Sample Access Manager Auto Scaling Deployment on AWS

AWS EC2 Auto Scaling helps you maintain the application availability and automatically adds and removes EC2 instances according to the conditions you define.

In this sample deployment, based on auto scaling configuration, new instances of Identity Server and Access Gateway are created automatically using custom AMIs. These instances are automatically imported into Administration Console and assigned to the respective clusters. These instances are also assigned automatically to AWS Load Balancer. Similarly, when not in use, the instances are removed from Administration Console and then get terminated automatically.

To automate these steps, Access Manager credentials are stored in the AWS EC2 Parameter Store as a secure string encrypted by KMS. All instances of Access Manager are assigned an IAM role to read the encrypted credentials and perform the auto scaling configuration automatically.

Scaling Options

You can configure Access Manager auto scaling in the following ways:

Option	Description
Target Tracking Scaling	Responds to the changing requirement. It reduces the need for manual provisioning Amazon EC2 capacity and enables you to meet the demand of Access Manager closely. For example, you use target tracking scaling policies to select a load metric for Access Manager, such as CPU utilization. Auto scaling automatically adjusts the number of EC2 instances required to maintain your target metric.
Scheduled Scaling	Automatically schedules the appropriate number of EC2 instances based on the predicted demand. It enables you to scale your application before the predicted load changes. For example, every week the traffic to Access Manager starts to increase on Monday, remains high on Thursday, and starts to decrease on Friday. You can plan your scaling activities based on the known traffic patterns.

IMPORTANT: The auto scaling configuration explained in this documentation is supported only for Identity Server and Access Gateway components of Access Manager. To auto scale LDAP user stores and web servers, see Amazon EC2 Auto Scaling (https://aws.amazon.com/ec2/autoscaling/).

Watch the following video to understand how the auto scaling of Access Manager works in AWS:

http://www.youtube.com/watch?v=IJYx3qbA1gQ

Watch the following video to understand the configuration of Access Manager auto scaling in AWS:

http://www.youtube.com/watch?v=X7OwBHuQFmU

In this Article

1 Prerequisites for Sample Access Manager Auto Scaling Deployment on AWS

- ❒ Access Manager 4.5 or later is deployed on AWS and contains working Administration Console, Access Gateway cluster, and Identity Server cluster.

  See Deploying Access Manager on Amazon Web Services EC2.
- ❒ An administrative Access Manager account is available.
- ❒ Load balancer is configured and the load balancer target type is specified as `instance`.

  For information about configuring the target group, see Target Groups for Your Network Load Balancers.
- ❒ The following details are available to be specified during the auto scaling configuration:
  - Amazon Virtual Private Cloud (VPC) ID of the existing Access Manager deployment.
  - The subnet ID of the existing Identity Server and Access Gateway deployments.

    The new instances of Identity Servers and Access Gateway servers will be deployed in the respective subnets.
  - A unique ARN of Identity Server and Access Gateway load balancer target group. The newly created Identity Servers and Access Gateway servers will be registered to these targets group automatically.
- ❒ To configure auto scaling for Access Gateway, the existing Access Gateway instances are identified and terminated before deploying the auto scaling configuration.

2 CloudFormation Templates

To deploy the Access Manager auto scaling configuration, you require CloudFormation templates. Download AccessManagerAutoscale.zip and unzip it.

NOTE: These CFT files work for Access Manager 4.5.x versions.

NOTE: When you use the sample cloud formation template (CFT) for Access Manager auto-scaling, Access Manager admin credentials are encrypted using AWS KMS and stored in the AWS parameter store. Therefore, any AWS user with the AWS IAM role of AdministratorAccess can read these parameter values.

The ZIP file contains the following files:

File	To Configure...
Schedule_Scaling_Access_Gateway.yml	Scheduled auto scaling for Access Gateway
Schedule_Scaling_Identity_Server.yml	Scheduled auto scaling for Identity Server
Schedule_Scaling_Identity_Server_Access_Gateway.yml	Scheduled auto scaling for both Identity Server and Access Gateway
Target_Tracking_Scaling_Access_Gateway.yml	Auto scaling for Access Gateway based on CPU utilization or memory utilization
Target_Tracking_Scaling_Identity_Server.yml	Auto scaling for Identity Server based on CPU utilization or memory utilization
Target_Tracking_Scaling_Identity_Server_Access_Gateway.yml	Auto scaling for both Access Gateway and Identity Server based on CPU utilization or memory utilization.

