**Steps for testing Crypto Attestation using KMS**

**Nitro enclave ensures data is processed in a controlled environment. Cryptography attestation means only authorized code can decrypt data.**

Login to account: devsecops-development (359973825921)

Create EC2 instance with AMI as (AMI ID: AWS Nitro Enclaves Developer AMI) and instance type as c5.xlarge. Select enclave when creating the instance. Select rest of options as appropriate and create new SSH PEM Key.

Tags:

Project: AWS Nitro Enclaves PoC

Name: Niro Enclave PoC

=>If creating EC2 instance with nitro enclave flag as enabled from console creates ec2 instance with nitro enclave disabled. Create with command line as follows:

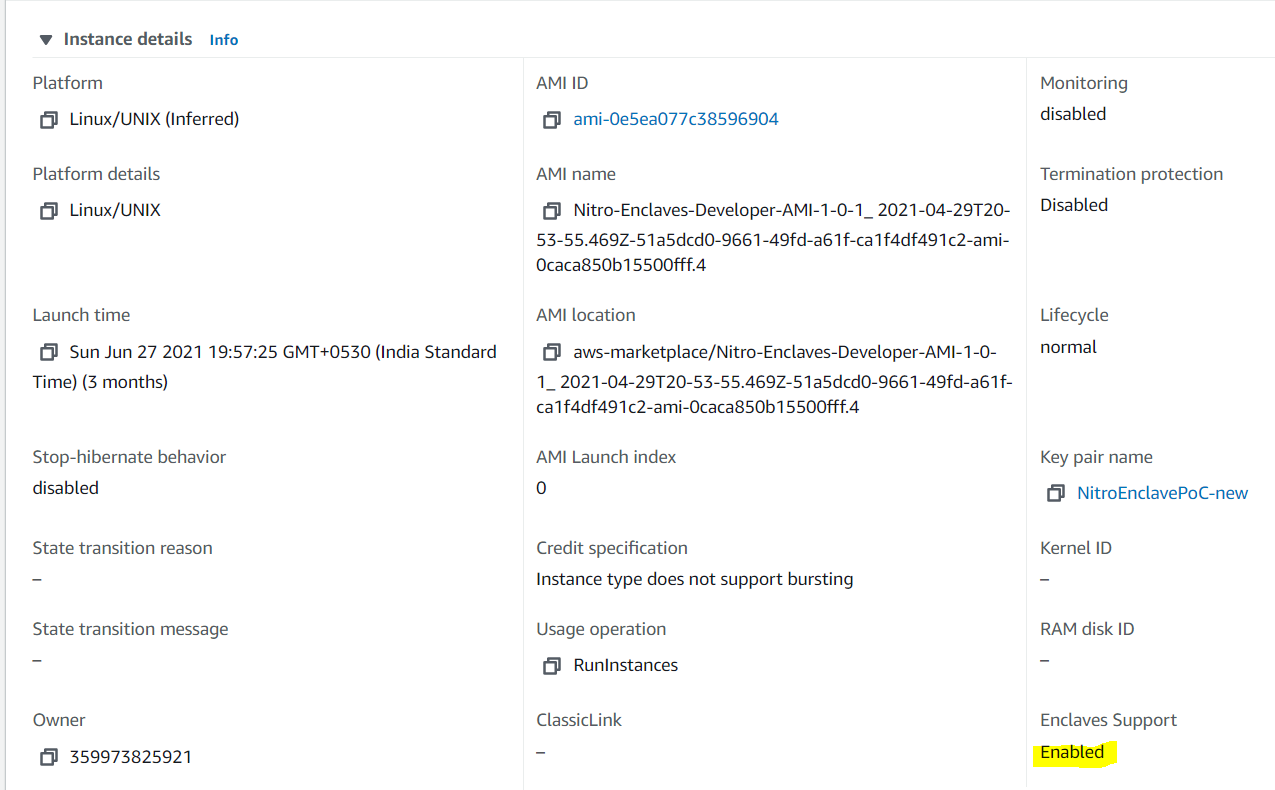
Connect to AWS instance devsecops (using Devops-Admin-Role)

