SAM: Open Port Check

Goals:

- Create a serverless application which will check if a TCP port is open on a remote target.
    - NOTE: Use python 3.9... python 3.6 is dead now for lambda
    - NOTE: Use Region of Ohio / us-east-2 in Student Accounts

Dependencies:

- Access to the Student Environment in AWS
- Cloud9 IDE was created previously, see previous lab entitled: "Cloud9 & SAM 101"
- Understanding the content within the lab: "HTTP GET Parameters"
- Understanding the content within the lab: "Local Debug & Testing"

Code & Files:

- https://github.com/TweekFawkes/train_intro_to_serverless

Login to the Student AWS Account

- AWS Login: https://console.aws.amazon.com/ (Links to an external site.)
- IAM Username: Hal
- IAM Password: <password>

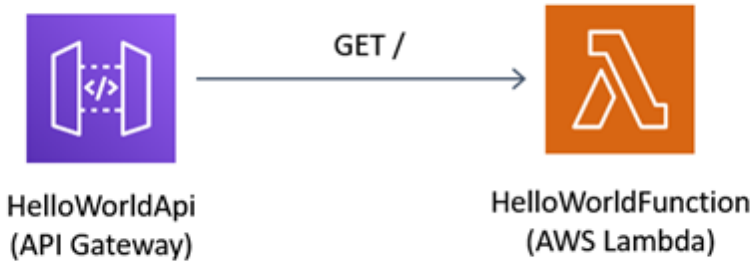
Login to the Cloud9 IDE Environment

Region: Ohio / us-east-2
Service: Cloud9

Locate the "HelloWorld101" Cloud9 environment
Click the "Open IDE" button

Download the Sample SAM App

We will build a simple SAM app with the following components:



In the terminal, run the following command(s) to create a new sam application:

```
cd ~/environment/

sam init

1

1

N

13

1

N

portcheck-app-001
```

We should see output similar to the following:

```
Hal:~/environment $ sam init

You can preselect a particular runtime or package type when using the `sam
init` experience.
```

```
Call `sam init --help` to learn more.

Which template source would you like to use?
        1 - AWS Quick Start Templates
        2 - Custom Template Location
Choice: 1

Choose an AWS Quick Start application template
        1 - Hello World Example
        2 - Multi-step workflow
        3 - Serverless API
        4 - Scheduled task
        5 - Standalone function
        6 - Data processing
        7 - Infrastructure event management
        8 - Lambda EFS example
        9 - Machine Learning
Template: 1

Use the most popular runtime and package type? (Python and zip) [y/N]: N

Which runtime would you like to use?
        1 - dotnet6
        2 - dotnet5.0
        3 - dotnetcore3.1
        4 - go1.x
        5 - graalvm.java11 (provided.al2)
        6 - graalvm.java17 (provided.al2)
        7 - java11
        8 - java8.al2
        9 - java8
        10 - nodejs16.x
        11 - nodejs14.x
        12 - nodejs12.x
        13 - python3.9
        14 - python3.8
        15 - python3.7
        16 - ruby2.7
        17 - rust (provided.al2)
Runtime: 13

What package type would you like to use?
```

```
        1 - Zip
        2 - Image
Package type: 1

Based on your selections, the only dependency manager available is pip.
We will proceed copying the template using pip.

Would you like to enable X-Ray tracing on the function(s) in your
application?  [y/N]: N

Project name [sam-app]: portcheck-app-001

Cloning from https://github.com/aws/aws-sam-cli-app-templates (process may
take a moment)

    -----------------------
    Generating application:
    -----------------------
    Name: portcheck-app-001
    Runtime: python3.9
    Architectures: x86_64
    Dependency Manager: pip
    Application Template: hello-world
    Output Directory: .

    Next steps can be found in the README file at
./portcheck-app-001/README.md


    Commands you can use next
    =========================
    [*] Create pipeline: cd portcheck-app-001 && sam pipeline init
--bootstrap
    [*] Validate SAM template: sam validate
    [*] Test Function in the Cloud: sam sync --stack-name {stack-name}
--watch

Hal:~/environment $
```