The ZIP file contains the following nested CFT:

File	Description
AMIofIDPInstance.yml	Automatically installs a temporary Identity Server instance for automatically creating Identity Server AMI.
AMIofMAGInstance.yml	Automatically installs a temporary Access Gateway instance for automatically creating Access Gateway AMI
ASGLifeCycleHooks.yml	Creates the AWS auto scaling life cycle hooks
ASScalingPolicy.yml	Creates the AWS auto scaling policy based on CPU or Memory utilization
ASScheduledAction.yml	Creates an AWS auto scaling scheduled action based on the specified time
AssignInstanceToASG.yml	Assigns a newly created primary Access Gateway to the auto scaling group
AutoScalingGroup	
IDPASLaunchConfig.yml	Creates the AWS auto scaling launch configuration for Identity Servers
LambdaIDPAMI.yml	Creates an AWS Lambda function based on Python that creates an AMI for Identity Server
LambdaMAGAMI.yml	Creates an AWS Lambda function based on Python that creates an AMI for Access Gateway
LambdaTerminateIDPInstance.yml	Creates an AWS Lambda function based on Python that terminates the temporary Identity Server after the AMI creation
LambdaTerminateMAGInstance.yml	Creates an AWS Lambda function based on Python that terminates the temporary Access Gateway server after the AMI creation
MAGASLaunchConfig.yml	Creates the AWS auto scaling launch configuration for Access Gateway
PrimaryAG.yml	Creates a new single instance of Access Gateway server with a new AMI. This new instance is used as the primary Access Gateway
RoleEC2toS3andKMS.yml	Creates AWS KMS key and AWS IAM Roles to be used for EC2 instances

File	Description
RoleLambdaExecution.yml	Creates the required AWS IAM roles for AWS Lambda functions
RoleLifecycleHook.yml	Creates the required AWS IAM roles for AWS auto scaling life cycle hooks

The ZIP file contains the following nested scripts:

Script File	Description
autoInstallIDPForAMI.sh	Automatically installs the temporary Identity Server for creating Identity Server AMI
autoInstallMAGForAMI.sh	Automatically installs the temporary Access Gateway Server for creating Access Gateway Server AMI
aws-check-state.service	Creates a Linux service to check the status of EC2 instance periodically to trigger scale down scripts whenever the instance enters TERMINATING_WAIT state
bootStrapImportIDP.sh	Automatically imports Identity Server to Administration Console, adds Identity Server to the cluster, updates existing nodes in the cluster one after the other, and performs the health check of the new server. Based on the health check response, it proceeds with the auto scaling process.
bootStrapImportMAG.sh	Automatically imports Access Gateway to Administration Console, adds Access Gateway to the cluster, updates all other nodes in the cluster, performs the health check of the new server. Based on the health check response, it proceeds with the auto scaling process.
json_parser.py (python utility script)	Parses the API response and validates a few details provided by Access Manager details, such as cluster name
probeInstanceMetadata.sh	Along with aws-check-status.service, periodically probes the EC2 instance state
scaleDownIDP.sh	Automatically removes Identity Server from Administration Console during the scale down process
scaleDownMAG.sh	Automatically removes Identity Server from Administration Console during the scale down process

3 AWS Services Deployed in the Access Manager Auto Scaling Configuration

AWS Service	Description
Custom Amazon Machine Image (AMIs)	AMI of Identity Server and Access Gateway are created that are used in the auto scaling process. For more information about AMI, see Amazon Machine Images.
AWS Auto Scaling	EC2 auto scaling launch configuration and auto scaling groups are created. For more information, see What Is Amazon EC2 Auto Scaling.