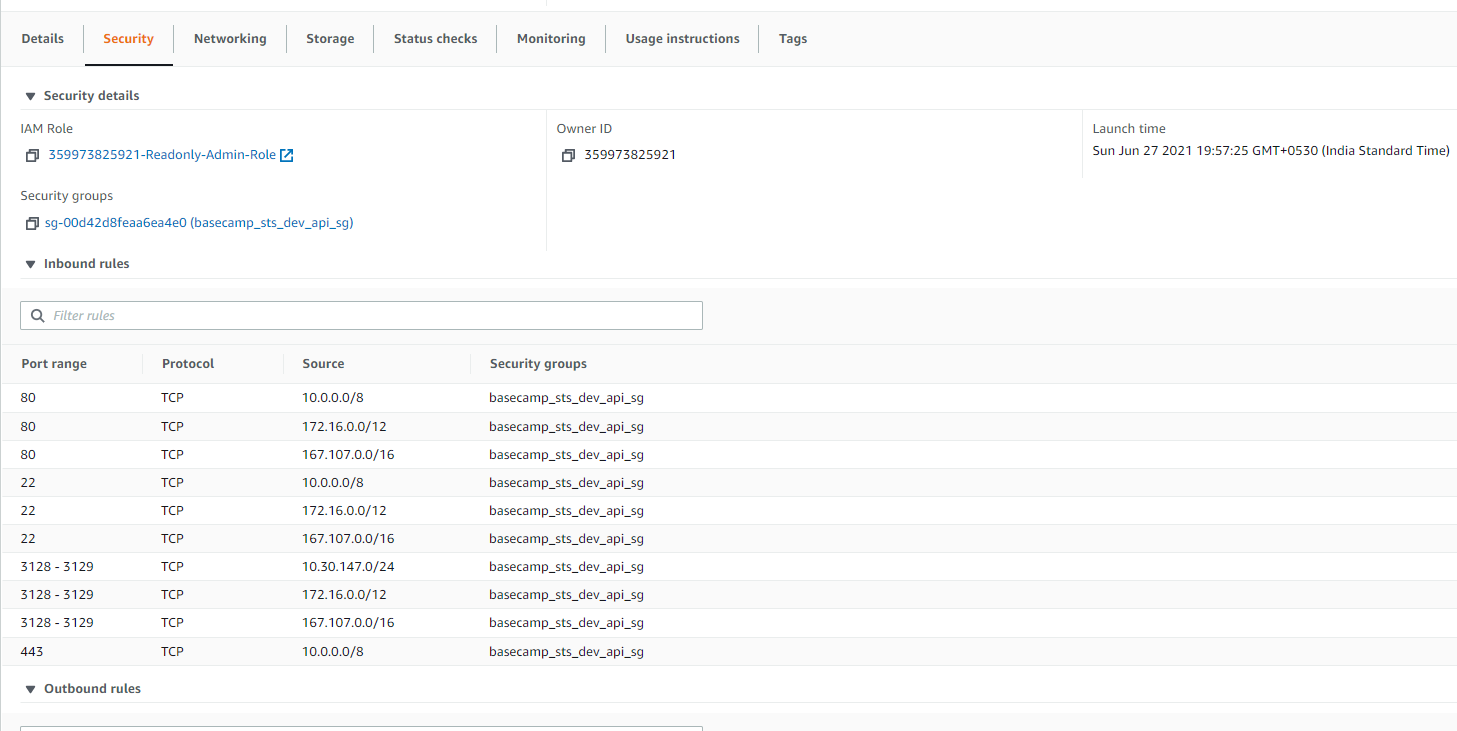
--------------------------------

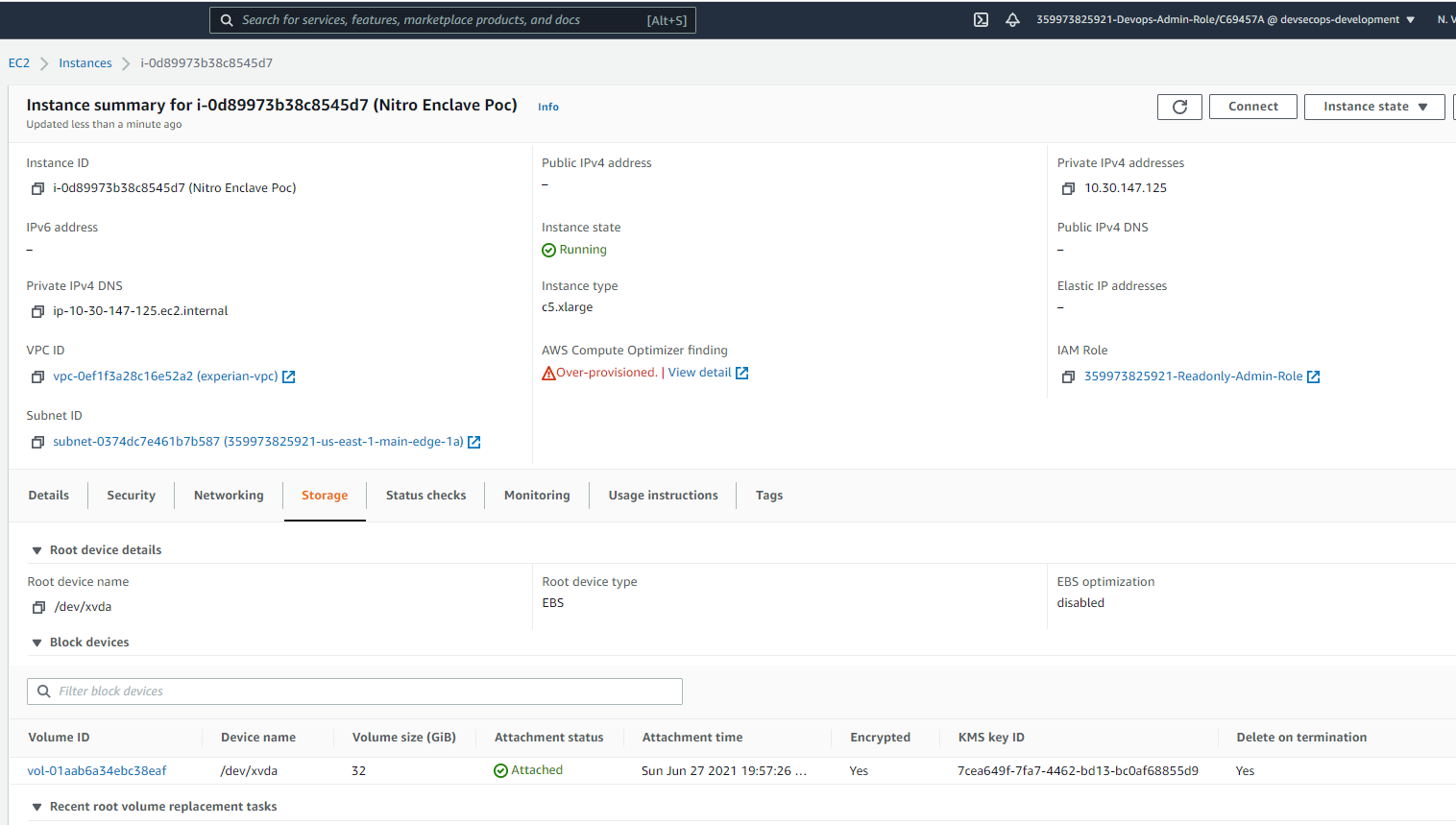
C:\Users\C69457A>gimme-aws-creds -p devsecops-development --insecure

C:\Users\C69457A>aws sts get-caller-identity

C:\Users\C69457A\pawan\Sai\Nitroenclave>aws ec2 run-instances --image-id ami-0e5ea077c38596904 --count 1 --instance-type c5.xlarge --key-name NitroEnclavePoC-new --security-group-ids sg-00d42d8feaa6ea4e0 --subnet-id subnet-0374dc7e461b7b587 --enclave-options "Enabled=true"







Login to AWS server:

C:\Users\C69457A\pawan\Sai\Nitroenclave>ssh -i "NitroEnclavePoC-new.pem" ec2-user@10.30.147.125

Install openjdk

$ sudo yum install java-1.8.0-openjdk

Install AWS Nitro Enclaves CLI

$ sudo amazon-linux-extras install aws-nitro-enclaves-cli

$ sudo yum install aws-nitro-enclaves-cli-devel -y

$ sudo usermod -aG ne ec2-user

$ sudo usermod -aG docker ec2-user

$ nitro-cli –version // nitro-cli -V

The allocated memory should be greater than four times of the EIF file size.

open /etc/nitro\_enclaves/allocator.yaml and set memory\_mib(3072) and cpu\_count (2).

#sudo nano /etc/nitro\_enclaves/allocator.yaml

#sudo systemctl start nitro-enclaves-allocator.service && sudo systemctl enable nitro-enclaves-allocator.service

#sudo systemctl start docker && sudo systemctl enable docker

reboot NitroEnclave EC2 instance using console/cli for changes to take affect

KMS Tool is a small example application for aws-nitro-enclaves-sdk-c that is able to connect to KMS and decrypt an encrypted KMS message.

To test decryption using KMS Tool (https://github.com/aws/aws-nitro-enclaves-sdk-c/blob/main/docs/kmstool.md)

#sudo yum install git

#git clone https://github.com/aws/aws-nitro-enclaves-cli.git

#cd aws-nitro-enclaves-sdk-c

#docker build --target kmstool-instance -t kmstool-instance -f containers/Dockerfile.al2 .

#docker build --target kmstool-enclave -t kmstool-enclave -f containers/Dockerfile.al2 .

