

Trigger rescan of cloud connectors for new threats in Threat Center

First published - May 2024

Revision 3 – July 2024

Overview:

Customer requested script that will trigger a rescan of selected cloud connectors only if new threats were published to the threat center. This is expected to reduce the cost of unneeded scans, specifically for NonOS disk scanning, by allowing the NonOS disk scanning interval to be set to a longer period, while still ensuring that the environment is scanned for new threats soon after they are announced.

Functionality:

Create a script that performs the following on a scheduled basis:

1. Retrieve parameters from storage – connection credentials, last scan date, list of connectors to scan, etc.
2. Get the date of the latest non-informational threat published to the Threat Center
3. Get the date of the most recent triggered scan
4. If the latest threat is newer
 - a. Update the most recent triggered scan date
 - b. Request a scan of the connectors in the supplied list

Implementation:

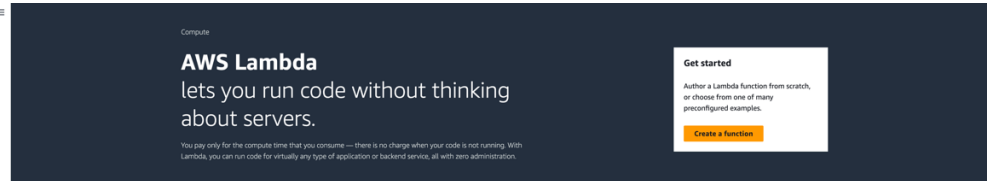
The script is implemented as an AWS Lambda function containing a single python file which is schedule to run on a desired repeating basis (e.g. daily). The Lambda function makes use of an SSM Parameter store to retrieve the following parameters. The function also updates the lastScanDate field based on the logic above:

Parameter	Purpose	Value
Client_id	Wiz Service Account ID	Populate from Wiz UI when creating Service Account
Client_secret	Wiz Service Account Secret	Populate from Wiz UI when creating Service Account
Connector_list	List of connectors to rescan	Comma-separated list, spaces allowed for readability
debugMode	Detailed debugging mode	True or False (default False)
lastScanDate	Date of last triggered scan	Must contain an initial value, will be rewritten with the current date if invalid.

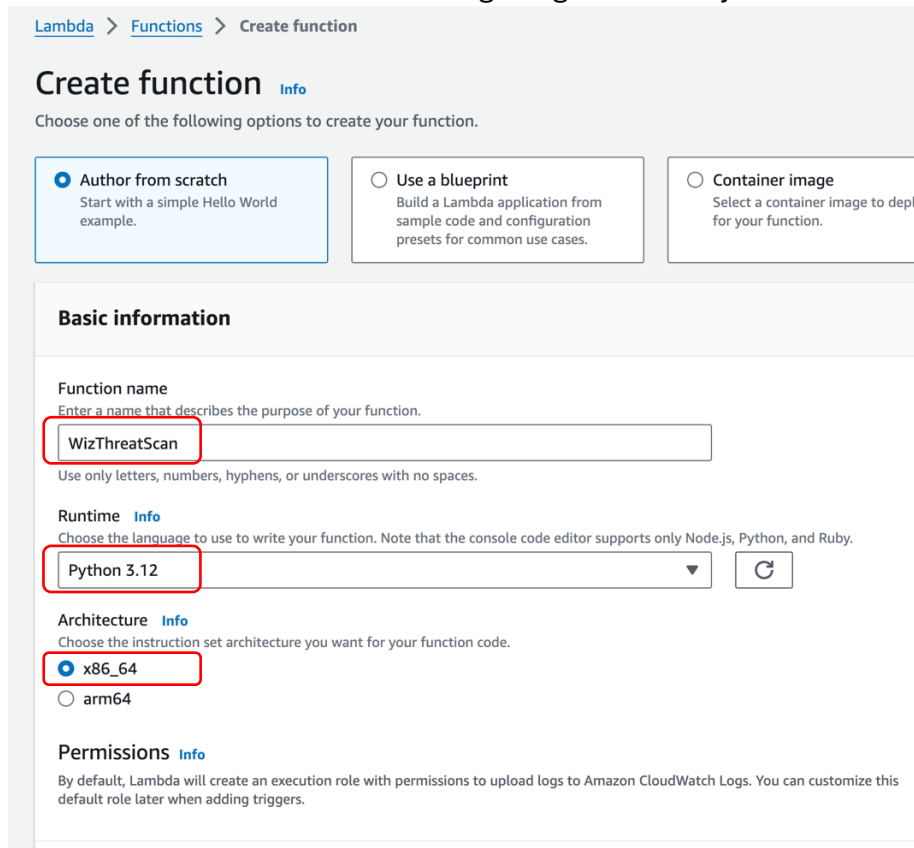
Installation and Setup:

AWS Lambda Setup:

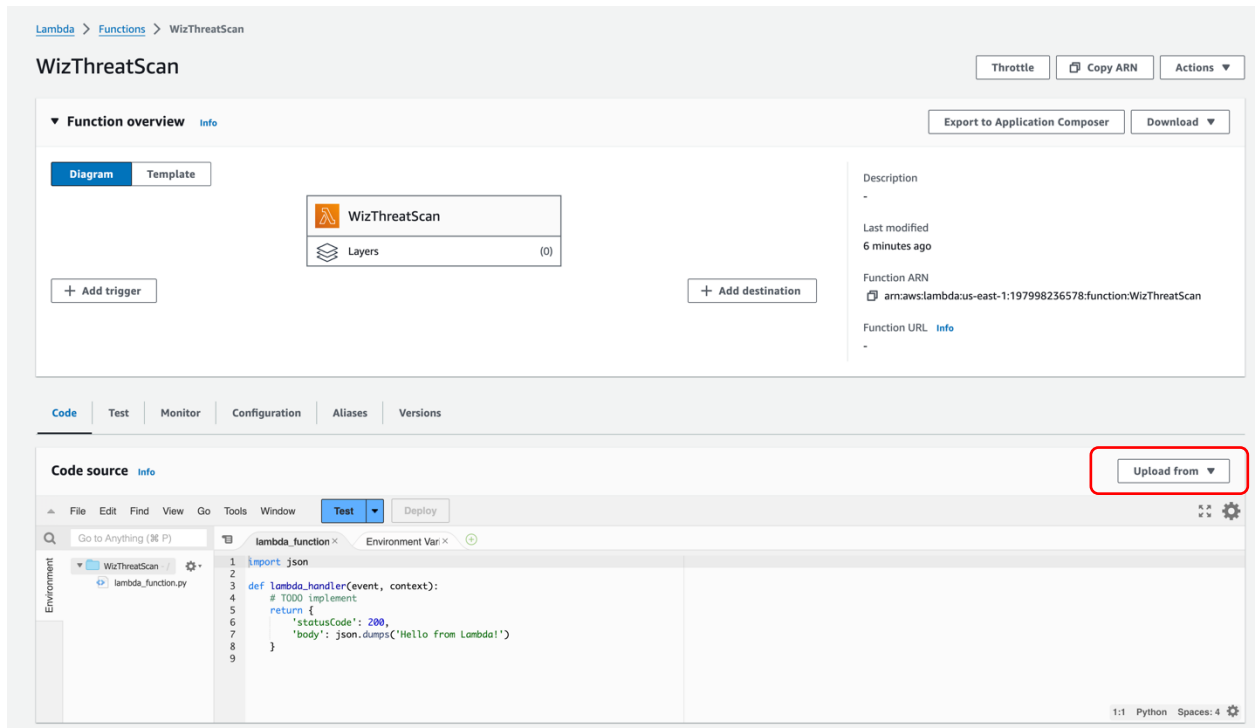
1. Download the WizThreatScan.zip and python_package.zip files (git clone <https://github.com/jsing3r/WizThreatScan.git>)
2. Login to AWS console with Admin rights (create roles, assign role, create SSM, create Lambda function, etc.)
3. Set your desired region (e.g. us-east-1)
4. Create a new Lambda function



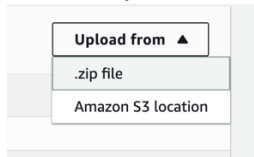
5. Use the information in the following image to create your Lambda function:

A screenshot of the AWS Lambda 'Create function' console. The breadcrumb trail is 'Lambda > Functions > Create function'. The main heading is 'Create function' with an 'Info' link. Below it, it says 'Choose one of the following options to create your function.' There are three radio button options: 'Author from scratch' (selected), 'Use a blueprint', and 'Container image'. The 'Basic information' section contains three fields: 'Function name' with the value 'WizThreatScan', 'Runtime' with the value 'Python 3.12', and 'Architecture' with the value 'x86_64'. The 'Permissions' section is partially visible at the bottom.

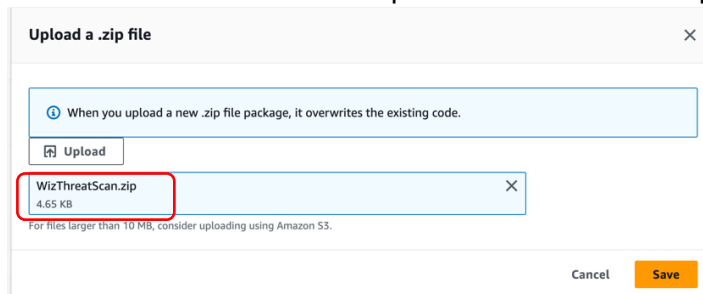
6. You should have a default Lambda function created as shown below:



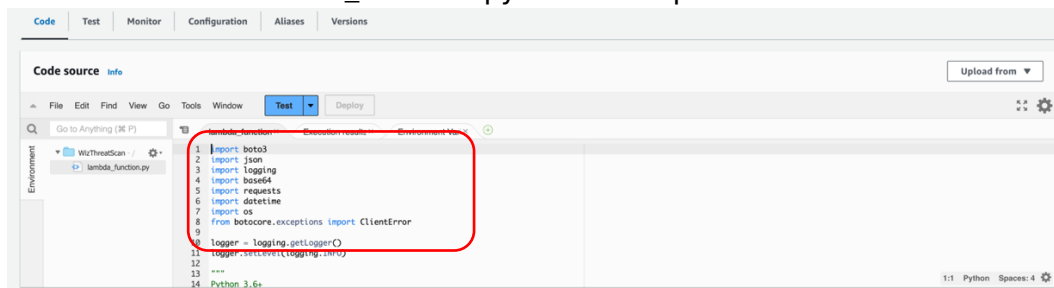
7. Select Upload from option and select .zip file



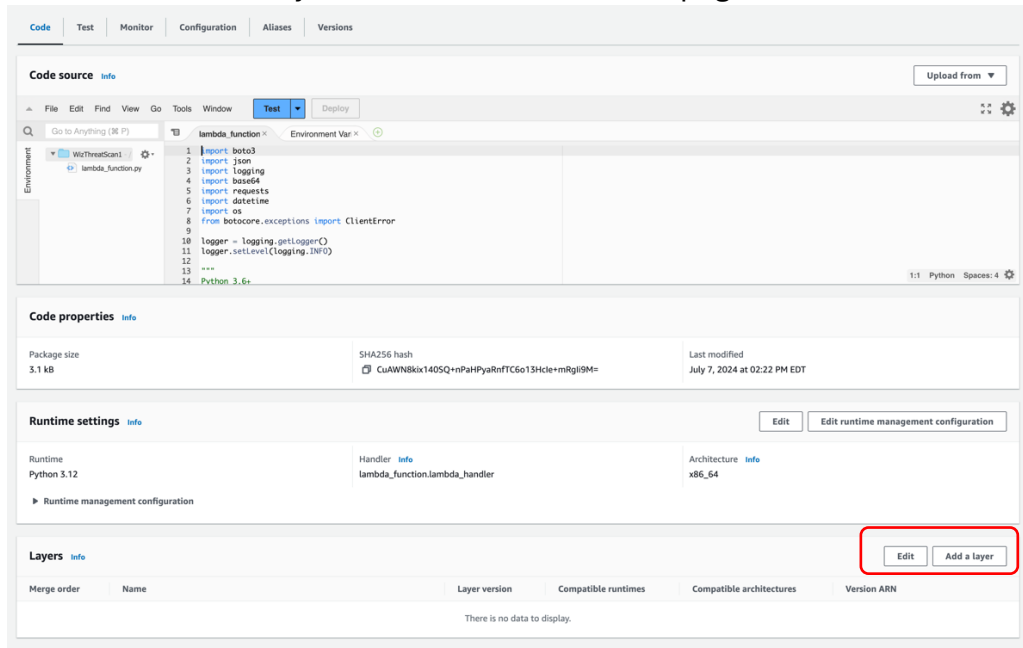
8. Select the WizThreatScan.zip file saved earlier to update the Lambda code