Passing Values via HTTP GET Params

Inspect the source code of the following files:

- template.yaml -> /home/ubuntu/environment/portcheck-app-001/template.yaml
    - SAM Template that defines your application's AWS resources

Change the "CodeUri" and "Path" to be the following values in the "template.yaml" file:

```
...

Globals:

Function:

Timeout: 900

...

Properties:

CodeUri: port_check/

Handler: app.lambda_handler

Runtime: python3.9

Timeout: 900

MemorySize: 512

...

Properties:

Path: /portcheck

Method: get
...
```

Click "File" -> "Save" or Ctrl+S on Windows, to save the "template.yaml" file

Next, move the "hello_world" directory to be called "port_check":

```
pwd

cd /home/ubuntu/environment/portcheck-app-001/

ls -alF

mv hello_world/ port_check/

ls -alF
```

We should see output similar to the following:

```
Hal:~/environment $ pwd
/home/ubuntu/environment

Hal:~/environment $ cd /home/ubuntu/environment/portcheck-app-001/

Hal:~/environment/portcheck-app-001 $ ls -alF
total 40
drwxrwxr-x 5 ubuntu ubuntu 4096 Oct 27 17:17 ./
drwxr-xr-x 7 ubuntu ubuntu 4096 Oct 27 17:17 ../
-rw-rw-r-- 1 ubuntu ubuntu 3730 Oct 27 17:17 .gitignore
-rw-rw-r-- 1 ubuntu ubuntu 8459 Oct 27 17:17 README.md
-rw-rw-r-- 1 ubuntu ubuntu    0 Oct 27 17:17 __init__.py
drwxrwxr-x 2 ubuntu ubuntu 4096 Oct 27 17:17 events/
drwxrwxr-x 2 ubuntu ubuntu 4096 Oct 27 17:17 hello_world/
-rw-rw-r-- 1 ubuntu ubuntu 1681 Oct 27 17:17 template.yaml
drwxrwxr-x 4 ubuntu ubuntu 4096 Oct 27 17:17 tests/

Hal:~/environment/portcheck-app-001 $ mv hello_world/ port_check/

Hal:~/environment/portcheck-app-001 $ ls -alF
total 40
drwxrwxr-x 5 ubuntu ubuntu 4096 Oct 27 17:19 ./
drwxr-xr-x 7 ubuntu ubuntu 4096 Oct 27 17:17 ../
-rw-rw-r-- 1 ubuntu ubuntu 3730 Oct 27 17:17 .gitignore
-rw-rw-r-- 1 ubuntu ubuntu 8459 Oct 27 17:17 README.md
-rw-rw-r-- 1 ubuntu ubuntu    0 Oct 27 17:17 __init__.py
drwxrwxr-x 2 ubuntu ubuntu 4096 Oct 27 17:17 events/
drwxrwxr-x 2 ubuntu ubuntu 4096 Oct 27 17:17 port_check/
-rw-rw-r-- 1 ubuntu ubuntu 1681 Oct 27 17:17 template.yaml
```

```
drwxrwxr-x 4 ubuntu ubuntu 4096 Oct 27 17:17 tests/

Hal:~/environment/portcheck-app-001 $
```

Inspect the source code of the following files:

- app.py -> /home/ubuntu/environment/portcheck-app-001/port_check/app.py
    - Contains the logic/code for your lambda application

When passing the lambda function information via the API gateway as HTTP GET Parameters, e.g.

```
red_team_040:~/environment/dirb-app-010 $ curl
https://EXAMPLE.execute-api.us-east-1.amazonaws.com/Prod/dirb/?AAAA=BBBB
```

