Automated Account Configuration

Implementation Guide

Automated Account Configuration: Implementation Guide

Table of Contents

Set up your AWS accounts using automated operational processes in an efficient, error-free, standardized, and consistent way

Publication date: *October 2021 (last update (p. 25): December 2021)*

When you create new AWS accounts, standard operational processes must be manually set up, including backup and patching services. Manual set up of these processes can take time, be prone to human errors, and result in inconsistent standards being established between accounts. The Automated Account Configuration solution helps you automate these operational processes, providing a standardized and consistent way to ensure that your AWS accounts are set up properly and with the necessary resources to meet your business and production needs.

This solution configures and deploys the following business critical services:

- AWS Backup to centrally manage the backups of AWS services including Amazon Elastic Compute Cloud Console (Amazon EC2) instances, Amazon Relational Database Service (Amazon RDS), and Amazon Elastic File System (Amazon EFS).
- AWS Systems Manager Patch Manager to automate the patching of managed instances such as EC2 instances.

This solution can be extended to add additional operational processes, including set up and maintenance updates for AWS Identity and Access Management (IAM) roles and AWS Key Management Service (AWS KMS). You can use this solution as a template or framework to set up the operational tasks that are essential to your organization.

This solution offers the following key features:

- An automated process to install core operational capabilities including backup and patching in all AWS accounts.
- A customizable configuration file that lets you control and manage the operational services that you want deployed in your AWS accounts.
- Supports AWS Managed Services (AMS) accounts, creating the request for change forms.
- The ability to extend the solution by adding additional configuration steps to meet your business requirements.

This implementation guide discusses architectural considerations and configuration steps for deploying Automated Account Configuration in the Amazon Web Services (AWS) Cloud. It includes links to a set of AWS CloudFormation templates that launch and configure the AWS services required to deploy this solution using AWS best practices for security and availability.

This solution is intended for cloud architects, operations engineers, and DevOps teams requiring an automated process that consistently builds and prepares AWS accounts for production use.

Cost

You are responsible for the cost of the AWS services used while running this solution. As of October 2021, the cost for running this solution with the default settings in US East (N. Virginia) is approximately **$2.30 per month**. This cost estimate accounts for an Amazon Simple Storage Service (Amazon S3) bucket, AWS Key Management Service, AWS Lambda functions, AWS Step Functions, AWS Systems Manager Patch Manager, an AWS Identity and Access Management (IAM) role, and AWS Backup services that are initiated in the solution's account, for up to 100 AWS accounts.

There is no charge to deploy the following AWS services: an IAM role, AWS Backup services including backup plan and backup vault, and AWS Systems Manager Patch Manager. However, usage charges apply for these services, for example, when you create a backup plan or when backups are being run.

Table 1: Cost estimate

AWS Service	Factors	Cost per month
Amazon S3	• Storage: 10 MB • 50,000 get requests/month	$0.25
AWS KMS	Number of AWS KMS key = 1 with 200 Symmetric requests. For more information about the KMS key, refer to Using symmetric and asymmetric keys in the *AWS KMS Developer Guide*. **Note** AWS KMS key is replacing the term customer master key (CMK). For more information, refer to AWS Key Management Service concepts in the *AWS Key Management Service Developer Guide*.	$1.01
AWS Lambda	10 GB x $0.00001667 per GB The monthly compute price is $0.00001667 per GB Monthly compute charges breakdown: • Total compute (seconds) = 1M * (1s) = 1,000,000 seconds • Total compute (GB) = 1,000,000 * 512 MB/1024 = 500,000 GB • Total compute – Free tier compute = Monthly billable compute GB	$1.667

AWS Service	Factors	Cost per month
	500,000 GB – 400,000 free tier GB = 100,000 GB	
AWS Step Functions	No cost for up to 4,000 transitions per month then 0.25 per 1,000 transitions	$0.00
IAM	No cost	$0.00
	Total:	**$2.297**

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.

Architecture overview

Deploying this solution with the default parameters builds the following environment in the AWS Cloud.

