AWS Landing Zone

AWS Developers Guide

May 2019

Notice: AWS Landing Zone must be deployed by your AWS Account team or a certified partner to ensure that your account meets the required prerequisites to successfully deploy this solution.

AWS Control Tower is the recommended option for self-service landing zones and will be generally available in 2019. For more information please visit AWS Control Tower.

Contents

About This Guide

This developer guide provides information about customizing and extending the AWS Landing Zone solution. It includes information about the AWS Landing Zone configuration

ZIP file structure, manifest schema, configuration templates, pipeline deployment stages and add-on products.

The guide is intended for IT infrastructure architects, administrators, DevOps professionals, systems integrators, or independent software vendors who want to customize and extend the AWS Landing Zone solution for their company or customers.

AWS Landing Zone Deployment

AWS Landing Zone is deployed and configured by processing a configuration ZIP file through AWS CodePipeline. The following sections describe this process in detail.

AWS CodePipeline Overview

The AWS Landing Zone configuration process leverages Amazon Simple Storage Service (Amazon S3) and AWS CodePipeline.

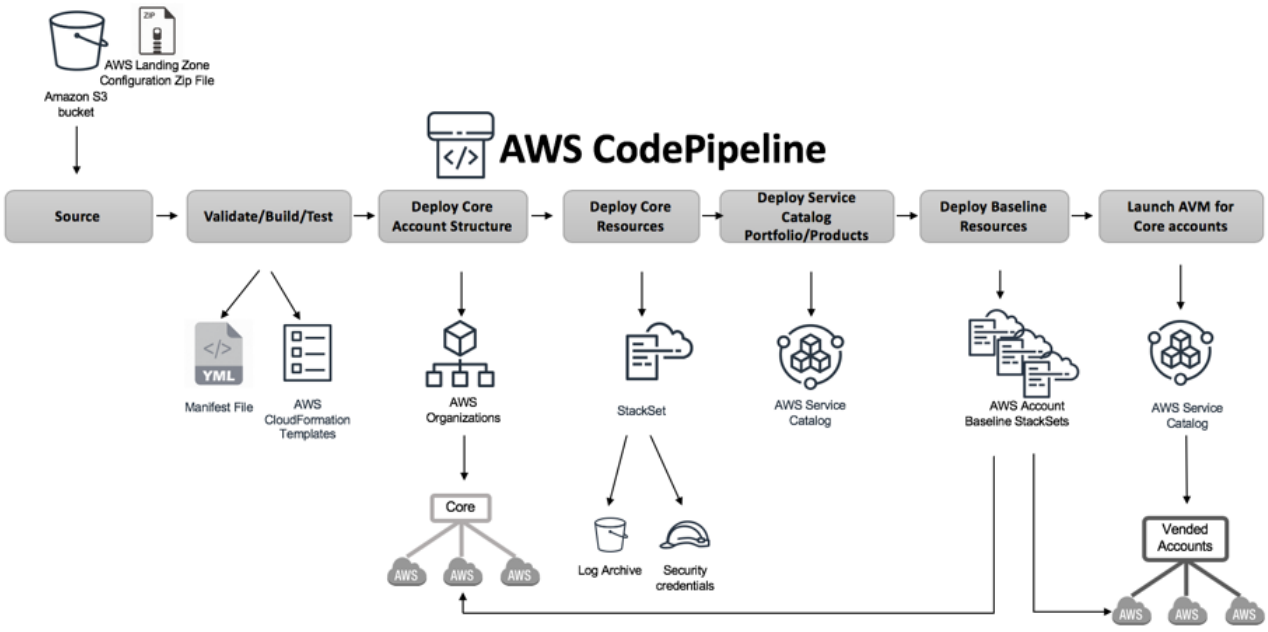


Figure 1: AWS Landing Zone configuration pipeline architecture

A configuration zip file (`aws-landing-zone-configuration.zip`) is loaded in an AWS Landing Zone Amazon S3 bucket (`aws-landing-zone-configuration-[account-id]-[region]`) which provides a manifest, and all related templates for describing and implementing a customer's landing zone environment. The manifest describes AWS account structures and dependencies required to implement a customer's account baseline for new and existing accounts. Updating this configuration file triggers the AWS Landing Zone configuration pipeline. The configuration pipeline extracts the manifest and related

templates, performs manifest and template validation, and uses sections in the manifest file to execute specific pipeline stages.

Code Pipeline Stages

The AWS Landing Zone configuration pipeline leverages several specific AWS CodePipeline stages for implementing and updating your AWS Landing Zone.

Source Stage

The AWS Landing Zone configuration pipeline source stage monitors a configuration zip file) in an AWS Landing Zone created Amazon S3 bucket for changes. Changes to the file start the additional pipeline stages.

