the managed rule IAM_USER_GROUP_MEMBERSHIP_CHECK to all accounts managed by the solution:

```
awsConfig:
  enableConfigurationRecorder: true
  enableDeliveryChannel: true
  ruleSets:
    - deploymentTargets:
        organizationalUnits:
          - Root
      rules:
        - name: accelerator-iam-user-group-membership-check
          complianceResourceTypes:
            - AWS::IAM::User
          identifier: IAM_USER_GROUP_MEMBERSHIP_CHECK
```

Central Security Services

This solution supports the concept of delegated administration for security services such as AWS Security Hub, Amazon GuardDuty, Amazon Macie, Amazon Detective, and AWS Audit Manager. The solution simplifies configuring your delegated administrator account and sets up organization member accounts to forward their findings to a central location. Use the following steps to activate central security services in the solution configuration.

1. Sign in to the AWS CodeCommit console.
2. Select the **aws-accelerator-config** repository.
3. Open the **security-config.yaml** file.
4. Add the following lines to the `centralSecurityServices` configuration block. This example turns on all supported central security services (except Amazon Detective):

> **Important**
>
> Amazon Detective isn't activated in our configuration sample due to a prerequisite that Amazon GuardDuty is activated for 48 hours before activating Amazon Detective. Refer to Amazon Detective prerequisites and recommendations in the *Amazon Detective User Guide* for additional information.

Note

When activating Audit Manager, the service creates an Amazon S3 bucket in your organization's Audit account specifically for Audit Manager reports. This is created for scenarios where third-party auditors require access to reports. This is a deviation from other central security services, which send logs to the central logging bucket in your organization's Log Archive account.

```
centralSecurityServices:
  delegatedAdminAccount: Audit
  ebsDefaultVolumeEncryption:
    enable: true
    excludeRegions: []
  s3PublicAccessBlock:
    enable: true
    excludeAccounts: []
  macie:
    enable: true
    excludeRegions: []
    policyFindingsPublishingFrequency: FIFTEEN_MINUTES
    publishSensitiveDataFindings: true
  guardduty:
    enable: true
    excludeRegions: []
    s3Protection:
      enable: true
      excludeRegions: []
    exportConfiguration:
      enable: true
      destinationType: S3
      exportFrequency: FIFTEEN_MINUTES
  auditManager:
    enable: true
    excludeRegions: []
    defaultReportsConfiguration:
      enable: true
      destinationType: S3
  detective:
    enable: false
```

```
    excludeRegions: []
  securityHub:
    enable: true
    regionAggregation: true
    excludeRegions: []
    standards:
      - name: AWS Foundational Security Best Practices v1.0.0
        enable: true
        controlsToDisable:[]
      - name: PCI DSS v3.2.1
        enable: true
        controlsToDisable: []
      - name: CIS AWS Foundations Benchmark v1.2.0
        enable: true
        controlsToDisable:[]
```

Adding an AWS Transit Gateway

Use the following steps to deploy a Transit Gateway to your multi-account environment.

1. Sign in to the AWS CodeCommit console.
2. Select the **aws-accelerator-config** repository.
3. Open the **network-config.yaml** file.
4. Add the following lines to the `transitGateways` configuration block. This example adds a Transit Gateway with an AWS Resource Access Manager (AWS RAM) share for the entire organization to the Network account in the US East (N. Virginia) Region:

```
transitGateways:
  - name: Network-Main
    account: Network
    region: us-east-1
    shareTargets:
      organizationalUnits:
        - Root
    asn: 65521
    dnsSupport: enable
    vpnEcmpSupport: enable
    defaultRouteTableAssociation: disable
    defaultRouteTablePropagation: disable
    autoAcceptSharingAttachments: enable
```

```
    routeTables:
      - name: Network-Main-Core
        routes: []
      - name: Network-Main-Segregated
        routes: []
      - name: Network-Main-Shared
        routes: []
      - name: Network-Main-Standalone
        routes: []
```

Adding an Amazon VPC

Use the following steps to deploy a VPC to your multi-account environment:

1. Sign in to the AWS CodeCommit console.
2. Select the **aws-accelerator-config** repository.
3. Open the **network-config.yaml** file.
4. Add the following lines to the `vpcs` configuration block. This example adds a VPC with routes for the Transit Gateway created in Adding an AWS Transit Gateway.

```
vpcs:
  - name: Network-Example
    account: Network
    region: us-east-1
    cidrs:
      - 10.1.0.0/22
    internetGateway: false
    enableDnsHostnames: true
    enableDnsSupport: true
    instanceTenancy: default
    routeTables:
      - name: Network-Endpoints-Tgw-A
        routes: []
      - name: Network-Endpoints-Tgw-B
        routes: []
      - name: Network-Endpoints-A
        routes:
          - name: TgwRoute
            destination: 0.0.0.0/0
            type: transitGateway
            target: Network-Main
```

```
      - name: Network-Endpoints-B
        routes:
          - name: TgwRoute
            destination: 0.0.0.0/0
            type: transitGateway
            target: Network-Main
    subnets:
      - name: Network-Endpoints-A
        availabilityZone: a
        routeTable: Network-Endpoints-A
        ipv4CidrBlock: 10.1.0.0/24
      - name: Network-Endpoints-B
        availabilityZone: b
        routeTable: Network-Endpoints-B
        ipv4CidrBlock: 10.1.1.0/24
      - name: Network-EndpointsTgwAttach-A
        availabilityZone: a
        routeTable: Network-Endpoints-Tgw-A
        ipv4CidrBlock: 10.1.3.208/28
      - name: Network-EndpointsTgwAttach-B
        availabilityZone: b
        routeTable: Network-Endpoints-Tgw-B
        ipv4CidrBlock: 10.1.3.224/28
    transitGatewayAttachments:
      - name: Network-Endpoints
        transitGateway:
          name: Network-Main
          account: Network
        routeTableAssociations:
          - Network-Main-Shared
        routeTablePropagations:
          - Network-Main-Core
          - Network-Main-Shared
          - Network-Main-Segregated
        subnets:
          - Network-EndpointsTgwAttach-A
          - Network-EndpointsTgwAttach-B
```

Important

Landing Zone Accelerator on AWS does not support creation of users and groups in Identity Center. Users and groups are required to be created in Identity Center before the solution can use those principals for permission set assignments.