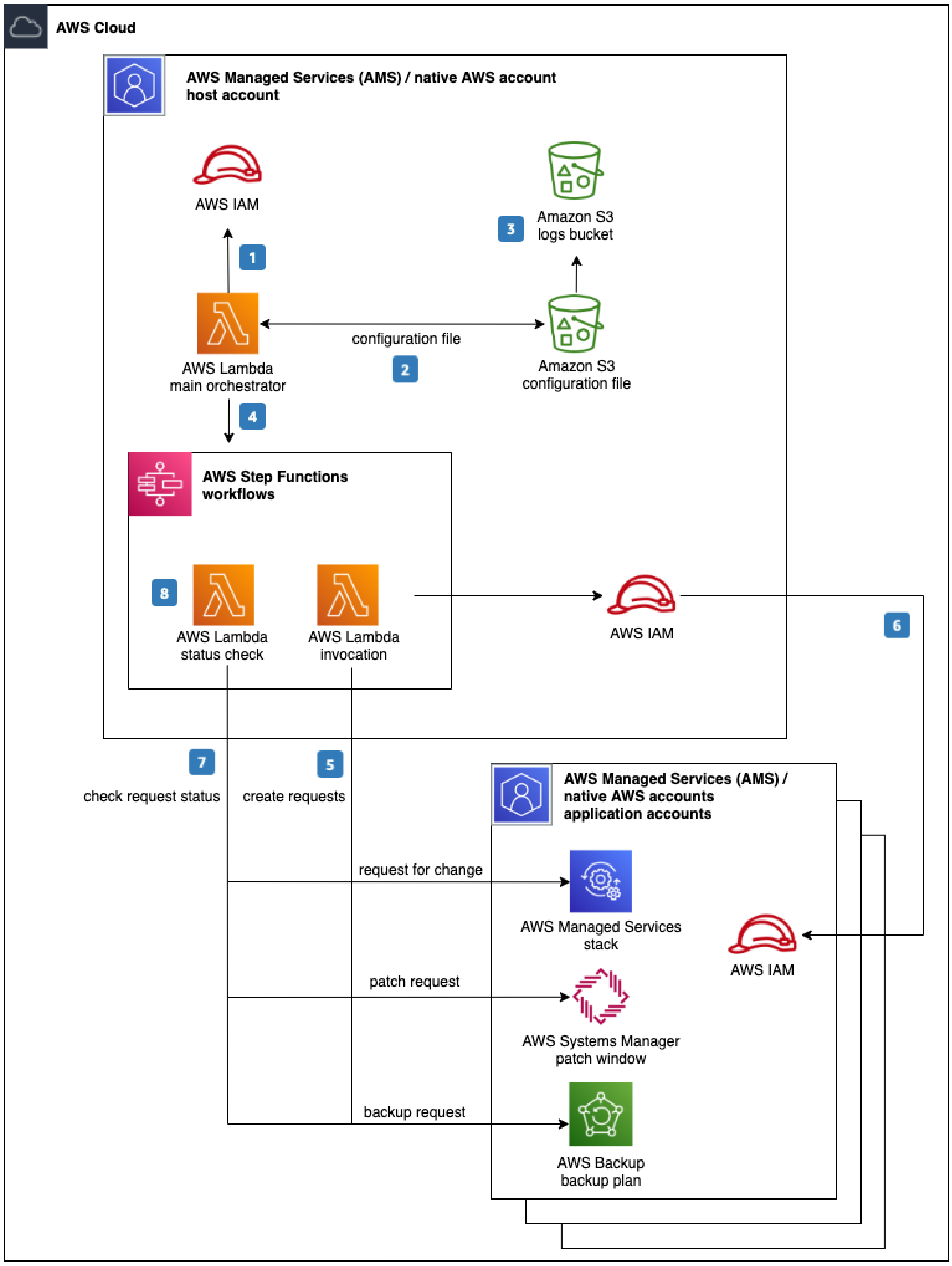
AWS Cloud
AWS Managed Services (AMS) / native AWS account host account
AWS IAM
1
Amazon S3 logs bucket
3
AWS Lambda main orchestrator
configuration file
2
Amazon S3 configuration file
4
AWS Step Functions workflows
8
AWS Lambda status check
AWS Lambda invocation
AWS IAM
6
7
check request status
5
create requests
AWS Managed Services (AMS) / native AWS accounts application accounts
request for change
AWS Managed Services stack
AWS IAM
patch request
AWS Systems Manager patch window
backup request
AWS Backup backup plan