#nitro-cli build-enclave --docker-uri kmstool-enclave --output-file kmstool.eif

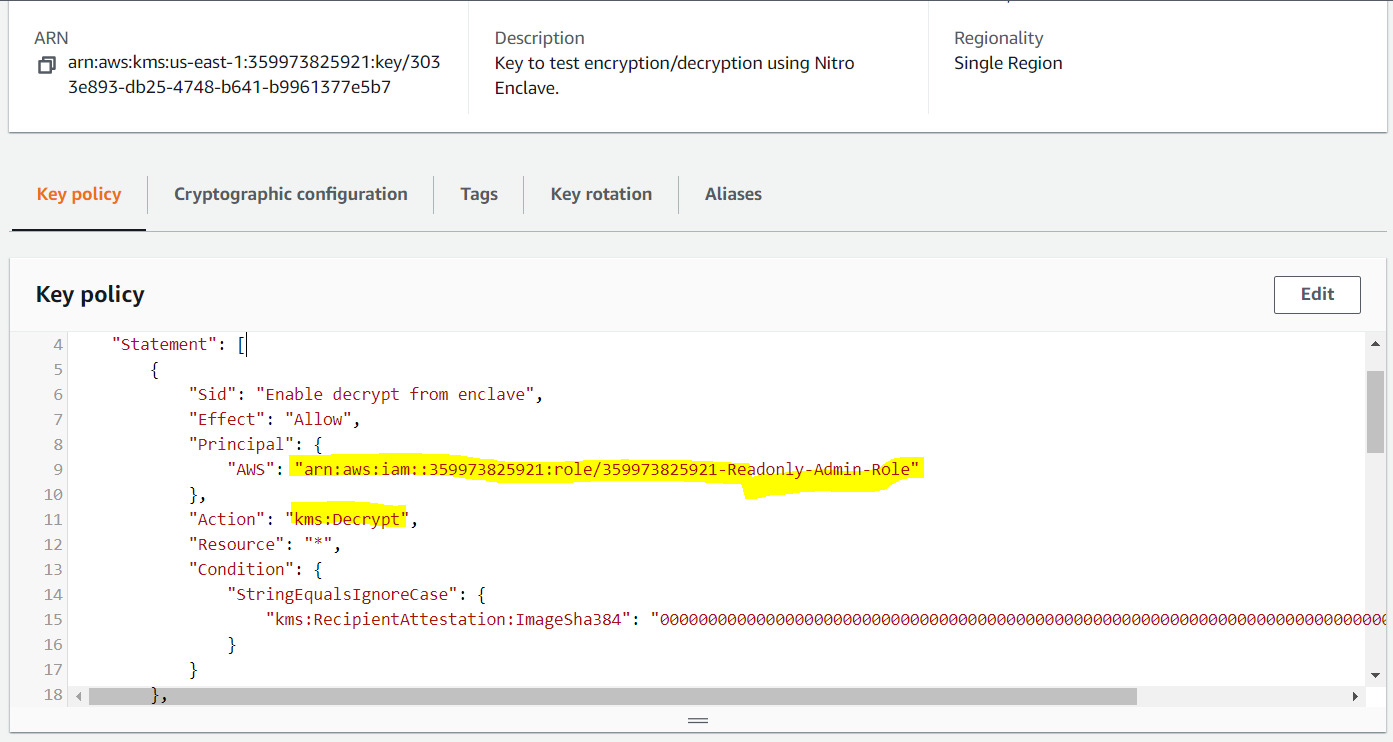
Create AWS KMS CMK (symmetric) using console with attached key policy.



OR use following command from GIT bash (using Devops-Admin-Role)

KMS\_KEY\_ARN=$(aws kms create-key --description "Nitro Enclaves Test Key" --policy test-enclave-policy.json --query KeyMetadata.Arn --output text)

echo $KMS\_KEY\_ARN



Login to AWS cli(gimme) with Readonly-Admin-Account

Run following in GIT Bash

#MESSAGE="Hello, KMS\!"

#CIPHERTEXT=$(aws kms encrypt --key-id "arn:aws:kms:us-east-1:359973825921:key/3033e893-db25-4748-b641-b9961377e5b7" --plaintext "$MESSAGE" --query CiphertextBlob --output text)

#echo $CIPHERTEXT

AQICAHgcVPJ4Y6JZ0rYkDJQtqn83HRz/htHG798sUDzcvTFzFAFALPIgVAsvuMqqb2gHLQg/AAAAZDBiBgkqhkiG9w0BBwagVTBTAgEAME4GCSqGSIb3DQEHATAeBglghkgBZQMEAS4wEQQMOhRDWVpiZLm0xSi+AgEQgCHdfuhci80TrRk0xtETTNqLnBYqhBItE1evCgEBouDK2a4=

Now try to decrypt message, it should fail since decryption is possible only from Nitro Enclave.

echo "$CIPHERTEXT" | base64 --decode > cipher-text

$aws kms decrypt --key-id "arn:aws:kms:us-east-1:359973825921:key/3033e893-db25-4748-b641-b9961377e5b7" --ciphertext-blob fileb://cipher-text

An error occurred (AccessDeniedException) when calling the Decrypt operation: The ciphertext refers to a customer master key that does not exist, does not exist in this region, or you are not allowed to access.

(Above error is correct behaviour)

Connect to EC2 instance (3 terminals):

In one instance:

#nitro-cli run-enclave --eif-path kmstool.eif --memory 3072 --cpu-count 2 --debug-mode

#nitro-cli describe-enclaves

#ENCLAVE\_ID=$(nitro-cli describe-enclaves | jq -r .[0].EnclaveID)

#nitro-cli console --enclave-id $ENCLAVE\_ID

To terminate (don’t do if not required)

#nitro-cli terminate-enclave --enclave-id $ENCLAVE\_ID

In separate instance:

#CMK\_REGION=us-east-1 # The region where you created your AWS KMS CMK

#vsock-proxy 8000 kms.$CMK\_REGION.amazonaws.com 443

In separate terminal

#CMK\_REGION=us-east-1

#ENCLAVE\_CID=$(nitro-cli describe-enclaves | jq -r .[0].EnclaveCID)

#docker run --network host -it kmstool-instance /kmstool\_instance --cid "$ENCLAVE\_CID" --region "$CMK\_REGION" "AQICAHgcVPJ4Y6JZ0rYkDJQtqn83HRz/htHG798sUDzcvTFzFAFALPIgVAsvuMqqb2gHLQg/AAAAZDBiBgkqhkiG9w0BBwagVTBTAgEAME4GCSqGSIb3DQEHATAeBglghkgBZQMEAS4wEQQMOhRDWVpiZLm0xSi+AgEQgCHdfuhci80TrRk0xtETTNqLnBYqhBItE1evCgEBouDK2a4="

OR

#CIPHERTEXT=AQICAHgcVPJ4Y6JZ0rYkDJQtqn83HRz/htHG798sUDzcvTFzFAFALPIgVAsvuMqqb2gHLQg/AAAAZDBiBgkqhkiG9w0BBwagVTBTAgEAME4GCSqGSIb3DQEHATAeBglghkgBZQMEAS4wEQQMOhRDWVpiZLm0xSi+AgEQgCHdfuhci80TrRk0xtETTNqLnBYqhBItE1evCgEBouDK2a4=

#docker run --network host -it kmstool-instance /kmstool\_instance --cid "$ENCLAVE\_CID" --region "$CMK\_REGION" "$CIPHERTEXT"

If successful:

-----------------

[INFO] [2021-08-20T15:06:40Z] [00007f7a3bfff700] [connection-manager] - id=0x292b480: Destroying self

[DEBUG] [2021-08-20T15:06:40Z] [00007f7a3bfff700] [channel-bootstrap] - id=0x292a7c0: releasing bootstrap reference

[DEBUG] [2021-08-20T15:06:40Z] [00007f7a3bfff700] [channel] - id=0x7f7a34018a90: destroying channel.

[DEBUG] [2021-08-20T15:06:40Z] [00007f7a3bfff700] [channel-bootstrap] - id=0x292a7c0: releasing bootstrap reference

Object = { "Status": "Ok" }

Object = { "Operation": "Decrypt", "Ciphertext": "AQICAHgcVPJ4Y6JZ0rYkDJQtqn83HRz\/htHG798sUDzcvTFzFAFALPIgVAsvuMqqb2gHLQg\/AAAAZDBiBgkqhkiG9w0BBwagVTBTAgEAME4GCSqGSIb3DQEHATAeBglghkgBZQMEAS4wEQQMOhRDWVpiZLm0xSi+AgEQgCHdfuhci80TrRk0xtETTNqLnBYqhBItE1evCgEBouDK2a4=" }

Object = { "Status": "Ok", "Message": "HelloKMS" }

**Troubleshooting: If failure (Could not get credentials error as shown below)**

If you encounter Could not get credentials error when running kmstool\_instance it means that you have not attached an IAM role to an instance profile or you have not associated the instance profile to your instance.

