AWS load balancer with auto scale quick setup

- Overview
- Launching AWS Auto Scaling group with classic load balancer from custom AMI
  - 1. Create new AMI
  - 2. Create new Load Balancer
  - 3. Create new Launch Configuration
  - 4. Create new Auto Scaling group
- Launching Application Load Balancer using existing instances
  - 1. Instances launching
  - 2. Application Load Balancer creation
  - 3. Websocket listener target group creation
  - 4. Listener parameters configuration
- Launching AWS Auto Scaling group with load balancing from AWS Marketplace AMI
  - 1. Create launch template
  - 2. Create Auto scaling group
- Testing

Overview

WCS Amazon instances support AWS load balancer.

WebSocket connections will be distributed between active load balancer instances. In case a scaling policy is executed (when the policy target – e.g., CPU load on instance - is reached) and new instances are launched, they will be added to the load balancer.

The following components would be required

- AMI on the basis of which new instances will be created for scaling out
- Load Balancer
- Launch Configuration
- Auto Scaling Group

Launching AWS Auto Scaling group with classic load balancer from custom AMI

Load balancer with autoscaling deployment from custom AMI can be useful for logn term projects (months and years). In this case, AWS Marketplace image will be more expensive due to hourly payment, therefore it is recommended to buy and activate WCS monthly subscription.

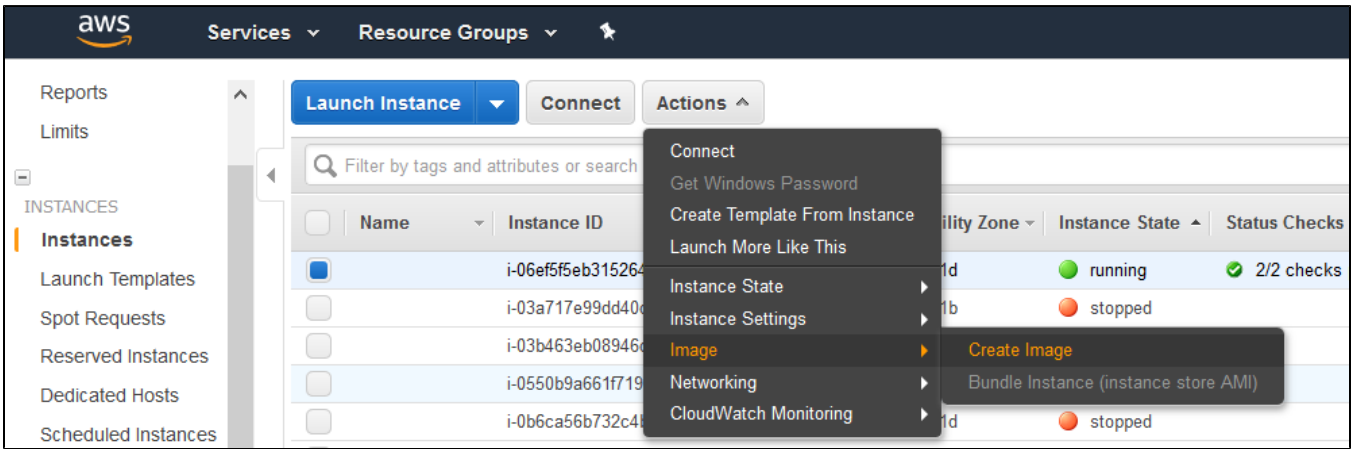
Note that classic load balancer will work till August 2022.