Figure 1: Automated Account Configuration architecture on AWS

The AWS CloudFormation template deploys the following infrastructure:

1. The `main_orchestrator` AWS Lambda function retrieves the IAM role required to complete the account configuration deployment.
2. This Lambda function then downloads the configuration file stored in the Amazon S3 bucket.
3. The S3 bucket hosting the configuration file logs an event to a dedicated logging S3 bucket.
4. After the configuration file is downloaded, the `main_orchestrator` Lambda function performs error checks on the inputs and the list of configuration steps. This Lambda function calls the appropriate AWS Step Functions workflow based on the change type defined in the configuration file.
5. AWS Step Functions call the appropriate `invocation` Lambda function to run the resource creation process in the destination AWS account.
6. The `invocation` Lambda function assumes the appropriate IAM role in the application AWS account containing the necessary permissions to create stacks.
7. Based on the change type, AWS Step Functions may initiate a follow up status check by calling the `status_check` Lambda function on a schedule until the process completes.
8. The `status_check` Lambda function returns the result of the job to AWS Step Functions once the job completes providing a status of the job.

Solution components

Configuration file

This solution uses a JSON configuration file to run the account set up steps. You customize this JSON file to identify the backup and patching activities that this solution automates for you.

By default, this solution supports the following account configuration activities: submit a request for change for AWS Managed Services accounts, set up a backup plan using AWS Backup, and set up a patching schedule using AWS Systems Manager Patch Manager.

This configuration file is set up to support both native AWS accounts and AWS Managed Services accounts as shown in Figure 2. When editing this file, you must comment out the non-relevant section based on your AWS account type (either the `aws_managed_list` or the `customer_managed_list` options).

```
{
  "app_accounts" : [ {
    "account_id" : {
      "app_acct_role"  : "<role_name>",
      "ams_managed_list": [],
      "customer_managed_list": []
    }
    }
  ]
}
```

Figure 2: Configuration file format

For guidance on how to edit the configuration file, refer to Update Account Configuration Files (p. 15) in the Automated deployment section of this guide.

AWS Lambda functions and AWS Step Functions

This solution uses the following AWS Lambda functions:

- `Main_orchestrator` Lambda function: This Lambda function downloads the configurations from the S3 file and performed input checks to ensure all inputs are valid. Once inputs are validated, the `main_orchestrator` Lambda function calls the invocation Lambda function to provision the AWS resources in the target account.
- `Invocation` Lambda functions: These Lambda functions manage the creation of each configuration step like AWS Managed Service RFC, AWS Backup, and AWS patch. When called, this Lambda function downloads the inputs for the configuration step from the Amazon S3 bucket, then assumes a role in the application account and creates the resource in the account.
- `Status_check` Lambda functions: These Lambda functions checks the deployment process and, when the set up process has completed, communicates the completion status back to the AWS Step Functions workflow.

The `main_orchestrator` Lambda function reads the customized configuration file and runs input validation and error handling to ensure the inputs are valid.

This Lambda function then determines the type of change requested by reading the list of changes and calls a set of AWS Step Functions to run the requested actions.

This solution provides the following AWS Step Functions workflows.

- Request for change (RFC): A request for change (RFC) is a request created by either you or AMS through the AMS interface to make a change to your managed environment. For more information about RFCs, Refer to What is AMS change management in the *AWS Managed Services Change Management User Guide*.
- Patch request: This solution creates an AWS System Manager patch baseline so that auto approving patches can be applied. For more information about a patch baseline, refer to AWS Systems Manager Patch Manager in the *AWS Systems Manager User Guide*.
- Backup request: In AWS Backup, you can create backups automatically using backup plans or manually by initiating an on-demand backup. For detailed information about creating backups, refer to Creating a backup in the *AWS Backup Developer Guide*.
- Service request: A request by the customer for an action that you want AWS Managed Service to take on your behalf. For more details, refer to Service request management in the *AWS Managed Services Change Management User Guide*.

Each AWS Step Functions workflow calls a Lambda function that assumes the appropriate IAM role in the destination account and performs the requested action.

The Lambda function downloads the inputs for the requested activity: a change request, a backup request, or a patch request from the configuration S3 bucket and uses those inputs to configure the change in the application account.

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 in the AWS Cloud. This solution creates IAM roles that grant the solution's AWS Lambda functions access to create Regional resources.