-----------------------------

[ERROR] [2021-07-06T07:34:44Z] [00007f87196de700] [AuthCredentialsProvider] - (id=0x16d2940) Default chain credentials provider failed to source credentials with error 1051(aws-c-io: AWS\_IO\_SOCKET\_CLOSED, socket is closed.)

[DEBUG] [2021-07-06T07:34:44Z] [00007f87196de700] [channel] - id=0x7f870c016860: Channel shutdown is already pending, not scheduling another.

[DEBUG] [2021-07-06T07:34:44Z] [00007f87196de700] [channel] - id=0x7f870c016860: handler 0x7f870c019128 shutdown in write dir completed.

[DEBUG] [2021-07-06T07:34:44Z] [00007f87196de700] [socket] - id=0x7f870c01b190 fd=7: closing

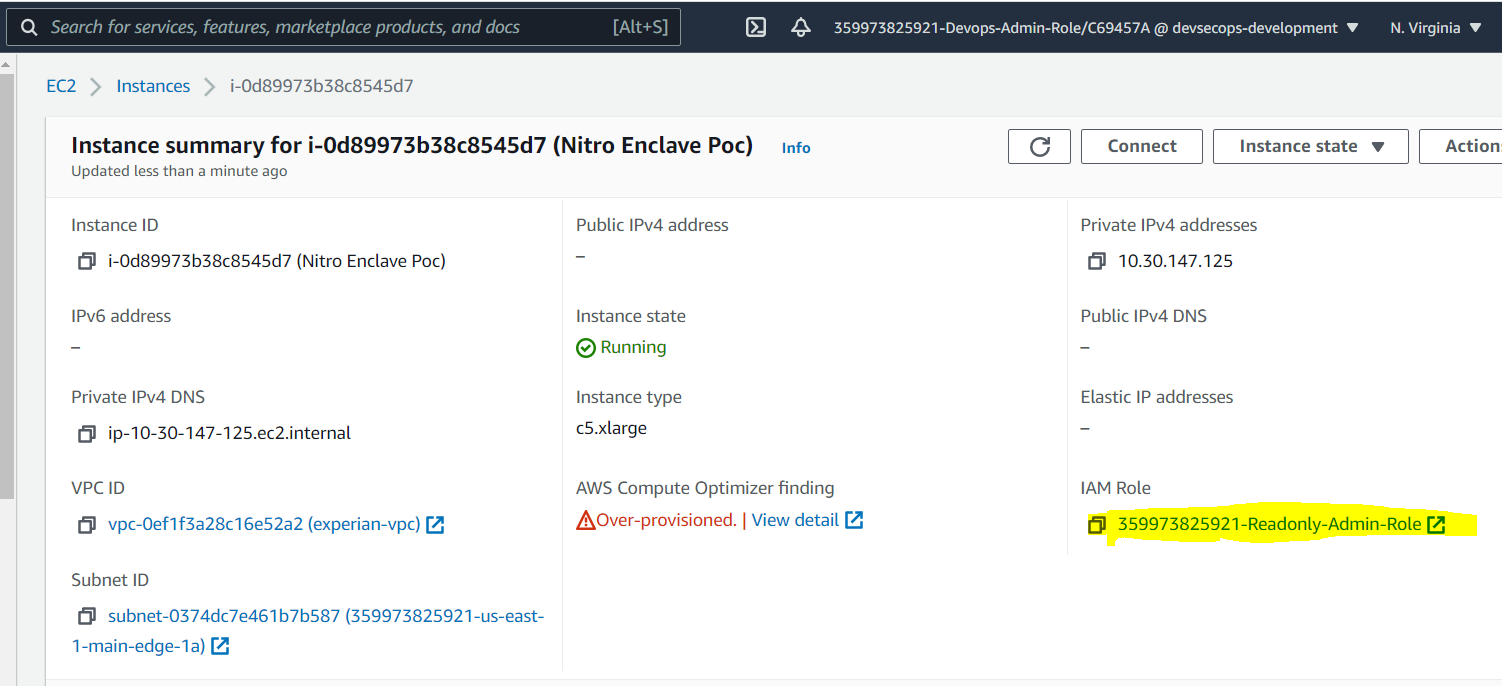
[DEBUG] [2021-07-06T07:34:44Z] [00007f87196de700] [task-scheduler] - id=0x7f870c019ed0: Scheduling epoll\_event\_loop\_unsubscribe\_cleanup task for immediate execution

Could not get credentials

**Solution:**

1. Attach role to EC2 instance

----------------------------------



-Login to aws cli using 359973825921-Devops-Admin-Role

#aws iam create-instance-profile --instance-profile-name Readonly-Admin-Role

#aws iam add-role-to-instance-profile --role-name 359973825921-Readonly-Admin-Role --instance-profile-name Readonly-Admin-Role

#aws ec2 associate-iam-instance-profile --instance-id i-0d89973b38c8545d7 --iam-instance-profile Name=Readonly-Admin-Role

Verify instance profile association

#aws ec2 describe-iam-instance-profile-associations

#aws iam get-instance-profile --instance-profile-name Readonly-Admin-Role

1. Get role to see trust policies attached to a role and add trust policy

-----------------------------------------------------

#aws iam get-role --role-name 359973825921-Readonly-Admin-Role

Update trust policy (add trust policy for "Principal": { "Service": "ec2.amazonaws.com"} either from console or from cli as shown below).

--------------------------

C:\>aws iam update-assume-role-policy --role-name 359973825921-Readonly-Admin-Role --policy-document file://C:\Users\C69457A\pawan\Sai\Nitroenclave\EC2trustpolicy.json



1. Dissasssociate instance profile and associate again

--------------------------------------------

get associate id using below command

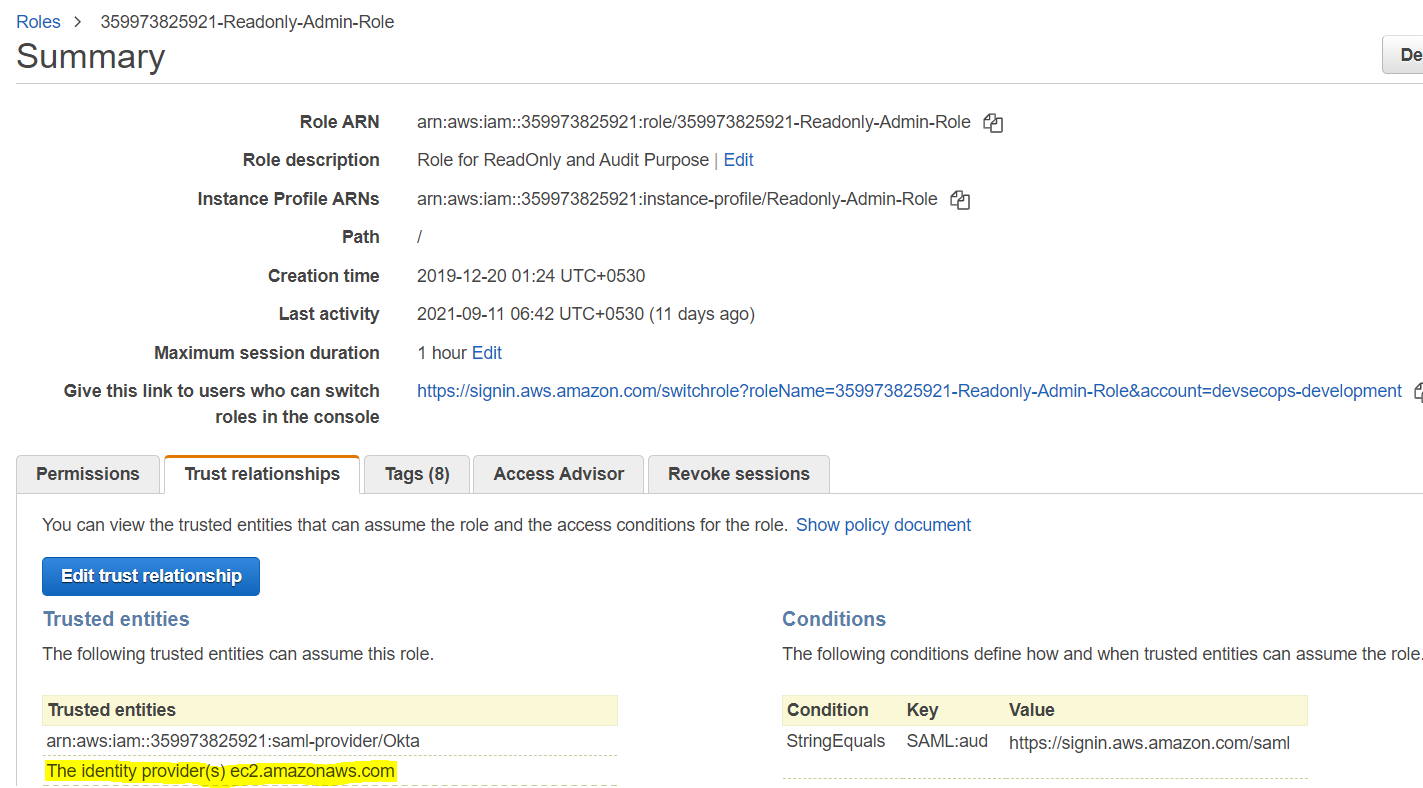
#aws ec2 describe-iam-instance-profile-associations

