Deploy Java Application on AWS 3-Tier Architecture

Goal of this project is to deploy scalable, highly available and secured Java application on 3-tier architecture and provide application access to the end users from public internet.

Pre-Requisites

1. Create AWS Free Tier account
2. Create a Bitbucket account and create a repository to keep Java source code.
3. Migrate Java Source Code to your own Bitbucket repository
4. Create an account in Sonarcloud.
5. Create an account in Jfrog cloud.

Pre-deployment

1. Create Global AMI

a. AWS CLI

This is installed by default on Amazon Linux 2023

b. Cloudwatch agent

So let's install CloudWatch agent.

Lets connect with ssh to our EC2 machine then:

```
# sudo yum install amazon-cloudwatch-agent
```

Run this cloudwatch config wizard and select the defaults, but ensure to select the memory option when prompted and the cwagent user

```
#/opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-config-wizard
```

Start the cloudwatch agent

```
#/opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a fetchconfig -m ec2 -c file:/opt/aws/amazon-cloudwatch-agent/bin/config.json -s
```

Verify the cloudwatch agent is running:

```
# systemctl status amazon-cloudwatch-agent.service
```

To Push custom memory metrics to Cloudwatch, attach an IAM role to the instance with this AWS managed policy named **CloudWatchFullAccess**

c. Install AWS SSM agent

This is already installed by default on Amazon Linux 2023. You can test it by attaching a IAM role **AmazonSSMFullAccess** and connect to the EC2 AWS console

Once all the components are installed we can build an image as global AMI.

2. Create Golden AMI using Global AMI for Nginx application

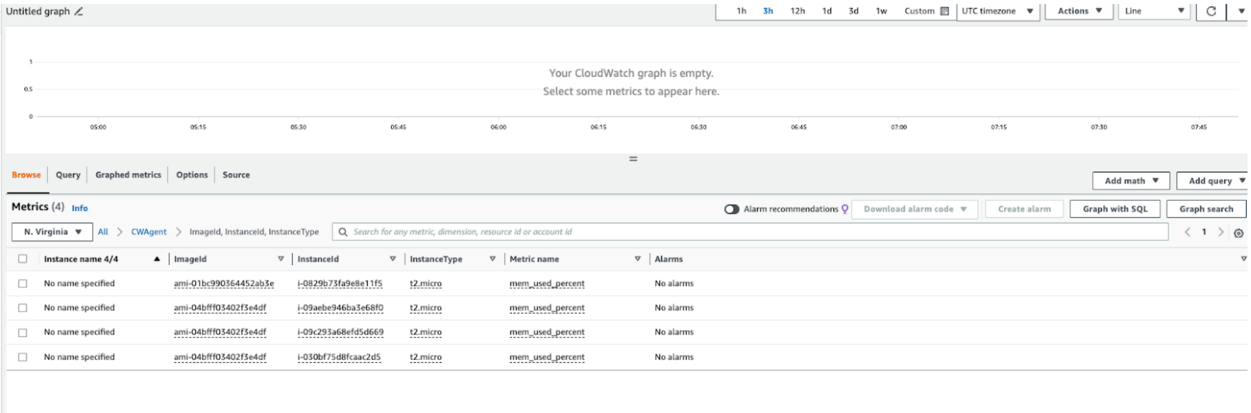
a. Install Nginx

```
# sudo dnf install nginx -y
```

b. Push custom memory metrics to Cloudwatch

To push custom memory metrics to cloudwatch the role that we talk about above need to be added to the ec2 machine

On Cloudwatch a new metric will be available: **CWagent**
And you will get metrics get from the EC2 instances.



Your CloudWatch graph is empty.
Select some metrics to appear here.

Metrics (4) Info

	Instance name 4/4	ImageId	InstanceId	InstanceType	Metric name	Alarms
□	No name specified	ami-01bc990364452ab3e	i-0829b73fa9e8e11f5	t2.micro	mem_used_percent	No alarms
□	No name specified	ami-04bfff03402f3e4df	i-09aebe946ba3e68f0	t2.micro	mem_used_percent	No alarms
□	No name specified	ami-04bfff03402f3e4df	i-09c293a68efd5d669	t2.micro	mem_used_percent	No alarms
□	No name specified	ami-04bfff03402f3e4df	i-030bf75d8fcaac2d5	t2.micro	mem_used_percent	No alarms

3. Create Golden AMI using Global AMI for Apache Tomcat application

a. Install Apache Tomcat

```
# wget
https://downloads.apache.org/tomcat/tomcat-9/v9.0.41/bin/apache-tomcat
-9.0.41.tar.gz
```

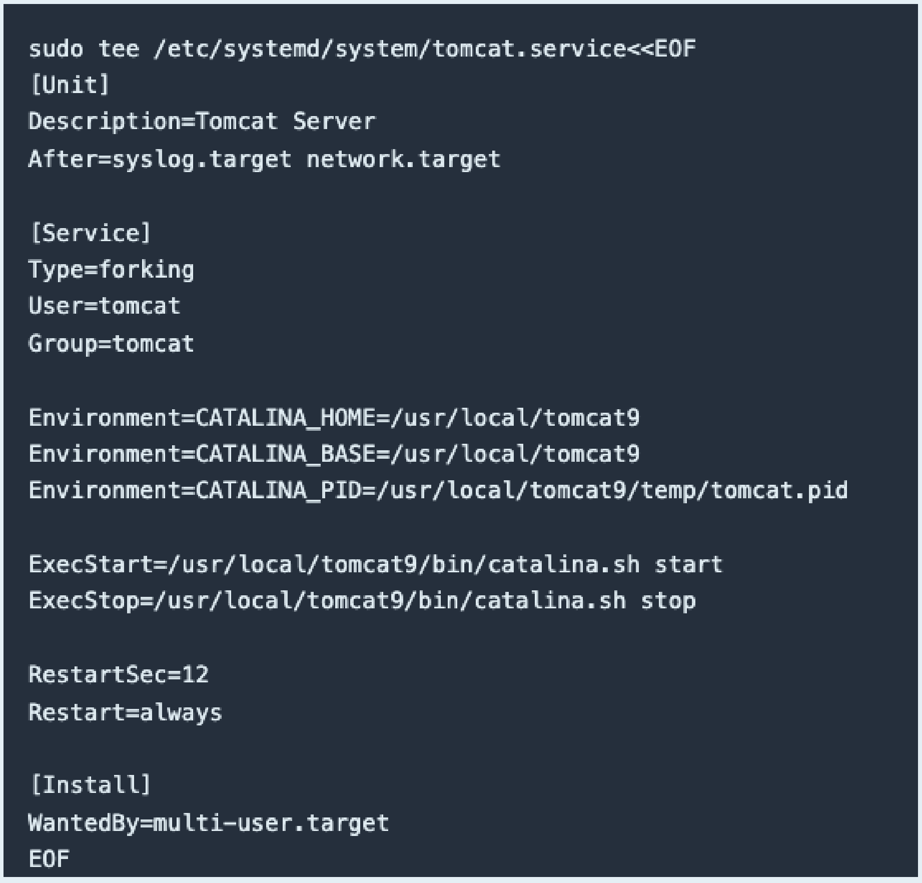
```
# tar -xvf /root/apache-tomcat-9.0.41.tar.gz
```

```
# mv apache-tomcat-9.0.41 tomcat9mv tomcat9 /usr/local
```

```
# useradd -r tomcat
```

```
# chown -R tomcat:tomcat /usr/local/tomcat9
```

b. Configure Tomcat as Systemd service



```
sudo tee /etc/systemd/system/tomcat.service<<EOF
[Unit]
Description=Tomcat Server
After=syslog.target network.target

[Service]
Type=forking
User=tomcat
Group=tomcat

Environment=CATALINA_HOME=/usr/local/tomcat9
Environment=CATALINA_BASE=/usr/local/tomcat9
Environment=CATALINA_PID=/usr/local/tomcat9/temp/tomcat.pid

ExecStart=/usr/local/tomcat9/bin/catalina.sh start
ExecStop=/usr/local/tomcat9/bin/catalina.sh stop

RestartSec=12
Restart=always

[Install]
WantedBy=multi-user.target
EOF
```

Reload tomcat service

```
# sudo systemctl daemon-reload
```

Restart/Start tomcat service

```
# sudo systemctl start tomcat
```

Check tomcat service status

```
# systemctl status tomcat.service
```

c. Install JDK 11

```
# sudo dnf install java-11-amazon-corretto-devel.x86_64
```

d. Push custom memory metrics to Cloudwatch.

We build the tomcat9 base on the global AMI, so cloudwatch agent is already installed.

Finally we check on our browser the Tomcat Server GUI via http://[AWS EC2 Public IP]:8080

4. Create Golden AMI using Global AMI for Apache Maven Build Tool

a. Install Apache Maven

```
# wget
https://downloads.apache.org/maven/maven-<Maven_Version>/binaries/apac
he-maven-<Maven_Version>-bin.tar.gz
```

```
# tar -xzvf apache-maven-<Maven_Version>-bin.tar.gz
```

```
# mv apache-maven-<Maven_Version> /opt/
```

b. Install Git

```
# yum install -y git
```

c. Install JDK 11

```
# sudo yum install java-11-amazon-corretto-devel
```

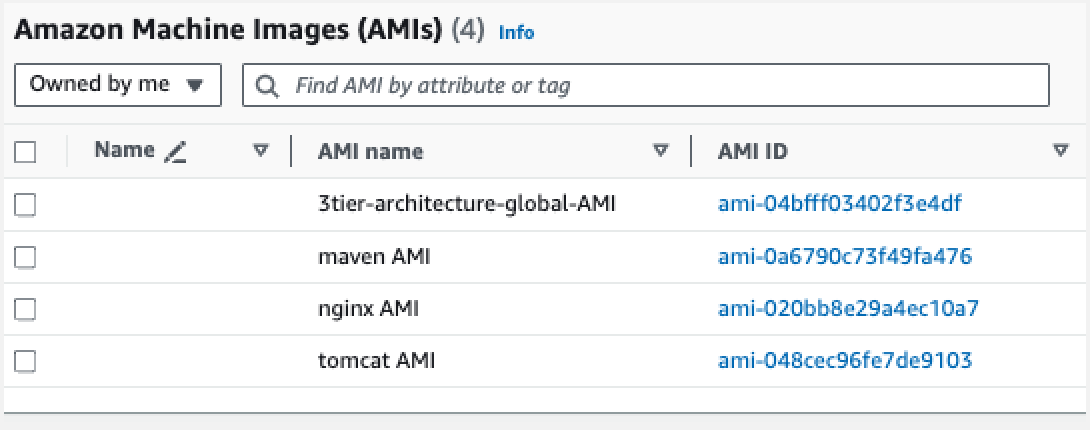
```
#java -version
```

d. Update Maven Home to the system PATH environment variable

```
# export
PATH='/opt/apache-maven-<version>':'/opt/apache-maven-<version>/bin':$
PATH
```

```
#mvn -version
```

Custom AMI should be created as follow:



Amazon Machine Images (AMIs) (4) Info

□	Name	AMI name	AMI ID
□		3tier-architecture-global-AMI	ami-04bfff03402f3e4df
□		maven AMI	ami-0a6790c73f49fa476
□		nginx AMI	ami-020bb8e29a4ec10a7
□		tomcat AMI	ami-048cec96fe7de9103

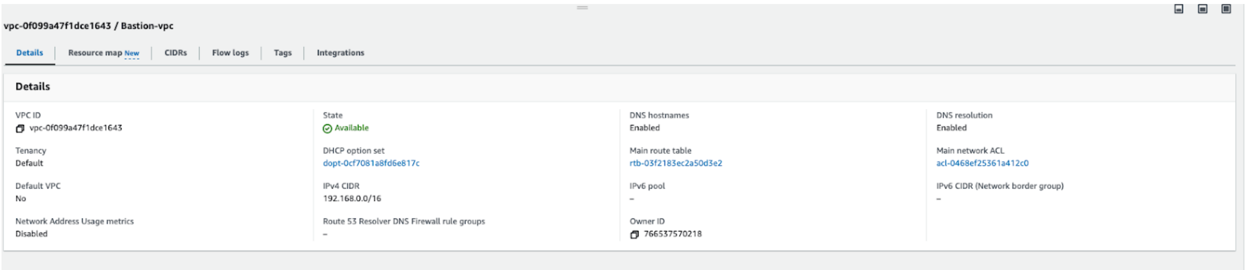
VPC Deployment

1. VPC (Network Setup)
    a. **Build VPC network (192.168.0.0/16) for Bastion Host deployment as per the architecture shown above.**
    b. **Build VPC network (172.32.0.0/16) for deploying Highly Available and Auto Scalable application servers as per the architecture shown above.**
    c. **Create NAT Gateway in Public Subnet and update Private Subnet associated Route Table accordingly to route the default traffic to NAT for outbound internet connection.**
    d. **Create Transit Gateway and associate both VPCs to the Transit Gateway for private communication.**
    e. **Create Internet Gateway for each VPC and update Public Subnet associated Route Table accordingly to route the default traffic to IGW for inbound/outbound internet connection.**

2. Bastion
   a. **Deploy Bastion Host in the Public Subnet with EIP associated.**
   b. **Create Security Group allowing port 22 from public internet**