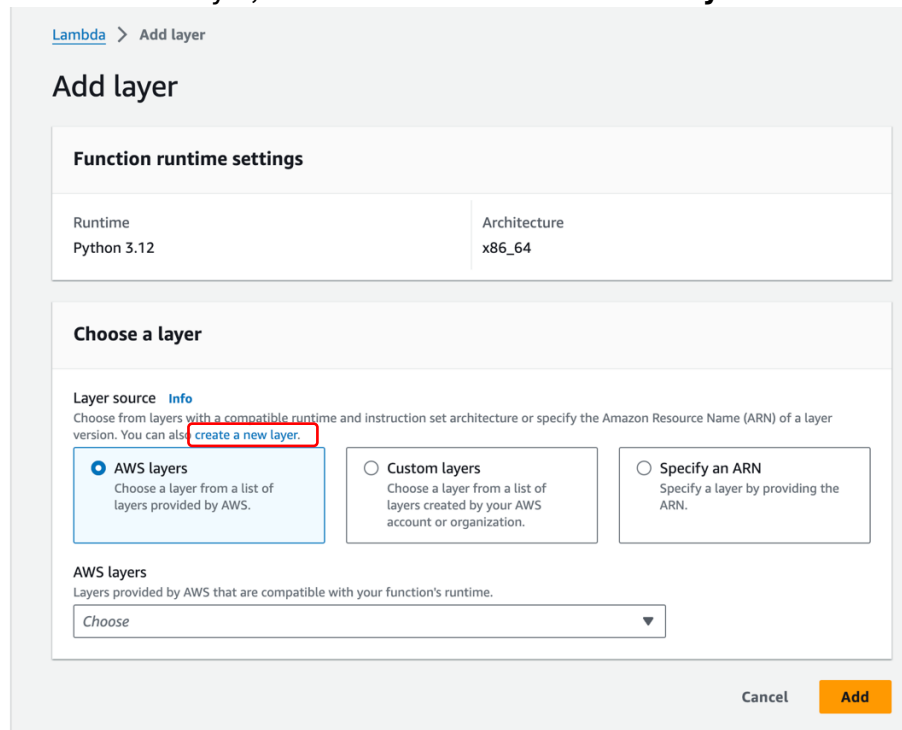
9. Confirm that the lambda_function.py has been updated as shown below



10. Scroll down to the Layers section on the Lambda page



11. Select Add a layer, and then select **create a new layer**



12. Create a new layer called WizThreatScanLayer and upload the python_packages.zip file to the layer. Also select the compatible architecture and runtimes as shown

Layer configuration

Name
WizThreatScanLayer

Description - optional
Description

☒ Upload a .zip file
☐ Upload a file from Amazon S3

Upload

python_packages.zip
17.58 MB

For files larger than 10 MB, consider uploading using Amazon S3.

Compatible architectures - optional Info
Choose the compatible instruction set architectures for your layer.

☒ x86_64
☐ arm64

Compatible runtimes - optional Info
Choose up to 15 runtimes.

Runtimes
Python 3.12

License - optional Info

Cancel Create

13. Return to the Add a layer dialog, select Custom layers, then select the newly created **WizThreatScanLayer** and Version 1 in the pulldown list:

Lambda > Add layer

Add layer

Function runtime settings

Runtime	Architecture
Python 3.12	x86_64

Choose a layer

Layer source Info
Choose from layers with a compatible runtime and instruction set architecture or specify the Amazon Resource Name (ARN) of a layer version. You can also [create a new layer](#).

☐ AWS layers
Choose a layer from a list of layers provided by AWS.

☒ Custom layers
Choose a layer from a list of layers created by your AWS account or organization.

☐ Specify an ARN
Specify a layer by providing the ARN.

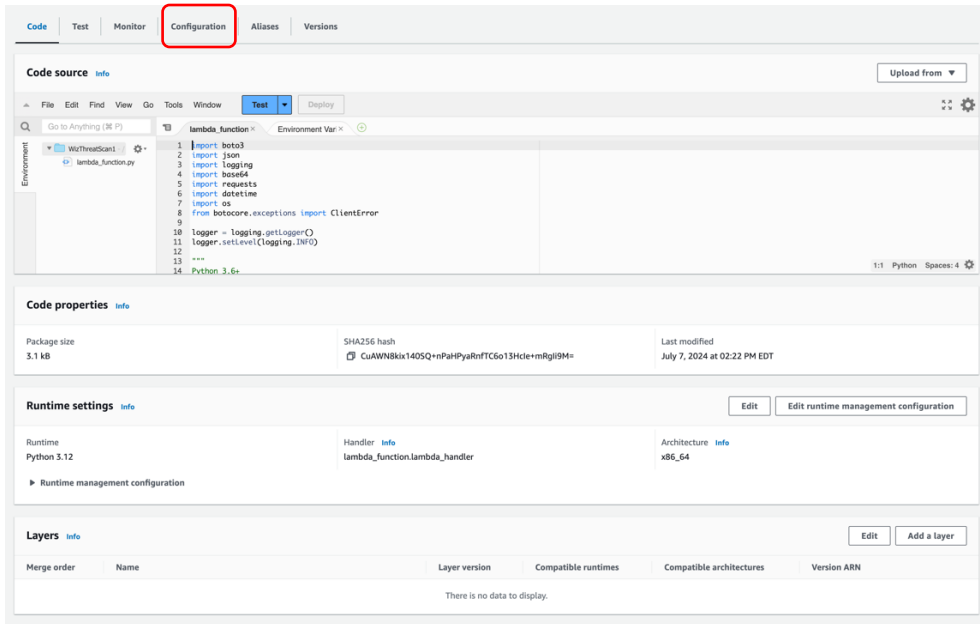
Custom layers
Layers created by your AWS account or organization that are compatible with your function's runtime.