AWS Service	Description
Key Management Service (KMS)	The AWS KMS key is created. This key is used for encrypting and decrypting Access Manager credentials stored in the AWS EC2 Parameter store. For more information about KMS, see AWS Key Management Service.
System Manger Parameter Store	Access Manager credentials are stored in the EC2 Parameter store as a secure string. For more information, see AWS Systems Manager Parameter Store.
Auto Scaling Life Cycle Hooks	Necessary auto scaling life cycle hooks are created to perform customized actions whenever auto scaling launches or terminates an instance. For more information, see Amazon EC2 Auto Scaling Life Cycle Hooks.
Identity and Access Management (IAM) Roles	Necessary AWS IAM roles are created to automate the tasks involved in auto scaling. For more information, see AWS Identity and Access Management.
AWS Lambda Functions	AWS Lambda functions are created that create AMI of Identity Servers and Access Gateway automatically. For more information, see AWS Lambda.
AWS Simple Notification Service (SNS)	An AWS SNS topic is created. Subscribe to this to receive notifications whenever an instance is launched and terminated during auto scaling. For more information, see Amazon Simple Notification Service.

4 Deploying the Sample Auto Scaling Configuration

Perform the following steps to deploy the Access Manager auto scaling configuration on AWS:

1. Download the CloudFormation templates package and extract the `AccessManagerAutoscale.zip` file.
2. Log in to AWS Console.
3. Click **S3 > Create bucket**.
4. Specify a bucket name.
5. Select a region and click **Next**.
6. Click the bucket name that you created.
7. Click **Upload**.
8. Copy the following folders from the extracted content to the bucket:
   - logs
   - source
9. Upload the Access Manger installers (`AM_45_AccessGatewayService_Linux64.tar.gz` and `AM_45_AccessManagerService_Linux64.tar.gz`) to the S3 bucket.
10. Browse to the S3 bucket > source folder.

The CloudFormation package contains the following six templates:

Filename	Description
`Schedule_Scaling_Identity_Server_Access_Gateway.yml`	To deploy scheduled auto scaling of Identity Server and Access Gateway.
`Target_Tracking_Scaling_Identity_Server_Access_Gateway.yml`	To deploy target tracking auto scaling of Identity Server and Access Gateway.
`Schedule_Scaling_Identity_Server.yml`	To deploy scheduled auto scaling of Identity Server.
`Target_Tracking_Scaling_Identity_Server.yml`	To deploy target tracking auto scaling of Identity Server.
`Schedule_Scaling_Access_Gateway.yml`	To deploy scheduled auto scaling of Access Gateway.
`Target_Tracking_Scaling_Access_Gateway.yml`	To deploy target tracking auto scaling of Access Gateway.

11 Click the required CloudFormation template and copy **Object URL**.

12 Navigate to **Services > CloudFormation** and click **Create Stack**.

13 Select **Specify an Amazon S3 template URL**.

14 Specify the Object URL that you copied in Step 11 and click **Next**.

15 In **Stack name**, specify a name for the auto scaling configuration.

16 Specify the following details in **Access Manager Configuration**:

Parameter	Description
Administrator Name	The name of the Access Manager administrator of the existing deployment.
Administrator Password	The password of the Access Manager administrator.
Administration Console IP Address	The IP address of Administration Console.
Identity Server Cluster Name	The name of the Identity Server cluster for which auto scaling needs to be configured.
Access Gateway Cluster Name	The name of the Access Gateway cluster for which auto scaling needs to be configured.
Platform	The name of the required operating system.

IMPORTANT: After specifying cluster details, ensure that you do not change the cluster name and cluster ID in the AWS Parameter Store. If you change the cluster ID, the auto scaling configuration will not work. If you change the cluster name, the cluster logs will display incorrect cluster name. This is applicable for both Identity Server cluster and Access Gateway cluster.

17 Specify the following details in **Access Manager Installer Location**:

Parameter	Description
S3 Bucket Name	The name of the S3 bucket that contains auto scaling files and Access Manager installer.
S3 Bucket Region	The AWS region where the S3 bucket is created. For example, if the region is Asia Pacific (Mumbai), specify `ap-south-1`. For the list of AWS regions and their codes, see Available Regions.