Infrastructure Solution

Bastion VPC:



vpc-0f099a47f1dce1643 / Bastion-vpc

Details

VPC ID
vpc-0f099a47f1dce1643

State
Available

DNS hostnames
Enabled

DNS resolution
Enabled

Tenancy
Default

DHCP option set
dopt-0cf7081a8fd6e817c

Main route table
rtb-03f2183ec2a50d3e2

Main network ACL
acl-0468ef25361a412c0

Default VPC
No

IPv4 CIDR
192.168.0.0/16

IPv6 pool
–

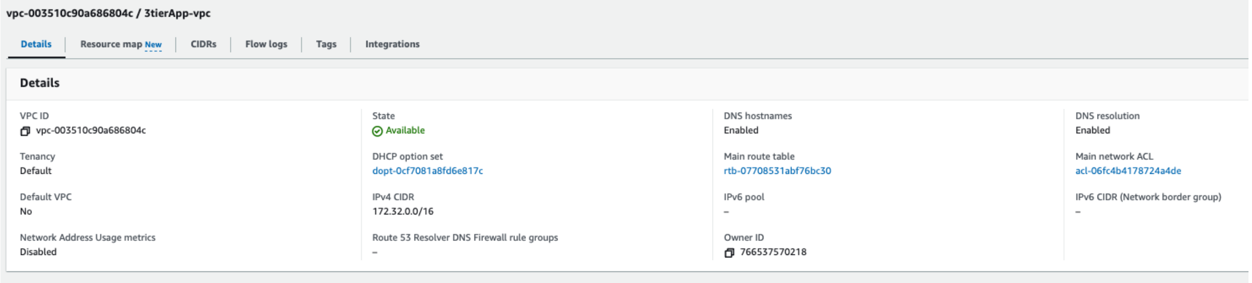
IPv6 CIDR (Network border group)
–

Network Address Usage metrics
Disabled

Route 53 Resolver DNS Firewall rule groups
–

Owner ID
766537570218

3tierApp VPC:



vpc-003510c90a686804c / 3tierApp-vpc

Details

VPC ID
vpc-003510c90a686804c

State
Available

DNS hostnames
Enabled

DNS resolution
Enabled

Tenancy
Default

DHCP option set
dopt-0cf7081a8fd6e817c

Main route table
rtb-07708531abf76bc30

Main network ACL
acl-06fc4b4178724a4de

Default VPC
No

IPv4 CIDR
172.32.0.0/16

IPv6 pool
–

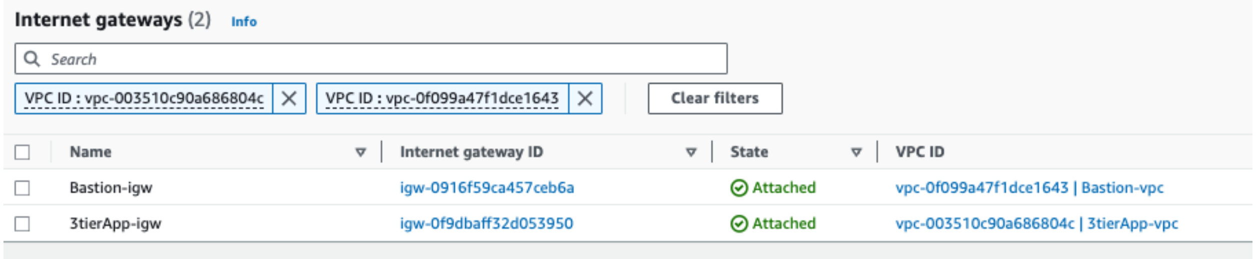
IPv6 CIDR (Network border group)
–

Network Address Usage metrics
Disabled

Route 53 Resolver DNS Firewall rule groups
–

Owner ID
766537570218

We need to create 2 IG for Bastion and 3tierApp VPC

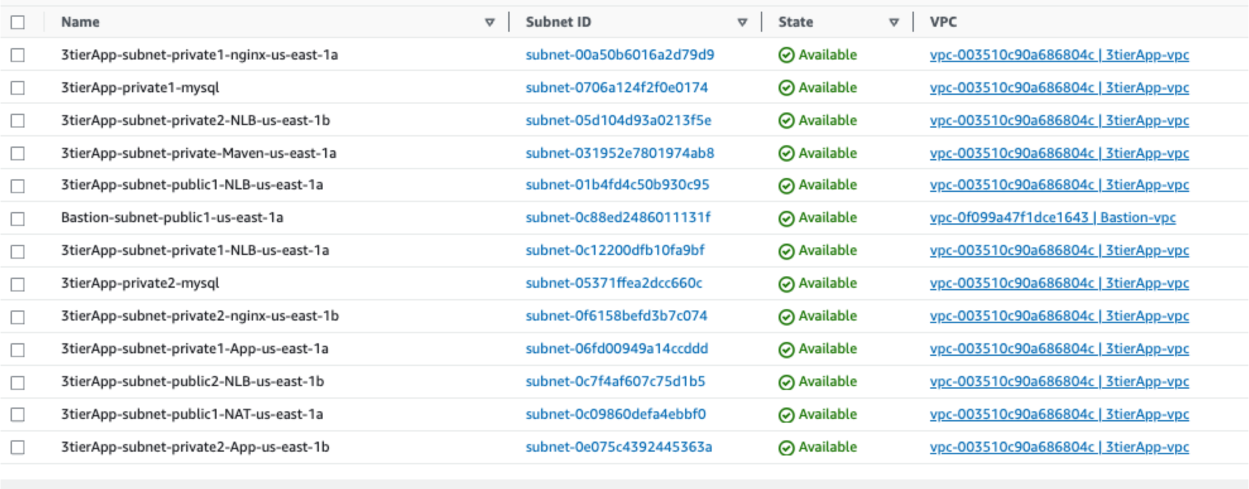


Internet gateways (2) Info

Search
VPC ID : vpc-003510c90a686804c
VPC ID : vpc-0f099a47f1dce1643
Clear filters

□	Name	Internet gateway ID	State	VPC ID
□	Bastion-igw	igw-0916f59ca457ceb6a	✓ Attached	vpc-0f099a47f1dce1643 | Bastion-vpc
□	3tierApp-igw	igw-0f9dbaff32d053950	✓ Attached	vpc-003510c90a686804c | 3tierApp-vpc

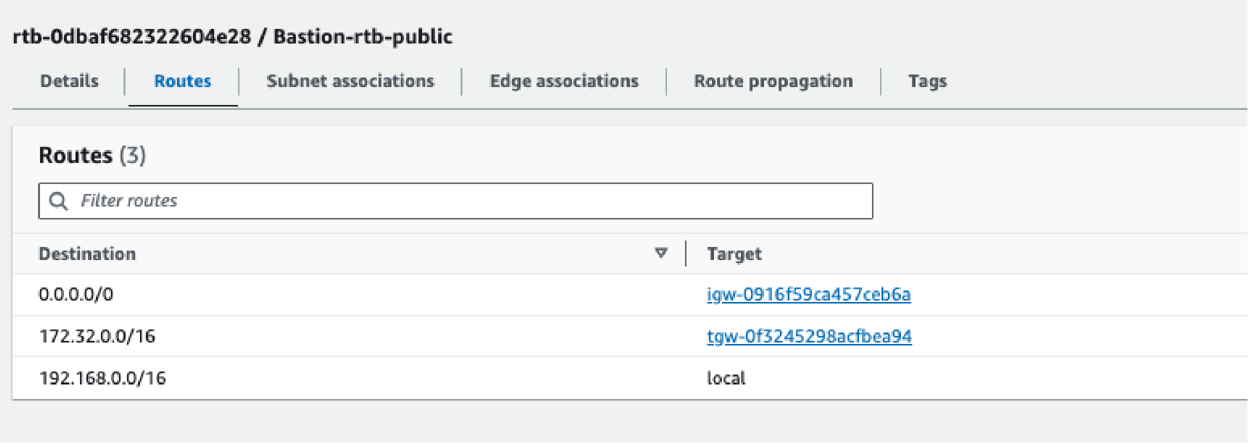
- 1 public subnet is create for the Bastion VPC because the purpose of the Bastion VPC is to accept connection from internet. Once a user is connected to the bastion, the user will connect to 3tierApp VPC via transit gateway
- 3tierApp VPC will have several privates and publics VPC



□	Name	Subnet ID	State	VPC
□	3tierApp-subnet-private1-nginx-us-east-1a	subnet-00a50b6016a2d79d9	✓ Available	vpc-003510c90a686804c | 3tierApp-vpc
□	3tierApp-private1-mysql	subnet-0706a124f2f0e0174	✓ Available	vpc-003510c90a686804c | 3tierApp-vpc
□	3tierApp-subnet-private2-NLB-us-east-1b	subnet-05d104d93a0213f5e	✓ Available	vpc-003510c90a686804c | 3tierApp-vpc
□	3tierApp-subnet-private-Maven-us-east-1a	subnet-031952e7801974ab8	✓ Available	vpc-003510c90a686804c | 3tierApp-vpc
□	3tierApp-subnet-public1-NLB-us-east-1a	subnet-01b4fd4c50b930c95	✓ Available	vpc-003510c90a686804c | 3tierApp-vpc
□	Bastion-subnet-public1-us-east-1a	subnet-0c88ed2486011131f	✓ Available	vpc-0f099a47f1dce1643 | Bastion-vpc
□	3tierApp-subnet-private1-NLB-us-east-1a	subnet-0c12200dfb10fa9bf	✓ Available	vpc-003510c90a686804c | 3tierApp-vpc
□	3tierApp-private2-mysql	subnet-05371ffea2dcc660c	✓ Available	vpc-003510c90a686804c | 3tierApp-vpc
□	3tierApp-subnet-private2-nginx-us-east-1b	subnet-0f6158befd3b7c074	✓ Available	vpc-003510c90a686804c | 3tierApp-vpc
□	3tierApp-subnet-private1-App-us-east-1a	subnet-06fd00949a14ccddd	✓ Available	vpc-003510c90a686804c | 3tierApp-vpc
□	3tierApp-subnet-public2-NLB-us-east-1b	subnet-0c7f4af607c75d1b5	✓ Available	vpc-003510c90a686804c | 3tierApp-vpc
□	3tierApp-subnet-public1-NAT-us-east-1a	subnet-0c09860defa4ebbf0	✓ Available	vpc-003510c90a686804c | 3tierApp-vpc
□	3tierApp-subnet-private2-App-us-east-1b	subnet-0e075c4392445363a	✓ Available	vpc-003510c90a686804c | 3tierApp-vpc

- 1 public route table for bastion VPC
- 2 route table for 3tierApp VPC: private and public

Bastion route table:



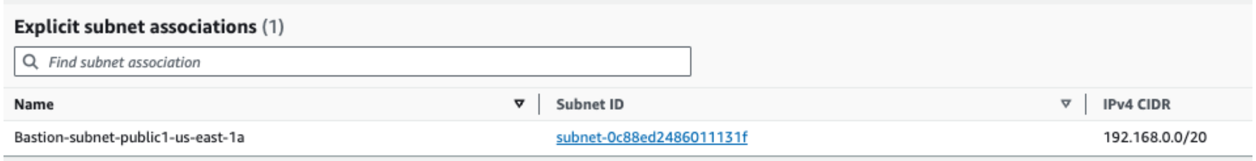
rtb-0dbaf682322604e28 / Bastion-rtb-public

Routes (3)

Destination	Target
0.0.0.0/0	igw-0916f59ca457ceb6a
172.32.0.0/16	tgw-0f3245298acfbea94
192.168.0.0/16	local

Bastion route table should include the transit gateway associated to the 3tierApp CIDR
Bastion subnet is a public subnet, so we associate the IG to 0.0.0.0/0

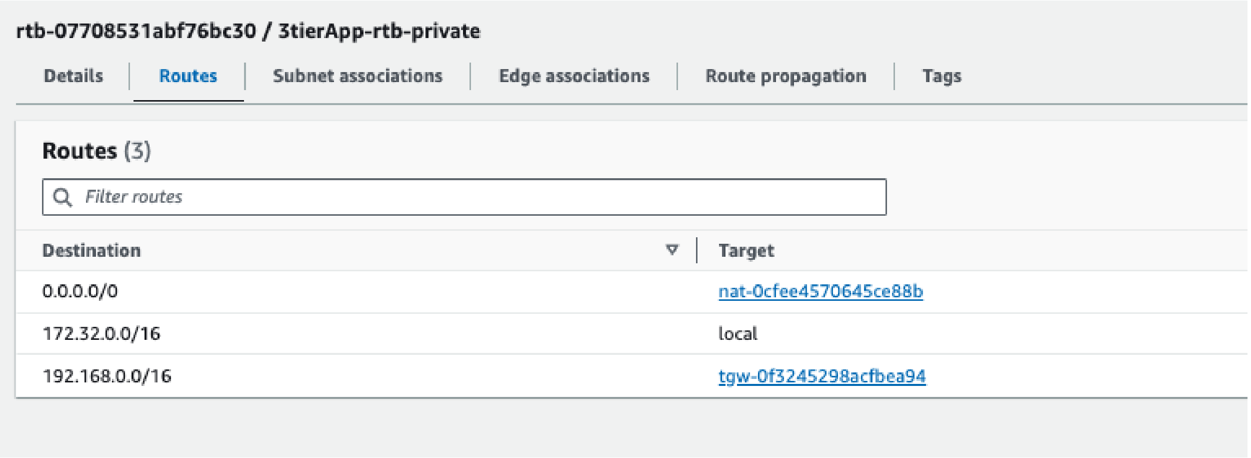
Bastion route table subnet associations:



Explicit subnet associations (1)

Name	Subnet ID	IPv4 CIDR
Bastion-subnet-public1-us-east-1a	subnet-0c88ed2486011131f	192.168.0.0/20

3tierApp private route table:



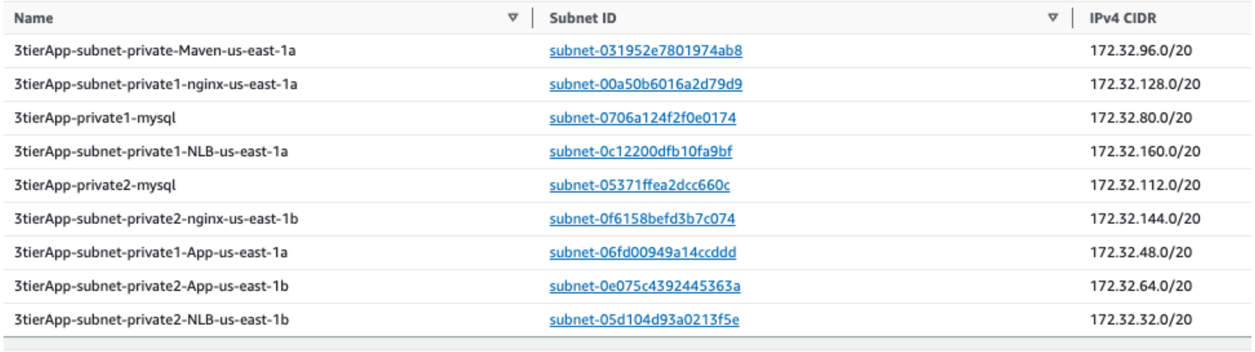
rtb-07708531abf76bc30 / 3tierApp-rtb-private