The IAM roles created in this solution follow the least privilege model, providing only access to the resources specifically used by this solution. These resources are tagged with unique identifiers, allowing IAM policies to use these tags to run the necessary operations in the specified AWS services.

AWS KMS keys

This solution uses AWS Key Management Service (AWS KMS) keys to run server-side encryption on the configuration file stored in the Amazon S3 bucket. This file can be decrypted only by authorized users, preventing unauthorized access or tampering of the file.

AWS S3 security

This solution implements the following security measures to secure the solution's Amazon S3 bucket, following Security Best Practices for Amazon S3 as provided in the *Amazon S3 User Guide*.

- Complies with the `s3-bucket-ssl-requests-only` rule to explicitly deny access to HTTP requests, allowing only HTTPS traffic.
- Restricts access to the Amazon S3 objects to only the solution-created IAM role.
- Uses the default Amazon S3 encryption at rest methodology for the files stored in the S3 bucket.
- Records all S3 bucket actions in server access logs which are stored in a separate S3 bucket. The solution's default IAM role does not have access to this logs S3 bucket. These access logs may be used in security and access audits, as needed by your organization.

Design considerations

Supported AWS accounts

This solution provides two CloudFormation templates: `automated-account-configuration.template` and `automated-account-configuration-application-acct.template`.

For the `automated-account-configuration.template`, we recommend deploying in a non-workload bearing AWS account to host the solution. This template provides the configuration file, initial Amazon S3 bucket to store the file, the IAM role and AWS KMS keys to provide access to the file, and the AWS Step Functions processes to run the account configurations.

Deploy the second template, `automated-account-configuration-application-acct.template`, to your AWS accounts that should receive the automated account configurations. These target AWS accounts are the application accounts that should receive the configurations that you set up in the JSON file. This template provides the IAM role needed to automated the account activities that you specified in the configuration file.

After you have deployed both templates to the target accounts, you must edit the configuration file to identify the activities that will be run in the application accounts. Configuring the JSON file deployed in your host account is required in order for the solution to run successfully.

Additionally, this solution supports the two types of AWS accounts – AWS Managed Services (AMS) accounts and native AWS accounts. If you are deploying this solution into AMS accounts, a change management process is used to complete the service setup. For native AWS accounts, this solution works directly with the AWS services.

Regional deployments

The Automated Account Configuration solution has been validated in the following AWS Regions:

• us-east-1 (N.Virginia) • us-east-2 (Ohio) • us-west-1 (N.California) • us-west-2 (Oregon) • eu-west-1 (Ireland) • eu-west-2 (London) • eu-west-3 (Paris) • eu-central-1 (Frankfurt)	• eu-north-1 (Stockholm) • sa-east-1 (Sao Paulo) • ap-northeast-1(Tokyo) • ap-northeast-2 (Seoul) • ap-south-1 (Mumbai) • ap-southeast-1 (Singapore) • ap-southeast-2 (Sydney) • ca-central-1 (Canada)

AWS CloudFormation templates

To automate deployment, this solution uses the following AWS CloudFormation templates, which you can download before deployment:

View Template **automated-account-configuration.template:** Use this template to launch the initial Amazon S3 bucket containing the JSON configuration file, IAM role, AWS KMS keys, and AWS Step Functions to your host AWS account.

View Template **automated-account-configuration-application-acct.template:** Use this template to launch the IAM role to the AWS accounts that is receiving the automated account configurations.

Automated deployment

Before you launch the solution, review the architecture in this guide. Follow the step-by-step instructions in this section to configure and deploy the solution into your account.

Time to deploy: Approximately 30 minutes

Prerequisites

This solution requires two or more AWS accounts or AWS Managed Services accounts. If you have not completed the set up of your AWS accounts yet, refer to How do I create and activate a new Amazon Web Services account?

Before deploying the CloudFormation templates, determine the AWS account that will host the `automated-account-configuration.template` template. Additionally, determine the AWS accounts that should contain the `automated-account-configuration-application-acct.template` template. For guidance on selecting the suitable AWS accounts, refer to Supported AWS accounts (p. 10) in the Design considerations section of this guide.