Adding an IAM Identity Center permission set

Use the following steps to deploy an IAM Identity Center permission set to your multi-account environment:

1. Sign in to the AWS CodeCommit console.
2. Select the **aws-accelerator-config** repository.
3. Open the **iam-config.yaml** file.
4. Add the `identityCenter` configuration block. This example creates two predefined permission sets that use a single AWS managed policy and modifies the session duration.

```
identityCenter:
  name: identityCenter
  identityCenterPermissionSets:
    - name: PowerAccessUser
      policies:
        awsManaged:
          - arn:aws:iam::aws:policy/PowerUserAccess
        customerManaged: []
      sessionDuration: 60
    - name: ViewOnlyAccess
      policies:
        awsManaged:
          - arn:aws:iam::aws:policy/job-function/ViewOnlyAccess
        customerManaged: []
      sessionDuration: 60
```

Important

Landing Zone Accelerator on AWS does not support creation of users and groups in Identity Center. Users and groups are required to be created in Identity Center before the solution can use those principals for permission set assignments.

Working with solution-specific variables

This section provides information about working with variables.

Policy replacement variables

This solution supports the concept of environment variables in policy documents. These variables are processed at CDK application runtime and are replaced with contextual values based on the runtime environment and, if applicable, user-defined configurations.

The following policy types support variable replacements:

- AWS Organization policies (SCPs, tag policies, backup policies)
- IAM policies
- AWS KMS key policies
- VPC endpoint policies (available as of v1.5.0)

Policy replacement variables support the following variable keys:

- `${ACCELERATOR_DEFAULT_PREFIX_SHORTHAND}` - Short version of the prefix applied to solution-provisioned resources (first four letters, capitalized)
- `${ACCELERATOR_PREFIX_ND}` - Prefix applied to solution-provisioned resources without dashes
- `${ACCELERATOR_PREFIX_LND}` - Prefix applied to solution-provisioned resources in lowercase and without dashes
- `${ACCELERATOR_PREFIX}` - Prefix applied to solution-provisioned resources
- `${ACCELERATOR_SSM_PREFIX}` - Prefix applied to solution-provisioned Parameter Store parameters (includes leading forward slash (/))
- `${ACCOUNT_ID}` - Account ID the policy is deployed to

- ${AUDIT_ACCOUNT_ID} - Account ID of the Audit account
- ${ACCELERATOR_CENTRAL_LOGS_BUCKET_NAME} - Central Log bucket name
- ${HOME_REGION} - AWS Region the solution is deployed to
- ${LOGARCHIVE_ACCOUNT_ID} - Account ID of the Log Archive account
- ${MANAGEMENT_ACCOUNT_ACCESS_ROLE} - Role name that the solution uses for cross-account access, for example, AWSControlTowerExecution
- ${MANAGEMENT_ACCOUNT_ID} - Account ID of the Management account
- ${PARTITION} - Partition that the policy is deployed to
- ${ORG_ID} - The ID of the AWS Organization
- ${REGION} - AWS Region that the policy is deployed to

> **Note**
>
> As of version 1.4.0 of this solution, the ACCELERATOR_PREFIX and associated variables listed are derived from the **Accelerator Resource name Prefix** value input in the installer stack parameter . For previous versions, the prefix value was AWSAccelerator.

The following policy sample shows how you can implement the variables:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAllAWSServicesExceptBreakglassRoles",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "ArnNotLike": {
          "aws:PrincipalArn": [
            "arn:${PARTITION}:iam::*:role/${MANAGEMENT_ACCOUNT_ACCESS_ROLE}",
            "arn:${PARTITION}:iam::*:role/aws*",
            "arn:${PARTITION}:iam::*:role/${ACCELERATOR_PREFIX}*",
            "arn:${PARTITION}:iam::*:role/cdk-accel-*"
          ]
        }
      }
```

```
    }
  ]
}
```

Parameter Store reference variables

This solution provides dynamic lookups from the Parameter Store within configuration files. This feature primarily meets two customer needs:

- Customers wishing to reuse a single generic Landing Zone Accelerator on AWS configuration across multiple Landing Zone Accelerator on AWS deployments
- Customers wishing to simplify their configuration files and decrease the number of manual configuration edits

> **Note**
>
> Parameter Store lookup functionality was made available starting with the v1.5.0 release.

You can leverage this pattern by creating an additional, optional configuration file in the configuration repository named `replacements-config.yaml`.

```
globalReplacements:
 - key: SnsEmail
   path: /accelerator/replacements/SnsEmailAddress
```

In the above example, we have a single *replacement* defined:

- The configuration files use the `key` in `SnsEmail` to indicate which value to replace.
- The path of `/accelerator/replacements/SnsEmailHigh` describes the Parameter Store path where the corresponding replacement value exists.

> **Note**
>
> Parameter Store parameters used for replacements must exist in the `homeRegion` of the management account.

You must explicitly create the Systems Manager parameter `/accelerator/replacements/SnsEmailHigh`. Create this Systems Manager parameter with an appropriate replacement value through Landing Zone Accelerator on AWS or by following this guide.

```
aws ssm put-parameter \
    --name "/accelerator/replacements/SnsEmailAddress" \
    --value "example_email@example.com" \
    --type String \
```

After you create the Systems Manager parameter, you can reference the value from within the configuration files. For example, we might create a new Amazon SNS topic in the `global-config.yaml` file where the email address value uses the replacement previously defined:

```
snsTopics:
  topics:
   - name: SampleTopic
      emailAddresses:
        - {{SnsEmail}}
```

Note

All YAML configuration files use replacements except `accounts-config.yaml`.

This feature also includes functionality to look up account IDs based on the account name without the use of Systems Manager parameters:

```
parameters:
 - name: TrustedAccounts
   value: {{account Management}}
```

The double-curly brace notation is reserved for usage in the LZA configuration files by LZA replacements and SSM dynamic references. Any instances of this syntax that do not follow the SSM dynamic reference syntax or exist in the `replacements-config.yaml` file will throw an error during configuration validation.

If you must use this syntax in your configuration files, you can skip static configuration validation by setting `cdkOptions.skipStaticValidation` to `true` in the `global-config.yaml` file.

$ACCEL_LOOKUP reference variable

This solution supports the concept of referencing variables within configuration files. This lookup is processed at CDK application runtime and are replaced with contextual values based on the runtime environment and, if applicable, user-defined configurations. These variables are used in specific cases within the configuration files for dynamically looking up specific values and are scoped to only the following use cases:

Systems Manager automation document variables

The following variables allow for defining Systems Manager Automation documents in the solution `security-config.yaml`:

- `${ACCEL_LOOKUP::KMS}` - Resolves to the main Amazon S3 AWS KMS key ID for the solution deployment
- `${ACCEL_LOOKUP::Bucket:elbLogs}` - Resolves to the centralized logging Amazon S3 bucket for Elastic Load Balancing
- `${ACCEL_LOOKUP::InstanceProfile:EC2-Default-SSM-AD-Role}` - Used for custom remediation AWS Config rules to allow automatic remediation to be performed using that role on certain rules

Launch template variables

The following variable allow for referencing local or public image IDs for Amazon EC2 launch templates in the solution `customization-config.yaml`:

- `${ACCEL_LOOKUP::ImageId:/path/to/ssm/parameter}` - Local or public SSM parameter store lookup for Image ID

Amazon EC2 firewall configuration variables

As of v1.5.0 of this solution, the following variables are available when deploying a custom third-party Amazon EC2 firewall in the solution `customization-config.yaml`. These variables require your firewall vendor to support bootstrapping the instance using a configuration file from an S3 bucket. For more information on the specific implementation, see the Ec2FirewallInstanceConfig or Ec2FirewallAutoScalingGroupConfig references.

Hostname replacement

Use this variable to lookup up the name of the firewall instance as defined in `customization-config.yaml`.

> **Note**
>
> This replacement is only available for standalone firewall instances defined under Ec2FirewallInstanceConfig. Firewall autoscaling groups are not supported.

`${ACCEL_LOOKUP::EC2:INSTANCE:HOSTNAME}`

VPC replacements

Use these variables to look up metadata about the VPC that the firewall is deployed to.

> **Note**
>
> `${ACCEL_LOOKUP::EC2:VPC:<METADATA_TYPE>_<INDEX>} - <METADATA_TYPE>` is a type listed below, and `<INDEX>` is the index number of the VPC CIDR range based on the order in which the CIDR was associated with the VPC. Index numbering is zero-based, so the index of the primary range associated with the VPC is `0`.

Metadata Type	Description
CIDR	The VPC CIDR range in CIDR notation (specifically, 10.0.0.0/16)
NETMASK	The network mask of the VPC CIDR (specifically, 255.255.0.0)
NETWORKIP	The network address of the VPC CIDR (specifically, 10.0.0.0)
ROUTERIP	The VPC router address of the VPC CIDR (specifically, 10.0.0.1)

Example usage: `${ACCEL_LOOKUP::EC2:VPC:CIDR_0}` - translates to the primary CIDR range of the VPC.

Subnet replacements

Use these variables to look up metadata about subnets in the VPC the firewall is deployed to.

> **Note**
>
> `${ACCEL_LOOKUP::EC2:SUBNET:<METADATA_TYPE>:<SUBNET_NAME>}` - `<METADATA_TYPE>` is a type listed below, and `<SUBNET_NAME>` is the logical name of the subnet as defined in your solution `network-config.yaml`.

Metadata Type	Description
CIDR	The subnet CIDR range in CIDR notation (specifically, 10.0.0.0/16)
NETMASK	The network mask of the subnet (specifically, 255.255.0.0)
NETWORKIP	The network address of the subnet (specific ally, 10.0.0.0)
ROUTERIP	The VPC router address of the subnet (specific ally, 10.0.0.1)

Example usage: `${ACCEL_LOOKUP::EC2:SUBNET:CIDR:firewall-subnet-a}` - translates to the CIDR range of a subnet named `firewall-subnet-a` in the VPC.

Network interface IP replacements

Use these variables to look up public and private IP addresses assigned to the firewall network interfaces.

Note

These replacements are only available for standalone firewall instances defined under Ec2FirewallInstanceConfig. Firewall autoscaling groups are not supported.

Note

`${ACCEL_LOOKUP::EC2:ENI_<ENI_INDEX>:<IP_TYPE>_<IP_INDEX>}` - `<ENI_INDEX>` is the device index of the network interface as defined in the firewall launch template, `<IP_TYPE>` is either a public or private IP of the interface, and `<IP_INDEX>` is the index of the interface IP address. Index numbering is zero-based, so the primary interface of the instance is `0` and its primary IP address is also `0`.

IP Type	Description
PRIVATEIP	A private IP address associated with the interface
PUBLICIP	A public IP address associated with the interface

Example usage: `${ACCEL_LOOKUP::EC2:ENI_0:PRIVATEIP_0}` - translates to the primary private IP address of the primary network interface.

Network interface subnet replacements

Use these variables to look up metadata about the subnet that a network interface is deployed to.

Note

These replacements are only available for standalone firewall instances defined under `Ec2FirewallInstanceConfig` . Firewall autoscaling groups are not supported.

`${ACCEL_LOOKUP::EC2:ENI_<ENI_INDEX>:SUBNET_<METADATA_TYPE>} - <ENI_INDEX>` is the device index of the network interface as defined in the firewall launch template and `<METADATA_TYPE>` is a type listed below. Index numbering is zero-based, so the primary interface of the instance is `0`.

Metadata Type	Description
CIDR	The subnet CIDR range in CIDR notation (specifically, 10.0.0.0/16)
NETMASK	The network mask of the subnet (specifically, 255.255.0.0)
NETWORKIP	The network address of the subnet (specific ally, 10.0.0.0)
ROUTERIP	The VPC router address of the subnet (specific ally, 10.0.0.1)

Example usage: `${ACCEL_LOOKUP::EC2:ENI_0:SUBNET_CIDR}` - translates to the subnet CIDR range of the primary network interface.

Site-to-Site VPN replacements

Use these variables to look up metadata about site-to-site VPNs that are created between a firewall network interface and AWS gateway.

> **Note**
>
> These replacements are only available for standalone firewall instances defined under Ec2FirewallInstanceConfig. To utilize the replacements, there must be a CustomerGatewayConfig that references a firewall instance network interface as its `ipAddress` property. You can reference any VPN connection `name` configured under this customer gateway item for dynamic configuration of the VPN connection(s) on your firewall. Firewall autoscaling groups are not supported for this replacement type.

Note

`${ACCEL_LOOKUP::EC2:VPN:<METADATA_TYPE>:<VPN_NAME>} - <METADATA_TYPE>` is a type listed below, and `<VPN_NAME>` is the logical name of the VPN connection as defined in your solution `network-config.yaml`. For metadata types with index values, index numbering is zero-based, so the primary tunnel of the VPN connection is `0`.

Metadata Type	Description
AWS_BGPASN	The BGP autonomous system number (ASN) of the AWS gateway device (specifically, 65000)
CGW_BGPASN	The BGP autonomous system number (ASN) of the customer gateway device (specifically, 65000)
CGW_OUTSIDEIP	The outside (public) IP address of the customer gateway device
AWS_INSIDEIP_<TUNNEL_INDEX>	The inside (link-local) IP address of the AWS gateway device, where <TUNNEL_INDEX> is the index number of the VPN tunnel (specific ally, 169.254.100.1)
CGW_INSIDEIP_`<TUNNEL_INDEX>`	The inside (link-local) IP address of the customer gateway device, where <TUNNEL_I NDEX> is the index number of the VPN tunnel (specifically, 169.254.100.2)
AWS_OUTSIDEIP_`<TUNNEL_INDEX>`	The outside (public) IP address of the AWS gateway device, where <TUNNEL_INDEX> is the index number of the VPN tunnel
INSIDE_CIDR_`<TUNNEL_INDEX>`	The inside (link-local) CIDR range of the tunnel, where <TUNNEL_INDEX> is the index number of the VPN tunnel

Metadata Type	Description
INSIDE_NETMASK_`<TUNNEL_INDEX>`	The inside (link-local) subnet mask of the tunnel, where <TUNNEL_INDEX> is the index number of the VPN tunnel (specifically, 255.255.255.252)
PSK_`<TUNNEL_INDEX>`	The pre-shared key of the tunnel, where <TUNNEL_INDEX> is the index number of the VPN tunnel

Example usage: `${ACCEL_LOOKUP::EC2:VPN:AWS_OUTSIDEIP_0:accelerator-vpn}` - translates to the AWS-side public IP of the primary VPN tunnel for a VPN named `accelerator-vpn`.

Firewall configuration S3 bucket replacement

To support the above dynamic configuration replacements, the solution creates an S3 bucket in the target account from which the Amazon EC2 firewall can read the resulting configuration file. To dynamically target this bucket in your Amazon EC2 firewall user data script, you may use the following variable:

```
${ACCEL_LOOKUP::S3:BUCKET:firewall-config}
```

Customer gateway IP replacement

To support the previous dynamic Amazon EC2 firewall customer gateway and site-to-site VPN connection creation, use the following variable as the CustomerGatewayConfig `ipAddress` property:

`${ACCEL_LOOKUP::EC2:ENI_<ENI_INDEX>:<FIREWALL_INSTANCE_NAME>}`, where <ENI_INDEX> is the device index of the network interface as defined in the firewall launch template and <FIREWALL_INSTANCE_NAME> is the name of the firewall instance as defined in EC2FirewallInstanceConfig. Index numbering is zero-based, so the primary interface of the instance is `0`.

> **Note**
>
> This reference variable is only available for Amazon EC2 firewall network interfaces that have the `associateElasticIp` property set to `true`. It is not supported for Amazon EC2 firewall autoscaling groups.

Example usage: `${ACCEL_LOOKUP::EC2:ENI_0:accelerator-firewall}` - translates to the primary public IP address of the primary network interface of a firewall named `accelerator-firewall`.

Working with existing landing zones

This solution can integrate with and manage your accounts and OUs in existing landing zone environments. Remember the following when deploying the solution to an existing environment.

Existing accounts and OUs

Landing Zone Accelerator on AWS requires that all accounts in the organization are defined in the `accounts-config.yaml` file unless they are contained in an ignored OU. Additionally, it requires that all OUs in the organization are defined in the `organization-config.yaml` file.

> **Note**
>
> If you're using the solution-provisioned configuration repository, your existing environment might not match the base configuration applied to the configuration files when the solution is initially deployed. This will cause your Core pipeline to fail environment validation during the **Prepare** stage until the configuration files are updated with these details.

As of version 1.3.1 of this solution, you can provide your own configuration repository during installation of the solution to overcome this initial pipeline failure. If you pre-load the repository with your landing zone's account and OU configuration, the solution will use it on the initial Core pipeline run. For more information, refer to the installer stack parameters.

For more information on adding an existing account to the solution, see Adding an existing account.

For more information on adding OUs to the solution configuration, see Adding an organizational unit (OU).

For information on troubleshooting environment validation errors, see Problem: Account enrollment and environment validation failures.

Existing resources

Most configurable services and features in this solution don't currently support the import and management of your existing resources in the Core pipeline. Therefore, most resources defined in the solution configuration files will deploy new resources to your environment.

> **Note**
>
> In some instances, deploying these new resources can cause conflicts with resource quotas or existing configurations in your environment. Consider this when deploying organization-wide configurations using the solution, such as centralized security services and organizational policies.

If you want to migrate from your existing resources to resources managed by the solution, do the following:

1. Identify a change window where it is acceptable for the resource(s) to be unavailable.
2. Deactivate the existing service(s) and resource(s).
3. Refer to the Services, Features, and Configuration References section of the solution's GitHub Pages website to configure the service in the solution configuration files.
4. Release a change to the Core pipeline.

If you encounter errors related to existing resource conflicts during your solution deployment, refer to Troubleshooting.

Existing service control policies (SCPs)

Existing SCPs in your environment can cause deployment of accelerator resources to fail. If the solution encounters an explicit deny from an SCP, ensure that the conditions block of your statements (if applicable) are updated to allow actions from the solution administrative role and roles using the **Accelerator Resource name prefix** parameter.

Alternatively, you can migrate the management of your SCPs to the solution. This provides you the added benefit of using our policy replacement variables in your policy documents to reference the aforementioned role names. For more information, see Adding a service control policy (SCP).

Configuration file best practices

Landing Zone Accelerator on AWS aims to abstract away most aspects of managing its underlying infrastructure as code (IaC) templates from the user. This is facilitated through the use of its configuration files to define your landing zone environment. However, it is important to keep some common IaC best practices in mind when modifying your configuration to avoid pipeline failure scenarios.

Manage accelerator resources strictly through configuration

Introducing manual, out-of-band changes to accelerator-managed resources in your environment can cause unexpected pipeline failures or resource updates to occur when you release updates using your Core pipeline. CDK and CloudFormation don't keep a record of these out-of-band changes, which leads to drift between the solution-generated template and actual resource configuration state. For this reason, we recommend managing changes to your landing zone resources strictly through the solution configuration files. For features and configurations that aren't supported by the solution, we recommend using custom stacks in the `customizations-config.yaml` file or externally-managed IaC to manage those updates.

Understanding the name property

Several configuration objects throughout the solution configuration files make use of the `name` property. This property has two important uses:

- It allows you to logically name and reference a configuration item in other configuration items. This allows you to easily build relationships between separate configuration items if there are dependencies between them.
- It is used to generate unique logical IDs for resources in the solution CDK application when it is synthesizing the underlying CloudFormation templates.

> **Important**
>
> Use caution when modifying the `name` property of any configuration item. In doing so, any other configuration items that reference that `name` value must also be updated.

Additionally, changing a `name` property value is treated as a resource replacement operation when the Core pipeline runs. Assess downstream dependencies before performing this action.

Managing resource create, update, and delete actions

This solution's resource orchestration follows the same create, update, and delete workflow as standard CloudFormation templates. These actions are defined in How does AWS CloudFormation work? in the *AWS CloudFormation User Guide*. This section provides specifics related to translating these behaviors to accelerator configuration file modifications.

Adding a net new configuration block to an accelerator configuration file causes a new resource to be created in the defined AWS Regions and target accounts or OUs. For the configuration items that use the `deploymentTargets` property, your configured resources will be automatically replicated to every Region configured under the `enabledRegions` property of the `global-config.yaml` file. You can explicitly remove target accounts and Regions for `deploymentTargets` by using the `excludedAccounts` and `excludedRegions` properties, respectively.

If you modify or remove a property value of an existing configuration item, an update of the previously deployed resource will occur. Specific update behaviors and constraints are dependent on the underlying resource that is being updated. Before you make updates, we recommend reviewing the configuration reference in the Services, Features, and Configuration References section of the solution's GitHub Pages website for property updates that result in resource replacement. Making updates that cause resource replacement prior to removing a resource's downstream dependencies can result in a dependency error during resource replacement.

Removing a configuration block from your configuration results in resource deletion. Please be aware that removing a configuration block prior to removing a resource's downstream dependencies may result in a dependency error during resource deletion.

Downstream resource dependencies

This solution uses a stack layering strategy for building your landing zone environment. This means that resources provisioned in later stages of the Core pipeline might depend on resources created in earlier stages. When updating the solution configuration files, consider downstream resource

dependencies. For additional information about strategizing your resource updates, see Managing resource dependencies.

Existing resources in your environment

If you want to migrate an existing landing zone to use Landing Zone Accelerator on AWS, you might have conflicts between your existing resources and your configured resources in the solution configuration files. See the details in Working with existing landing zones for more information.

Managing resource dependencies

This solution uses a layered approach to deploying your landing zone infrastructure resources. Each stage after the **Build** stage produces at least one CloudFormation stack, with some stages producing multiple stacks. For more information on the stage ordering and stacks produced by each stage, refer to the details in Core pipeline.

This stage-based approach provides the flexibility to manage the ordering of resource deployment. It also maximizes the number of resources that you can deploy for each stage while logically grouping similar resources. Finally, it provides the solution a consistent means to reference resources created in previous stages, including those created in different AWS accounts and Regions. When required, the solution uses lookup methods to retrieve details for resources with upstream dependencies.

Updating existing resources might require knowledge of the stack ordering and resource dependencies between them. The following section provides detail about identifying resources with dependencies in your accelerator configuration files.

Identifying resources with dependencies

The following sections provide example VPC configurations to demonstrate how nested configurations and logical references create a downstream dependency tree for your resources. A helpful mental model to follow when updating resources with dependency trees is to start with the lowest nested item. In these examples, interface endpoints are the lowest nested item since they are dependent on subnets that, in turn, are dependent on the base VPC configuration item.

Modifying (including deleting) the lowest nested item first doesn't result in dependency issues. However, if you the change properties of the subnet or VPC, you might encounter issues if those modifications cause the resource to be replaced (for example, changing the CIDR property of the resource) or deleted. Modifications that don't result in resource replacement shouldn't

impact downstream dependencies. Refer to the Services, Features, and Configuration References section of the solution's GitHub Pages website before modifying configurations with downstream dependencies to determine the impact of updating a property value.

Configuration item containing nested configuration items

The following is an example basic VPC configuration and its nested dependencies. Comments are denoted with hash (#) marks:

```
vpcs:
  - name: accelerator-vpc
    account: Network
    region: us-east-1
    cidrs:
      - 10.0.0.0/24
    enableDnsHostnames: true
    enableDnsSupport: true
    instanceTenancy: default
    routeTables:
      # Items under this block are dependent
      # on the base VPC configuration item
      - name: accelerator-default
        routes: []
    subnets:
      # Items under this block are dependent
      # on the base VPC configuration item
      - name: accelerator-subnet-a
        availabilityZone: a
        routeTable: accelerator-default
        ipv4CidrBlock: 10.0.0.0/26
      - name: accelerator-subnet-b
        availabilityZone: b
        routeTable: accelerator-default
        ipv4CidrBlock: 10.0.0.64/26
```

Configuration items referencing the logical names of other configuration items

Building from our previous example, the following adds interface endpoints to the VPC. Comments are denoted with hash (#) marks:

```
vpcs:
  - name: accelerator-vpc
```

```
    account: Network
    region: us-east-1
    cidrs:
      - 10.0.0.0/24
    enableDnsHostnames: true
    enableDnsSupport: true
    instanceTenancy: default
    routeTables:
      - name: accelerator-default
        routes: []
    subnets:
      - name: accelerator-subnet-a
        availabilityZone: a
        # Referencing the route table name
        routeTable: accelerator-default
        ipv4CidrBlock: 10.0.0.0/26
      - name: accelerator-subnet-b
        availabilityZone: b
        # Referencing the route table name
        routeTable: accelerator-default
        ipv4CidrBlock: 10.0.0.64/26
    interfaceEndpoints:
      # Referencing a policy name defined
      # in another config block
      defaultPolicy: Default
      endpoints:
        - service: ec2
      # Referencing subnet names defined
      # under this VPC configuration.
      subnets:
        - accelerator-subnet-a
        - accelerator-subnet-b
```

Modifying resources with dependencies

If your update or deletion will impact downstream resources, you can use the following strategies to avoid dependency errors.

Multiple Core pipeline runs

1. Starting with the lowest nested item in your dependency tree, remove or comment out the items that will be affected by your change. The following example comments out interface endpoints from the previous section:

```
vpcs:
  - name: accelerator-vpc
    account: Network
    region: us-east-1
    cidrs:
      - 10.0.0.0/24
    enableDnsHostnames: true
    enableDnsSupport: true
    instanceTenancy: default
    routeTables:
      - name: accelerator-default
        routes: []
    subnets:
      - name: accelerator-subnet-a
        availabilityZone: a
        routeTable: accelerator-default
        ipv4CidrBlock: 10.0.0.0/26
      - name: accelerator-subnet-b
        availabilityZone: b
        routeTable: accelerator-default
        ipv4CidrBlock: 10.0.0.64/26
    # interfaceEndpoints:
    #  defaultPolicy: Default
    #  endpoints:
    #    - service: ec2
    #  subnets:
    #    - accelerator-subnet-a
    #    - accelerator-subnet-b
```

2. Release your changes to the Core pipeline. This removes the interface endpoints from the VPC subnets.
3. Modify the subnet CIDRs. The following example makes the subnet ranges smaller:

```
vpcs:
  - name: accelerator-vpc
    account: Network
    region: us-east-1
    cidrs:
      - 10.0.0.0/24
    enableDnsHostnames: true
    enableDnsSupport: true
    instanceTenancy: default
```