Routes (3)

Destination	Target
0.0.0.0/0	nat-0cfee4570645ce88b
172.32.0.0/16	local
192.168.0.0/16	tgw-0f3245298acfbea94

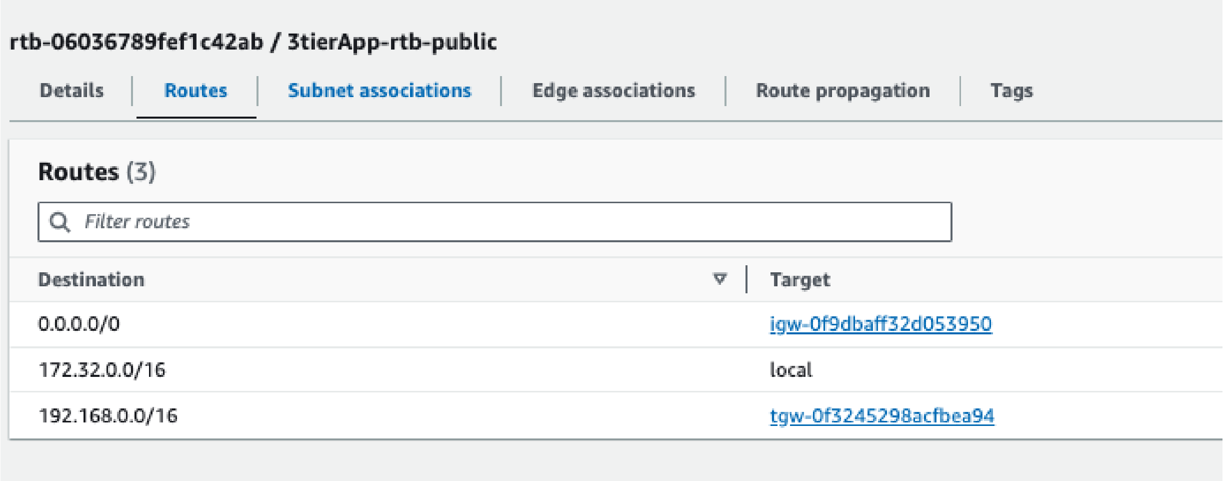
The private routetable should associate the transit gateway with the Bastion VPC CIDR
The NAT gateway should be associated to 0.0.0.0/0 because this routetable is associated with a private subnet. Private subnets cannot access directly to internet. External resources cannot access to private subnets, but resources inside the private subnets can access internet to make update for example.

3tierApp private route table subnet associations:



Name	Subnet ID	IPv4 CIDR
3tierApp-subnet-private-Maven-us-east-1a	subnet-031952e7801974ab8	172.32.96.0/20
3tierApp-subnet-private1-nginx-us-east-1a	subnet-00a50b6016a2d79d9	172.32.128.0/20
3tierApp-private1-mysql	subnet-0706a124f2f0e0174	172.32.80.0/20
3tierApp-subnet-private1-NLB-us-east-1a	subnet-0c12200dfb10fa9bf	172.32.160.0/20
3tierApp-private2-mysql	subnet-05371ffea2dcc660c	172.32.112.0/20
3tierApp-subnet-private2-nginx-us-east-1b	subnet-0f6158befd3b7c074	172.32.144.0/20
3tierApp-subnet-private1-App-us-east-1a	subnet-06fd00949a14ccddd	172.32.48.0/20
3tierApp-subnet-private2-App-us-east-1b	subnet-0e075c4392445363a	172.32.64.0/20
3tierApp-subnet-private2-NLB-us-east-1b	subnet-05d104d93a0213f5e	172.32.32.0/20

3tierApp public routetable



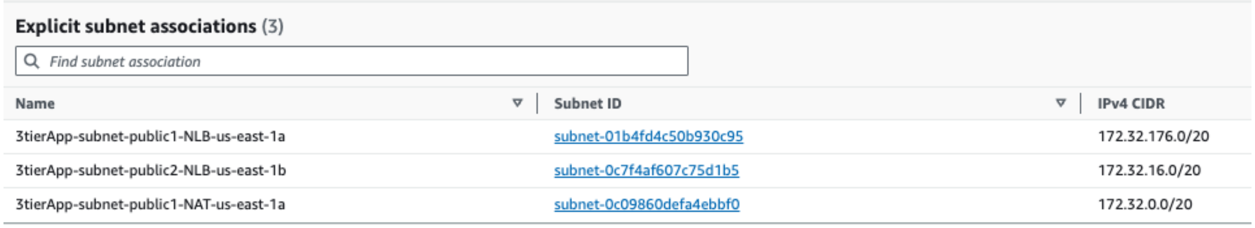
rtb-06036789fef1c42ab / 3tierApp-rtb-public

Routes (3)

Destination	Target
0.0.0.0/0	igw-0f9dbaff32d053950
172.32.0.0/16	local
192.168.0.0/16	tgw-0f3245298acfbea94

Public route table associate transit gateway with the Bastion VPC CIDR.

3tierApp public route table subnets association:

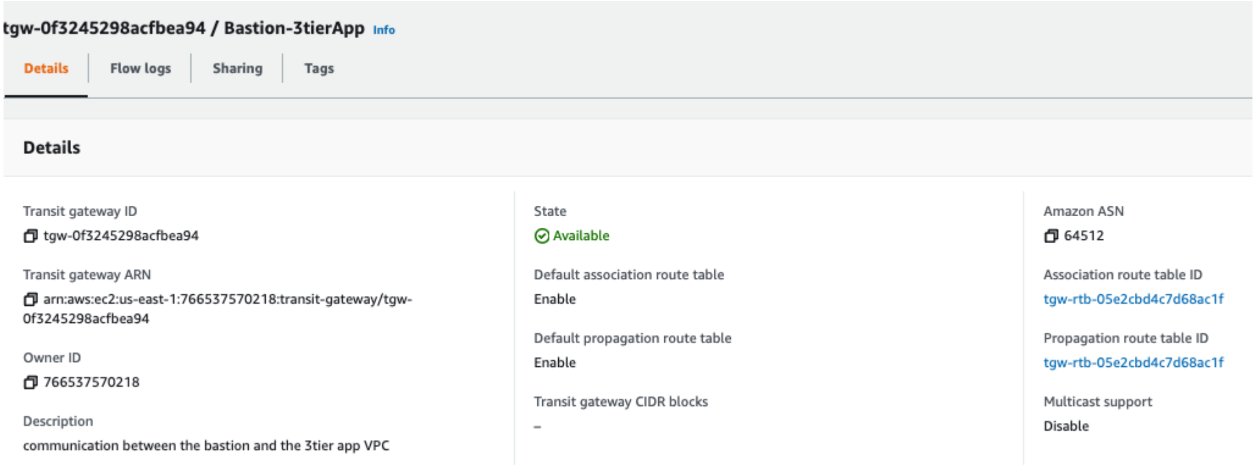


Explicit subnet associations (3)

Find subnet association

Name	Subnet ID	IPv4 CIDR
3tierApp-subnet-public1-NLB-us-east-1a	subnet-01b4fd4c50b930c95	172.32.176.0/20
3tierApp-subnet-public2-NLB-us-east-1b	subnet-0c7f4af607c75d1b5	172.32.16.0/20
3tierApp-subnet-public1-NAT-us-east-1a	subnet-0c09860defa4ebbf0	172.32.0.0/20

Transit gateway provides a hub to connect VPC and on-premise network to VPC



tgw-0f3245298acfbea94 / Bastion-3tierApp Info

Details

Transit gateway ID
tgw-0f3245298acfbea94

Transit gateway ARN
arn:aws:ec2:us-east-1:766537570218:transit-gateway/tgw-0f3245298acfbea94

Owner ID
766537570218

Description
communication between the bastion and the 3tier app VPC

State
✓ Available

Default association route table
Enable

Default propagation route table
Enable

Transit gateway CIDR blocks
–

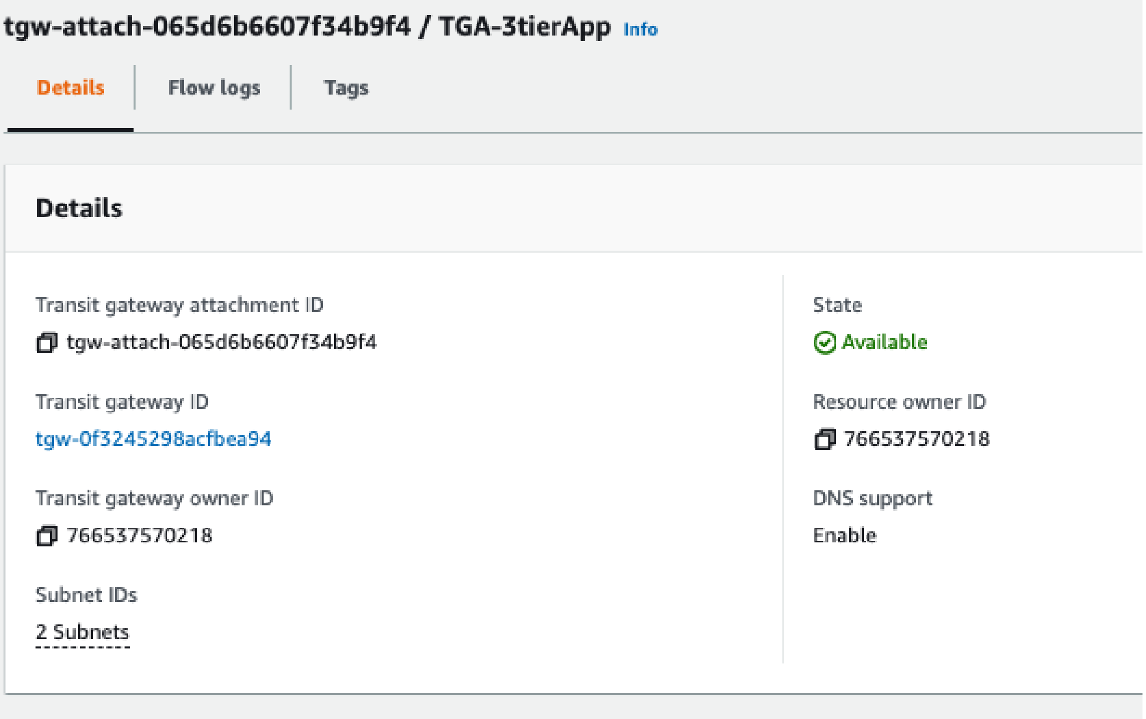
Amazon ASN
64512

Association route table ID
tgw-rtb-05e2cbd4c7d68ac1f

Propagation route table ID
tgw-rtb-05e2cbd4c7d68ac1f

Multicast support
Disable

Then we will attach 2 transit gateway attachment to link bastion and 3tierApp VPC:



tgw-attach-065d6b6607f34b9f4 / TGA-3tierApp

Details

Transit gateway attachment ID
tgw-attach-065d6b6607f34b9f4

Transit gateway ID
tgw-0f3245298acfbea94

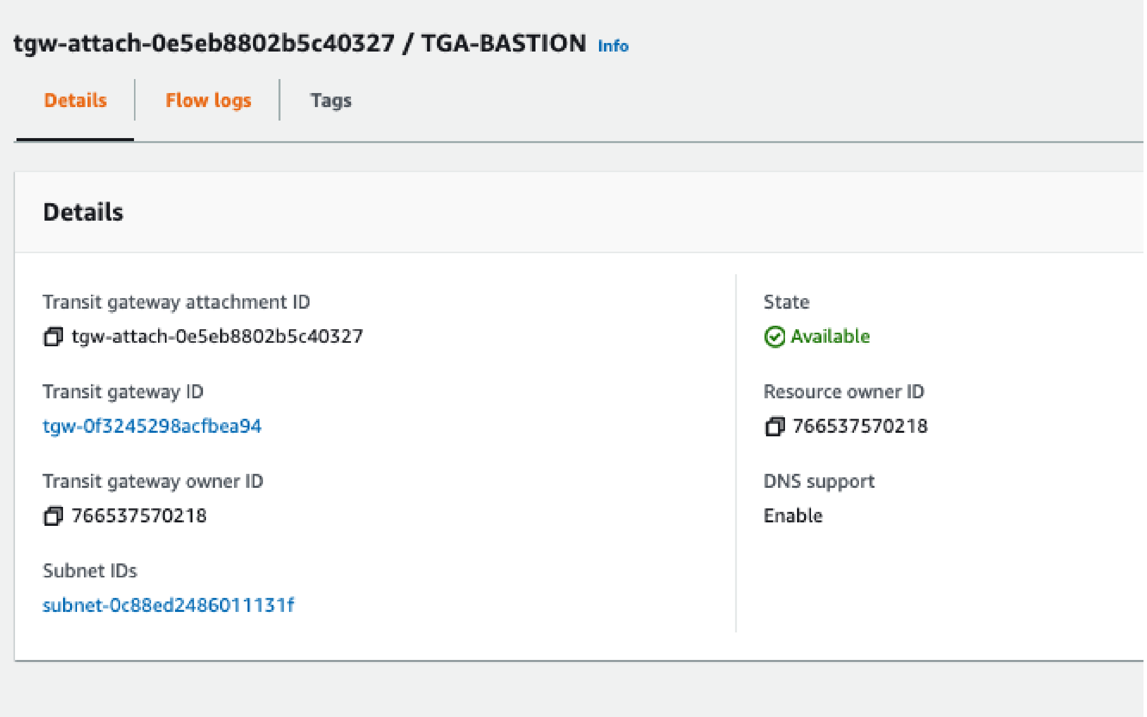
Transit gateway owner ID
766537570218

Subnet IDs
2 Subnets

State
Available

Resource owner ID
766537570218

DNS support
Enable



tgw-attach-0e5eb8802b5c40327 / TGA-BASTION

Details

Transit gateway attachment ID
tgw-attach-0e5eb8802b5c40327

Transit gateway ID
tgw-0f3245298acfbea94

Transit gateway owner ID
766537570218

Subnet IDs
subnet-0c88ed2486011131f

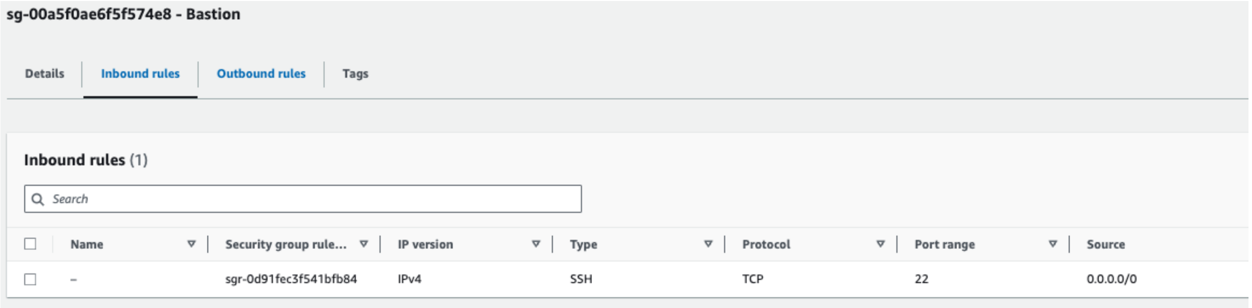
State
Available

Resource owner ID
766537570218

DNS support
Enable

We also configure SG:

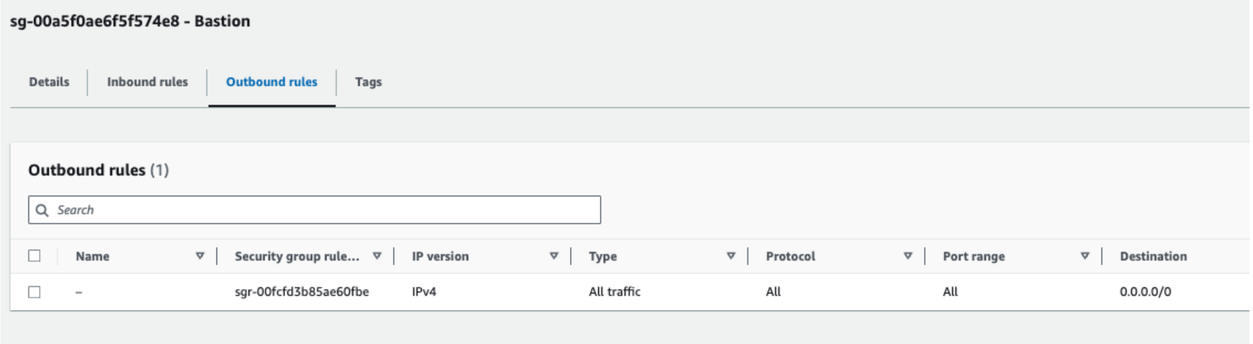
Bastion SG:



sg-00a5f0ae6f5f574e8 - Bastion

Inbound rules (1)

□	Name	Security group rule...	IP version	Type	Protocol	Port range	Source
□	–	sgr-0d91fec3f541bfb84	IPv4	SSH	TCP	22	0.0.0.0/0

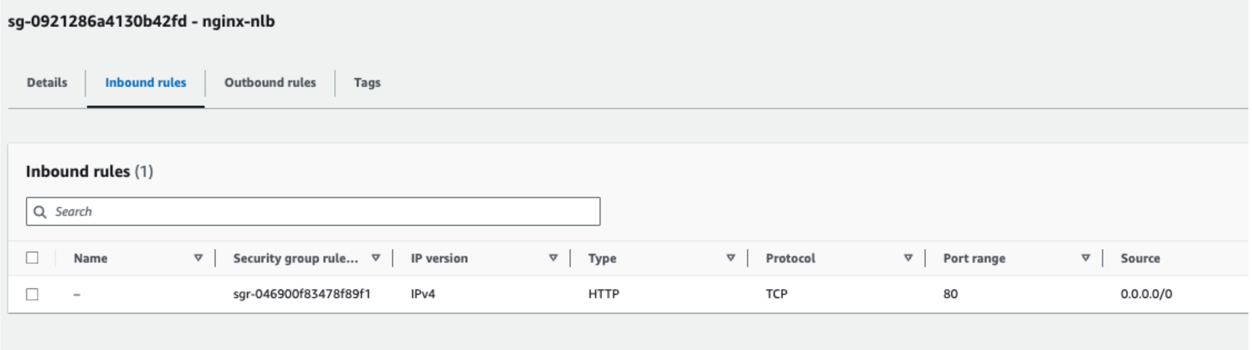


sg-00a5f0ae6f5f574e8 - Bastion

Outbound rules (1)

□	Name	Security group rule...	IP version	Type	Protocol	Port range	Destination
□	–	sgr-00fcfd3b85ae60fbe	IPv4	All traffic	All	All	0.0.0.0/0

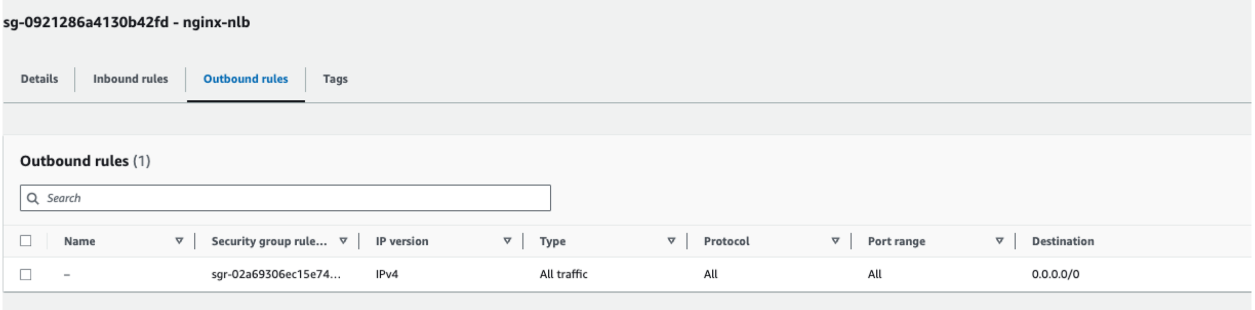
Nginx NLB SG:



sg-0921286a4130b42fd - nginx-nlb

Inbound rules (1)

□	Name	Security group rule...	IP version	Type	Protocol	Port range	Source
□	–	sgr-046900f83478f89f1	IPv4	HTTP	TCP	80	0.0.0.0/0

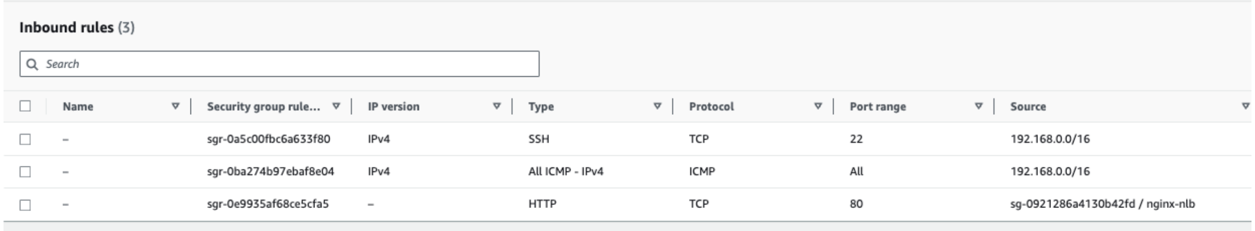


sg-0921286a4130b42fd - nginx-nlb

Outbound rules (1)

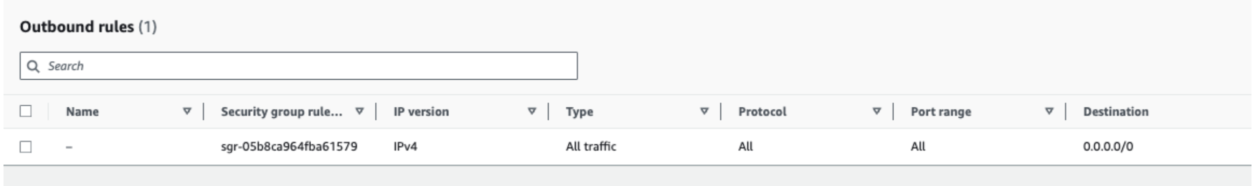
□	Name	Security group rule...	IP version	Type	Protocol	Port range	Destination
□	–	sgr-02a69306ec15e74...	IPv4	All traffic	All	All	0.0.0.0/0

Nginx EC2 SG:



Inbound rules (3)

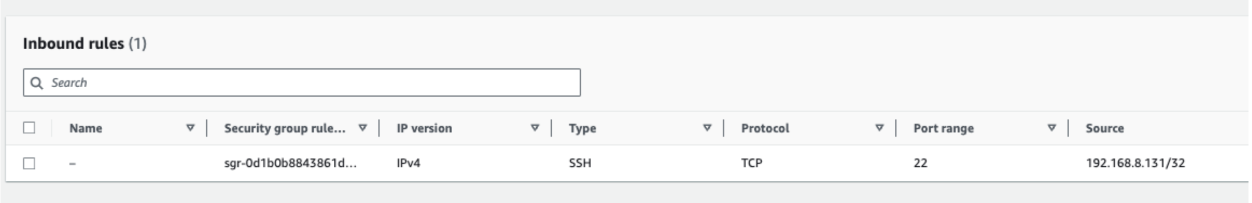
☐	Name	Security group rule…	IP version	Type	Protocol	Port range	Source
☐	–	sgr-0a5c00fbc6a633f80	IPv4	SSH	TCP	22	192.168.0.0/16
☐	–	sgr-0ba274b97ebaf8e04	IPv4	All ICMP - IPv4	ICMP	All	192.168.0.0/16
☐	–	sgr-0e9935af68ce5cfa5	–	HTTP	TCP	80	sg-0921286a4130b42fd / nginx-nlb



Outbound rules (1)

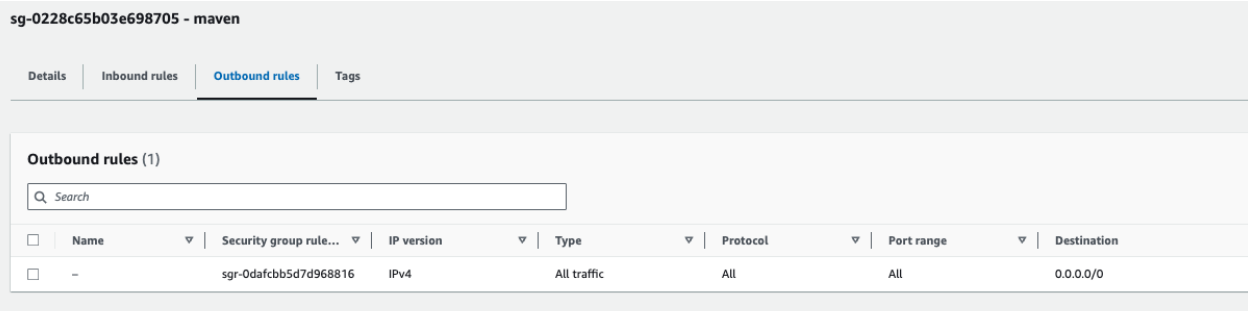
☐	Name	Security group rule…	IP version	Type	Protocol	Port range	Destination
☐	–	sgr-05b8ca964fba61579	IPv4	All traffic	All	All	0.0.0.0/0

Maven SG:



Inbound rules (1)

☐	Name	Security group rule…	IP version	Type	Protocol	Port range	Source
☐	–	sgr-0d1b0b8843861d…	IPv4	SSH	TCP	22	192.168.8.131/32

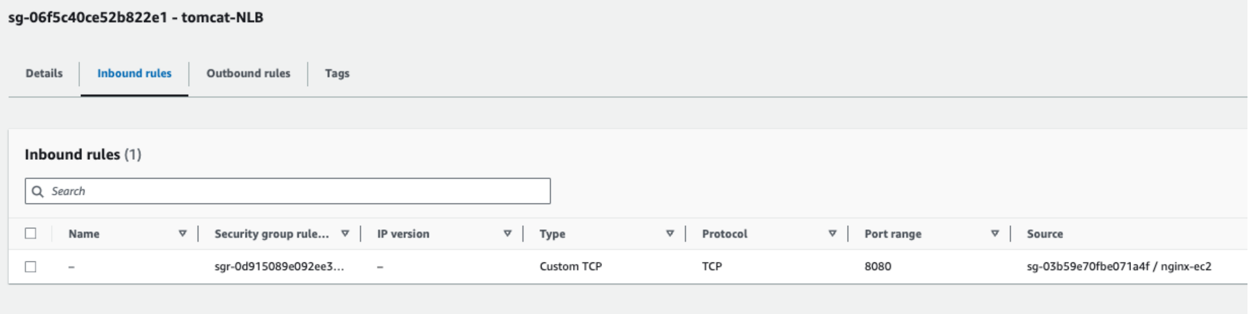


sg-0228c65b03e698705 - maven

Outbound rules (1)

☐	Name	Security group rule…	IP version	Type	Protocol	Port range	Destination
☐	–	sgr-0dafcbb5d7d968816	IPv4	All traffic	All	All	0.0.0.0/0

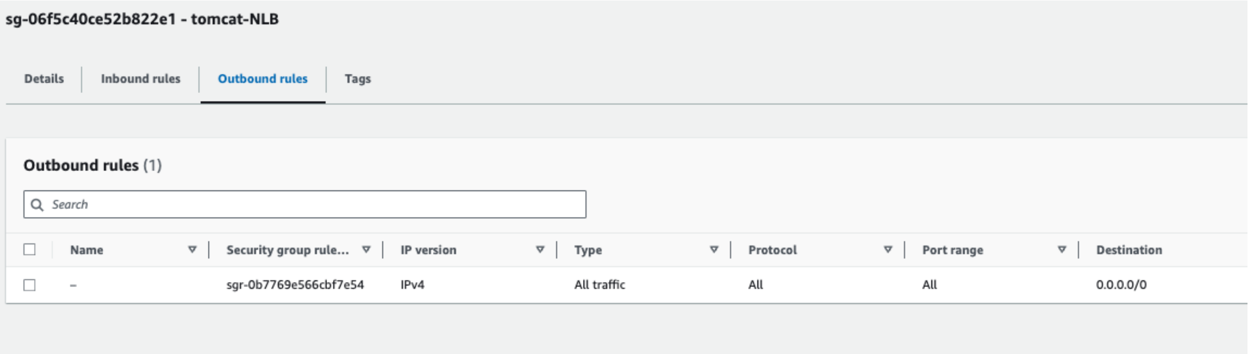
Tomcat NLB SG:



sg-06f5c40ce52b822e1 - tomcat-NLB

Inbound rules (1)