Build Stage

This stage leverages AWS CodeBuild to perform the following tasks:

- Merges any add-on micro-configuration ZIP file(s) that are uploaded inside the add-on folder of the AWS Landing Zone configuration ZIP file. For more information about merging the add-on micro-configuration ZIP file, see AWS Landing Zone Add-On.
- Generates the AWS Cloudformation Template for the Account Vending Machine Product.
- Validates the contents of AWS Landing Zone configuration ZIP file. These checks include testing the `manifest.yaml` file syntax and schema, and all AWS CloudFormation templates included in the zip file or remotely hosted using CloudFormation validate-template and cfn_nag. If the manifest file and AWS CloudFormation templates pass the tests, the pipeline continues to the next stage.

(Optional) Manual Approval Stage

If enabled, during AWS Landing Zone initialization, a manual approval step is added to the configuration pipeline. This optional stage provides additional control over the configuration pipeline execution by pausing the pipeline until additional approval is provided to proceed.

Core Accounts Stage

The core accounts stage triggers the AWS Organizations State Machine to make AWS Organizations API calls to create organizational units, and core accounts specified in the Organizational Units section of the manifest file.

Service Control Policy Stage

The service control policy stage triggers the Service Control Policy State Machine to make AWS Organizations API calls to create service control policies specified in the Organization Policies section of the manifest file.

Core Resource Stage

The core resource stage triggers the StackSet State Machine to deploy the core resources specified in the Core Accounts section of the manifest file. Core resources are created in the order in which they appear in the manifest file.

Service Catalog Stage

The service catalog stage triggers the AWS Service Catalog State Machine to create AWS Service Catalog portfolios and products specified in the Portfolios section of the manifest file.

Baseline Resource Stage

The baseline resource stage triggers the StackSet State Machine to deploy account baseline resources specified in the Baseline Resources section of the manifest file.

Launch AVM Stage

The launch AVM stage triggers the Launch AVM State Machine to automatically apply account baselines to managed accounts, by creating or updating AWS Service Catalog. AVM products for each account as specified in the Baseline Resources section of the manifest file.

AWS Landing Zone Configuration

The AWS Landing Zone Configuration is defined by the manifest file and accompanying set of templates and other JSON files. The manifest file (`manifest.yaml`) is a YAML-formatted text file that describes your AWS Landing Zone core accounts, core resources, service control policies, AWS Service Catalog portfolios and products, and configuration baseline resources. These files are packaged into a folder structure and put in as a ZIP file into the Amazon S3 bucket.

Landing Zone Configuration Folder Structure

The Landing Zone Configuration folder structure is shown below:

```
- manifest.yaml
- parameters/
   - aws_baseline/
      - parameter files for Baseline Resources (*.json)
   - core_accounts/
      - parameter files for Core Resources (*.json)
- policies/
   - service control policies files (*.json)
- templates/
   - aws_baseline/
      - template files for Baseline Resources (*.template)
   - core_accounts/
      - template files for Core Resources (*.template)
```

```
- template_constraints/
   - template constraint rules files (*.json)
```

Manifest Overview

The following examples show the manifest file structure and its sections:

```
---
region: String
version: 2018-06-14
lock_down_stack_sets_role: Boolean

organizational_units:
  set of AWS Organization OUs and related core accounts

organization_policies:
  set of managed AWS Organization SCPs

portfolios:
  set of AWS Service Catalog portfolios and products

resources:
  set of account baseline resources
```

Organizational Units

This manifest file section describes the AWS Organizations structure of your AWS core accounts including related templates that define what core resources you want deployed into these accounts. Core accounts are AWS accounts which contain resources upon which all of your AWS Landing Zone managed accounts will depend. For example, implementing a centralized logging account for securely storing all access logs creates dependencies between the account storage resources (i.e. Amazon S3 bucket), and all other managed accounts.

Organization Policies

This manifest file section controls the AWS Organizations Security Control Policies (SCPs) that are applied to the accounts in your organization. This section allows you to specify which Organization Units (OUs) to use when applying SCPs to accounts, however, SCPs are applied at the account level rather than the OU level. This allows SCPs to be added and removed from specific accounts while baseline configurations are applied rather than to the OU, which would affect all accounts in the OU, whether or not changes are being applied to a particular account.

Portfolios

This manifest file section defines the AWS Service Catalog portfolio and products for account baselining and add-on products. Account baselining products are used to apply and update

managed account baselines using resources defined in the baseline resource section and account OU membership. Add-on products allow AWS administrators to enhance their AWS Landing Zone by deploying optional resources such as Amazon Elasticsearch Service (Amazon ES) for log analytics and reporting.

Baseline Resources

This manifest file section defines the baseline resources that will be automatically configured for OU-grouped managed accounts. For a description of out-of-the-box example configuration baseline resources for many AWS services, see the AWS Landing Zone User Guide.

Manifest Sections

The manifest file is described in detail in this section. Sections in the file can be in any order. However, the order of core resources is used to determine the execution order for creating AWS Landing Zone core account resource dependencies. For more information, see the Core Accounts section.