WizThreatScanLayer

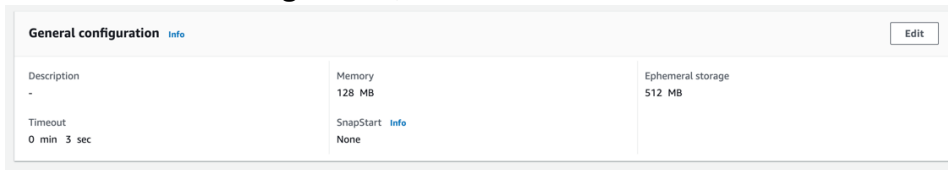
Version
1

Cancel Add

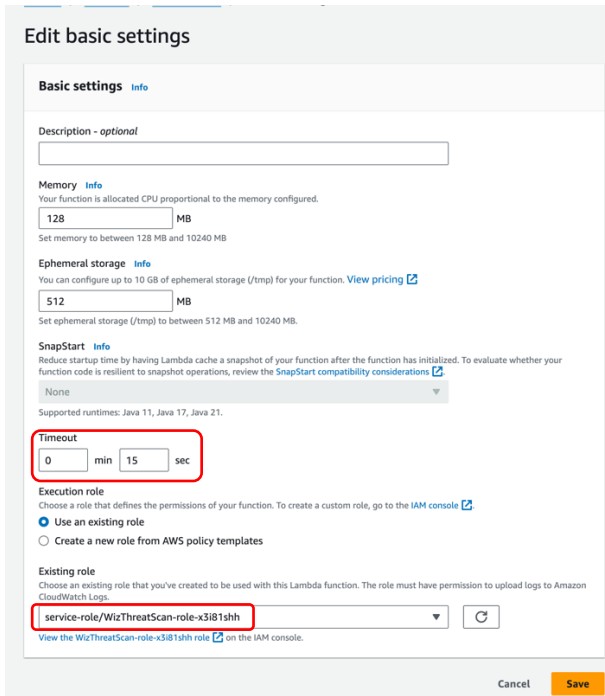
14. Navigate to the **Configuration** tab



15. Under General configuration, select Edit



16. Set the Timeout value to 15 sec instead of the default of 3 sec. Also make note of role that will be used to execute the Lambda function



17. Navigate to IAM in the AWS console, and select Roles. Locate the **WizThreatScan** role identified above:

The screenshot shows the AWS IAM console 'Roles' page. A table lists various roles, with 'WizThreatScan-role-x3i81shh' highlighted by a red rectangle. Below the table, the 'Roles Anywhere' section is visible.

Role name	Trusted entities	Last activity
Amazon_EventBridge_Scheduler_LAMBDA_Role	AWS Service: scheduler	2 hours ago
AWSServiceRoleForAmazonEK	AWS Service: eks (Service-Linked Rol	375 days ago
AWSServiceRoleForAutoScaling	AWS Service: autoscaling (Service-Li	375 days ago
AWSServiceRoleForCloudFormationStackSetOrgMember	AWS Service: member.org.stacksets-	423 days ago
AWSServiceRoleForOrganizations	AWS Service: organizations (Service-	423 days ago
AWSServiceRoleForSupport	AWS Service: support (Service-Linker	365 days ago
AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service	363 days ago
NewThreatScan-role-ib40vcc	AWS Service: lambda	-
OrganizationAccountAccessRole	Account: 555857844237	-
stacksets-exec-c5723ba6ca987d56f8a479923a7d99	Account: 555857844237	423 days ago
Wiz-Remediation-Stack-Role	AWS Service: lambda	19 minutes ago
WizAccess-Role	Account: 197171649850	1 hour ago
WizAccess-SNS-Integration	Account: 197171649850	289 days ago
WizThreatScan-role-1k5cnp5	AWS Service: lambda	-
WizThreatScan-role-gxkxmdv	AWS Service: lambda	27 minutes ago
WizThreatScan-role-x3i81shh	AWS Service: lambda	-

18. Select the **WizThreatScan** role and confirm the permission policies are like that shown below

The screenshot shows the AWS IAM console 'WizThreatScan-role-x3i81shh' page. The 'Permissions policies' section is expanded, showing a list of policies. The policy 'AWSLambdaBasicExecutionRole-7e053c37-7d75-4b36-8b2a-6fcd41178518' is highlighted with a red rectangle.