18 Specify the following details in **EC2 Configuration**:

Parameter	Description
Instance Type	The AWS EC2 instance type for Access Manager components. To understand the instance types, see Amazon EC2 Instance Types and to understand the pricing for each instances, see Amazon EC2 Pricing.
SSH KeyPair Name	The existing EC2 key pair to enable SSH access to deployed servers.
VPC	ID of the Virtual Private Cloud (VPC) where Access Manager is deployed.
Identity Server Subnet	The existing subnets for Identity Servers deployment in the Access Manager VPC. Ensure that you select the subnets which are part of the selected VPC.
Access Gateway Server Subnet	The existing subnets for the Access Gateway deployment in the Access Manager VPC. Ensure that you select the subnets which are part of the selected VPC.
AWS Administrator	The AWS Administrator username with required rights to deploy auto scaling. This value is case-sensitive.
Http Proxy Server	This auto scaling deployment needs Internet access to configure few AWS services. In your environment, if you rely on any http_proxy server for the Internet access, specify the http_proxy server address in the `http://proxy_server_ip:port` format. For example, `http://192.168.56.200:3128`

19 Specify the following details in **Auto Scaling Configuration**:

Parameter	Description
Initial Size	The initial size of the auto scaling group.
Minimum Size	The minimum required size for the auto scaling group. Access Manager does not scale in after this limit. It must be a numerical value.
Maximum Size	The maximum required size for the auto scaling group. Access Manager does not scale out beyond this limit. It must be a numerical value.
Scale Up Time	Access Manager nodes scale out to the maximum size at this time. Specify the time in the `UNIX cron` format.
Scale Down Time	Access Manager nodes scale in to the minimum size at this time. Specify the time in the `UNIX cron` format.

Parameter	Description
Metric Threshold Value	The minimum required size for the auto scaling group. Access Manager does not scale in after this limit. It must be a numerical value. This option is available only when you use a CloudFormation template for target tracking auto scaling.
Metric Value	The maximum required size for the auto scaling group. Access Manager does not scale out beyond this limit. It must be a numerical value. This option is available only when you use a CloudFormation template for target tracking auto scaling.
Load Balancer Target Group ARN of Identity Server	ARN of the Identity Server target group for load balancing in the `"arn:aws:elasticloadbalancing:region:account-id:targetgroup/target-group-name/target-group-id"` format.
Load Balancer Target Group ARN of Access Gateway	ARN of the Access Gateway target group for load balancing in the `"arn:aws:elasticloadbalancing:region:account-id:targetgroup/target-group-name/target-group-id"` format.

20 Click Next.

21 (optional) In Tags, specify the Key and Value for the resources available in the stack. This value is applied to all resources created by the CloudFormation template.

22 In Advanced, specify additional details based on your requirements.

23 Click Next and review the stack details that you specified.

24 Select the acknowledgment check boxes and click Create.

5 Post-Deployment Tasks

When the auto scaling configuration is deployed, new instances of Identity Server and Access Gateway are created. Auto scaling works seamlessly with new instances that are created as part of the auto scaling group. The existing Identity Server and Access Gateway instances must be deleted.

Perform the following steps after the auto scaling configuration is deployed:

For Identity Servers

1 Log in to AWS EC2 Console.

2 Identify and terminate the AWS EC2 Identity Server instances that are created manually before deploying the auto scaling group.

3 Log in to Administration Console.

4 Go to the Identity Server cluster, identify, and delete Identity Server nodes that are added manually before deploying the auto scaling configuration.

IMPORTANT: If the initial size of the Identity Server auto scaling is two or more than two, all scaled-up nodes might not appear with a green health status immediately. If the nodes do not appear with a green health status even after one hour, perform the following steps:

1 Log in to Administration Console.

2 Go to the Identity Server cluster and stop all nodes.

3 Identify and delete Identity Server nodes that are added manually before deploying the auto scaling configuration.

4 Restart all nodes.

For Access Gateway

1 Log in to AWS EC2 Console.

2 Go to the Access Gateway auto scaling group.

3 Ensure that the primary Access Gateway instance is protected against scale-in. To verify, select the primary instance and right-click **Instance Protection**. Ensure that the **Enable Set Scale In Protection** option is enabled.

4 Log in to Administration Console.

5 Go to the Access Gateway cluster.

6 Ensure that the Access Gateway nodes that are created as part of the auto scaling group are listed in the Access Gateway cluster.

7 Ensure that the primary Access Gateway node IP is the same IP that is listed in the auto scaling group in AWS EC2 Console.

6 Troubleshooting Auto Scaling

The auto scaling configuration automatically creates log files and uploads the log files to the S3 bucket in the following scenarios:

- During AMI creation of Identity Server and Access Gateway
- Before terminating instances of Identity Server and Access Gateway

You can analyze these logs to identify the problem areas and troubleshoot.