Configuration steps

This solution provides set up and configuration details in the JSON configuration file for creating a backup plan using AWS Backup and for providing a patching baseline in Amazon EC2 instances. In order to activate these set ups, you must provide the input values for the backup plan and the patching application using AWS CLI and Git.

AWS CLI version 2 is required. To install, refer to Installing, updating, and uninstalling the AWS CLI in the *AWS CLI User Guide*.

If you do not have Git, refer to Getting Started – Installing Git in the Git documentation.

Launch the automated-account-configuration stack

Important
This solution includes an option to send anonymous 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 Policy.
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 template and deploy the solution. For more information, refer to the Collection of operational metrics (p. 23) section of this guide.

This automated AWS CloudFormation template deploys the solution in the AWS Cloud. You must complete the prerequisites before launching the stack.

Note
You are responsible for the cost of the AWS services used while running this solution. For more details, visit to the Cost (p. 2) section in this guide, and refer to the pricing webpage for each AWS service used in this solution.

Use the following steps to deploy this stack to your host AWS account.

Note
For guidance on selecting the AWS account to be the host account, refer to Supported AWS accounts (p. 10) in the Design considerations section of this guide.

1. Sign in to the AWS Management Console and select the button to launch the `automated-account-configuration.template` AWS CloudFormation template.

   Launch Solution

   Alternatively, you can download the template as a starting point for your own implementation.
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.
3. On the **Specify stack details** page, assign a name to your solution stack. For information about naming character limitations, refer to IAM and STS Limits in the *AWS Identity and Access Management User Guide*.
4. Under **Parameters**, review the parameters for this solution template and modify them as necessary. This solution uses the following default values.

Parameter	Default	Description
ExternalIamID	*`<Requires input>`*	Unique IAM identifier used for cross account role assumption. For information about obtaining the IAM ID, refer to How to use an external ID in the *AWS IAM User Guide*.
IAMPermissionsBoundary	<Optional input>	For AMS accounts, this is the ARN of the policy used to set the permissions boundary for the role. Provide this value if your change management process requires one. For information about AWS Permission boundary, refer to Permissions boundaries for IAM entities in the *AWS IAM User Guide*.

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 will create AWS Identity and Access Management (IAM) resources.
8. Choose **Create stack** 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 five minutes.

Launch the automated-account-configuration-application-acct stack

Note
You are responsible for the cost of the AWS services used while running this solution. For more details, visit to the Cost (p. 2) section in this guide, and refer to the pricing webpage for each AWS service used in this solution.

Use the following steps to deploy this stack to your target AWS accounts.

Note
For guidance on selecting the AWS account to be the application accounts, refer to Supported AWS accounts (p. 10) in the Design considerations section of this guide.

1. Sign in to the AWS Management Console and select the button to launch the `automated-account-configuration-application-acct.template` AWS CloudFormation template.

   Launch Solution

   Alternatively, you can download the template as a starting point for your own implementation.
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.
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. For information about naming character limitations, refer to IAM and STS Limits 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
ToolsAccountID	*`<Requires input>`*	Enter the AWS account ID hosting the solution, where the `automated-account-configuration.template` was launched. To find your AWS account ID, refer to Finding your AWS account ID in the *AWS IAM User Guide*.
IAMPermissionsBoundary	<Optional input>	For AWS Managed Services accounts only, this is the ARN of the policy used to set the permissions boundary for the role. Provide this value if your change management process requires one. For information

Parameter	Default	Description
		about identifying this value, refer to the Permissions boundaries for IAM entities in the *AWS IAM User Guide*.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.
8. On the **Review** page, review and confirm the settings. Check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources.
9. Choose **Create stack** 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 five minutes.

Post-configuration tasks

Return to your AWS account hosting the solution, where the `automated-account-configuration.template` stack was deployed. Before you can initiate the automated configurations for your application accounts, you must set up the account configuration options and input the appropriate values for the AWS services (AWS Backup and/or AWS Systems Manager Patch Manager) that you want to automate. Then, you must upload the configuration file to the solution's Amazon S3 bucket.

Update the account configuration file

The `Account_Config.json` configuration file used for the account set up.

Use the following steps to update and upload the file:

1. From your terminal, run the following command:

```
git clone https://github.com/awslabs/automated-account-configuration.git
```

Note
You are cloning the Automated Account Configuration GitHub repository into your local directory. For more information, refer to Getting a Git Repository.