Summary

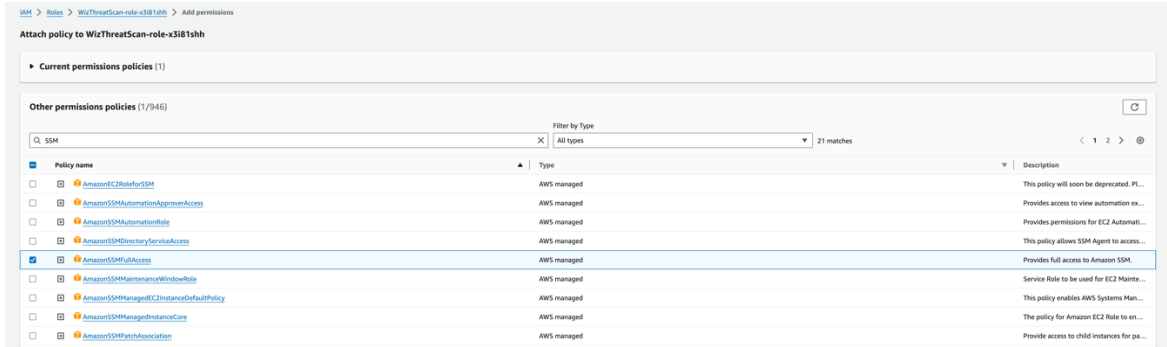
Creation date: July 07, 2024, 13:13 (UTC-04:00)
Last activity: -
ARN: arn:aws:iam:197998236578:role/service-role/WizThreatScan-role-x3i81shh
Maximum session duration: 1 hour

Permissions policies (1)

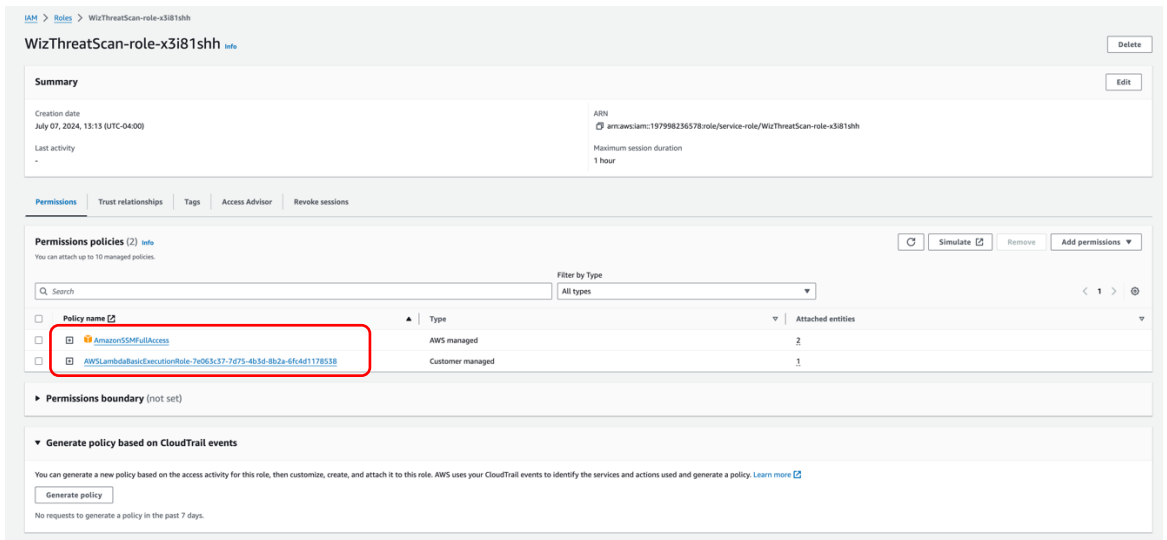
Filter by Type: All types

Policy name	Type	Attached entities
AWSLambdaBasicExecutionRole-7e053c37-7d75-4b36-8b2a-6fcd41178518	Customer managed	1

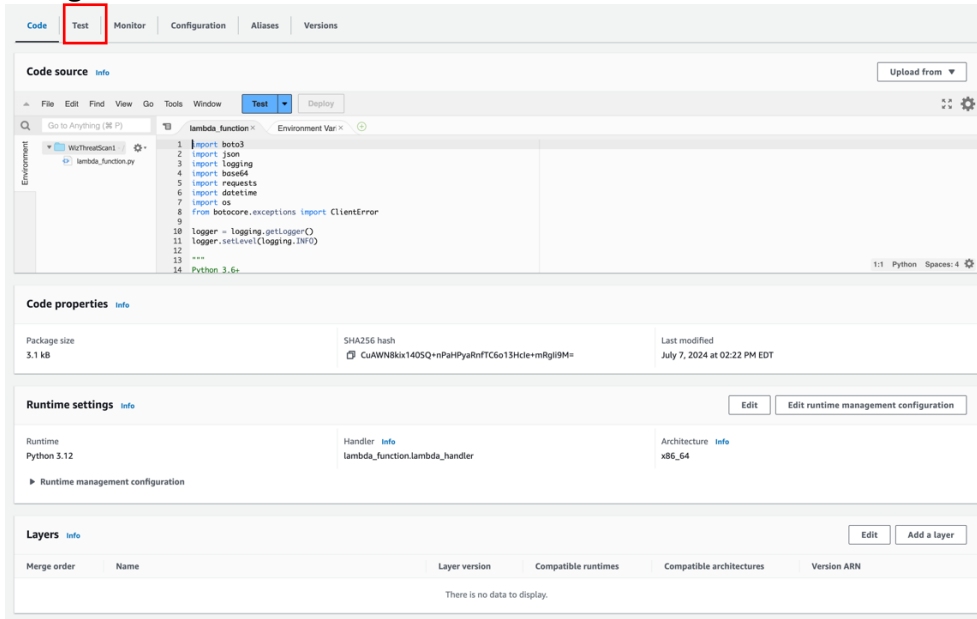
19. Select Add permissions and Attach policy, then select **AmazonSSMFullAccess** policy



20. Confirm that the WizThreatScan role now has the **AmazonSSMFullAccess** policy attached.



21. Navigate to the Test tab



22. Create a new test event called **myTestEvent** as shown. This test data will not be used but it allows the Test button to be used to test our function.

Configure test event

A test event is a JSON object that mocks the structure of requests emitted by AWS services to invoke a Lambda function. Use it to see the function's invocation result.

To invoke your function without saving an event, configure the JSON event, then choose Test.

Test event action

Create new event

Edit saved event

Event name

myTestEvent

Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.