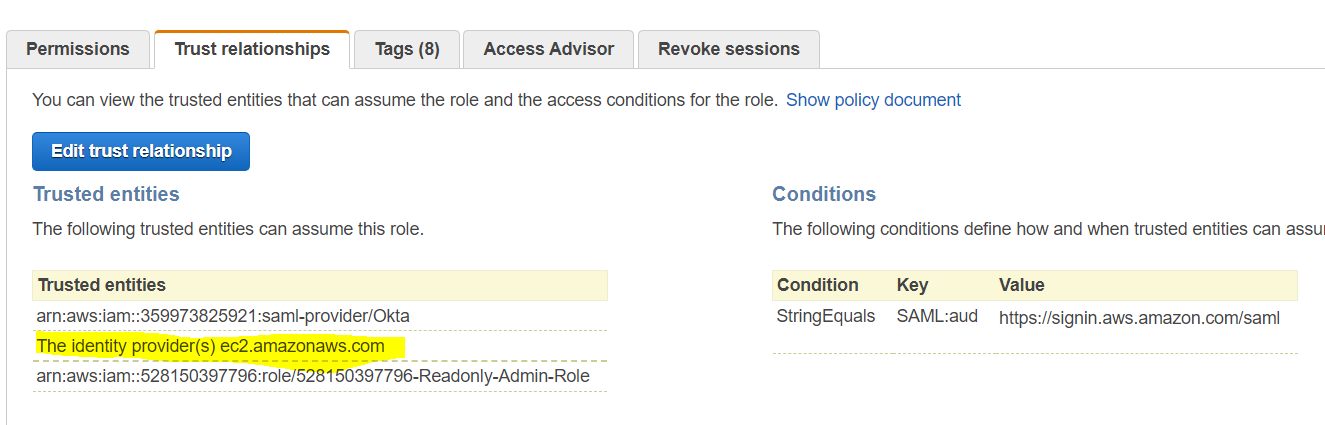
#aws ec2 disassociate-iam-instance-profile --association-id iip-assoc-03da8d52454553622

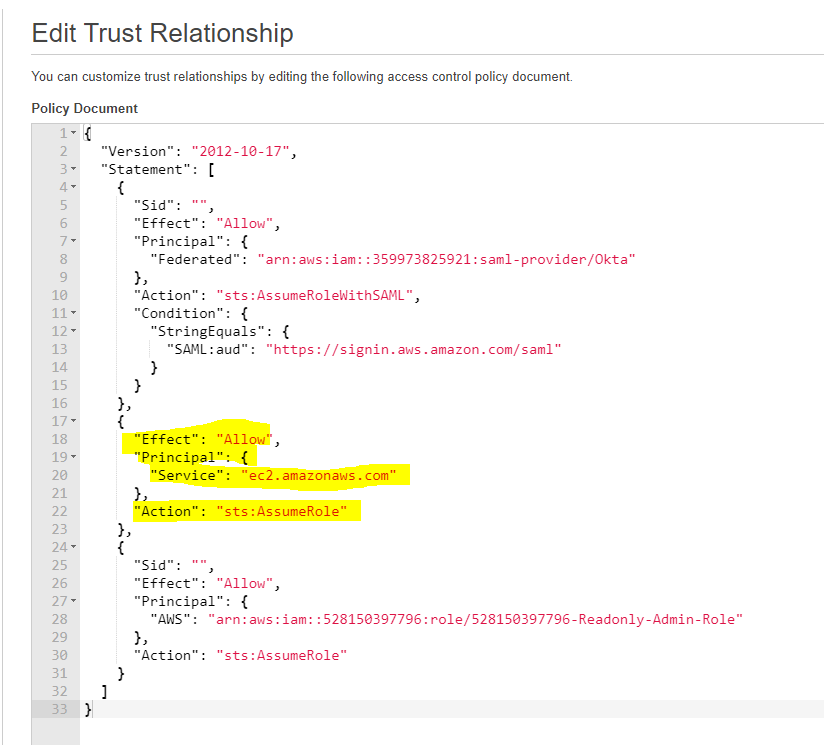
check it is disassociated using #aws ec2 describe-iam-instance-profile-associations

associate again:

#aws ec2 associate-iam-instance-profile --instance-id i-0d89973b38c8545d7 --iam-instance-profile Name=Readonly-Admin-Role







**References:**

<https://docs.aws.amazon.com/enclaves/latest/user/getting-started.html>

<https://nitro-enclaves.workshop.aws/en/my-first-enclave/cryptographic-attestation.html>

<https://github.com/aws/aws-nitro-enclaves-sdk-c/blob/main/docs/kmstool.md>

**Integrating SafeNet HSM and Nitro Enclave using Java Encryption Libraries**

<https://nitro-enclaves.workshop.aws/en/my-first-enclave/secure-local-channel.html>

In this section, you will set up communication between the Nitro Enclave and an external endpoint. Out of the box, Nitro Enclaves can communicate only with parent instance and only through secure local communication channel - vsock. To allow the Nitro Enclave to communicate with the external endpoint, you will run a proxy server that runs on the host instance and Traffic-Forwarder in the Nitro Enclave. They will forward traffic from an Enclave coming through vsock to an external endpoint.

In this example, you will call a sample server application running inside the Nitro Enclave and configure a forwarder with vsock-proxy to allow the enclave to reach out to the external website. The client application will be reaching out to a server, running inside the Nitro Enclave, and listening on port 5005. Local loopback will forward all traffic from the server to the Traffic-Forwarder (on port 443) and from there to the vsock-proxy (on port 8001). The vsock-proxy routes the traffic to the target external endpoint. And rolling back response will trickle back to the client application. Both types of calls (client-server and enclave-endpoint) will go through the vsock socket.