Region

A text string for the AWS Landing Zone default region. This value must be a valid AWS Region name (i.e. us-east-1, eu-west-1, ap-southeast-1). The default region will be used for creating AWS Landing Zone resources (i.e. AWS CloudFormation StackSets, AWS Service Catalog portfolios and products), unless a more resource-specific region is specified.

Version

The AWS Landing Zone manifest schema version number. The current version is `2018-06-14`.

Lock_down_stack_sets_role

When an account is created, the AWS Organizations preconfigured role is created (`AWSCloudFormationStackSetExecutionRole`), and used by AWS Landing Zone to manage StackSet instances. By default, this role allows any IAM principal in the AWS Organizations account with `sts:AssumeRole` permissions to assume the role in a member account. The `lock_down_stack_sets_role` parameter is required, and configures the role permissions, to only allow AWS Landing Zone provisioning roles to assume the StackSets execution role. We highly recommend setting this to `true` to avoid granting StackSet execution access to users in the AWS Organizations account.

Nested_OU_Delimiter

An organizational unit is like a container for accounts within a root. An OU can also contain other OUs, enabling you to create a hierarchy that resembles an upside-down tree, with a

root at the top and branches of OUs that reach down, ending in accounts that are the leaves of the tree. The AWS Landing Zone solution version 2.1 supports the nested OU hierarchy. When you attach a policy to one of the nodes in the hierarchy, it flows down and affects all the branches (OUs) and leaves (accounts) beneath it. Note that only blacklists are inheritable.

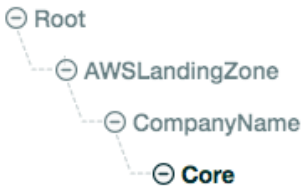
The nested OU delimiter is the special character that can be used in the OU names and allows the solution to create the hierarchy under the root. The allowed values for this key are as follows:

- Colon (:)
- Dot (.)
- Hyphen (-)
- Underscore (_)
- Semicolon (;)
- Hash (#)
- Pipe (|)

For example, if you use colon (:) as the delimiter the manifest.yaml should look as follows:

```
nested_ou_delimiter: ':' # the value must be in single quotes
...
# Landing Zone Core Account Structure
organizational_units:
  # Landing Zone OU for Core accounts
  - name: AWSLandingZone:CompanyName:Core
    include_in_baseline_products:
      - AWS-Landing-Zone-Account-Vending-Machine
```

This setting should create the following example structure in your organization.



Organizational Units

Specifies the AWS Organizations Organizational Units (OUs), related core accounts to be created in the OU, and core account resources that should be created in each core account.

Syntax

The OUs section consists of the key name `organizational_units`, a set of organizational units, and account definitions. The following pseudo template outlines the OUs section:

```
organizational_units: # List of OUs and related accounts
  - name: String
    include_in_baseline_products: # List of Service Catalog Product
Names
      - String
    core_accounts:
      - List of core accounts
```

Name

AWS Organizations organization unit (OU) name to be created.

Type: String

Required: Yes

include_in_baseline_products

List of AWS Service Catalog product names to determine which AWS Landing Zone AVM products to update to include the ability to deploy accounts into this OU.

Condition: This determines which AVM products will be able to select this OU when creating new accounts.

Type: String

Required: Yes

core_accounts

List of Core Accounts to create in the OU.

Type: Core Accounts

Required: Optional

Core Accounts

AWS Landing Zone core accounts are defined under the Organizational Units section of the manifest file using the key name `core_accounts`. The following pseudo template outlines the accounts object:

```
core_accounts: # List of accounts
  - name: String
    email: String
    ssm_parameters: # List of SSM parameters
      - name: String
        value: String
    core_resources: # List of resources
```

Name

Name of the core account.

Type: String

Required: Yes

Valid Values: 'a-zA-Z0-9._-' Any other character is automatically replaced with '_'

Email

Email address for the core account.

Type: String

Required: Yes

ssm_parameters

List of SSM parameter key name and value pairs for storing AWS Organizations account creation outputs in SSM parameter store for reference by other core or baseline resources.

```
ssm_parameters: # List of SSM parameters
  - name: String
    value: String
```

Type: List of name and value key pairs where name is an SSM parameter store key name string and value is the parameter value string.

Required: Optional

Account variables: When an account is created, the following table lists the variables that can be used to store the new account's ID, email, and AWS Organizations ID in the SSM parameter store.

Variable	Description
$[AccountId]	Account ID for the related AWS account.
$[AccountEmail]	Account email address for the related AWS account.
$[OrganizationId]	AWS Organizations organization ID.

For example:

```
ssm_parameters: # List of SSM parameters
  - name: /org/member/sharedservices/account_id
    value: $[AccountId]
```

core_resources

List of Core Resources to deploy into the core account. Core resources are deployed in the order in which they are provided. This allows dependencies between core resources to be managed by listing dependent resources later in the list than the resources they depend upon.