Event sharing settings

Private

This event is only available in the Lambda console and to the event creator. You can configure a total of 10. [Learn more](#)

Shareable

This event is available to IAM users within the same account who have permissions to access and use shareable events. [Learn more](#)

Template - optional

hello-world

Event JSON

Format JSON

```
1 {  
2   "key1": "value1",  
3   "key2": "value2",  
4   "key3": "value3"  
5 }
```

Cancel

Invoke

Save

Wiz UI Setup:

1. Navigate to Wiz – Settings / Access Management / Service Accounts

2. Create a new Service Account of type Custom Integration (GraphQL API)

New Service Account

Name

Lambda Service Account

What's this service account for?

Type

Select which type of software component will use this Service Account

</> Custom Integration (GraphQL API) ▾

Description optional

Projects optional

Limit access to selected projects only

Select Projects... ▾

Select up to 50 projects. Leave empty to grant access to all projects

Expiration Date optional

Set an optional expiration date for this service account. After this date, the service account will no longer be able to access the Wiz API.

No date selected ▾

API Scopes

Scopes define the read and write permissions for this account.

3. Select all API Scopes

API Scopes

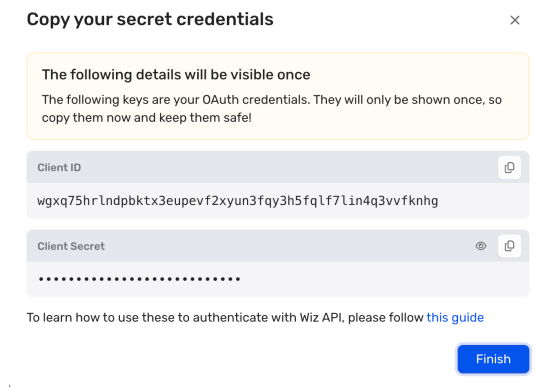
Scopes define the read and write permissions for this account.

- ☒ All
- | | |
|---|--------------|
| <input checked="" type="checkbox"/> Administer all admin entities
admin:all | Dec 31, 2022 |
| <input checked="" type="checkbox"/> Read all entities
read:all | Dec 31, 2022 |
| <input checked="" type="checkbox"/> Update all entities
update:all | Dec 31, 2022 |
| <input checked="" type="checkbox"/> Create all entities
create:all | Dec 31, 2022 |
| <input checked="" type="checkbox"/> Delete all entities
delete:all | Dec 31, 2022 |
| <input checked="" type="checkbox"/> Create, update and delete all entities
write:all | Dec 31, 2022 |

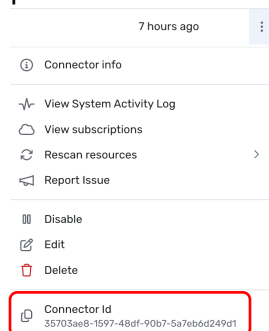
4. Select all Security Scan Scopes

- ☒ Security Scans
- | | |
|--|--------------|
| <input checked="" type="checkbox"/> List security scans
read:security_scans | Dec 31, 2022 |
| <input checked="" type="checkbox"/> Update a security scan
update:security_scans | Dec 31, 2022 |
| <input checked="" type="checkbox"/> Upload new security scan
create:security_scans | Dec 31, 2022 |
| <input checked="" type="checkbox"/> Delete a security scan
delete:security_scans | Dec 31, 2022 |
| <input checked="" type="checkbox"/> Create, update and delete security scans
write:security_scans | Dec 31, 2022 |

- Copy your secret credentials to a secure location as they will not be displayed again. These will be entered in the SSM parameter list

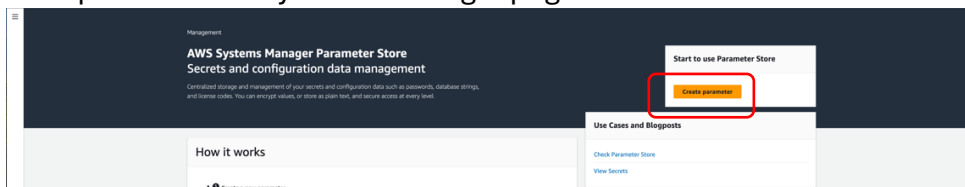


- Open the Wiz Settings / Deployments page and identify the connectors that should be rescanned for new threats. Click the menu at the end of the connector line to find the connector ID. These IDs will be entered as comma-separated values in the SSM parameter list.



AWS SSM Setup:

- Open the AWS Systems Manager page and select Parameter Store

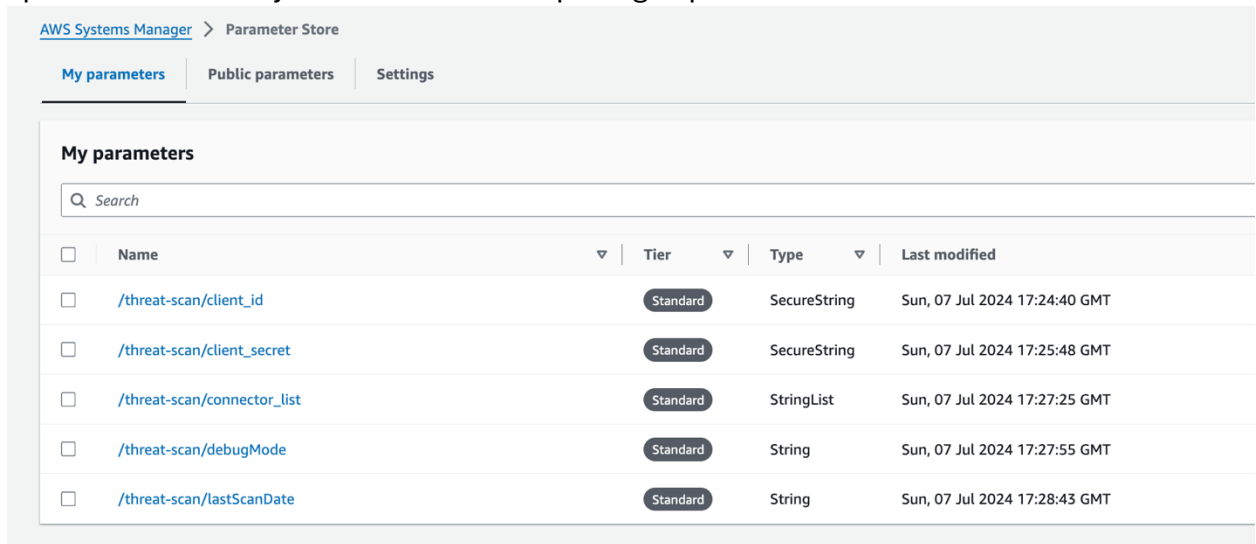


- Create the following parameters :