2. Navigate to your local copy of the source code and locate the following file: `source/S3_Files/Account_Configuration/Account_Config.json`.

The `Account_Config.json` file is set up to support both native AWS accounts and AWS Managed Services accounts. When editing this file, you must comment out the non-relevant section based on your AWS account type (either the `aws_managed_list` or the `customer_managed_list` options).

```
{
  "app_accounts" : [ {
    "account_id" : {
      "app_acct_role" : "<role_arn>",
      "ams_managed_list": [],
      "customer_managed_list": []
    }
    }
  ]
}
```

This file contains three main sections:

- **account_id:** The AWS account ID where resources are created. To find your AWS account ID, refer to Finding your AWS account ID In the *IAM User Guide*.
- **ams_managed_list:** This section defines a sample of AWS Managed Services (AMS) change types for the different bootstrap steps for AMS accounts. This solution includes the following change types: AMS Service Request creation, creating SSM Maintenance Window RFC, and creation AWS Backup plan RFC. This is not a comprehensive list of AMS Change Types. For more info, refer to AMS Change Type Reference Guide. For examples supported by this solution, refer to Example 1 (p. 17).
- **customer_managed_list:** This section defines a sample for the different bootstrap steps for native AWS accounts (accounts that are not managed by AMS). The supported automations include: creating a backup plan and working with custom patch baselines. For examples, refer to Example 2 (p. 17).

Within each of the `ams_managed_list` and `customer_managed_list` sections there are parameters that you may need to modify per specific use case. The following table shows these different parameters that are part of `Account_Config` of the two corresponding sections (`ams_managed_list` and `customer_managed_list`)

Parameter	Values	Description
Key	• `Service_Request` • `ct-0el2j07llrxs7` • `ct-2hyozbpa0sx0m` • `backup` • `patch`	Specifies the type of bootstrapping or operational steps that are needed to provision for the AWS account where resources are created (the application account). Refer to Example 1 and Example 2 for supported key values.
Exec_Params	• `Service_Request.json` • `ct-0el2j07llrxs7_Params.json` • `ct-2hyozbpa0sx0m_Params.json` • `customer-managed-backup-default.json` • `customer-managed-patch-baseline.json`	Specifies the file name containing the input for the request type. Files are stored in the `source/S3_Files/JSON_Template` file. Refer to Example 1 and Example 2 for supported Exec_Params values.
Wait_Time	`60`	Sets the wait time, in seconds, for the AWS Step Functions workflow before checking status. This allows the time needed for the change type to run. The recommended range is from 30 to 180 (seconds).
Comment	<Optional input>	Provides a text description of the AWS services you are configuring in this solution – either AWS Backup or AWS Systems Manager Patch Manager. Refer to Example 1 and Example 2 for example descriptions.

Example 1

This `Account_Config.json` file example shows the input needed to create both AMS and native AWS operational bootstrapping steps.

```
"app_accounts" : [ {
    "<account-id>" : {
      "app_acct_role" : "ApplicationAccountRole",
      "ams_managed_list": [
        {
          "Key": "Service_Request",
          "Exec_Params" : "Service_Request.json",
          "Wait_Time": 60,
          "Comment": "Create Service Request"
        },
        {
          "Key": "ct-2hyozbpa0sx0m",
          "Exec_Params" : "ct-2hyozbpa0sx0m_Params.json",
          "Wait_Time": 60,
          "Comment": "Create Default Backup Plan - Note: to check latest value for key,
 visit AWS Managed Services https://aws.amazon.com/managed-services/"
        }
      ]
    }
  }
  ]
}
```

Example 2

This `Account_Config.json` file example shows the provisioning of native AWS accounts only. The input creates an AWS Backup plan and an AWS Systems Manager patch baseline.

```
{
  "app_accounts" : [ {
    "<account-id>" : {
      "app_acct_role" : "ApplicationAccountRole",
      "customer_managed_list": [
        {
          "Key": "backup",
          "Exec_Params" : "customer-managed-backup-default.json",
          "Wait_Time": 60,
          "Comment": "Create backup Plan for customer managed stacks"
        },
        {
          "Key": "patch",
          "Exec_Params" : "customer-managed-patch-baseline.json",
          "Wait_Time": 60,
          "Comment": "Create patch baseline for customer managed stacks"
        }
      ]
    }
  }
  ]
}
```