□	Name	Security group rule...	IP version	Type	Protocol	Port range	Source
□	–	sgr-0d915089e092ee3...	–	Custom TCP	TCP	8080	sg-03b59e70fbe071a4f / nginx-ec2

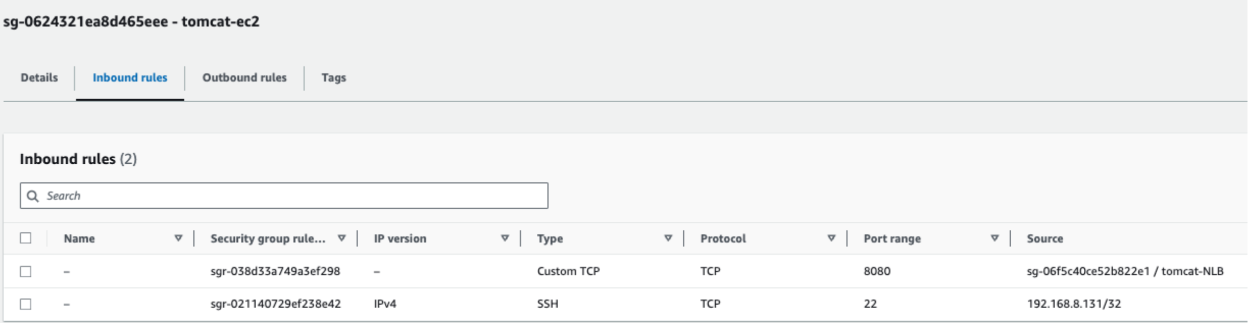


sg-06f5c40ce52b822e1 - tomcat-NLB

Outbound rules (1)

□	Name	Security group rule...	IP version	Type	Protocol	Port range	Destination
□	–	sgr-0b7769e566cbf7e54	IPv4	All traffic	All	All	0.0.0.0/0

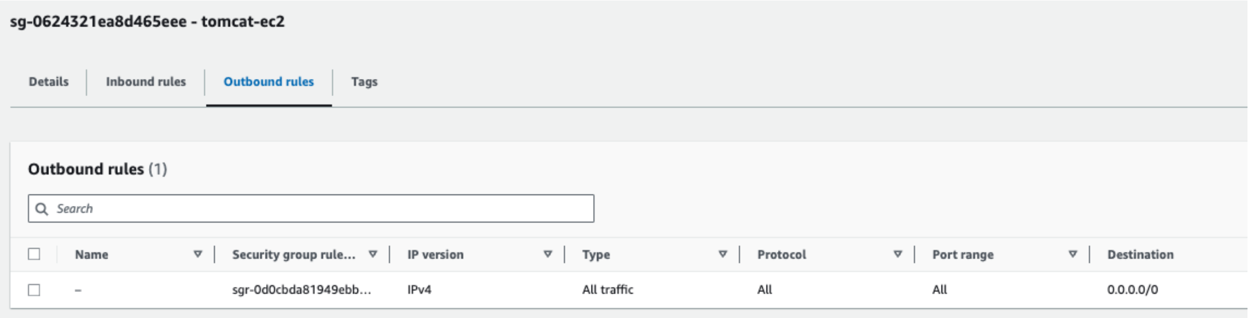
Tomcat EC2 SG:



sg-0624321ea8d465eee - tomcat-ec2

Inbound rules (2)

□	Name	Security group rule...	IP version	Type	Protocol	Port range	Source
□	–	sgr-038d33a749a3ef298	–	Custom TCP	TCP	8080	sg-06f5c40ce52b822e1 / tomcat-NLB
□	–	sgr-021140729ef238e42	IPv4	SSH	TCP	22	192.168.8.131/32

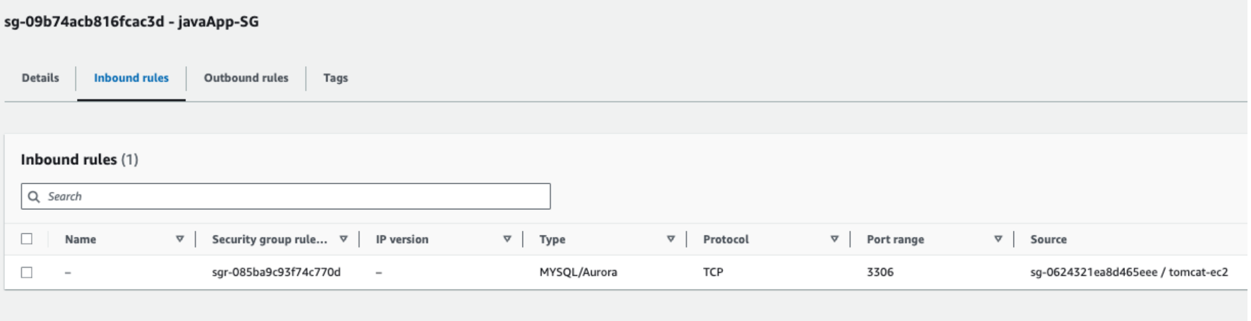


sg-0624321ea8d465eee - tomcat-ec2

Outbound rules (1)

□	Name	Security group rule...	IP version	Type	Protocol	Port range	Destination
□	–	sgr-0d0cbda81949ebb...	IPv4	All traffic	All	All	0.0.0.0/0

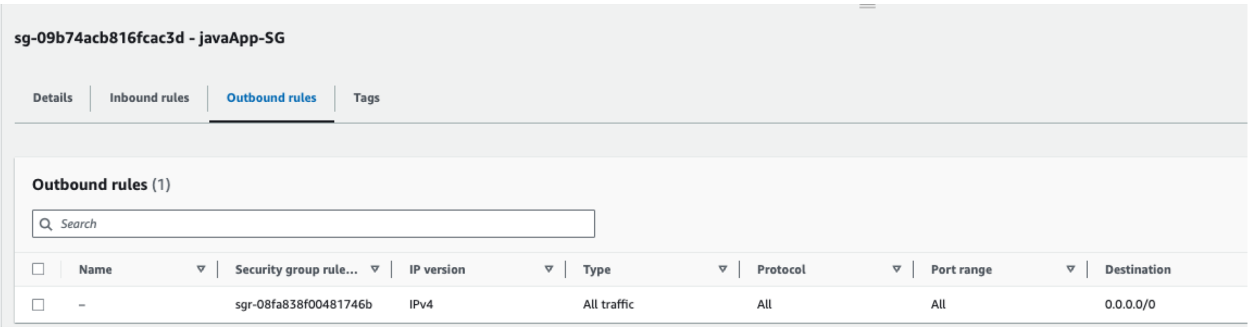
RDS mysql SG:



sg-09b74acb816fcac3d - javaApp-SG

Inbound rules (1)

☐	Name	Security group rule...	IP version	Type	Protocol	Port range	Source
☐	–	sgr-085ba9c93f74c770d	–	MYSQL/Aurora	TCP	3306	sg-0624321ea8d465eee / tomcat-ec2



sg-09b74acb816fcac3d - javaApp-SG

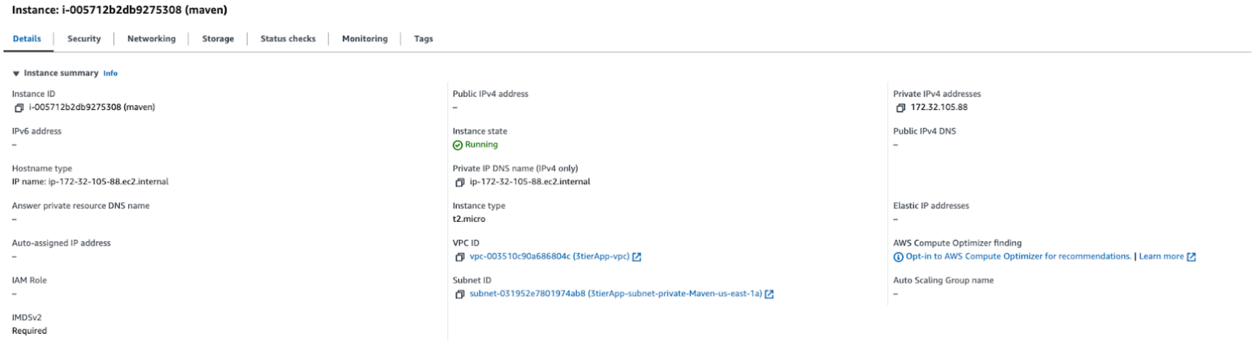
Outbound rules (1)

☐	Name	Security group rule...	IP version	Type	Protocol	Port range	Destination
☐	–	sgr-08fa838f00481746b	IPv4	All traffic	All	All	0.0.0.0/0

Maven Build

1. Create EC2 instance using Maven Golden AMI

Maven instance is launched in 3tierApp VPC, in a private subnet:



Instance: i-005712b2db9275308 (maven)

Instance ID
i-005712b2db9275308 (maven)

IPv6 address
–

Hostname type
IP name: ip-172-32-105-88.ec2.internal

Answer private resource DNS name
–

Auto-assigned IP address
–

IAM Role
–

IMDSv2
Required

Public IPv4 address
–

Instance state
Running

Private IP DNS name (IPv4 only)
ip-172-32-105-88.ec2.internal

Instance type
t2.micro

VPC ID
vpc-003510c90a686804c (3tierApp-vpc)

Subnet ID
subnet-031952e7801974ab8 (3tierApp-subnet-private-Maven-us-east-1a)

Private IPv4 addresses
172.32.105.88

Public IPv4 DNS
–

Elastic IP addresses
–

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more

Auto Scaling Group name
–

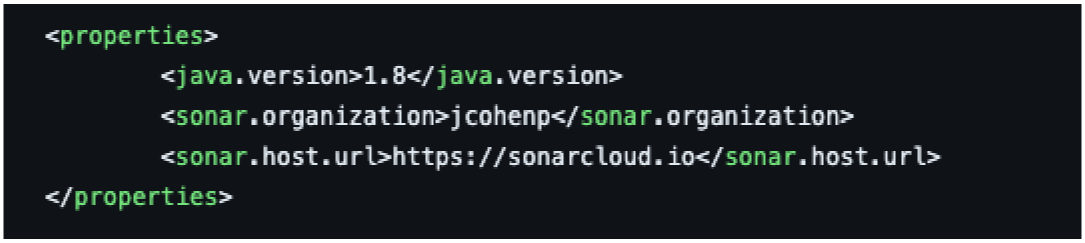
2. Clone Bitbucket repository to VSCode and update the pom.xml with Sonar and JFROG deployment details.

I forked instructor’s repo and cloned from my bitbucket repo

```
# git clone remote_url && cd java-login-app
# git branch feature
# git checkout feature
```

For sonarcloud integration

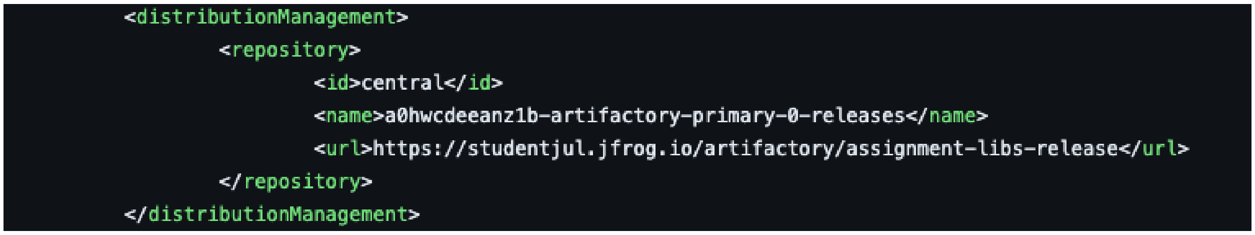
- create an organization and a project in sonar cloud account.
- After which, instructions are provided for integration. Execute them on maven ec2 instance.
- Amongst other instructions this includes updating the pom.xml with organization name and sonar host url as shown below



```
<properties>
        <java.version>1.8</java.version>
        <sonar.organization>jcohenp</sonar.organization>
        <sonar.host.url>https://sonarcloud.io</sonar.host.url>
</properties>
```

For jfrog integration:

- First create a repository on jfrog.
- Afterwards use the ‘Quick Setup” option to generate deployment configuration.
- Click 'set me up' for your ‘local’ type repo. I=In this case, local repo is named ‘assignment-libs-release-local’ .
- click "deploy" tab on jfrog Web UI. This generates configuration to use at maven to upload generated artifact to jfrog local respository.
- Afterwards update the pom.xml file with generated distributionManagement config.



```
    <distributionManagement>
            <repository>
                    <id>central</id>
                    <name>a0hwcdeeanz1b-artifactory-primary-0-releases</name>
                    <url>https://studentjul.jfrog.io/artifactory/assignment-libs-release</url>
            </repository>
    </distributionManagement>
```

3. Add settings.xml file to the root folder of the repository with the JFROG credentials and JFROG repo to resolve the dependencies.

- To generate settings.xml, use the ‘Quick Setup” option in jfrog
- Select ‘default-maven-virtual’ repo for downloading dependencies
- Click 'configure' using ‘default-maven-virtual’ repo
- A settings configuration for maven to connect to jfrog and download dependencies is auto-generated
- Place configuration in /root/.m2/settings.xml file on maven instance Settings.xml file should include credentials and reference to default-mavenvirtual jfrog repo.