**External Endpoint = SafeNet HSM dev servers running on IP (10.8.74.129,10.8.74.130,10.28.241.24) and port 9000**

Diagram

Description automatically generated

You need to request for opening firewall port to HSM IP (10.8.74.129,10.8.74.130,10.28.241.24) from your subnet of EC2 instance where nitro enclave is installed.

<https://experian.service-now.com/itservices?id=sc_cat_item&sys_id=6865efd3dbbce3442511fa910f961993>

After firewall port is opened, check connectivity using #telnet IP port command e.g.

#telnet 10.8.74.129 9000

**Java Encryption Libraries**

You can find the documentation for using the library here :-

<https://code.experian.local/projects/APIE/repos/basecamp-crypto/browse?at=refs%2Fheads%2Ffeature%2F0.0.5>

**Preparing Java program for encryption and decryption using the onprem HSM.**

You need to generate jar file (**nitroenclave-poc-1.0-SNAPSHOT.jar**) for Java code for encryption/decryption using onprem HSM. Create Java maven project in Eclispe and copy attached files (NitroApp.java, pom.xml).

You need to use the attached properties file and change the path of the keystore.jks in the file for your local test.  You can use the following creds for testing: -

"PAJ\_AscendSandbox\_7801"

"FkC6Uswyra2LCWnz"

  The library is in artifactory :-

**Artifact :**

 <dependency>  
    <groupId>com.experian.basecamp.crypto</groupId>  
    <artifactId>basecamp-crypto-core</artifactId>  
    <version>0.0.5-SNAPSHOT</version>  
</dependency>

**Repo Url:**

        <repository>

          <snapshots>

            <enabled>false</enabled>

          </snapshots>

          <id>basecamp-central</id>

          <name>basecamp-central</name>

          <url>[https://artifacts.experian.local/artifactory/cis-ascend-platform-infrastructure-local</url>](https://artifacts.experian.local/artifactory/cis-ascend-platform-infrastructure-local%3c/url%3e)

        </repository>

        <repository>

          <snapshots>

            <enabled>true</enabled>

          </snapshots>

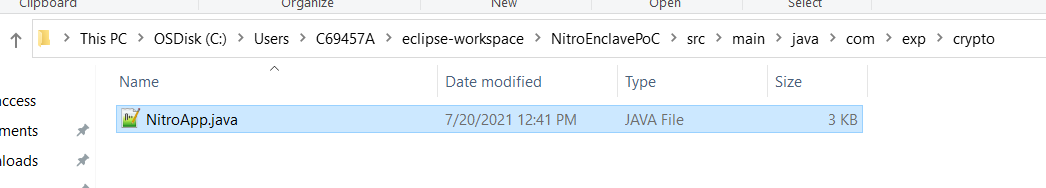
          <id>basecamp-snapshot</id>

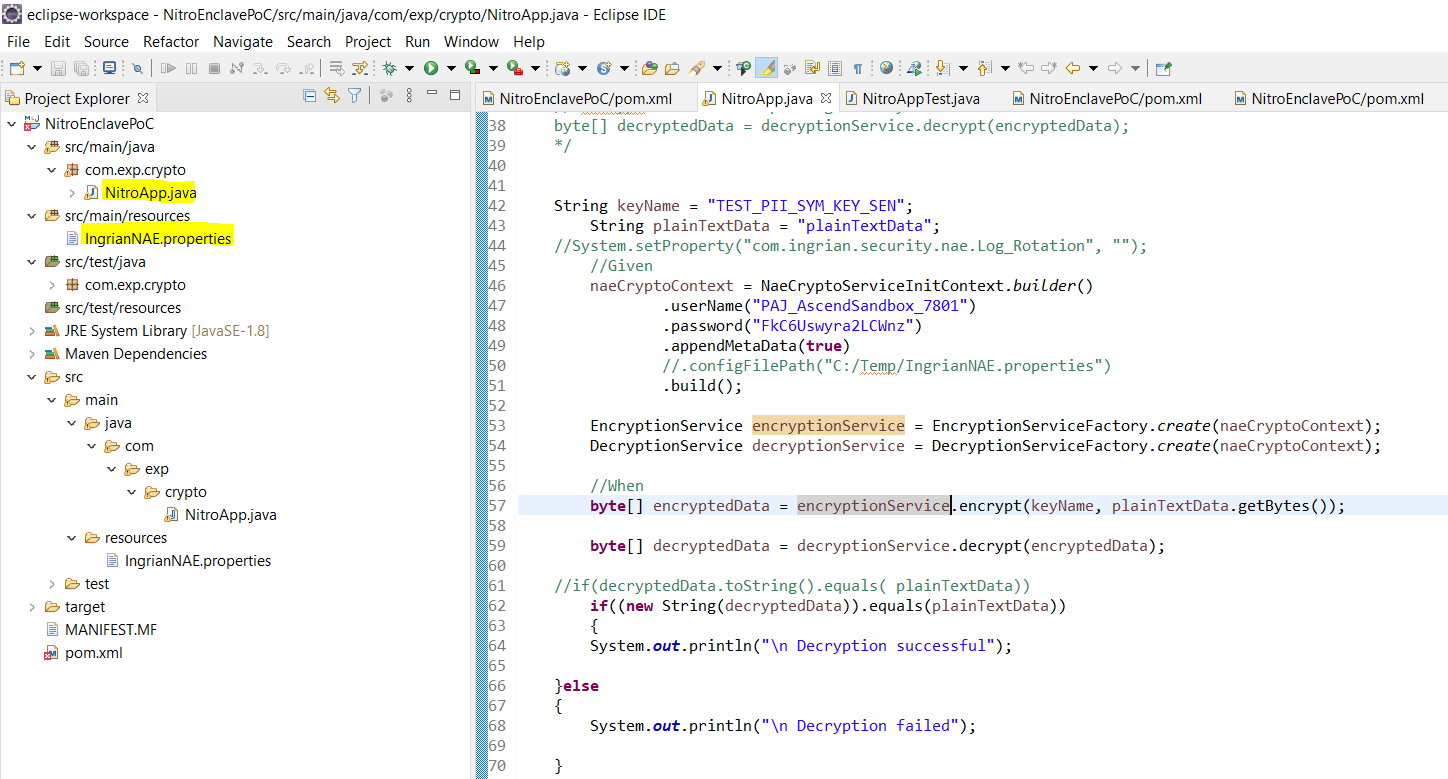
          <name>basecamp-snapshot</name>

          <url>[https://artifacts.experian.local/artifactory/cis-ascend-platform-infrastructure-local</url>](https://artifacts.experian.local/artifactory/cis-ascend-platform-infrastructure-local%3c/url%3e)

        </repository>



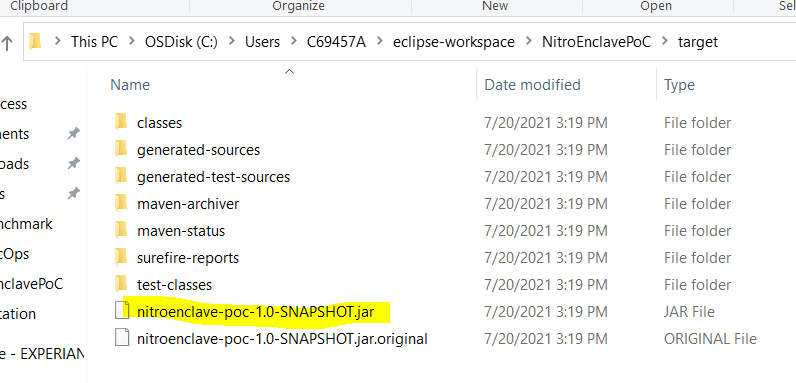




Path for Eclipse project (use eclipse to build project instead of command line): C:\Users\C69457A\eclipse-workspace\NitroEnclavePoC