Type: List of Core Resources

Required: Yes

Core Resources

AWS Landing Zone core account resources are defined under the Core Accounts section of the manifest file using the key name `core_resources`. The following pseudo template outlines the Core Resources object:

```
core_resources: # List of resources
  - name: String
    template_file:  String
    parameter_file: String
    deploy_method:  stack_set
    ssm_parameters: # List of SSM parameters
      - name: String
      - value: String
```

name

Name to associate with the core account resource.

The provided name is used to provide a more user-friendly name for an account.

Type: String

Required: Yes

Valid Values: 'a-zA-Z0-9._-' Any other character is automatically replaced with '_'

template_file

This can be either relative location to the manifest file or an Amazon S3 URL that points to an AWS CloudFormation template for creating core resources.

Type: String

Required: Yes

For example:

```
core_resources:
  - name: SecurityRoles
    template_file:templates/core_accounts/aws-landing-zone-
security.template
```

Or

```
core_resources:
  - name: SecurityRoles
    template_file:s3://my-bucket/templates/aws-landing-zone-
security.template
```

Note: If providing Amazon S3 URL, verify that the Bucket policy provides the read access for the Organizations Master account deploying the Landing Zone solution.

Example S3 Bucket policy:

```
{
   "Version": "2012-10-17",
   "Statement": [
   {
        "Effect": "Allow",
        "Principal": {"AWS": "arn:aws:iam::xxxxxxxxxxx:root"},
        "Action": "s3:GetObject",
        "Resource": "arn:aws:s3:::my-bucket/*"
   }
   ]
}
```

```
Where xxxxxxxxxxx is AWS Account ID of Organizations Master account
deploying the Landing Zone solution.
```

parameter_file

This can be either relative location to the manifest file or an Amazon S3 URL that points an AWS CloudFormation template parameter file defining the input parameters to use when launching **template_file**.

Type: String

Required: Optional, if the associated AWS CloudFormation template does not have any input parameters.

Example:

```
core_resources:
  - name: SecurityRoles
    parameter_file:parameters/core_accounts/aws-landing-zone-
security.json
```

Or

```
core_resources:
  - name: SecurityRoles
    parameter_file:s3://my-bucket/parameters/aws-landing-zone-
security.template
```

deploy_method

Deployment method for deploying resource(s) into the account. Currently, deploy_method supports deploying resources using the stack_set option for deployment through StackSets.

Type: String

Valid Values: `stack_set`

Required: Yes

ssm_parameters

List of SSM parameter key name and value pairs for storing template outputs in SSM parameter store for reference by other core or baseline resources.

```
ssm_parameters: # List of SSM parameters
```

```
    - name: String
      value: String
```

Type: List of name and value key pairs where name is an SSM parameter store key name string and value is the parameter value string.

Valid Values: Any string or the `$[output_CfnOutputVariable]` variable where **CfnOutputVariable** corresponds to the template output variable.

Required: Optional

For example, the following template snippet will store the template `VPCID` output variable into the `/org/member/sharedservices/vpc_id` SSM parameter key.

```
ssm_parameters: # List of SSM parameters
  - name: /org/member/sharedservices/vpc_id
    value: $[output_VPCID]
```

Organizational Policies

Specifies the AWS Organizations SCPs to be created in each core account.

Syntax

The Organizational Policies section consists of the key name `organizational_policies`. The following pseudo template outlines the Organizational Policies section:

```
organization_policies: # List of policies
  - name: String
    description: String
    policy_file: String
    apply_to_accounts_in_ou: # List of Strings
        - String
```

name

Name for the AWS Organizations SCP.

Type: String

Required: Yes

Valid Values: 'a-zA-Z0-9._-' Any other character is automatically replaced with '_'

description

Description for the AWS Organizations SCP.

Type: String

Required: Optional

policy_file

Location relative to the manifest file for a file containing the SCP policy to apply.

Type: String

Required: Yes

apply_to_accounts_in_ou

List of OUs to apply to accounts in this OU. AWS Landing Zone applies SCPs at the account level rather than the OU level. This allows SCPs to be added and removed to specific accounts while baseline configurations are applied rather than to the OU (which would affect all accounts in the OU). This option allows OU membership to determine which accounts the SCP will be applied to. If the nested OU name is provided with the defined delimeter character, the SCP will be applied on the last level of the OU hierarchy. For example: if the user provides '**AWSLandingZone:CompanyName'** in the list, the SCP will be applied to OU name 'CompanyName' and all the OU under it.

Type: List of OU names

Required: Optional

Portfolios

Specifies the AWS Service Catalog portfolios and products to be created in the organizations account.

Syntax

The Portfolios section consists of the key name `portfolios`. The following pseudo template outlines the Portfolios section:

```
portfolios: # List of portfolios
  - name: String
    description: String
    owner: String
    principal_role: String
```

```
      products: # List of products to add to portfolio
        - List of products
```

name

Name for the AWS Service Catalog portfolio.

Type: String

Required: Yes

Valid Values: 'a-zA-Z0-9._-' Any other character is automatically replaced with '_'

description

Description for the AWS Service Catalog portfolio.