The `source/S3_Files/JSON_Template/` directory contains files providing the inputs to the RFCs, backups, and patching specifications. This directory is used for configuring the input to AWS and AWS Managed Services (AMS) accounts. There are three different configuration file type\s under the `JSON_Template` directory:

- **AMS RFC configuration files:** This solution contains examples of AWS Backup and SSM Patch Window. Refer to Example 3 and Example 4.
- **AMS Service Requests configuration:** AMS customers create service requests to communicate with the AMS Operations Team. Refer to Example 4.
- **Customer managed configuration:** The solution includes examples of creating an AWS Backup plan and AWS Systems Manager patch baselines. Refer to Example 5.

Example 3:

```
{
  "ChangeTypeVersion": "1.0",
  "ChangeTypeId": "ct-0el2j07llrxs7",
  "Title": "Patch-Window-Create-RFC"
}
```

```
{
  "ChangeTypeId" : "ct-2hyozbpa0sx0m",
  "ChangeTypeVersion" : "2.0",
  "Title": "AWS Backup create Bronz backup plan"
}
```

Example 4:

```
// Example of a patch window
{
  "Cutoff": 4,
  "Description": "WindowsDevelopment",
  "MaxConcurrency": "33%",
  "MaxErrors": "100%",
  "ScheduleTimeZone": "America/New_York",
  "Duration": 6,
  "Name": "WindowsDevelopment",
  "NotificationEmails": [
    "email"
  ],
  "PatchGroup": "myPatchGroup",
  "Schedule": "cron(0 11 ? FEB,MAR,JUN,AUG,SEP,NOV,DEC 6#2 *)"
}

//Example of Backup Plan
{
  "VpcId": "Vpc-id",
  "Description": "Bronze Backup Plan" ,
  "Parameters": {
    "BackupPlanName": "Bronze" ,
    "ResourceTagKey": "RPO" ,
    "ResourceTagValue": "Bronze" ,
    "BackupRule1Name": "BronzeBackupRule1" ,
    "BackupRule1Vault": "ams-custom-backups" ,
    "BackupRule1CompletionWindowMinutes": 180 ,
    "BackupRule1DeleteAfterDays": 360 ,
    "BackupRule1MoveToColdStorageAfterDays": 180 ,
    "BackupRule1StartWindowMinutes": 60 ,
    "BackupRule1RecoveryPointTagKey": "" ,
    "BackupRule1RecoveryPointTagValue": "" ,
    "BackupRule1ScheduleExpression": "cron(0 10 ? * * *)"
  },
  "StackTemplateId": "stm-sc68a620000000000",
  "TimeoutInMinutes": 60,
  "Name": "TEST_STACK"
}
// Example of a service request
{
  "subject" : "Request to <request-details-here> to AWS Managed Services",
  "serviceCode": "sentinel-service-request",
  "severityCode": "normal",
  "categoryCode": "feature-request",
  "communicationBody" :"Request to <request-details-here> to AWS Managed Services",
  "ccEmailAddresses" : "email"
}
```

Example 5:

```
//Example of creating customer managed AWS backup plan
{
  "VpcId": "vpc_id",
  "Description": "My test AWS Backup plan" ,
  "Parameters": {
    "BackupPlanName": "TestBackupPlan" ,
    "ResourceTagKey": "RPO" ,
    "ResourceTagValue": "CustomerManagedBronze" ,
    "BackupRule1Name": "BronzeBackupRule1" ,
    "BackupRule1Vault": "Customer-custom-backups" ,
    "BackupRule1CompletionWindowMinutes": 180 ,
    "BackupRule1DeleteAfterDays": 360 ,
    "BackupRule1MoveToColdStorageAfterDays": 180 ,
    "BackupRule1StartWindowMinutes": 60 ,
    "BackupRule1RecoveryPointTagKey": "" ,
    "BackupRule1RecoveryPointTagValue": "" ,
    "BackupRule1ScheduleExpression": "cron(0 10 ? * * *)"
  }
}

//Example of creating customer managed patch baseline

{
  "OperatingSystem": "WINDOWS",
  "Name": "TestPatchBaseline",
  "GlobalFilters": {
    "PatchFilters": [
      {
        "Key": "PRODUCT",
        "Values": [
          "Windows7"
        ]
      }
    ]
  },
  "ApprovalRules": {
    "PatchRules": [
      {
        "PatchFilterGroup": {
          "PatchFilters": [
            {
              "Key": "PRODUCT",
              "Values": [
                "Windows8"
              ]
            }
          ]
        },
        "ComplianceLevel": "CRITICAL",
        "ApproveUntilDate": "2021-12-30",
        "EnableNonSecurity": false
      }
    ]
  }
}
```

Note
Before running the solution, verify that the inputs are updated to reflect your environment.