The "event" object will contain data similar to the following...

```
{'resource': '/dirb', 'path': '/dirb/', 'httpMethod': 'GET', 'headers':
{'Accept': '*/*', 'CloudFront-Forwarded-Proto': 'https',
'CloudFront-Is-Desktop-Viewer': 'true', 'CloudFront-Is-Mobile-Viewer':
'false', 'CloudFront-Is-SmartTV-Viewer': 'false',
'CloudFront-Is-Tablet-Viewer': 'false', 'CloudFront-Viewer-Country': 'US',
'Host': '7ierqt1j17.execute-api.us-east-1.amazonaws.com', 'User-Agent':
'curl/7.58.0', 'Via': '2.0 237bd7e86f7f99cead16dc4ecb5fed20.cloudfront.net
(CloudFront)', 'X-Amz-Cf-Id':
'2_HaEWlB9X5fOnGYWnQJVj09JvA9ztuSZ7h9fGCLECepvTzOoZBRJw==',
'X-Amzn-Trace-Id': 'Root=1-614a56b1-0bf806fa6a43613863d243b4',
'X-Forwarded-For': '3.226.252.96, 70.132.60.74', 'X-Forwarded-Port': '443',
'X-Forwarded-Proto': 'https'}, 'multiValueHeaders': {'Accept': ['*/*'],
'CloudFront-Forwarded-Proto': ['https'], 'CloudFront-Is-Desktop-Viewer':
['true'], 'CloudFront-Is-Mobile-Viewer': ['false'],
'CloudFront-Is-SmartTV-Viewer': ['false'], 'CloudFront-Is-Tablet-Viewer':
['false'], 'CloudFront-Viewer-Country': ['US'], 'Host':
['7ierqt1j17.execute-api.us-east-1.amazonaws.com'], 'User-Agent':
['curl/7.58.0'], 'Via': ['2.0
237bd7e86f7f99cead16dc4ecb5fed20.cloudfront.net (CloudFront)'],
'X-Amz-Cf-Id':
['2_HaEWlB9X5fOnGYWnQJVj09JvA9ztuSZ7h9fGCLECepvTzOoZBRJw=='],
'X-Amzn-Trace-Id': ['Root=1-614a56b1-0bf806fa6a43613863d243b4'],
'X-Forwarded-For': ['3.226.252.96, 70.132.60.74'], 'X-Forwarded-Port':
['443'], 'X-Forwarded-Proto': ['https']}, 'queryStringParameters': {'AAAA':
```

```
'BBBB'}, 'multiValueQueryStringParameters': {'AAAA': ['BBBB']},
'pathParameters': None, 'stageVariables': None, 'requestContext':
{'resourceId': '80a50y', 'resourcePath': '/dirb', 'httpMethod': 'GET',
'extendedRequestId': 'GCJ7tETQIAMF4jg=', 'requestTime':
'21/Sep/2021:22:03:29 +0000', 'path': '/Prod/dirb/', 'accountId':
'580299357056', 'protocol': 'HTTP/1.1', 'stage': 'Prod', 'domainPrefix':
'7ierqt1j17', 'requestTimeEpoch': 1632261809163, 'requestId':
'f8e77801-303a-4aa3-b32c-2149491d6f66', 'identity':
{'cognitoIdentityPoolId': None, 'accountId': None, 'cognitoIdentityId':
None, 'caller': None, 'sourceIp': '3.226.252.96', 'principalOrgId': None,
'accessKey': None, 'cognitoAuthenticationType': None,
'cognitoAuthenticationProvider': None, 'userArn': None, 'userAgent':
'curl/7.58.0', 'user': None}, 'domainName':
'7ierqt1j17.execute-api.us-east-1.amazonaws.com', 'apiId': '7ierqt1j17'},
'body': None, 'isBase64Encoded': False}
```

We can see from this output that the GET parameter "AAAA" value of "BBBB" is contained within the following object:

```
event['queryStringParameters']
event['queryStringParameters']['AAAA']
```

Add the following imports to the top of the "app.py" file:

```
import json
import socket
```

Add the following logic to the application's lambda_handler() function to process the input via a GET query parameter called "RootDomainName":

```
sReturn = "NULL"

try:
sTargetIp = str(event['queryStringParameters']['TargetIp'])
sTcpPort = str(event['queryStringParameters']['TcpPort'])

print("[~] sTargetIp: " + sTargetIp)
print("[~] sTcpPort: " + sTcpPort)

sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