Type: String

Required: Optional

owner

Name to provide AWS Service Catalog for the portfolio owner field.

Type: String

Required: Optional

principal_role

IAM Role Arn to grant initial access to the portfolio in AWS Service Catalog.

Type: String

Required: Optional

products

List of AWS Landing Zone AWS Service Catalog Products.

Type: List of Products

Required: Optional

Products

AWS Service Catalog products are defined under the Portfolios section of the manifest file using the key name `products`. The following pseudo template outlines the Products object:

```
products: # List of products to add to portfolio
  - name: String
    description: String
    product_type: String
    template_file: String
    parameter_file: String
    skeleton_file: String
    ssm_parameters: # List of SSM parameters
      - name: String
        value: String
    hide_old_versions: Boolean
    launch_constraint_role: String
    apply_to_accounts_in_ou: #List of Strings
      - String
```

name

Name to use for the AWS Service Catalog product name.

Type: String

Required: Yes

Valid Values: 'a-zA-Z0-9._-' Any other character is automatically replaced with '_'

description

Description for the AWS Service Catalog product.

Type: String

Required: Optional

product_type

Determines whether the product is an account configuration `baseline` or an `optional (Deprecated)` product containing optional AWS Landing Zone resources.

Type: String

Required: Yes

Valid Values: `baseline` or `optional`

Note: Optional Product Type is deprecated in Landing Zone v2.0 and replaced with the Landing Zone Add-On

template_file

Location relative to the manifest file for a base AWS CloudFormation template for creating the AWS Service Catalog product.

Type: String

Required: Required if `product_type = optional`

Note: Optional Product Type is deprecated in Landing Zone v2.0 and replaced with the Landing Zone Add-On

parameter_file

Relative location to the AWS CloudFormation template parameter file defining the input parameter defaults for the AWS Service Catalog product.

Type: String

Required: Required if `product_type = baseline`

Note: When you add new parameter to the parameter file for the `product_type = baseline`, i.e. `parameters/aws_baseline/aws-landing-zone-avm.json`, provide the default parameter value, which will be used for updating the baseline for the existing accounts.

Example:

```
{
  "ParameterKey": "foo",
  "ParameterValue": "bar"
}
```

skeleton_file

Relative location of a Jinja2 skeleton template that will be used to create the final AWS Service Catalog product (Account Vending Machine) if `product_type = baseline`.

Type: String

Required: Yes

rules_file

Relative location of a Template constraint rules file for AWS Service Catalog product (Account Vending Machine) if `product_type` = `baseline`.

Type: String

Required: Yes

hide_old_versions

Configures whether or not AWS Landing Zone will hide previous versions of the product in AWS Service Catalog when a new product version is deployed.

Type: String

Required: Yes

launch_constraint_role

IAM role ARN to be used for the AWS Service Catalog product launch constraint.

Type: String

Required: Yes

apply_to_accounts_in_ou *(deprecated in Version 2.1)*

List of OUs to automatically create or update product instances for each account in the OU. AWS Landing Zone applies baseline configurations by launching AWS Service Catalog baseline products for accounts. This option allows OU membership to determine which accounts baseline products will be applied to.

Type: List of OU name strings

Required: Optional

Baseline Resources

Specifies the AWS Landing Zone account baseline configuration for managed accounts.

Syntax

The baseline resources section consists of the key name `baseline_resources`. The following pseudo template outlines the Resources section:

```
baseline_resources: # List of account baseline resources
```

```
    - name: String
      baseline_products: #List of SSM parameter key names
        - String
      depends_on: # List of account baseline resource names
        - String
      template_file: String
      parameter_file: String
      deploy_method: String
      regions: # List of Strings
        - String
```

name

Name to associate with the account baseline resources. The provided name is used as part of creating the StackSet name for this baseline configuration.

Type: String

Required: Yes

Valid Values: 'a-zA-Z0-9._-' Any other character is automatically replaced with '_'

baseline_products

List of key names for AWS Landing Zone AWS Service Catalog AVM products to associate with this configuration resource. This option allows customers to maintain different account baselines by associating a configuration resource with unique, multiple, or different accounts created by different AVM products.

Type: List of AWS Service Catalog name strings

Required: Yes

depends_on

List of baseline resource names that this resource depends on. This option is used to define baseline resource dependencies to control the order in which baseline resources are deployed to managed accounts.

Type: String

Required: Optional

template_file

This can be either relative location to the manifest file or an Amazon S3 URL that points to an AWS CloudFormation template for creating baseline resources.

Type: String

Required: Yes

Example:

```
baseline_resources:
  - name: EnableCloudTrail
    template_file:templates/aws_baseline/aws-landing-zone-enable-
cloudtrail.template
```

Or

```
core_resources:
  - name: EnableCloudTrail
    template_file:s3://my-bucket/templates/aws-landing-zone-enable-
cloudtrail.template
```

parameter_file

This can be either relative location to the manifest file or an Amazon S3 URL that points an AWS CloudFormation template parameter file defining the input parameters to use when launching **template_file**.