Parameter	Type	Value
/threat-scan/client_id	SecureString	Client_id from Wiz Service Account created above
/threat-scan/client_secret	SecureString	Client_secret from Wiz Service Account created above
/threat-scan/connector_list	StringList	Comma-separated list of connector IDs copied above, spaces allowed for readability

/threat-scan/debugMode	String	True or False (Set to True for detailed debugging information)
/threat-scan/lastScanDate	String	Must contain some initial value which will be rewritten with the current date if invalid. A good starting example is NEVER.

- Ensure that the following parameters are created and have the correct values as specified above. Pay attention to exact spelling of parameters:

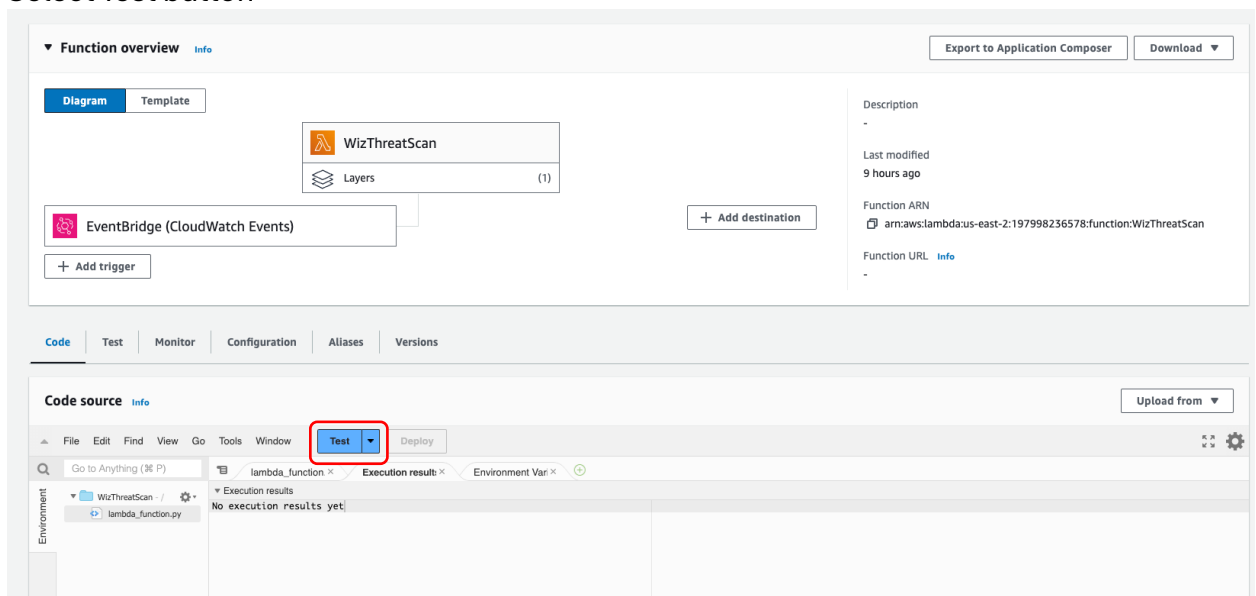


The screenshot shows the AWS Systems Manager Parameter Store interface. Under 'My parameters', there is a table listing five parameters:

Name	Tier	Type	Last modified
/threat-scan/client_id	Standard	SecureString	Sun, 07 Jul 2024 17:24:40 GMT
/threat-scan/client_secret	Standard	SecureString	Sun, 07 Jul 2024 17:25:48 GMT
/threat-scan/connector_list	Standard	StringList	Sun, 07 Jul 2024 17:27:25 GMT
/threat-scan/debugMode	Standard	String	Sun, 07 Jul 2024 17:27:55 GMT
/threat-scan/lastScanDate	Standard	String	Sun, 07 Jul 2024 17:28:43 GMT

Testing the script:

- Configure all setup elements in AWS and Wiz as described above
- Select Test button



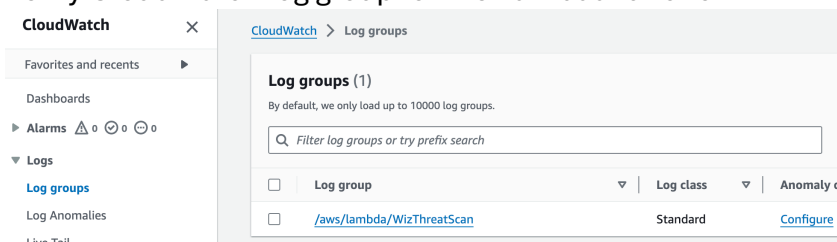
The screenshot shows the AWS Lambda console for a function named 'WizThreatScan'. The 'Function overview' tab is active, showing the function's configuration, including the trigger 'EventBridge (CloudWatch Events)' and the 'Layers' section. The 'Code source' tab is also visible, showing the 'Test' button highlighted with a red box. The 'Test' button is located in the 'Code source' tab, next to the 'Deploy' button. The 'Test' button is a blue button with a white 'Test' label. The 'Deploy' button is a grey button with a white 'Deploy' label. The 'Test' button is highlighted with a red box.