-------------------------

Run as>Maven clean>Maven install



**Login to ec2 instance**

----------------------------

C:\Users\C69457A\pawan\Sai\Nitroenclave>ssh -i "NitroEnclavePoC-new.pem" [ec2-user@10.30.147.125](mailto:ec2-user@10.30.147.125)

Git clone workshop directory (https://github.com/aws-samples/aws-nitro-enclaves-workshop) using following command:

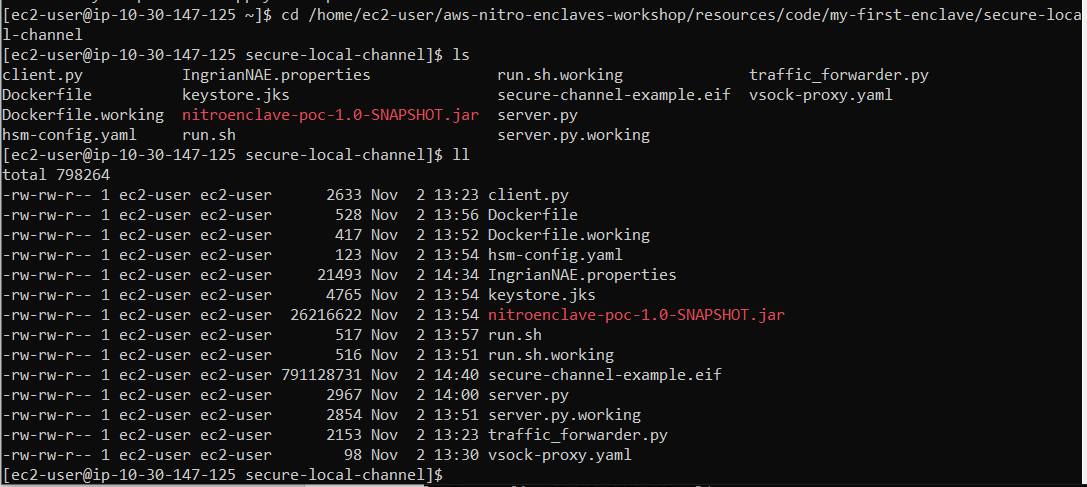
#git clone https://github.com/aws-samples/aws-nitro-enclaves-workshop.git

**Go to secure-local-channel folder**

#cd /home/ec2-user/aws-nitro-enclaves-workshop/resources/code/my-first-enclave/secure-local-channel

**Copying files (IngrianNAE.properties, keystore.jks and nitroenclave-poc-1.0-SNAPSHOT.jar) to secure-local-channel folder.**

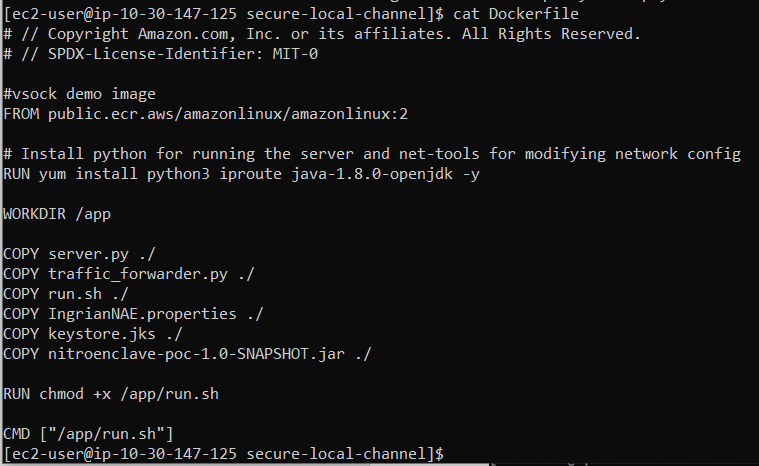
#C:\Users\C69457A\pawan\Sai\Nitroenclave>scp -i "NitroEnclavePoC-new.pem" IngrianNAE.properties [ec2-user@10.30.147.125:\home\ec2-user\aws-nitro-enclaves-workshop\resources\code\my-first-enclave\secure-local-channel](mailto:ec2-user@10.30.147.125:\home\ec2-user\aws-nitro-enclaves-workshop\resources\code\my-first-enclave\secure-local-channel)



**Edit Dockerfile as follows:**

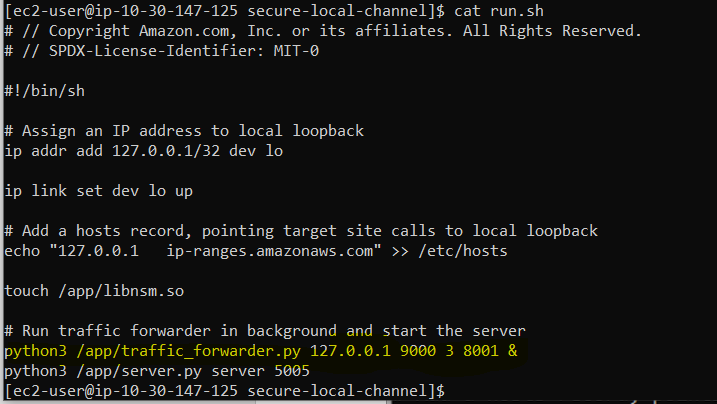
Add IngrianNAE.properties, keystore.jks and nitroenclave-poc-1.0-SNAPSHOT.jar.

Add yum install java-1.8.0-openjdk

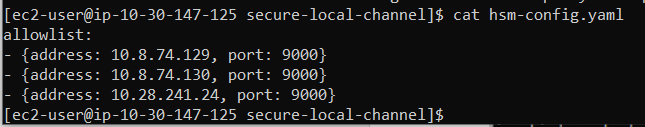


**Edit run.sh (change port 443 to 9000) as follows:**

python3 /app/traffic\_forwarder.py 127.0.0.1 **9000** 3 8001 &



**Create new file hsm-config.yaml and add HSM IP addresses and port no. as follows:**



**Edit server.py as follows:**

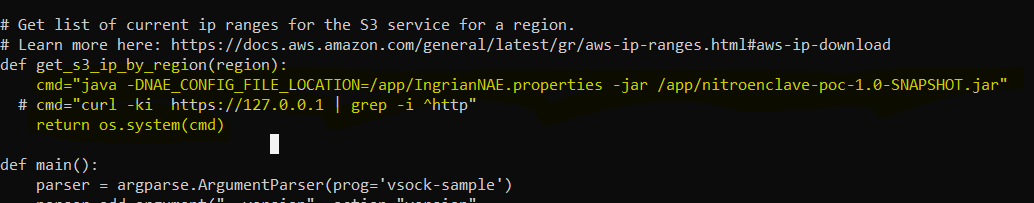
import os

def get\_s3\_ip\_by\_region(region):

cmd="java -DNAE\_CONFIG\_FILE\_LOCATION=/app/IngrianNAE.properties -jar /app/nitroenclave-poc-1.0-SNAPSHOT.jar"

# cmd="curl -ki https://127.0.0.1 | grep -i ^http"

return os.system(cmd)



**Changes in IngrianNAE.properties**

Edit IngrianNAE.properties

Comment

#NAE\_IP.1=10.8.74.129:10.8.74.130:10.28.241.24

NAE\_IP.1=127.0.0.1

Key\_Store\_Location=/app/keystore.jks





**To test encryption/decryption using Nitro Enclave**

Open 3 windows and login to parent instances of Nitroenclave

C:\Users\C69457A\pawan\Sai\Nitroenclave>ssh -i "NitroEnclavePoC-new.pem" ec2-user@10.30.147.125