Type: String

Required: Optional, if the associated AWS CloudFormation template does not have any input parameters.

Example:

```
baseline_resources:
  - name: EnableCloudTrail
    parameter_file:parameters/aws_baseline/aws-landing-zone-enable-
cloudtrail.json
```

Or

```
baseline_resources:
  - name: EnableCloudTrail
    template_file:s3://my-bucket/parameters/aws-landing-zone-enable-
cloudtrail.json
```

deploy_method

Deployment method for deploying the associated AWS CloudFormation template.

Conditional: Currently `deploy_method` supports deploying AWS CloudFormation templates using the `stack_set` option for deployment through StackSets.

Type: String

Required: Yes

Valid Values: stack_set

regions

List of regions where this baseline resource should be deployed.

Type: Any AWS commercial region names as well as **All** to indicate that this resource should be deployed into all regions.

Required: Yes

AWS Landing Zone Add-On

The add-on feature allows customers to extend their Landing Zone implementation by dropping in the add-on Micro-configuration into your existing Landing Zone Configuration. The default implementation creates the Service Catalog Portfolio: **AWS Landing Zone - Add-On Products**, and deploys add-on products.

When the add-on product is launched, it modifies the existing Landing Zone Configuration ZIP file inside the Amazon S3 Bucket used as the source for Landing Zone pipeline.

> **Note:** If CodeCommit is configured as the source for the Landing Zone pipeline, it will create the add-on Micro-configuration ZIP file in an Amazon S3 bucket. This file must be added inside the add-on folder of CodeCommit Landing Zone repo. For more information, see Appendix B.

For more information about deploying additional products, see AWS Landing Zone Add-On Products.

Landing Zone Add-on Configuration Folder Structure

The add-on folder inside the Landing Zone Configuration folder structure is where add-on Micro-configurations can be placed as a ZIP file, or the contents of the zip file under a unique folder for each micro-configuration.

For more information on folder structure see, AWS Landing Zone Configuration section for The Landing Zone Configuration folder structure is shown below:

```
- manifest.yaml
- add-on/
   - add-on1.ZIP (Add-On Micro-configuration ZIP files)
   - add-on2/
       - add_on_manifest.yaml
       - user-input.yaml
       - parameters/
            - parameter files (*.json)
       - templates/
            - template files (*.template)
- parameters/…
- policies/…
- templates/…
- template_constraints/…
```

The add-on must have the following set of files:

add_on_manifest.yaml

The add-on manifest file (`add_on_manifest.yaml`) follows the same syntax as that of the main Manifest file (`manifest.yaml`) as shown in Manifest overview section, with an exception that it cannot have the region, version & lock_down_stack_sets_role attributes.

Example:

The `add_on_manifest.yaml` is adding one core resource into the customer's choice of core account and one baseline resource to baseline product for the Centralized Logging Solution:

```
---

# Landing Zone Core Account Structure
organizational_units:
  # Landing Zone OU for Core accounts
  - name: {{ core_ou }}
    core_accounts:
      - name: {{ core_account }}
        core_resources:
        - name: Centralized-Logging-Primary
          template_file:templates/core_accounts/aws-landing-zone-
centralized-logging-primary.template
          parameter_file:parameters/core_accounts/aws-landing-zone-
centralized-logging-primary.json
          deploy_method: stack_set
          ssm_parameters:
            - name: /org/member/centrallogging/es_domain
```

```
              value: $[output_DomainEndpoint]
            - name: /org/member/centrallogging/master_role
              value: $[output_MasterRole]
          regions:
            - {{ region }}

# Landing Zone Service Baseline Resources
baseline_resources:
   - name: CentralizedLoggingSpoke
     baseline_products:
      {%- for avm_product in avm_products %}
      - {{ avm_product }}
      {%- endfor %}
     template_file:templates/aws_baseline/aws-landing-zone-centralized-
logging-spoke.template
     parameter_file:parameters/aws_baseline/aws-landing-zone-
centralized-logging-spoke.json
     deploy_method: stack_set
     regions:
     {%- for region in spoke_regions %}
     - {{ region }}
     {%- endfor %}
```

The add-on manifest (`add_on_manifest.yaml`) contains the snippet that will be added to the customer's master manifest file (`manifest.yaml`).

Below, the add-on manifest leverages Jinja2 markup language to find, replace and loop through the customer provided user inputs and dynamically generate the add-on manifest file that will then be merged with the master manifest file.