Upload the account configuration file

After completing the setup of the `Account_Config.json` file, upload or sync your local copy with the configuration Amazon S3 bucket. You can upload the files either using the AWS Management Console or via AWS CLI.

Using the AWS Management Console:

1. Sign in to the host AWS account where you deployed the `automated-account-configuration.template` stack.
2. Navigate to the Amazon S3 console.
3. Location the solution's configuration S3 bucket.
4. Create the following folders:
   - `Account_Configuration`
   - `JSON_Template`
5. Upload the configuration files to the appropriate directory:
   - Under the `/source/S3_Files/Account_Configuration` directory, upload the `Account_Config.json` file.
   - Under the `/source/S3_Files/JSON_Template` directory, upload the edited files, including the AMS RFC configuration files, the AWS Service Requests configuration file, and the Customer managed configuration file.

Using the AWS CLI v2:

1. Refresh your credentials on your AWS CLI by running: `aws configure`.
2. Upload the following files to the destination folder using the AWS Management Console or by running the command in the source directory.

```
aws s3 sync .Source/S3_Files/ s3://<name-of-S3—bucket>
```

The `main_orchestrator` Lambda function is ready to run using the configuration values that you entered.

Starting the solution

To start the solution, you need to invoke the `main_orchestrator` Lambda function. For information about invoking Lambda functions, refer to Invoke the Lambda function in the *AWS Lambda Developer Guide*.

Additional resources

AWS services

• AWS CloudFormation	• AWS Step Functions
• AWS Key Management Service	• Amazon S3
• AWS Lambda	• AWS Identity & Access Management
• AWS Managed Services	• AWS Backup

AWS documentation

- About AWS Systems Manager Patch Manager

Uninstall the solution

To uninstall the Automated Account Configuration solution, you must delete the stack from the host AWS account and all AWS accounts where the resources are deployed. Use the following procedure to uninstall this solution from all pertinent AWS accounts.

Using the AWS Management Console

1. Sign in to the host AWS account where you deployed the `automated-account-configuration.template` stack.
2. Navigate to the Amazon S3 console and delete the files in the configuration Amazon S3 bucket.
3. Navigate to the AWS CloudFormation console.
4. Select this solution's installation stack.
5. Choose **Delete**.
6. Sign in to AWS account where you deployed the `automated-account-configuration-application-acct.template` stack (the application account).
7. Navigate to the AWS CloudFormation console.
8. Select this solution's installation stack.
9. Choose **Delete**.

Note
Repeat steps 6 through 9 to delete the solution stack from every application account.

Collection of operational metrics

This solution includes an option to send anonymous 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:

- **Unique ID (UUID):** Randomly generated, unique identifier for each aws-automated-configuration 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 `automated-account-configuration.template` 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:

```
"Send" : {
   "AnonymousUsage" : { "Data" : "Yes" }
},
```

   to:

```
"Send" : {
   "AnonymousUsage" : { "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**, choose **Choose file** and select the edited template from your local drive.
8. Choose **Next** and follow the steps in Launch the automated account configuration stack (p. 12) in the Automated Deployment section of this guide.

Source code

Visit our GitHub repository download the source files for this solution and to share your customizations with others. The Automated Account Configuration templates are generated using the AWS Cloud Development Kit (CDK) (AWS CDK). Refer to the README.md file for additional information.

Revisions

Date	Change
October 2021	Initial release
December 2021	Release version 1.0.1: Added local build instructions and unit tests. For more information, refer to the CHANGELOG.md file in the GitHub repository.

Contributors

- Nadim Rabbani, Principal, Cloud Application Architect
- Barb Malik, Principal, Specialist Solutions Architect
- Rafal Aljuboori, Cloud Architect

Notices