4. Update application.properties file with JDBC connection string to authenticate with MySQL.
5. Push the code changes to feature branch of Bitbucket repository

Push all changes in the feature branch

6. Raise Pull Request to approve the PR and Merge the changes to Master branch.

Accept the PR and merge with master

7. Login to EC2 instance and clone the Bitbucket repository

```
# git clone remote_repo_url && cd java-login-app
```

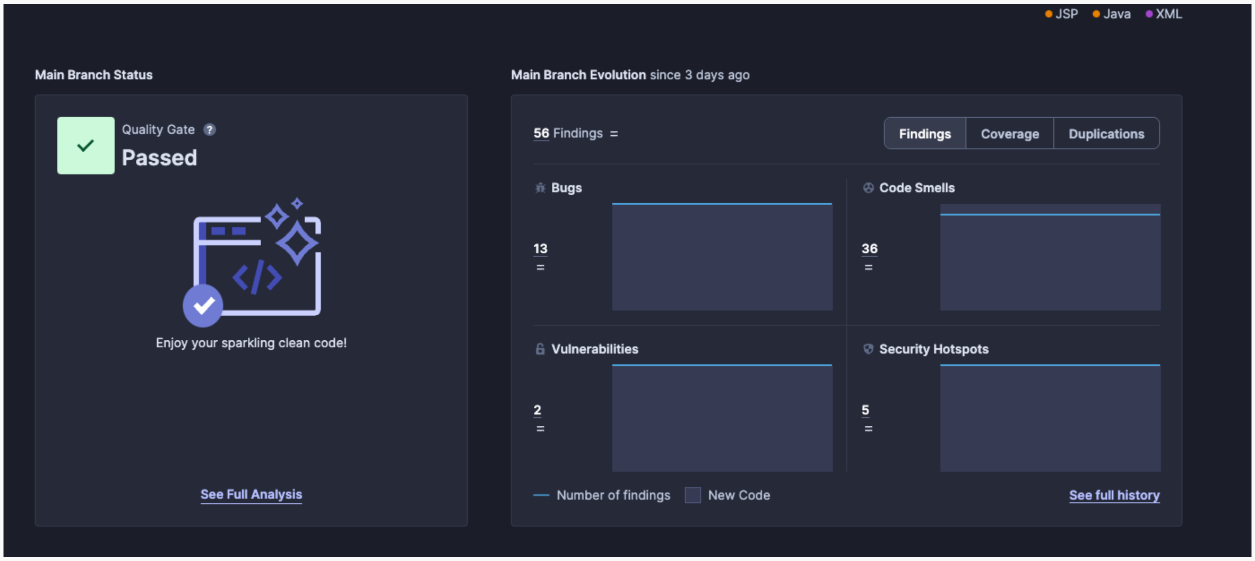
8. Build the source code using maven arguments “-s settings.xml”

```
# mvn -s ~/.m2/settings.xml deploy
```

9. Integrate Maven build with Sonar Cloud and generate analysis dashboard with default Quality Gate profile.

Export environment variable and run mvn verify command

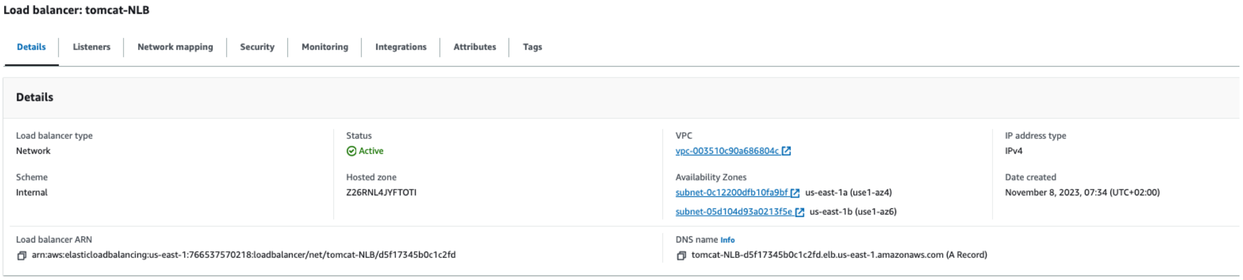
```
# export SONAR_TOKEN=xxxxxxxxx
# mvn verify org.sonarsource.scanner.maven:sonar-maven-plugin:sonar -Dsonar.projectKey=assignment5
```



Tomcat Backend

1. Create private facing Network Load Balancer and Target Group.

The tomcat NLB is on the 3tierApp VPC on private subnet.



Load balancer: tomcat-NLB

Details

Load balancer type
Network

Status
Active

VPC
vpc-003510c90a686804c

IP address type
IPv4

Scheme
Internal

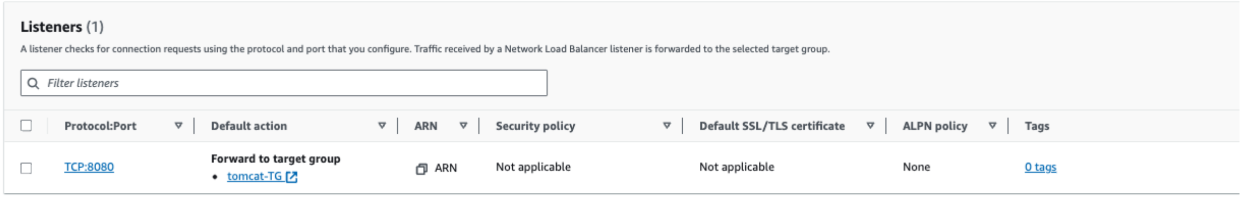
Hosted zone
Z26RNL4JYFTOTI

Availability Zones
subnet-0c12200dfb10fa9bf us-east-1a (use1-az4)
subnet-05d104d93a0213f5e us-east-1b (use1-az6)

Date created
November 8, 2023, 07:34 (UTC+02:00)

Load balancer ARN
arn:aws:elasticloadbalancing:us-east-1:766537570218:loadbalancer/net/tomcat-NLB/d5f17345b0c1c2fd

DNS name Info
tomcat-NLB-d5f17345b0c1c2fd.elb.us-east-1.amazonaws.com (A Record)

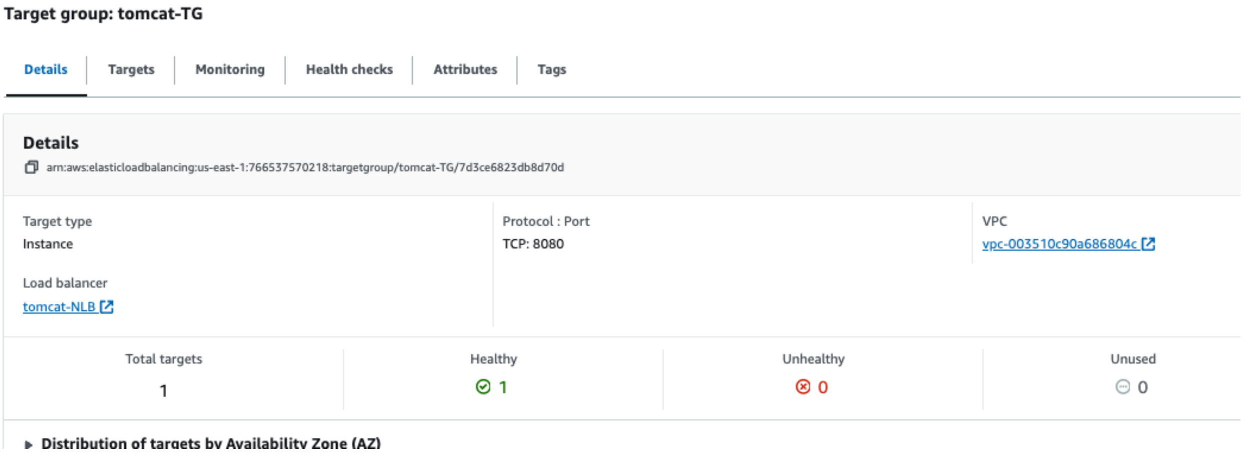


Listeners (1)

A listener checks for connection requests using the protocol and port that you configure. Traffic received by a Network Load Balancer listener is forwarded to the selected target group.

□	Protocol:Port	Default action	ARN	Security policy	Default SSL/TLS certificate	ALPN policy	Tags
□	TCP:8080	Forward to target group • tomcat-TG	ARN	Not applicable	Not applicable	None	0 tags

App target group:



Target group: tomcat-TG

Details

arn:aws:elasticloadbalancing:us-east-1:766537570218:targetgroup/tomcat-TG/7d3ce6823db8d70d

Target type
Instance

Protocol : Port
TCP: 8080

VPC
vpc-003510c90a686804c

Load balancer
tomcat-NLB

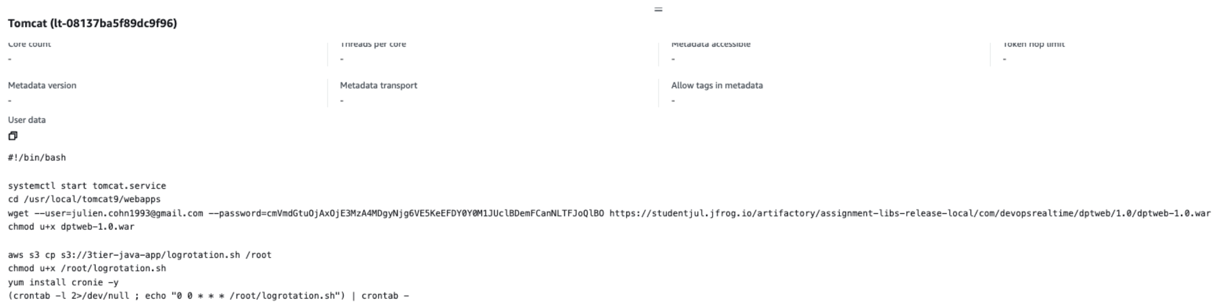
Total targets	Healthy	Unhealthy	Unused
1	1	0	0

▸ Distribution of targets by Availability Zone (AZ)

2. Create Launch Configuration with below configuration.
    1. Tomcat Golden AMI
    2. User Data to deploy .war artifact from JFROG into webapps folder.
    3. Security Group allowing Port 22 from Bastion Host and Port 8080 from private NLB.

When creating the launch template we need to specify:

- the Tomcat Golden AMI
- Keypair
- Security group
- User data
- Role to give permission to the EC2 machine to access S3, cloudwatch and Session Manager



Tomcat (lt-08137ba5f89dc9f96)

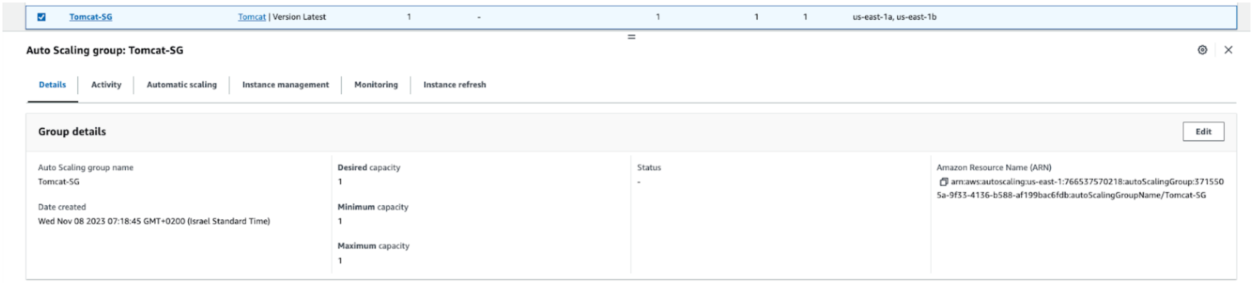
Core count	Threads per core	Metadata accessible	Token hop limit
-	-	-	-
Metadata version	Metadata transport	Allow tags in metadata	
-	-	-	

User data

```
#!/bin/bash

systemctl start tomcat.service
cd /usr/local/tomcat9/webapps
wget --user=julien.cohn1993@gmail.com --password=cmVmdGtuOjAxOjE3MzA4MDgyNjg6VE5KeEFDY0Y0M1JUclBDemFCanNLTFJoQlBO https://studentjul.jfrog.io/artifactory/assignment-libs-release-local/com/devopsrealtime/dptweb/1.0/dptweb-1.0.war
chmod u+x dptweb-1.0.war

aws s3 cp s3://3tier-java-app/logrotation.sh /root
chmod u+x /root/logrotation.sh
yum install cronie -y
(crontab -l 2>/dev/null ; echo "0 0 * * * /root/logrotation.sh") | crontab -
```

3. Create Auto Scaling Group

Create an ASG for the tomcat that defines the number of ec2 that will be run.
Set the Tracking scaling policy that will scale the load if the cpu utilization is too high.



☑	Tomcat-SG	Tomcat | Version Latest	1	-	1	1	1	us-east-1a, us-east-1b

Auto Scaling group: Tomcat-SG

Group details

Auto Scaling group name
Tomcat-SG

Date created
Wed Nov 08 2023 07:18:45 GMT+0200 (Israel Standard Time)

Desired capacity
1

Minimum capacity
1

Maximum capacity
1

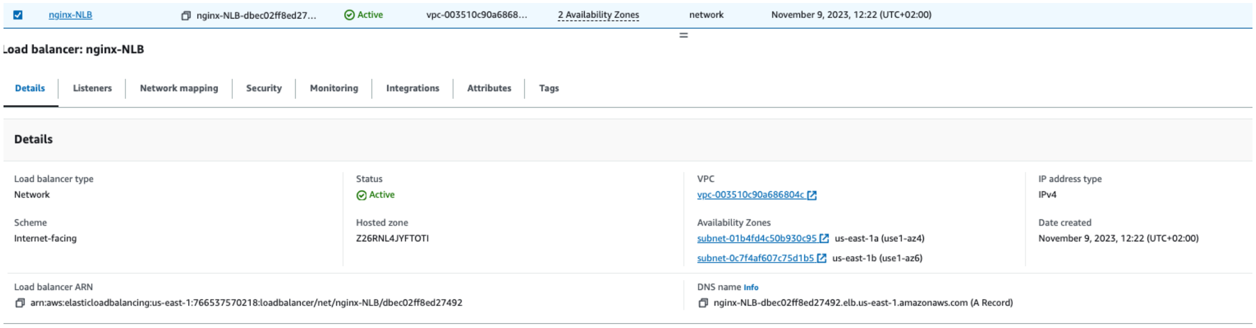
Status
-

Amazon Resource Name (ARN)
arn:aws:autoscaling:us-east-1:766537570218:autoScalingGroup:3715505a-9f33-4136-b588-af199bac6fdb:autoScalingGroupName/Tomcat-SG

Nginx (Frontend)

1. Create a public facing Network Load Balancer and Target Group.

Create a network load balancer with a listener on port 80



☑	nginx-NLB	nginx-NLB-dbec02ff8ed27…	Active	vpc-003510c90a6868…	2 Availability Zones	network	November 9, 2023, 12:22 (UTC+02:00)