Example:

The input in the above add-on manifest file is processed through the Jinja2 preprocessor will follow the example below:

```
---

# Landing Zone Core Account Structure
organizational_units:
  # Landing Zone OU for Core accounts
  - name: core
    core_accounts:
      - name: shared-services
        core_resources:
        - name: Centralized-Logging-Primary
          template_file:templates/core_accounts/aws-landing-zone-
centralized-logging-primary.template
          parameter_file:parameters/core_accounts/aws-landing-zone-
centralized-logging-primary.json
          deploy_method: stack_set
          ssm_parameters:
```

```
            - name: /org/member/centrallogging/es_domain
              value: $[output_DomainEndpoint]
            - name: /org/member/centrallogging/master_role
              value: $[output_MasterRole]
          regions:
            - us-east-1

# Landing Zone Service Baseline Resources
baseline_resources:
   - name: CentralizedLoggingSpoke
     baseline_products:
       -  AWS-Landing-Zone-Account-Vending-Machine
     template_file:templates/aws_baseline/aws-landing-zone-centralized-
logging-spoke.template
     parameter_file:parameters/aws_baseline/aws-landing-zone-
centralized-logging-spoke.json
     deploy_method: stack_set
     regions:
        - us-east-1
        - us-east-2
        - us-west-2
```

Note: When merging the add-on manifest files into the main manifest file, the build stage follows the first write wins logic whenever there is a conflict. For example, in the above case, if the customer already has a core resources named **Centralized-Logging-Primary** in the master manifest (`manifest.yaml`) file, it will NOT be overwritten by this add-on.

user-input.yaml

The user input YAML file is used to capture all user inputs required for the add-on in one file. This file is then used by the build stage to dynamically update the target files with the user provided input values using Jinja2.

Example:

The `user-input.yaml` file for the Centralized Logging Solution is as follows:

```
input_parameters:
  - file_name: add_on_manifest.yaml
    parameters:
      core_ou: core
      core_account: shared-services
      region: us-east-1
      avm_products: AWS-Landing-Zone-Account-Vending-Machine
      spoke_regions: ['us-east-1','us-east-2','us-west-1']
  -   file_name:   parameters/core_accounts/aws-landing-zone-centralized-
logging-primary.json
    parameters:
```

```
        domain_name: centralized-logging
        domain_admin_email: domain@example.com
        cognito_admin_email: cofgnito-admin@example.com
        cluster_size: small
    -    file_name:    parameters/aws_baseline/aws-landing-zone-centralized-
  logging-spoke.json
      parameters:
        cloud_trail_region: us-east-1
```

The **file_name** section references the relative path to the target file inside the add-on folder or ZIP file. The **parameters** section references the `key: value pair` used for find & replace by Jinja2. For example, the `core_account` inside the `add_on_manifest.yaml` file will be replaced with the user provided value `shared-services` in the above example.

Optionally, the add-on can have the following folders or files:

Add-On Templates

The `add_on_manifest.yaml` file refers to the relative template file(s) for Core and Baseline resources. Optionally, the add-on manifest can refer the remote Amazon S3 template file, in which case the templates are not bundled into the add-on ZIP file.

For more information, see the Core resources or Baseline resources sub-section for template_file.

Add-On Parameters

The `add_on_manifest.yaml` file can refers to the relative parameters file(s) for Core and Baseline resources. Optionally, the add-on manifest can refer the remote Amazon S3 parameter files, in which case the parameter files are not bundled into the add-on ZIP file.

For more information, see the Core resources or Baseline resources sub-section for parameter_file.

Appendix A: Solution Extensibility

The AWS Landing Zone solution allows you to modify the `manifest.yaml` file to add custom resources, add or delete AWS Service Catalog add-on products, add, update, or delete core and baseline resources, and add or remove core accounts. You can add or modify the templates in the ZIP file folders, create your own folders, and reference the templates or folders in the `manifest.yaml` file. Doing this can help you distinguish your customized templates and security baselines from the out-of-the-box AWS Landing Zone configurations.

Note that if you update the solution provided templates, you don't have to update the manifest file. However, if you add or delete templates and folders to the manifest file, you must ZIP the manifest file and all associated templates and upload the ZIP file to the AWS Landing Zone Amazon S3 configuration bucket for your changes to be applied.

Add or Remove Organizational Unit

To add a new OU, you must add the OU in the manifest file **organizational_units**: section:

```
organizational_units:
   - name: production
    include_in_baseline_products:
      - AWS-Landing-Zone-Account-Vending-Machine
```

- To add or remove an OU, create or delete the **- name: ou_name** section.

Add or Remove Core Accounts

To add new Core account, you must add the account information in the manifest file **accounts**: section:

```
organizational_units:
   - name: core
      - core_accounts:
         - compliance:
              email: email+compliance@company.com
              ssm_parameter:
                  - name: /accounts/compliance/account_id
                    value: $[AccountId]
```

- To remove the compliance account, delete the **-compliance**: section.