The following table lists log files that are created and uploaded automatically to the **S3 bucket > Logs** location:

Filename	Description
`/logs/idp_ami_<log creation date>_<xxxxxx>.tar.gz`	Created during the AMI creation for Identity Server
`/logs/ag_ami_<log creation date>_<xxxxxx>.tar.gz`	Created during the AMI creation for Access Gateway
`/logs/idp_i-<instance ID>_<instance IP address>.tar.gz`	Created during the termination of an Identity Server instance
`/logs/ag_i-<instance ID>_<instance IP address>.tar.gz`	Created during the termination of an Access Gateway instance

7 Auditing Access Manager Auto Scaling

You can configure Administrator Console to generate audit events whenever a new server is imported or deleted as part of the auto scaling process.

Perform the following steps:

1. In Administration Console Dashboard, click **Auditing**.
2. Under **Management Console Audit Events**, select the following options:
   - **Server Imports**: An event is generated whenever a server is imported into Administration Console.
   - **Server Deletes**: An event is generated whenever a server is deleted from Administration Console.
3. Click **OK**.

8 Monitoring Access Manager Auto Scaling

8.1 Monitoring the Activity History

1. Log in to AWS Console.
2. Click **Services > Auto Scaling > Auto Scaling Groups**.
3. Select an auto scaling group.
4. Click **Activity History**.
5. On the **Activity History** page, you can see various activity details of the automatically scaled instances, such as start time, end time, and reason of scaling.

8.2 Subscribing to Auto Scaling Notifications

The auto scaling configuration creates two Simple Notification Service (SNS) topics automatically after a successful deployment. These SNS topics are created for Identity Server and Access Gateway. You can subscribe to these topics to receive notifications whenever a change occurs in the auto scaling deployment.

Perform the following steps to configure SNS:

1. Log in to AWS Console.
2. Click **Services.**
3. In **Find Services**, search for `Simple Notification Service`.
4. Select the SNS topic for which you want to receive notifications.
5. Click **Create subscription**.
6. In **Protocol**, select a protocol of your choice. For example, **Email**.
7. In **Endpoint**, specify your email address.
8. Click **Create Subscription**.
9. To confirm the subscription, Amazon SNS sends you an email. Click the link in the email to confirm your subscription.

9 Deleting a CloudFormation Stack

You can delete the CloudFormation stacks if required. Beware that deleting a CloudFormation stack deletes all configuration that you created as part of the deployment.

You can do it in the AWS CloudFormation console by selecting the main stack > **Actions** > **Delete Stack**. This action deletes all nested CloudFormation stack along with the main stack. However, some nested stacks may fail to be deleted and that prevents deleting the main stack. In such cases, you need to delete the failed nested CloudFormation stack first and then the main stack.

In some scenarios, the nested stacks fail to delete. In such scenarios, you must delete the resources created by these nested stacks. Perform the following steps to delete the resources.

1. Log in to AWS Console.
2. Navigate to **Service > System manager > Parameter store**.
3. Delete the parameters that nested stacks created.
4. Log in to Administration Console and delete the nodes that nested stacks created.

For information about how to delete a CloudFormation stack, see Deleting a Stack on the AWS CloudFormation Console.

10 Limitations of Sample Access Manager Auto Scaling on AWS

- You can use the auto scaling configuration for scheduled scaling or target tracking policy for CPU utilization of the auto scaling group.
- During scale in, the auto scaling process terminates an Identity Server instance immediately if this instance does not have any active session. If Identity Server has an active session, it waits for a maximum of 60 minutes for session timeout before performing a periodical check for active sessions. After this duration, auto scaling forcibly terminates the instances.

  However, for an Access Gateway instance, auto scaling does not wait for even session timeout and terminates the instances immediately during scale in. In this scenario, Identity Server keeps a copy of the session to prevent the user to re-authenticate.
- If you delete an Identity Server stack and do not remove all the nodes that the stack created, then the new stacks to be created might not work properly. It causes a number of non-responsive IP addresses to appear in the Identity Server cluster. These IP addresses do not belong to old or new stack and prevent the new nodes to have a green health status.

  To avoid this issue, you must delete all old nodes before you create a new stack.
- Multiple proxy services listening on different IP addresses is not supported in Auto Scaling.