

AppSentinels API Security Platform

Lambda Layer Deployment

Revision	Date Modified	Author	Comments
1.0	14-Aug-24	Arun	Initial spec for lambda layer deployment
1.1	05-Nov-24	Arun	Removed redundancy IAM permissions

Contents

AppSentinels Layers for AWS Lambda Functions	4
Deploying Layers	4
Deploying Layers Manually	4
Update Layers	5
Automated Deployment of Layers.....	5
Manage Layers	5
Roles and permission requirements	6
Layers configuration	6

AppSentinels Layers for AWS Lambda Functions

This document covers the steps to deploy the AppSentinels wrapper with AWS Lambda function. AppSentinels wrapper captures the HTTP request and response headers and payload and forwards that to AppSentinels OnPrem Controller. AppSentinels wrapper consists of two components - a tech stack dependent component, called *appsentinels-wrapper*, and an external lambda extension, called *appsentinels-log-server*. Following points summarize the operations done by the Appsentinels wrapper:

- The *appsentinels-wrapper* wraps the lambda function calls to capture HTTP request and response. It depends on the tech stack used by the lambda function - python wrapper is used if lambda function is implemented using python and a Nodejs wrapper is required if the lambda function is implemented using Nodejs.
- AWS provides the wrapper mechanism with the environment variable *AWS_LAMBDA_EXEC_WRAPPER*, which can be used to define the wrapper function that is invoked instead of the actual lambda handler.
- The *appsentinels-wrapper* bundles the HTTP request and response headers and payload in the form of a message and sends that message over UDP or TCP to *appsentinels-log-server*, running as lambda extension running in the same environment.
- The log server pushes the message received in a queue.
- A concurrent task (Go routine) picks the message from queue, converts that into a format expected by the controller and post that to controller.

Please note

- A layer version is always accessed using layer ARN.
- Contact support@appsentinels.ai for the latest ARN and the regions in which it is available.
- Current ARN:
 - `arn:aws:lambda:ap-south-1:488922454646:layer:appsentinels-wrapper:53`
 - `arn:aws:lambda:ap-south-1:488922454646:layer:appsentinels-log-server:59`

Deploying Layers

Deploying Layers Manually

Please execute following steps for both *appsentinels-wrapper* and *appsentinels-log-server*. After running these steps for both layers, lambda function should display both layers.

- Go to *Lambda > Functions > YourLambdaFunction*.
- Scroll down and click on **Add a layer**.
- Select **Specify an ARN**.
- Specify the ARN of the published AppSentinels layer (*appsentinels-wrapper* and *appsentinels-log-server*).
- Click on **Add**.

Update Layers

If a newer version of AppSentinels wrapper (*appsentinels-wrapper* or *appsentinels-log-server*) is available, following steps can be used to upgrade to the newer version of that component.

- Go to *Lambda > Functions > YourLambdaFunction*.
- Click on **Edit Layers**.
- Check the **Layer version** displayed along with the AppSentinels layer (*appsentinels-wrapper* or *appsentinels-log-server*).
- Click on the version number and select the new version to deploy from the drop-down.
- Click on **Save**.

Automated Deployment of Layers

- Customer can deploy AppSentinels wrapper on single or all Lambda function using the script `deploy_wrapper.sh` https://sample-config.appsentinels.ai/appsentinels-deployment/aws-lambda/deploy_wrapper.sh
- This script requires following command line arguments:
 - AppSentinels wrapper layer `appsentinels-wrapper` ARN
 - AppSentinels log-server layer `appsentinels-log-server` ARN
 - Region (AppSentinels has to publish the layers in this region before customer can deploy it).
 - Lambda Function: all or name of the lambda function.
 - all: deploy layer on all the Lambda function in the given account in the given region.
 - lambda function name: deploy layer on specific lambda function with given name.
 - Controller URL (AppSentinels edge controller URL).
- Example: Deploy lambda wrapper on `testHttpServer` function.

```
./deploy_wrapper.sh "arn:aws:lambda:ap-south-1:488922454646:layer:appsentinels-wrapper:53" "arn:aws:lambda:ap-south-1:488922454646:layer:appsentinels-log-server:59" ap-south-1 testHttpServer Appsentinels-Controller-NLB-14d455ada1d88f50.elb.ap-south-1.amazonaws.com
```

Manage Layers

- The script `manage_extensions.sh` https://sample-config.appsentinels.ai/appsentinels-deployment/aws-lambda/manage_extensions.sh can be used to perform following operations:
 - List all the extension or layer across all Lambda functions.
 - List all functions where an extension or layer with given ARN is deployed.
 - Remove the extension or layer with given ARN from all the functions.
- Please note that `appsentinels-wrapper` and `appsentinels-log-server` are separate layers and need to be managed separately, so this command needs to be executed separately to manage the `appsentinels-wrapper` and `appsentinels-log-server` layers.

- This script can be used to disable all instances of AppSentinels wrapper by running the command twice, once for each layer.

Roles and permission requirements

- Lambda permissions
 - `lambda:PublishLayerVersion`
 - `lambda:AddLayerVersionPermission`
 - `lambda:GetLayerVersion`
 - `lambda:UpdateFunctionConfiguration`
 - `lambda:GetFunction`
 - `lambda>DeleteLayerVersion`
 - `lambda:GetFunctionConfiguration`
- Lambda function CloudWatch logs
 - `logs:CreateLogGroup`
 - `logs:CreateLogStream`
 - `logs:PutLogEvents`
 - `logs:DescribeLogStreams`
 - `logs:GetLogEvents`

Layers configuration

Following table lists the environment variables used to configure the AppSentinels extension (mandatory marked in green)

Environment Variable	Description
<code>AS_EXTENSION_LOG_LEVEL</code>	Log level - Debug/Info/Warn/Error/Fatal, default Info
<code>AS_SERVER_ADDR</code>	Address of TCP/UDP log server (default 127.0.0.1)
<code>AS_SERVER_PORT</code>	Log server port number (default 9015)
<code>AS_SERVER_PROTOCOL</code>	Transport protocol to post messages to log server (udp)
<code>AS_LOGGING_MODE</code>	Logging mode - default/metadata/nologging
<code>AS_CONTROLLER_ENDPOINT</code>	AppSentinels Controller URL or IP address
<code>AS_CONTROLLER_PORT</code>	AppSentinels Controller port number
<code>AS_MAX_BATCH_SIZE</code>	Maximum number of logs in a batch (default 16)

Environment Variable	Description
AS_MAX_BODY_SIZE	Maximum request/response body size sent in the logs (60K)
AS_RELAY_PROTOCOL	Protocol use to relay the logs (http/https)
AS_SKIP_METHODS	HTTP methods skipped (default HEAD and OPTIONS)
AS_SKIP_URIS	Comma separated list of URIs to be skipped
AS_CONTENT_TYPE_REGEX	Regex defining content types process (json/xml/graphql/form)
AS_SERVER_CERT_VERIFY	Enable server certificate verification in HTTPS (default: no)

- Please note that AS_CONTROLLER_ENDPOINT is mandatory configuration parameter which needs to be specified during extension deployment.
- AS_LOGGING_MODE is set automatically based on the configuration.