Load balancer: nginx-NLB

Details

Load balancer type
Network

Scheme
Internet-facing

Status
✓ Active

Hosted zone
Z26RNL4JYFTOTI

VPC
vpc-003510c90a686804c

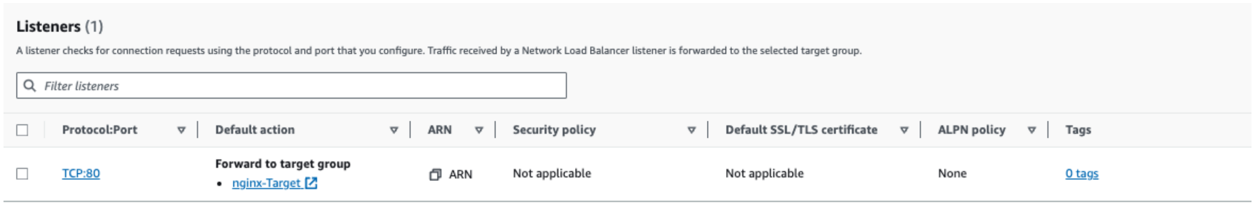
Availability Zones
subnet-01b4fd4c50b930c95 us-east-1a (use1-az4)
subnet-0c7f4af607c75d1b5 us-east-1b (use1-az6)

IP address type
IPv4

Date created
November 9, 2023, 12:22 (UTC+02:00)

Load balancer ARN
arn:aws:elasticloadbalancing:us-east-1:766537570218:loadbalancer/net/nginx-NLB/dbec02ff8ed27492

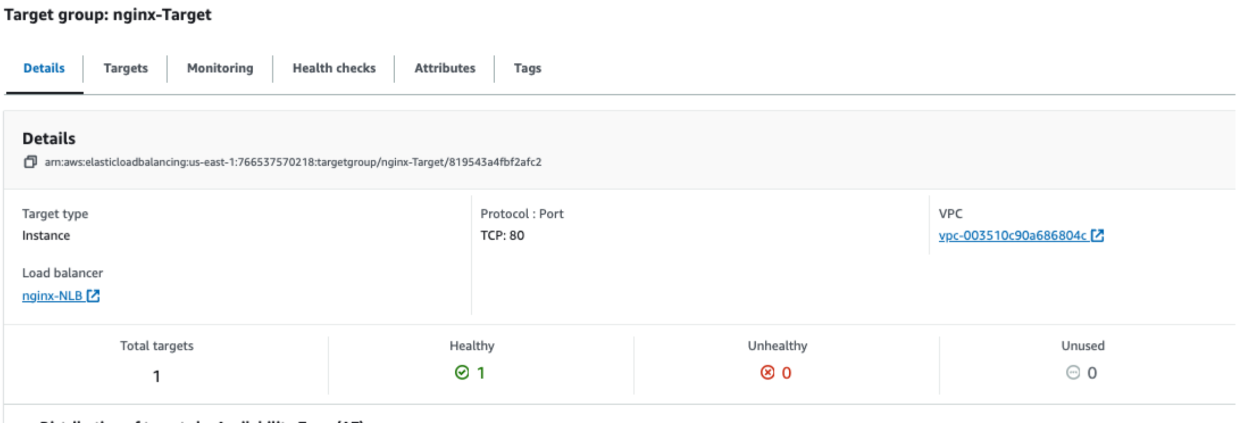
DNS name Info
nginx-NLB-dbec02ff8ed27492.elb.us-east-1.amazonaws.com (A Record)



Listeners (1)

A listener checks for connection requests using the protocol and port that you configure. Traffic received by a Network Load Balancer listener is forwarded to the selected target group.

□	Protocol:Port	Default action	ARN	Security policy	Default SSL/TLS certificate	ALPN policy	Tags
□	TCP:80	Forward to target group • nginx-Target	ARN	Not applicable	Not applicable	None	0 tags



Target group: nginx-Target

Details

arn:aws:elasticloadbalancing:us-east-1:766537570218:targetgroup/nginx-Target/819543a4fbf2afc2

Target type
Instance

Load balancer
nginx-NLB

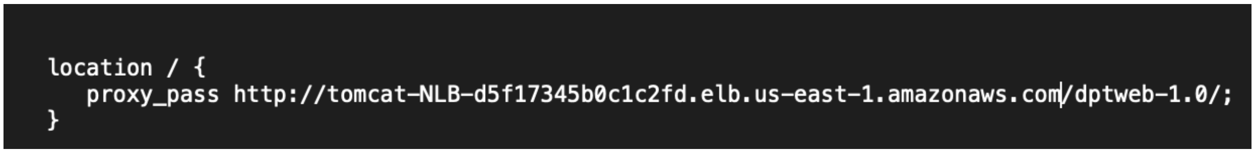
Protocol : Port
TCP: 80

VPC
vpc-003510c90a686804c

Total targets	Healthy	Unhealthy	Unused
1	✓ 1	⊗ 0	⊖ 0

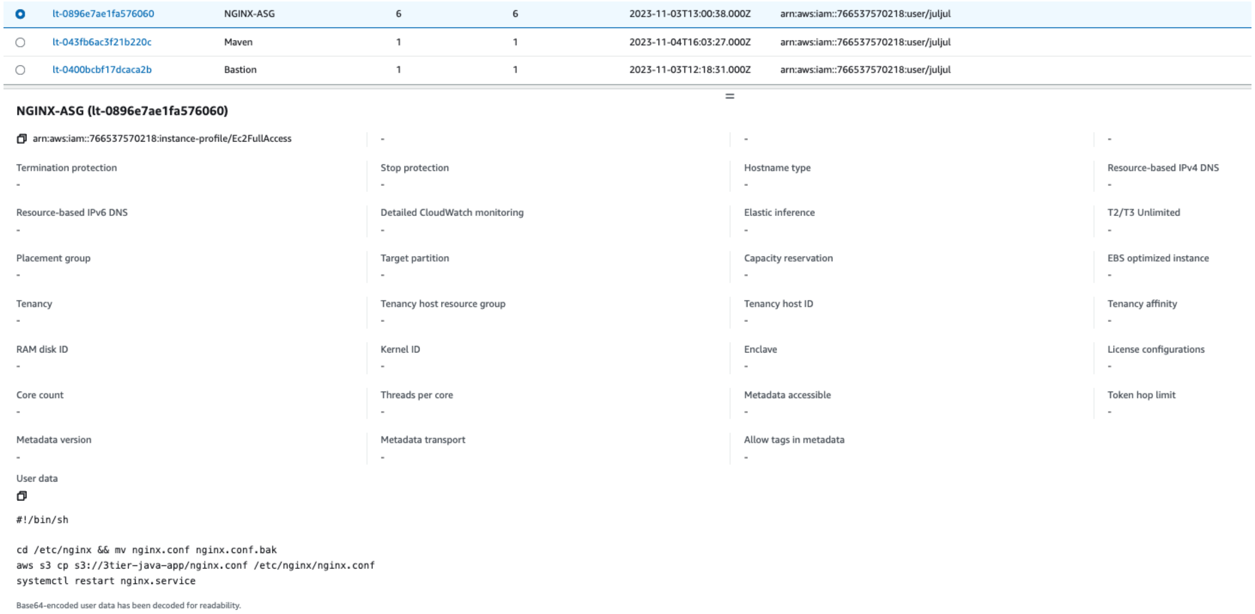
2. Create Launch Configuration with below configuration
    - **Nginx Golden AMI**
    - **User Data to update proxy_pass rules in nginx.conf file and reload nginx service.**
    - **Security Group allowing Port 22 from Bastion Host and Port 80 from Public NLB.**

Nginx.conf file is updated:



```
location / {
   proxy_pass http://tomcat-NLB-d5f17345b0c1c2fd.elb.us-east-1.amazonaws.com/dptweb-1.0/;
}
```

Proxy_pass value should have the tomcat network load balancer (backend of the app).



●	lt-0896e7ae1fa576060	NGINX-ASG	6	6	2023-11-03T13:00:38.000Z	arn:aws:iam::766537570218:user/juljul
○	lt-043fb6ac3f21b220c	Maven	1	1	2023-11-04T16:03:27.000Z	arn:aws:iam::766537570218:user/juljul
○	lt-0400bcbf17dcaca2b	Bastion	1	1	2023-11-03T12:18:31.000Z	arn:aws:iam::766537570218:user/juljul

NGINX-ASG (lt-0896e7ae1fa576060)

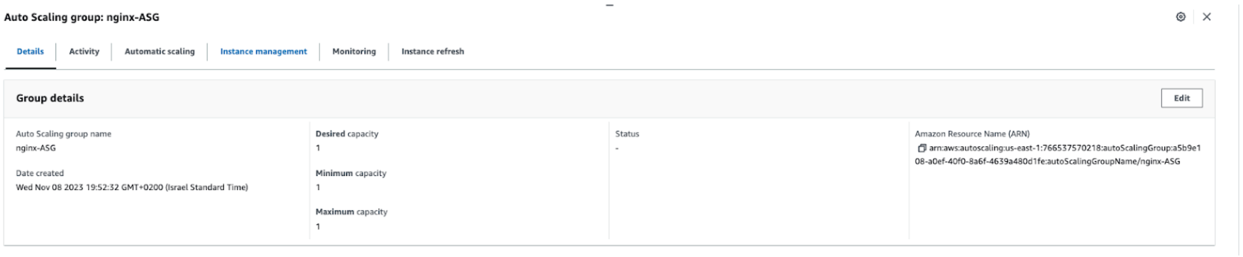
arn:aws:iam::766537570218:instance-profile/Ec2FullAccess	-	-	-
Termination protection -	Stop protection -	Hostname type -	Resource-based IPv4 DNS -
Resource-based IPv6 DNS -	Detailed CloudWatch monitoring -	Elastic inference -	T2/T3 Unlimited -
Placement group -	Target partition -	Capacity reservation -	EBS optimized instance -
Tenancy -	Tenancy host resource group -	Tenancy host ID -	Tenancy affinity -
RAM disk ID -	Kernel ID -	Enclave -	License configurations -
Core count -	Threads per core -	Metadata accessible -	Token hop limit -
Metadata version -	Metadata transport -	Allow tags in metadata -	

User data

```
#!/bin/sh

cd /etc/nginx && mv nginx.conf nginx.conf.bak
aws s3 cp s3://3tier-java-app/nginx.conf /etc/nginx/nginx.conf
systemctl restart nginx.service
```

Base64-encoded user data has been decoded for readability.

3. Create Auto Scaling Group



Group details

Auto Scaling group name
nginx-ASG

Date created
Wed Nov 08 2023 19:52:32 GMT+0200 (Israel Standard Time)

Desired capacity
1

Minimum capacity
1

Maximum capacity
1

Status
-

Amazon Resource Name (ARN)
arn:aws:autoscaling:us-east-1:766537570218:autoScalingGroup:a5b9e108-a0ef-40f0-8a6f-4639a480d1fe:autoScalingGroupName/nginx-ASG

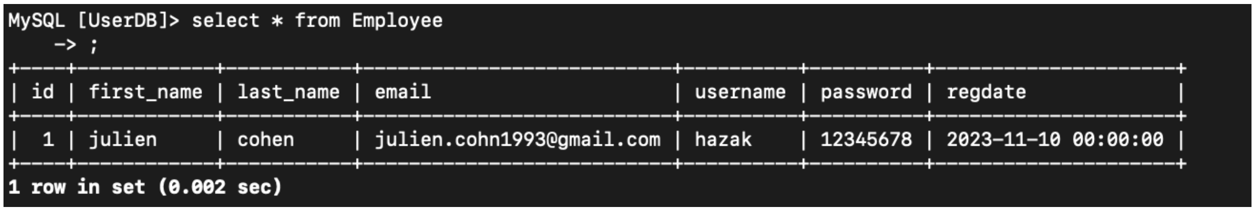
Application Deployment

1. Artifact deployment taken care by User Data script during Application tier EC2 instance launch process.

2. Login to MySQL database from Application Server using MySQL CLI client and create database and table schema to store the user login data (Instructions are update in README.md file in the Bitbucket repo)

Login to tomcat server, install mysql client and configure DB schema

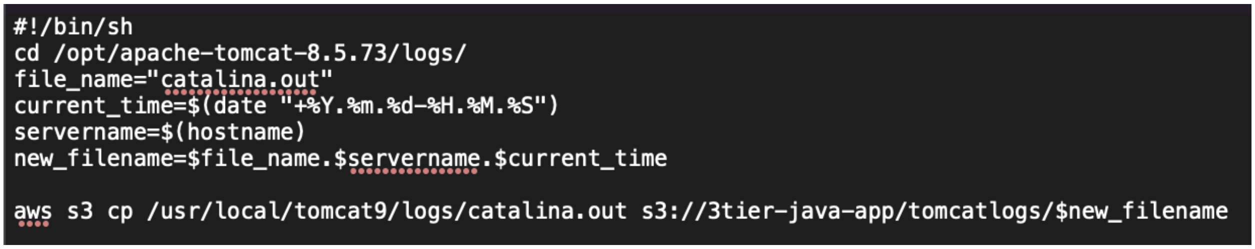
```
# sudo dnf install mariadb105-server

# mysql -u admin -p -h
javaapp-db1.cmvadjxoeeuc.us-east-1.rds.amazonaws.com

# create database UserDB;

# use UserDB;
# CREATE TABLE Employee (id int unsigned auto_increment not null,
first_name varchar(250), last_name varchar(250), email varchar(250),
username varchar(250), password varchar(250), regdate timestamp,
primary key (id) );
```



```
MySQL [UserDB]> select * from Employee
    -> ;
+----+------------+-----------+---------------------------+----------+----------+---------------------+
| id | first_name | last_name | email                     | username | password | regdate             |
+----+------------+-----------+---------------------------+----------+----------+---------------------+
|  1 | julien     | cohen     | julien.cohn1993@gmail.com | hazak    | 12345678 | 2023-11-10 00:00:00 |
+----+------------+-----------+---------------------------+----------+----------+---------------------+
1 row in set (0.002 sec)
```