#cd /home/ec2-user/aws-nitro-enclaves-workshop/resources/code/my-first-enclave/secure-local-channel

From any one windows, terminate any existing enclave

#nitro-cli describe-enclaves

#nitro-cli terminate-enclave --enclave-id i-0d89973b38c8545d7-enc17a4def5c44dc23

In one window:

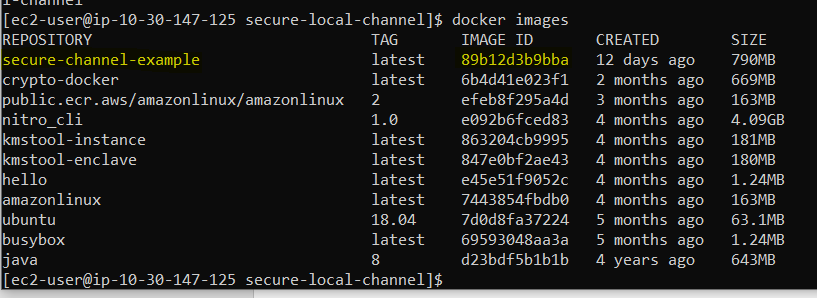
----------------

#vsock-proxy 8001 10.8.74.129 9000 --config hsm-config.yaml

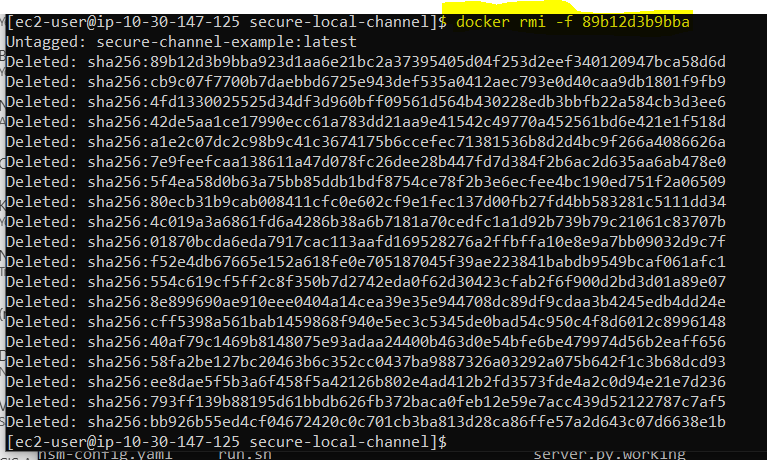
In other window:

------------------

#docker images



#docker rmi -f ${imageid of secure-channel-example}



# docker build ./ -t secure-channel-example

#rm secure-channel-example.eif if already existing.

[ec2-user@ip-10-30-147-125 secure-local-channel]$ nitro-cli build-enclave --docker-uri secure-channel-example:latest --output-file secure-channel-example.eif

Start building the Enclave Image...

Using the locally available Docker image...

Enclave Image successfully created.

{

"Measurements": {

"HashAlgorithm": "Sha384 { ... }",

"PCR0": "6693f9ee4eea60ab30d558c5f7d5dffc3587861db45fadfec057f965fb71ef49ffca1762a74ce8ccaa60c10c23f61e86",

"PCR1": "c35e620586e91ed40ca5ce360eedf77ba673719135951e293121cb3931220b00f87b5a15e94e25c01fecd08fc9139342",

"PCR2": "25ca0496e40f78c0e4f0743baae4cc00f0da1b83358badff7ca673f59630fdfbd87a463fc41ed302cfcdc1045ac03e98"

}

}

[ec2-user@ip-10-30-147-125 secure-local-channel]$

[ec2-user@ip-10-30-147-125 secure-local-channel]$ nitro-cli run-enclave --cpu-count 2 --memory 3036 --eif-path secure-channel-example.eif --debug-mode

Start allocating memory...

Started enclave with enclave-cid: 16, memory: 3072 MiB, cpu-ids: [1, 3]

{

"EnclaveID": "i-0d89973b38c8545d7-enc17bedb03c799f08",

"ProcessID": 25481,

"EnclaveCID": 16,

"NumberOfCPUs": 2,

"CPUIDs": [

1,

3

],

"MemoryMiB": 3072

}

[ec2-user@ip-10-30-147-125 secure-local-channel]$nitro-cli describe-enclaves

[ec2-user@ip-10-30-147-125 secure-local-channel]$ ENCLAVE\_ID=$(nitro-cli describe-enclaves | jq -r ".[0].EnclaveID")

[ec2-user@ip-10-30-147-125 secure-local-channel]$ [ "$ENCLAVE\_ID" != "null" ] && nitro-cli console --enclave-id ${ENCLAVE\_ID}

11)In other window:

---------------------

[ec2-user@ip-10-30-147-125 secure-local-channel]$ ENCLAVE\_CID=$(nitro-cli describe-enclaves | jq -r ".[0].EnclaveCID")

[ec2-user@ip-10-30-147-125 secure-local-channel]$ python3 client.py client ${ENCLAVE\_CID} 5005 "us-east-1"

=======================================================================================

**Output**

Let's accept stuff

Connection from <socket.socket fd=4, family=AddressFamily.AF\_VSOCK, type=SocketKind.SOCK\_STREAM, proto=0, laddr=(22, 5005), raddr=(3, 3767881867)>33767881867

Message received: us-east-1

SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".

SLF4J: Defaulting to no-operation (NOP) logger implementation

SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.

**Decryption successful**

Client call closed

Let's accept stuff

=========================================================================================

**Troubleshooting:**

1)To connect to EC2 instance

C:\Users\C69457A\pawan\Sai\Nitroenclave>ssh -i "NitroEnclavePoC-new.pem" ec2-user@10.30.147.125

2) Vsock error

[ec2-user@ip-10-30-147-125 secure-local-channel]$ vsock-proxy 8001 10.8.74.129 9000 --config hsm-config.yaml

Error: "Could not listen for connections: Could not bind to Vsock(cid: 3 port: 8001)"

[ec2-user@ip-10-30-147-125 secure-local-channel]$

Solution:

#ps -ef | grep vsock

#kill -9 vsock-pid (1st pid)

start vsock proxy again

To stop vsock-proxy command, press Ctrl-C.

3)TO terminate enclave

#nitro-cli describe-enclaves

#nitro-cli terminate-enclave --enclave-id i-0d89973b38c8545d7-enc17a4def5c44dc23

4) To test java encryption library from within parent instance

#java -DNAE\_CONFIG\_FILE\_LOCATION=IngrianNAE.properties -jar nitroenclave-poc-1.0-SNAPSHOT.jar

(you need to modify IngrianNAE.properties to set path of keystore.jks and set IP of HSM as follows:

Key\_Store\_Location=keystore.jks

#Key\_Store\_Location=/app/keystore.jks

NAE\_IP.1=10.8.74.129:10.8.74.130:10.28.241.24

#NAE\_IP.1=127.0.0.1

5) Insufficient memory error

The allocated memory should be greater than four times of the EIF file size.

open /etc/nitro\_enclaves/allocator.yaml and set memory\_mib(3072) and cpu\_count (2).

#sudo nano /etc/nitro\_enclaves/allocator.yaml

#sudo systemctl start nitro-enclaves-allocator.service && sudo systemctl enable nitro-enclaves-allocator.service

#sudo systemctl start docker && sudo systemctl enable docker

reboot NitroEnclave EC2 instance using console/cli for changes to take affect