```
sock.settimeout(2) #2 Second Timeout

result = sock.connect_ex((sTargetIp,int(sTcpPort)))

if result == 0:
print("Port is open")
sReturn = sTargetIp + ":" + sTcpPort + "/TCP is open"
else:
print("Port is not open")
sReturn = sTargetIp + ":" + sTcpPort + "/TCP is closed"

sock.close()
except Exception as e:
print("[!] Exception (e):" + str(e))

return {
"statusCode": 200,
"body": sReturn,
}
```

Click "File" -> "Save" or Ctrl+S on Windows, to save the "app.py" file

Run the following commands to build and deploy the application...

```
cd /home/ubuntu/environment/portcheck-app-001

sam build

sam deploy --guided

portcheck-app-001

[ENTER]

y

Y


N
```

```
y

Y

[ENTER]

[ENTER]

y
```

We should see output similar to the following...

```
Hal:/ $ cd /home/ubuntu/environment/portcheck-app-001
Hal:~/environment/portcheck-app-001 $ sam build
Your template contains a resource with logical ID "ServerlessRestApi",
which is a reserved logical ID in AWS SAM. It could result in unexpected
behaviors and is not recommended.
Building codeuri: /home/ubuntu/environment/portcheck-app-001/port_check
runtime: python3.9 metadata: {} architecture: x86_64 functions:
HelloWorldFunction
Running PythonPipBuilder:ResolveDependencies
Running PythonPipBuilder:CopySource

Build Succeeded

Built Artifacts  : .aws-sam/build
Built Template   : .aws-sam/build/template.yaml

Commands you can use next
=========================
[*] Validate SAM template: sam validate
[*] Invoke Function: sam local invoke
[*] Test Function in the Cloud: sam sync --stack-name {stack-name} --watch
[*] Deploy: sam deploy --guided

Hal:~/environment/portcheck-app-001 $ sam deploy --guided

Configuring SAM deploy
======================

        Looking for config file [samconfig.toml] :  Found
        Reading default arguments  :  Success
```

```
        Setting default arguments for 'sam deploy'
        =========================================
        Stack Name [portcheck-app-001]: portcheck-app-001
        AWS Region [us-east-2]:
        #Shows you resources changes to be deployed and require a 'Y' to
initiate deploy
        Confirm changes before deploy [Y/n]: Y
        #SAM needs permission to be able to create roles to connect to the
resources in your template
        Allow SAM CLI IAM role creation [Y/n]: Y
        #Preserves the state of previously provisioned resources when an
operation fails
        Disable rollback [y/N]: N
        HelloWorldFunction may not have authorization defined, Is this
okay? [y/N]: y
        Save arguments to configuration file [Y/n]: y
        SAM configuration file [samconfig.toml]:
        SAM configuration environment [default]:

        Looking for resources needed for deployment:
         Managed S3 bucket:
aws-sam-cli-managed-default-samclisourcebucket-142o3zytl001y
         A different default S3 bucket can be set in samconfig.toml

        Saved arguments to config file
        Running 'sam deploy' for future deployments will use the parameters
saved above.
        The above parameters can be changed by modifying samconfig.toml
        Learn more about samconfig.toml syntax at

https://docs.aws.amazon.com/serverless-application-model/latest/developergu
ide/serverless-sam-cli-config.html

Uploading to portcheck-app-001/b9fb3518a3f1403521f80648cf3a39c8  466576 /
466576  (100.00%)

        Deploying with following values
        ===============================
        Stack name                   : portcheck-app-001
        Region                       : us-east-2
        Confirm changeset            : True
```

```
        Disable rollback             : False
        Deployment s3 bucket         :
aws-sam-cli-managed-default-samclisourcebucket-142o3zytl001y
        Capabilities                 : ["CAPABILITY_IAM"]
        Parameter overrides          : {}
        Signing Profiles             : {}

Initiating deployment
=====================
Uploading to portcheck-app-001/b48f480ad3b527f3c238904c26ab9809.template
1257 / 1257  (100.00%)