- Observe output in execution log window. For first invocation, lastScanDate (set to NEVER in this example) is overwritten with the current date. For every following run, the lastScanDate will be compared with the date of the most recent threat center notice, and a rescan will be triggered as required.

```
Execution results
WizThreatScan
lambda_function.py
Execution results
{
  "title": "regreSSHion: RCE Vulnerability in OpenSSH",
  "type": null,
  "workInProgress": null
},
{
  "pageInfo": {
    "endCursor": "eyJ0dWJsaXNoZWRRbDcIGlJlMjQ4MDctMDFUMDA6MDA6MDBoIiwuSUQlOiJyZWdyZWNTSGlvdjI3",
    "hasNextPage": true
  }
},
{
  "totalCount": 292
}
}
publishedAt 2024-07-01T00:00:00
id regreSSHion
type None
publishedDate 2024-07-01 00:00:00
key /threat-scan/lastScanDate value NEVER
content NEVER
Unable to parse the content: time data 'NEVER' does not match format '%Y-%m-%d %H:%M:%S'
current date 2024-07-08T02:35:22Z
Set lastScanDate 2024-07-08 02:35:22
lastScanDate later than publishedDate. No need to scan
[INFO] 2024-07-08T02:35:22.960Z ab2df328-a5b2-403e-9f21-3083c5007c60 CloudWatch logs group: /aws/lambda/WizThreatScan
END RequestId: ab2df328-a5b2-403e-9f21-3083c5007c60
REPORT RequestId: ab2df328-a5b2-403e-9f21-3083c5007c60 Duration: 7065.36 ms Billed Duration: 7066 ms Memory Size: 128 MB Max Memory Used: 80 MB Init Duration: 825.30 ms

Request ID
ab2df328-a5b2-403e-9f21-3083c5007c60
```

- Verify CloudWatch log group for the Lambda function



- Examine CloudWatch logs for the execution of the Lambda function

Log stream	Last event time
2024/07/08/[\$LATEST]3803098484a849199d9f9b4b96514ad9d	2024-07-08 02:35:16 (UTC)
2024/07/08/[\$LATEST]j876a5fddc964c7e9044037fbae59283	2024-07-08 02:29:11 (UTC)
2024/07/07/[\$LATEST]e0c3adf188f1422c8903885e7327f319	2024-07-07 19:04:57 (UTC)
2024/07/07/[\$LATEST]3677ae1ad0104269a11e1f65d610d1bd	2024-07-07 19:04:43 (UTC)
2024/07/07/[\$LATEST]0b2a078eda8433cbe993158d0f4077	2024-07-07 17:52:28 (UTC)
2024/07/07/[\$LATEST]5e92fcd3eef4b89a95ae7de84163f9	2024-07-07 17:41:57 (UTC)

Log events

Actions

Start tailing

Create metric filter

You can use the filter bar below to search for and match terms, phrases, or values in your log events. [Learn more about filter patterns](#)

Q

Filter events - press enter to search

Clear

1m

3m

1h

12h

Custom

UTC timezone

Display

Timestamp	Message
	No older events at this moment. View
2024-07-08T02:12:37.733Z	ENIT_START Runtime Version: python:3.12.x28 Runtime Version: ARN: arn:aws:lambda:us-east-1:runtime:7776de64298f7460/833b784949416d248d40408c7c4756f30c5ef28
2024-07-08T02:12:38.555Z	START RequestId: 0a0b0a57-4768-4767-89a0-89f4d550a0ba Version: \$LATEST
2024-07-08T02:12:38.769Z	[INFO] 2024-07-08T02:12:38.769Z 0a0b0a57-4768-4767-89a0-89f4d550a0ba Found credentials in environment variables.
2024-07-08T02:12:43.009Z	debugMode True
2024-07-08T02:12:43.009Z	Getting token.
2024-07-08T02:12:43.809Z	checkTrustCenter initiated
2024-07-08T02:12:44.549Z	{ "data": { "threatCenterItems": { "nodes": [{ "id": "regression", "publishedAt": "2024-07-08T00:00:00Z", "description": "A critical unauthenticated RCE-as-root vulnerability was identified in OpenSSH server (sshd), assigned CVE-2024-6387 and dubbed regression." }] } } }
2024-07-08T02:12:44.549Z	{ "data": { "threatCenterItems": { "nodes": [{ "id": "regression", "publishedAt": "2024-07-08T00:00:00Z", "description": "A critical unauthenticated RCE-as-root vulnerability was identified in OpenSSH server (sshd), assigned CVE-2024-6387 and dubbed regression." }] } } }
2024-07-08T02:12:44.549Z	{ "data": { }
2024-07-08T02:12:44.549Z	{ "threatCenterItems": { }
2024-07-08T02:12:44.549Z	{ "nodes": [}
2024-07-08T02:12:44.549Z	{ }
2024-07-08T02:12:44.549Z	{ "alwaysShowFindingsCount": null,
2024-07-08T02:12:44.549Z	"description": "A critical unauthenticated RCE-as-root vulnerability was identified in OpenSSH server (sshd), assigned CVE-2024-6387 and dubbed regression. The vulnerability is a signal handler race condition that is exploitable in the default configuration." }
2024-07-08T02:12:44.549Z	"docUrl": "https://docs.wiz.io/wiz-docs/docs/wiz-adv-2024-006",
2024-07-08T02:12:44.549Z	"findingsLinks": [}

Code

Test

Monitor

Configuration

Aliases

Versions

General configuration

Triggers

Permissions

Destinations

Function URL

Environment variables

Tags

VPC

RDS databases

Monitoring and operations tools

Concurrency

Asynchronous invocation

Triggers (1)

Find triggers

EventBridge (CloudWatch Events): [WizThreatScanSchedule](#)

arn:aws:events:us-east-1:197998236578:rule/WizThreatScanSchedule

Rule state: **ENABLED**

▼ Details

Event bus: **default**

isComplexStatement: **No**

name: **WizThreatScanSchedule**

Schedule expression: **rate(1 day)**

Service principal: **events.amazonaws.com**

Statement ID: **lambda-9f532550-ecb-4cc9-9462-dde5fa7fb2fc**

uri: **events/home#/rules/WizThreatScanSchedule**