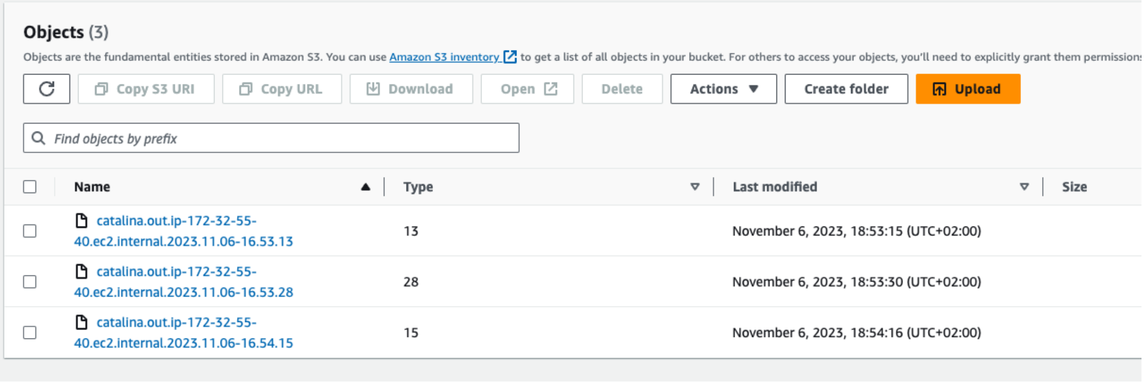
Post Deployment

1. Configure Cronjob to push the Tomcat Application log data to S3 bucket and also rotate the log data to remove the log data on the server after the data pushed to S3 Bucket.



```
#!/bin/sh
cd /opt/apache-tomcat-8.5.73/logs/
file_name="catalina.out"
current_time=$(date "+%Y.%m.%d-%H.%M.%S")
servername=$(hostname)
new_filename=$file_name.$servername.$current_time

aws s3 cp /usr/local/tomcat9/logs/catalina.out s3://3tier-java-app/tomcatlogs/$new_filename
```

Logs are pushed in the S3 bucket:



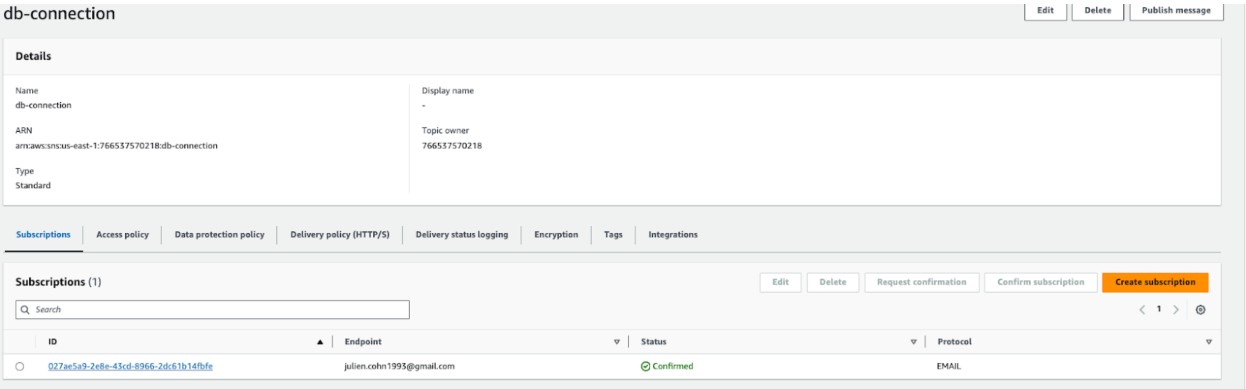
Objects (3)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permission

	Name	Type	Last modified	Size
□	catalina.out.ip-172-32-55-40.ec2.internal.2023.11.06-16.53.13	13	November 6, 2023, 18:53:15 (UTC+02:00)	
□	catalina.out.ip-172-32-55-40.ec2.internal.2023.11.06-16.53.28	28	November 6, 2023, 18:53:30 (UTC+02:00)	
□	catalina.out.ip-172-32-55-40.ec2.internal.2023.11.06-16.54.15	15	November 6, 2023, 18:54:16 (UTC+02:00)	

2. Configure Cloudwatch alarms to send E-Mail notification when database connections are more than 100 threshold.

Below the SNS topic used to send en email each time the threshold is reached.



db-connection

Details

Name
db-connection

ARN
arn:aws:sns:us-east-1:766537570218:db-connection

Type
Standard

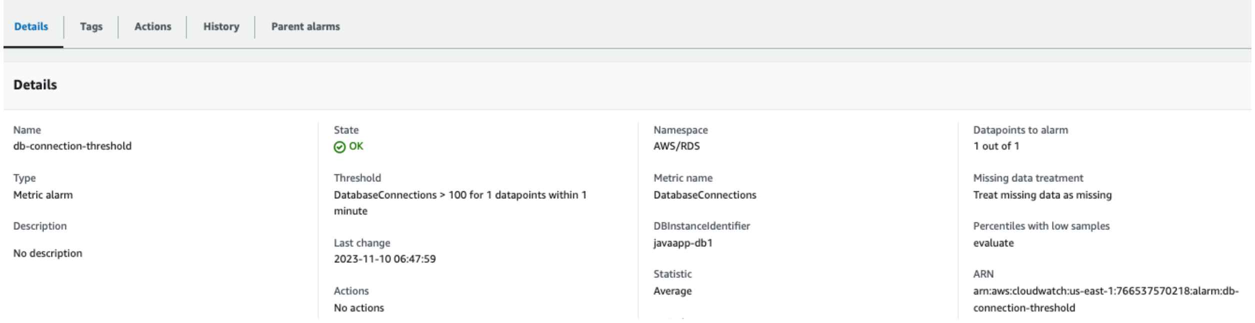
Display name
-

Topic owner
766537570218

Subscriptions (1)

	ID	Endpoint	Status	Protocol
○	027ae5a9-2e8e-43cd-8966-2dc61b14fbfe	julien.cohn1993@gmail.com	✓ Confirmed	EMAIL

Cloudwatch alarm created if more that 100 connections are performed on the DB, an email will be sent



Details

Name
db-connection-threshold

Type
Metric alarm

Description
No description

State
✓ OK

Threshold
DatabaseConnections > 100 for 1 datapoints within 1 minute

Last change
2023-11-10 06:47:59

Actions
No actions

Namespace
AWS/RDS

Metric name
DatabaseConnections

DBInstanceIdentifier
javaapp-db1

Statistic
Average

Datapoints to alarm
1 out of 1

Missing data treatment
Treat missing data as missing

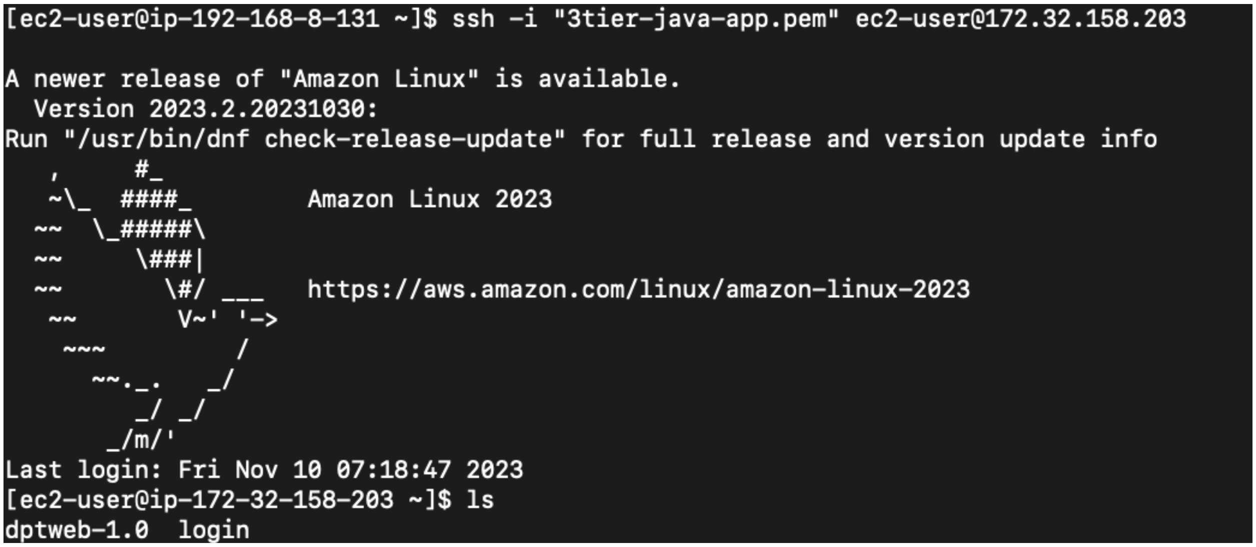
Percentiles with low samples
evaluate

ARN
arn:aws:cloudwatch:us-east-1:766537570218:alarm:db-connection-threshold

Validation

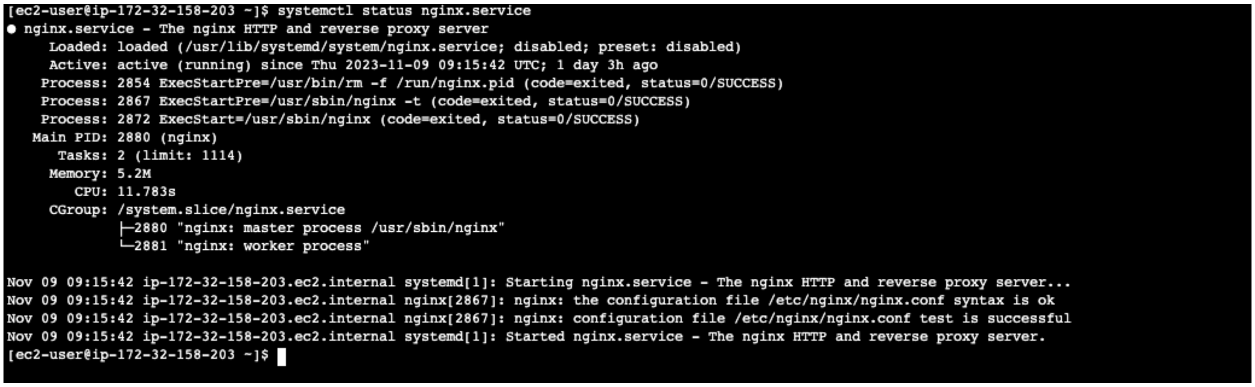
1. Verify you as an administrator able to login to EC2 instances from session manager & from Bastion Host.

SSH from bastion to EC2 instances in 3tierApp is working



```
[ec2-user@ip-192-168-8-131 ~]$ ssh -i "3tier-java-app.pem" ec2-user@172.32.158.203

A newer release of "Amazon Linux" is available.
  Version 2023.2.20231030:
Run "/usr/bin/dnf check-release-update" for full release and version update info
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Fri Nov 10 07:18:47 2023
[ec2-user@ip-172-32-158-203 ~]$ ls
dptweb-1.0  login
```

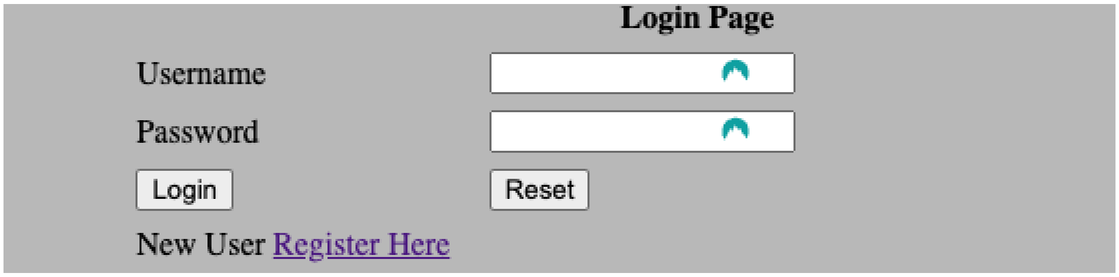
From session manager:



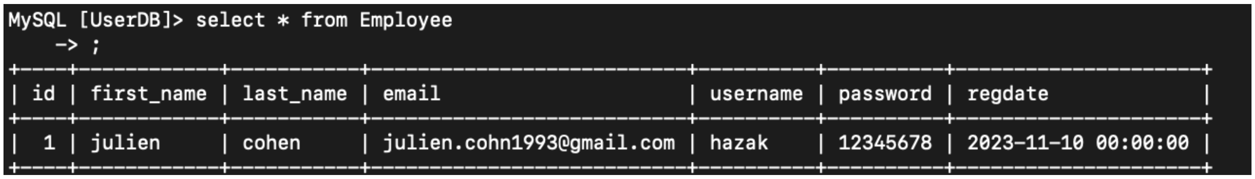
```
[ec2-user@ip-172-32-158-203 ~]$ systemctl status nginx.service
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; preset: disabled)
     Active: active (running) since Thu 2023-11-09 09:15:42 UTC; 1 day 3h ago
    Process: 2854 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
    Process: 2867 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
    Process: 2872 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
   Main PID: 2880 (nginx)
      Tasks: 2 (limit: 1114)
     Memory: 5.2M
        CPU: 11.783s
     CGroup: /system.slice/nginx.service
             ├─2880 "nginx: master process /usr/sbin/nginx"
             └─2881 "nginx: worker process"

Nov 09 09:15:42 ip-172-32-158-203.ec2.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Nov 09 09:15:42 ip-172-32-158-203.ec2.internal nginx[2867]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Nov 09 09:15:42 ip-172-32-158-203.ec2.internal nginx[2867]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Nov 09 09:15:42 ip-172-32-158-203.ec2.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
[ec2-user@ip-172-32-158-203 ~]$ █
```

2. Verify if you as an end user able to access application from public internet browser.

When I requested the nginx load balancer I got the tomcat app.
The nginx load balancer redirect to the tomcat load balancer thanks to proxy_pass.



Then I am able to login with:



```
MySQL [UserDB]> select * from Employee
    -> ;
+----+------------+-----------+---------------------------+----------+----------+---------------------+
| id | first_name | last_name | email                     | username | password | regdate             |
+----+------------+-----------+---------------------------+----------+----------+---------------------+
|  1 | julien     | cohen     | julien.cohn1993@gmail.com | hazak    | 12345678 | 2023-11-10 00:00:00 |
+----+------------+-----------+---------------------------+----------+----------+---------------------+
```