Waiting for changeset to be created..
CloudFormation stack changeset
---------------------------------------------------------------------------
---------------------------------------------------------------------------
-------------------------------------------------
Operation                                         LogicalResourceId
ResourceType                                      Replacement
---------------------------------------------------------------------------
---------------------------------------------------------------------------
-------------------------------------------------
+ Add
HelloWorldFunctionHelloWorldPermissionProd        AWS::Lambda::Permission
N/A
+ Add                                             HelloWorldFunctionRole
AWS::IAM::Role                                    N/A
+ Add                                             HelloWorldFunction
AWS::Lambda::Function                             N/A
+ Add
ServerlessRestApiDeployment9a29b48186
AWS::ApiGateway::Deployment                       N/A
+ Add
ServerlessRestApiProdStage                        AWS::ApiGateway::Stage
N/A
+ Add                                             ServerlessRestApi
AWS::ApiGateway::RestApi                          N/A
---------------------------------------------------------------------------
---------------------------------------------------------------------------
-------------------------------------------------

Changeset created successfully.
```

```
arn:aws:cloudformation:us-east-2:013109453517:changeSet/samcli-deploy166689
1430/2c072aa2-612f-44de-b0fc-1307fe990366


Previewing CloudFormation changeset before deployment
======================================================
Deploy this changeset? [y/N]: y

2022-10-27 17:23:58 - Waiting for stack create/update to complete

CloudFormation events from stack operations (refresh every 0.5 seconds)
---------------------------------------------------------------------------
---------------------------------------------------------------------------
-------------------------------------------------
ResourceStatus                                   ResourceType
LogicalResourceId                                ResourceStatusReason
---------------------------------------------------------------------------
---------------------------------------------------------------------------
-------------------------------------------------
CREATE_IN_PROGRESS                               AWS::IAM::Role
HelloWorldFunctionRole                           -
CREATE_IN_PROGRESS                               AWS::IAM::Role
HelloWorldFunctionRole                           Resource creation
Initiated
CREATE_COMPLETE                                  AWS::IAM::Role
HelloWorldFunctionRole                           -
CREATE_IN_PROGRESS                               AWS::Lambda::Function
HelloWorldFunction                               -
CREATE_IN_PROGRESS                               AWS::Lambda::Function
HelloWorldFunction                               Resource creation
Initiated
CREATE_COMPLETE                                  AWS::Lambda::Function
HelloWorldFunction                               -
CREATE_IN_PROGRESS                               AWS::ApiGateway::RestApi
ServerlessRestApi                                -
CREATE_IN_PROGRESS                               AWS::ApiGateway::RestApi
ServerlessRestApi                                Resource creation
Initiated
CREATE_COMPLETE                                  AWS::ApiGateway::RestApi
ServerlessRestApi                                -
CREATE_IN_PROGRESS
AWS::ApiGateway::Deployment
```

```
ServerlessRestApiDeployment9a29b48186              -
CREATE_IN_PROGRESS                                 AWS::Lambda::Permission
HelloWorldFunctionHelloWorldPermissionProd         -
CREATE_IN_PROGRESS                                 AWS::Lambda::Permission
HelloWorldFunctionHelloWorldPermissionProd         Resource creation
Initiated
CREATE_IN_PROGRESS
AWS::ApiGateway::Deployment
ServerlessRestApiDeployment9a29b48186              Resource creation
Initiated
CREATE_COMPLETE
AWS::ApiGateway::Deployment
ServerlessRestApiDeployment9a29b48186              -
CREATE_IN_PROGRESS                                 AWS::ApiGateway::Stage
ServerlessRestApiProdStage                         -
CREATE_IN_PROGRESS                                 AWS::ApiGateway::Stage
ServerlessRestApiProdStage                         Resource creation
Initiated
CREATE_COMPLETE                                    AWS::ApiGateway::Stage
ServerlessRestApiProdStage                         -
CREATE_COMPLETE                                    AWS::Lambda::Permission
HelloWorldFunctionHelloWorldPermissionProd         -
CREATE_COMPLETE
AWS::CloudFormation::Stack                         portcheck-app-001
-
------------------------------------------------------------------------------
------------------------------------------------------------------------------
------------------------------------------------------
CloudFormation outputs from deployed stack
------------------------------------------------------------------------------
------------------------------------------------------------------------------
------------------------------------------------------
Outputs
------------------------------------------------------------------------------
------------------------------------------------------------------------------
------------------------------------------------------
Key                 HelloWorldFunctionIamRole
Description         Implicit IAM Role created for Hello World function
Value
arn:aws:iam::013109453517:role/portcheck-app-001-HelloWorldFunctionRole-6F4
5YQXQ17YD
```