```
    routeTables:
      - name: accelerator-default
        routes: []
    subnets:
      - name: accelerator-subnet-a
        availabilityZone: a
        routeTable: accelerator-default
        ipv4CidrBlock: 10.0.0.0/27
      - name: accelerator-subnet-b
        availabilityZone: b
        routeTable: accelerator-default
        ipv4CidrBlock: 10.0.0.32/27
    # interfaceEndpoints:
    #  defaultPolicy: Default
    #  endpoints:
    #    - service: ec2
    #  subnets:
    #    - accelerator-subnet-a
    #    - accelerator-subnet-b
```

4. Release your changes to the Core pipeline. This replaces the existing /26 subnets with new /27 subnets.
5. If you want to redeploy your lowest nested item, remove the comment from the item and release the change to the Core pipeline again. In the previous example, the interface endpoint would be deployed to the newly-provisioned subnets.

Using the solution command line interface (CLI)

Using the solution CLI can help you perform the steps in the previous section more quickly. Rather than performing a full pipeline run for each change, you can run targeted changes to the specific stage, account, and AWS Region that requires the changes.

> **Note**
>
> Using the solution CLI requires a development toolchain to be installed on your workstation. Our development dependencies and documentation for using the CLI are included in the Developer Guide section of the solution's GitHub Pages website.
> In our example, the stage for these changes is `network-vpc`. The stage you target for CLI changes might differ depending on the resources that you are targeting for updates.

1. Make the configuration changes from step 1 in the Multiple Core pipeline runs section.
2. Run the following CLI command, replacing <LOCAL_CONFIG_DIR> and <ACCOUNT_ID> with your environment details:

```
yarn run ts-node --transpile-only cdk.ts deploy --stage network-vpc --require-
approval any-change --config-dir <LOCAL_CONFIG_DIR> --partition aws --region us-
east-1 --account <ACCOUNT_ID>
```

   The CLI indicates when the stack deployment has completed, meaning your interface endpoints have been deleted.
3. Make the configuration changes from step 3 in the Multiple Core pipeline runs section.
4. Run the following CLI command again, replacing <LOCAL_CONFIG_DIR> and <ACCOUNT_ID> with your environment details:

```
yarn run ts-node --transpile-only cdk.ts deploy --stage network-vpc --require-
approval any-change --config-dir <LOCAL_CONFIG_DIR> --partition aws --region us-
east-1 --account <ACCOUNT_ID>
```

5. The CLI indicates when the stack deployment has completed, meaning your subnets have been replaced.
6. If you want to redeploy your lowest nested item, remove the comment from the item and run the CLI command again. In our example, the interface endpoint would be deployed to the newly-provisioned subnets.

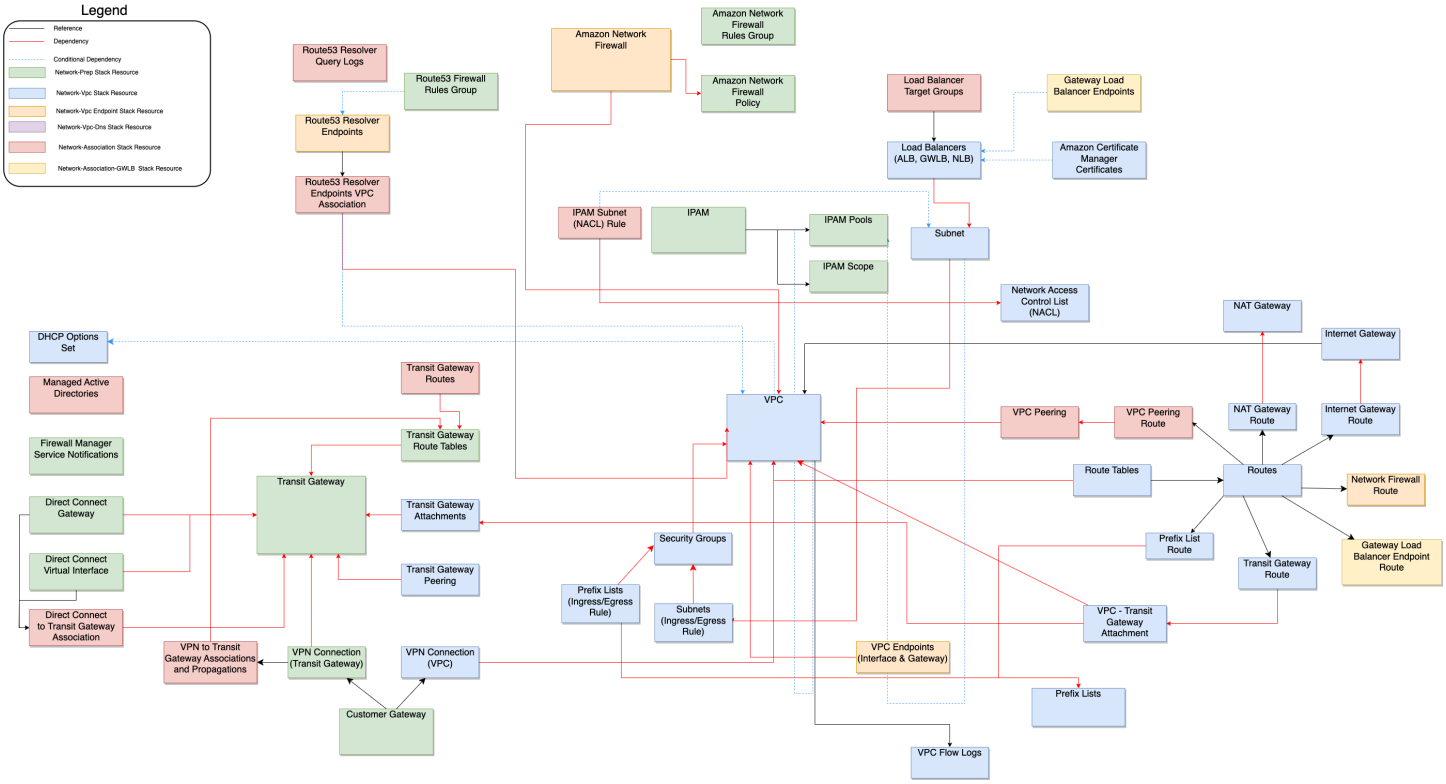
Strategizing network resource updates

This solution has multiple stages and stacks associated with building your core network topology. Many of the building blocks associated with landing zone networks have tightly coupled dependencies. Some of these dependencies for this solution cross stack barriers. For this reason, updating network resources using Landing Zone Accelerator on AWS requires additional knowledge of which stage and stack specific resources are deployed in.

The following diagram is a visual aid for understanding the network resources that the solution can deploy, and how their dependencies map together. When planning updates or removal of network resources with downstream dependencies, reference this dependency map along with the guidance in the previous sections of Managing resource dependencies.

> **Note**
>
> If the image is difficult to read on your screen, open it in a new tab.

Sample diagram of resource dependencies.



Developer guide

This section addresses the source code, configuration files, and administrator tasks for this solution.

Source code

Visit our GitHub repository to download the source files for this solution and to share your customizations with others. The Landing Zone Accelerator on AWS templates are generated using the AWS CDK. Refer to the README.md file for additional information.

Accessing solution outputs through Parameter Store

This solution provides configuration management for resources provisioned through Parameter Store. The solution records the following resources types and their respective Parameter Store paths.

Application resources

Metadata Type	Description	Path
Target Group ARN	The Amazon Resource Name (ARN) of the Target Group where `${0}${1}#` is replaced with the VPC name, and `${2}` is replaced with the target group name	`/application/targetGroup/ ${0}/${1}/${2}/arn`

AWS CloudFormation stacks

Metadata Type	Description	Path
AWS CloudFormation Stack ID	The solution CloudFormation stack ID where `${0}` is replaced with the stack name	`//stack-id`

AWS Organization resources

Metadata Type	Description	Path
Accelerator Service Control Policy ID	The ID of the Service Control Policy where `${0}` is replaced with the SCP name	`/organizations/scp/ ${0}/id`

Central Network resources

Metadata Type	Description	Path
VPC IP Address Manager ID	The ID of the VPC IP Address Manager (IPAM) where `${0}` is replaced with the IPAM name	`/network/ipam/ ${0}/id`
VPC IP Address Manager Pool ID	The ID of the VPC IP Address Manager (IPAM) Pool where `${0}` is replaced with the IPAM Pool name	`/network/ipam/pools/ ${0}/id`
VPC IP Address Manager Scope ID	The ID of the VPC IP Address Manager (IPAM) scope where `${0}` is replaced with the IPAM scope name	`/network/ipam/scopes/ ${0}/id`
Amazon Network Firewall ARN	The Amazon Resource Name (ARN) of the Amazon Network Firewall where `${0}$${1}#` is replaced with the network firewall name	`/network/vpc/ ${0}/networkFirewall/ ${1}/arn`
Amazon Network Firewall Policy ARN	The Amazon Resource Name (ARN) of the Amazon Network Firewall policy where `${0}`	`/network/networkFirewall/policies/ ${0}/arn`

Metadata Type	Description	Path
	is replaced with the network firewall policy name	
Amazon Network Firewall Rule Group ARN	The Amazon Resource Name (ARN) of the Amazon Network Firewall Rule Group where `${0}` is replaced with the rule group name	`/network/networkFirewall/ruleGroups/${0}/arn`

Direct Connect resources

Metadata Type	Description	Path
Direct Connect Virtual Interface (VIF) ID	The ID of the Direct Connect VIF where `${0}` is replaced with the Direct Connect gateway name; `${1}` is replaced with the VIF name	`/network/directConnectGateways/${0}/virtualInterfaces/${1}/id`
Direct Connect Gateway ID	The ID of the Direct Connect gateway where `${0}` is replaced with the Direct Connect gateway name	`/network/directConnectGateways/${0}/id`

Global Network resources

Metadata Type	Description	Path
ACM Certificate ARN	The Amazon Resource Name (ARN) of an ACM certificate where `${0}` is replaced with the certificate name	`/acm/${0}/arn`

Metadata Type	Description	Path
Prefix List ID	The ID of the prefix list where `${0}` is replaced with the prefix list name	`/network/prefixList/ ${0}/id`

IAM resources

Metadata Type	Description	Path
IAM Role ARN	The ARN of the IAM role where `${0}` is replaced with the IAM role name	`/iam/role/ ${0}/arn`
IAM Management Policy ARN	The ARN of the IAM managed policy where `{0}` is replaced with the IAM managed policy name	`/iam/policy/ ${0}/arn`
IAM Group ARN	The ARN of the IAM group where `${0}` is replaced with the IAM group name	`/iam/group/ ${0}/arn`
IAM User ARN	The ARN of the IAM user where `${0}` is replaced with the IAM user name	`/iam/user/ ${0}/arn`

Load Balancer resources

Metadata Type	Description	Path
Application Load Balancer ID	The ID of the Application Load Balancer (ALB) where `${0}` is replaced with the VPC name; `${1}` is replaced with the ALB name	`/network/vpc/ ${0}/alb/${1}/id`

Metadata Type	Description	Path
Network Load Balancer ID	The ID of the Network Load Balancer (NLB) where `${0}` is replaced with the VPC name; `${1}` is replaced with the NLB name	`/network/vpc/ ${0}/ nlb/${1}/id`
Gateway Load Balancer ARN	The ARN of the Gateway Load Balancer (GWLB) where `${0}` is replaced with the GWLB name	`/network/gwlb/ ${0}/ arn`
Gateway Load Balancer Endpoint Service ID	The ID of the GWLB service endpoint where `${0}$` is replaced with the GWLB name	`/network/gwlb/ ${0}/ endpointService/id`

Route 53 resources

Metadata Type	Description	Path
Route 53 DNS Firewall Rule Group ID	The ID of the Route 53 DNS firewall rule group ID where `${0}` is replaced with the DNS firewall rule group name.	`/network/route53Re solver/firewall/ru leGroups/ ${0}/id`
Interface Endpoint DNS name	The DNS name of the interface endpoint where `${0}` is replaced with the VPC name; `${1}` is replaced with the interface endpoint service name.	`/network/vpc/ ${0}/ endpoints/ ${1}/dns`
Interface Endpoint Hosted Zone ID	The hosted zone ID of the interface endpoint `${0}` is replaced with the VPC name; `${1}` is replaced with the	`/network/vpc/ ${0}/ endpoints/ ${1}/ hostedZoneId`

Metadata Type	Description	Path
	interface endpoint service name.	
Route 53 Private Hosted Zone ID	The ID of the private hosted zone where `${0}` is replaced with the VPC name; `${1}` is replaced with the interface endpoint service name.	`/network/vpc/ ${0}/ route53/hostedZone/ [.red]#${1}/id`
Route 53 Query Logs	The configuration ID of Route 53 query logs where `${0}` is replaced with the query logs configuration name.	`/network/route53Re solver/queryLogCon figs/ ${0}/id`
Route 53 Resolver Endpoint ID	The ID of the Route 53 resolver endpoint where `${0}` is replaced with the resolver endpoint name.	`/network/route53Re solver/en dpoints/ ${0}/id`

Transit Gateway resources

Metadata Type	Description	Path
Transit Gateway ID	The ID of the transit gateway where `${0}` is replaced with the transit gateway name	`/network/transitGa teways/ ${0}/id`
Transit Gateway Peering ID	The ID of the transit gateway peering ID where `${0}` is replaced with the transit gateway name for either the requester or accepter transit gateway*; `${1}` is replaced	`/network/transitGa teways/ ${0}/peering/ ${1}/id`

Metadata Type	Description	Path
	with the transit gateway peering name.	
Transit Gateway Route Table ID	The ID of the transit gateway route table where `${0}` is replaced with the transit gateway name; `${1}` is replaced with the route table name.	`/network/transitGateways/ ${0}/routeTables/ ${1}/id`
Transit Gateway VPN attachment ID	The ID of the transit gateway VPN attachment where `${0}` is replaced with the VPN Connection name.	`/network/vpnConnection/ ${0}/id`

- This depends on the account that the parameter is being put in.

VPC resources

Metadata Type	Description	Path
Virtual Private Cloud (VPC) ID	The ID of the VPC where `${0}` is replaced with the VPC name.	`/network/vpc/ ${0}/id`
VPC Peering ID	The ID of the VPC peering connection where `${0}` is replaced with the VPC peering name.	`/network/vpcPeering/ ${0}/id`
Internet Gateway ID	The ID of the internet gateway where `${0}` is replaced with the VPC name.	`/network/vpc/ ${0}/internetGateway/id`

Metadata Type	Description	Path
Virtual Private Gateway ID	The ID of the virtual private gateway where ${0} is replaced with the VPC name.	/network/vpc/ ${0}/ virtualPrivateGat eway/id
Subnet ID	The ID of the subnet where ${0} is replaced with the VPC name; `${1} is replaced with the subnet name	/network/vpc/ ${0}/ subnet/${1}/id
Route Table ID	The ID of the route table where ${0} is replaced with the VPC name; ${1} is replaced with the route table name	/network/vpc/ ${0}/ routeTable/ ${1}/id
Security Group ID	The ID of the security group where ${0} is replaced with the VPC name; ${1} is replaced with the security group name	/network/vpc/ ${0}/ securityGroup/ ${1}/id
Network ACL ID	The ID of the network ACL (NACL) where ${0} is replaced with the VPC name; ${1} is replaced with the NACL name	/network/vpc/ ${0}/ networkAcl/ ${1}/id
NAT Gateway ID	The ID of the NAT Gateway where ${0} is replaced with the VPC name; ${1} is replaced with the NAT Gateway name	/network/vpc/ ${0}/ natGateway/ ${1}/id

Metadata Type	Description	Path
Transit Gateway VPC Attachment ID	The ID of the transit gateway VPC attachment where `${0}` is replaced with the VPC name; `${1}` is replaced with the transit gateway attachment name	`/network/vpc/ ${0}/ transitGatewayAtt achment/ ${1}/id`

VPN resources

Metadata Type	Description	Path
Customer Gateway ID	The ID of the customer gateway where `${0}` is replaced with the customer gateway name	`/network/customerG ateways/ ${0}/id`

Reference

This section includes information about an optional feature for collecting unique metrics for this solution, pointers to related resources, and a list of builders who contributed to this solution.

Anonymized data collection

This solution includes an option to send anonymized operational metrics to AWS. We use this data to better understand how customers use this solution and related services and products. When invoked, the following information is collected and sent to AWS:

- **Solution ID** − The AWS solution identifier
- **Unique ID (UUID)** − Randomly generated, unique identifier for the Landing Zone Accelerator on AWS deployment
- **Timestamp** − Data-collection timestamp

AWS owns the data gathered though this survey. Data collection is subject to the AWS Privacy Policy. To opt out of this feature, complete the following steps before launching the AWS CloudFormation template.

1. Download the AWS CloudFormation template to your local hard drive.
2. Open the AWS CloudFormation template with a text editor.
3. Modify the AWS CloudFormation template mapping section from:

```
AnonymizedData:
    SendAnonymizedData:
      Data: Yes
```

   to:

```
AnonymizedData:
    SendAnonymizedData:
      Data: No
```

4. Sign in to the AWS CloudFormation console.
5. Select **Create stack**.

6. On the **Create stack** page, **Specify template** section, select **Upload a template file**.
7. Under **Upload a template file**, select **Choose file** and select the edited template from your local drive.
8. Choose **Next** and follow the steps in Launch the stack.

Related resources

- Landing Zone Accelerator on AWS is a fully automated implementation of the architectural guidelines documented in the AWS Security Reference Architecture (SRA).
- Landing Zone Accelerator on AWS incorporates features and lessons learned from previous accelerator solutions such as the Compliant Framework for Federal and DoD Workloads in GovCloud (US) and the AWS Secure Environment Accelerator.

Contributors

- James Armitage
- Mark Burr
- Jimmy Clem
- Brian Crissup
- Partha Debnath
- Randy Domingo
- Dustin Hickey
- Jason Johnson
- Nagmesh Kumar
- Bo Lechangeur
- Ryan Cerrato
- Eric Waxler
- Melinda Mosholder
- John Reynolds
- Aasim Sayani
- Jeremy Spell

United States (US) Federal and Department of Defense (DoD)

- Bhavish Khatri
- Nagmesh Kumar

US aerospace

- Tim Sills

US state and local government Central IT

- Jason Hammett
- Brian Stucker

Canadian Centre for Cyber Security (CCCS) Cloud Medium

- Brian Stucker
- James Kierstead
- Brian Mycroft
- Donny Wilson
- Tim Sills
- Ryan Jaeger
- Dave Liggat
- JD Lynch
- Lawrence Gohar
- Sohaib Tahir
- David Schmidt
- Olivier Gaumond
- Martin Guy Lapointe
- Joel Desaulniers
- Brent Fox
- Dave Wood

Trusted Secure Enclaves Sensitive Edition (TSE-SE) for National Security, Defence, and National Law Enforcement reference architecture

- Brian Mycroft
- Dave Liggat

United Kingdom (UK) National Cyber Security Centre (NCSC)

- Charlie Llewellyn
- Muhammad Khas

Healthcare

- Donny Wison
- Cate Hennard
- Parthiban Dhayalan
- Brian Stucker
- Jason Hammett

Education

- Brian Stucker
- Justin Haydt
- Leo Zhadanovsky

Elections

- Lawrence Gohar

Finance (tax)

- Sohaib Tahir
- David Schmidt
- Brian Stucker

Revisions

Publication date: *May 2022 (last update: October 2024)*

Refer to the CHANGELOG.mdfile in the GitHub repository.

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents AWS current product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers, or licensors. AWS products or services are provided "as is" without warranties, representations, or conditions of any kind, whether express or implied. AWS responsibilities and liabilities to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.