Add, Update, or Remove Core Account Resources

To add or update core account resources, use the following procedure:

- Update the template file in the **templates/core_accounts** folder (or add a new template in a folder of your choosing)
- Update the parameters file in the **parameters/core_accounts** folder (or add a new parameter file in a folder of your choosing)
- Add/update the section in the manifest file **organizational_units:name: core_accounts:core_resources** section

```
organizational_units:
   - name: core
      - core_accounts:
         - compliance:
              email: email+compliance@company.com
              ssm_parameter:
                  - name: /accounts/complaince/account_id
                    value: $[AccountId]
           core_resources:
```

```
              - name: ComplianceMonitoring
                template_file:           templates/custom/compliance-
monitoring.template
                parameter_file:         parameters/custom/compliance-
monitoring.json
                deploy_method: stack_set
                regions:
                    - us-east-1
                ssm_parameters:
                    - name: /org/member/compliance/resource_name
                      value: $[output_ComplianceResource]
```

Add, Update, or Remove Account Baseline Resources

To add or update account baseline configurations, use the following procedure:

- Update the template file in the **templates/aws_baseline** folder (or add a new template in a folder of your choosing)
- Update the parameters file in the **parameters/aws_baseline** folder (or add a new parameter file in a folder of your choosing)
- Add/update the section in the manifest file **baseline_resources** section

```
baseline_resources:
  - name: EnableCloudTrail
    # This resource is part of which baseline(s) product
    baseline_products:
      - AWS-Landing-Zone-Account-Vending-Machine
    template_file:    templates/aws_baseline/aws-landing-zone-enable-
cloudtrail.template
    parameter_file:  parameters/aws_baseline/aws-landing-zone-enable-
cloudtrail.json
    deploy_method: stack_set
```

Add, Update, or Remove AWS Service Catalog Products

To add or delete optional AWS Service Catalog products, use the following procedure:

- Add or update the template and template skeleton files in the **templates/optional_products** folder (or add a new template and skeleton file in a folder of your choosing).
- Update the manifest file **portfolios** section.

```
portfolios:
  - name: My_Portfolio_Name
    description: My awesome portfolio of products
    owner: My Company
```

```
    principal_role:
$[alfred_ssm_/org/primary/service_catalog/principal/role_arn]
    # These products will prompt the user to select target Account
Email and Region
    products:
      - name: My_Product_Name
        description: Description for my product
        template_file: templates/my_products/my-product.template
        skeleton_file:              templates/my_products/my-product-
skeleton.template.j2
        ssm_parameters:
          - name: /ssm_parameter_to_store
            value: $[output_MyProductOutput]
        # Hide/Disable the old version of the product in Service
Catalog
        hide_old_versions: true
        product_type: optional
        launch_constraint_role:
$[alfred_ssm_/org/primary/service_catalog/constraint/role_arn]
```

Add, Update, or Remove AWS Organizations Policies

To add, modify or delete optional AWS Organizations policies, use the following procedure:

- Add or update the policy file in the **policies** folder (or add a new policy file in any folder)
- Update the manifest file **organization_policies** section.

```
organization_policies:
  - name: policy-name
    description: Description for my policy
    policy_file: policies/my_new_scp_for_production.json
    apply_to_accounts_in_ou:
        - production
```

Appendix B: Using Git for Configuration Source Control

> **Note:** Do not delete the AWS Landing Zone configuration bucket after changing the source control.

AWS Landing Zone configuration and updates are managed through a configuration ZIP file stored in a configuration Amazon S3 bucket. Changes to this file triggers the configuration and update pipeline to make changes to your AWS Landing Zone. Customers may want to leverage a source control system like Amazon CodeCommit, or GitHub for managing their configuration files. To move from an Amazon S3 bucket to Git for configuration source control, use the following procedure:

1. Create an Amazon CodeCommit in your AWS Organizations account, or a GitHub repository.
2. Connect to your new Git repository.
3. Download and extract your AWS Landing Zone configuration zip file from the AWS Landing Zone configuration bucket to the new Git repository. Verify that you extract the file contents to the base of the Git repository and **not** in the folder: `aws-landing-zone-configuration/`.
4. Navigate to the AWS CodePipeline console in your AWS Organizations account.
5. Select the **AWS-Landing-Zone-CodePipeline** pipeline.
6. Select **Edit**.
7. In the first stage **Source**, edit the **Source** action.
8. **Change** the source provider to Amazon CodeCommit or GitHub.
9. **Configure** Amazon CodeCommit repository name, and branch or connect to GitHub.
10. Configure **Output artifact #1** = `SourceApp`.
11. Select **Update**.
12. Changes you commit to your Git repo, will trigger the update pipeline to push changes to your AWS Landing Zone configuration.

Notices

This document is provided for informational purposes only. It represents AWS's current product offerings and practices as of the date of issue of this document, which are subject to change without notice. Customers are responsible for making their own independent assessment of the information in this document and any use of AWS's products or services, each of which is provided "as is" without warranty of any kind, whether express or implied. This document does not create any warranties, representations, contractual commitments, conditions or assurances from AWS, its affiliates, suppliers or licensors. The responsibilities and liabilities of AWS 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.

The AWS Landing Zone solution is licensed under the terms of the Amazon Software License available at https://aws.amazon.com/asl/.