```
Key                 HelloWorldApi
Description         API Gateway endpoint URL for Prod stage for Hello World
function
Value
https://zehecvjteh.execute-api.us-east-2.amazonaws.com/Prod/hello/

Key                 HelloWorldFunction
Description         Hello World Lambda Function ARN
Value
arn:aws:lambda:us-east-2:013109453517:function:portcheck-app-001-HelloWorld
Function-LFh7VlQ4iImI
-------------------------------------------------------------------------
-------------------------------------------------------------------------
-----------------------------------------------------

Successfully created/updated stack - portcheck-app-001 in us-east-2

Hal:~/environment/portcheck-app-001 $
```

Test the deployment via the following command (replacing the URL with the URL from your deployment):

```
curl
"https://zehecvjteh.execute-api.us-east-2.amazonaws.com/Prod/portcheck/?Tar
getIp=34.209.82.230&TcpPort=22"

curl
"https://zehecvjteh.execute-api.us-east-2.amazonaws.com/Prod/portcheck/?Tar
getIp=34.209.82.230&TcpPort=23"

curl
"https://zehecvjteh.execute-api.us-east-2.amazonaws.com/Prod/portcheck/?Tar
getIp=34.209.82.230&TcpPort=80"

curl
"https://zehecvjteh.execute-api.us-east-2.amazonaws.com/Prod/portcheck/?Tar
getIp=34.209.82.230&TcpPort=443"

curl
"https://zehecvjteh.execute-api.us-east-2.amazonaws.com/Prod/portcheck/?Tar
getIp=34.209.82.230&TcpPort=10000"
```

We should see output similar to the following...

```
Hal:~/environment/portcheck-app-001 $ curl
"https://zehecvjteh.execute-api.us-east-2.amazonaws.com/Prod/portcheck/?Tar
getIp=34.209.82.230&TcpPort=22"

34.209.82.230:22/TCP is open


Hal:~/environment/portcheck-app-001 $ curl
"https://zehecvjteh.execute-api.us-east-2.amazonaws.com/Prod/portcheck/?Tar
getIp=34.209.82.230&TcpPort=23"

34.209.82.230:23/TCP is closed


Hal:~/environment/portcheck-app-001 $ curl
"https://zehecvjteh.execute-api.us-east-2.amazonaws.com/Prod/portcheck/?Tar
getIp=34.209.82.230&TcpPort=80"

34.209.82.230:80/TCP is open


Hal:~/environment/portcheck-app-001 $ curl
"https://zehecvjteh.execute-api.us-east-2.amazonaws.com/Prod/portcheck/?Tar
getIp=34.209.82.230&TcpPort=443"

34.209.82.230:443/TCP is open


Hal:~/environment/portcheck-app-001 $ curl
"https://zehecvjteh.execute-api.us-east-2.amazonaws.com/Prod/portcheck/?Tar
getIp=34.209.82.230&TcpPort=10000"
34.209.82.230:10000/TCP is closed


Hal:~/environment/portcheck-app-001 $
```

References

- Tutorial: Deploying a Hello World application - https://docs.aws.amazon.com/serverless-application-model/latest/developerguide/serverless-getting-started-hello-world.html
- https://stackoverflow.com/questions/6817640/catch-any-error-in-python
- https://stackoverflow.com/questions/19196105/how-to-check-if-a-network-port-is-open