1. Create new AMI

1.1.Launch an instance from a FlashphonerWebCallServer AMI and configure the WCS

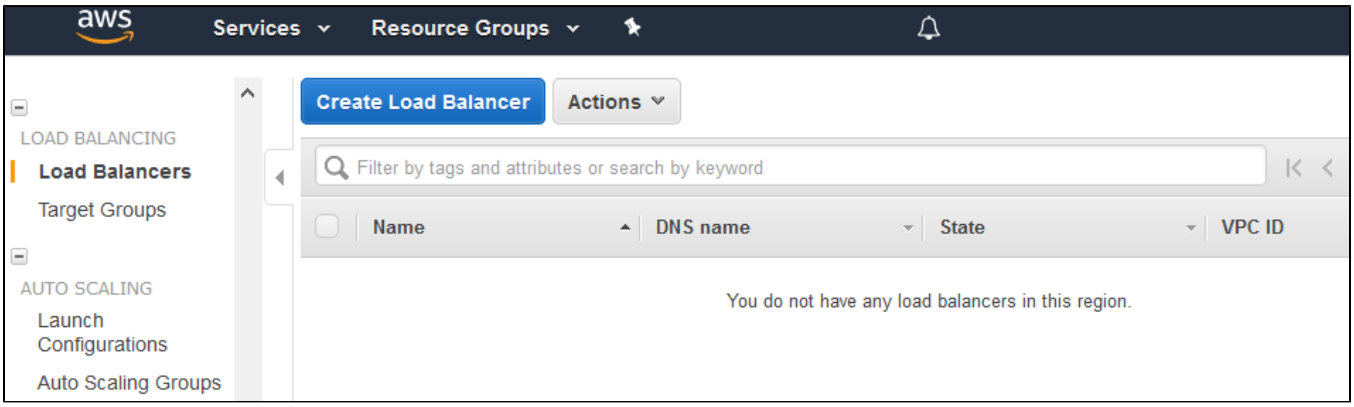
- activate license
- import certificates
- change configuration settings as required

1.2. In AWS console, select the instance and then “Actions” | “Image” | “Create Image” and create a new image:

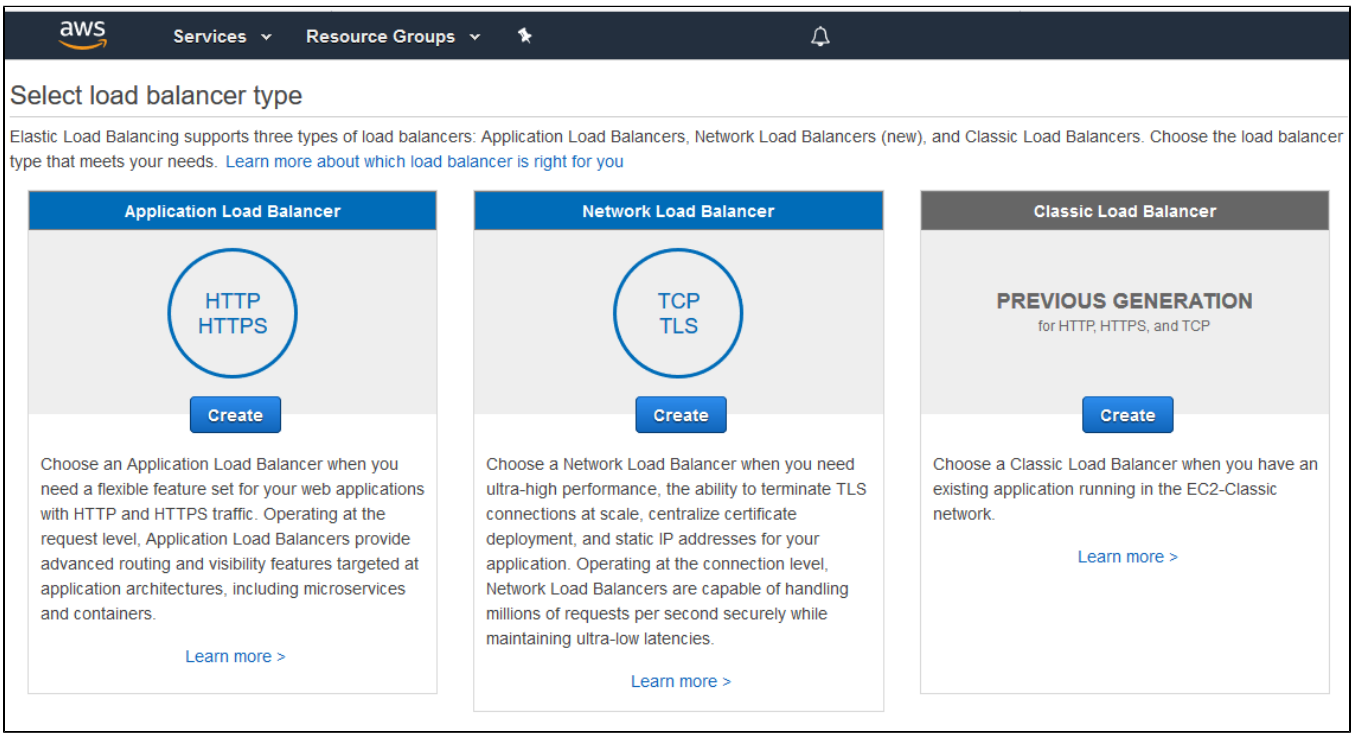


2. Create new Load Balancer

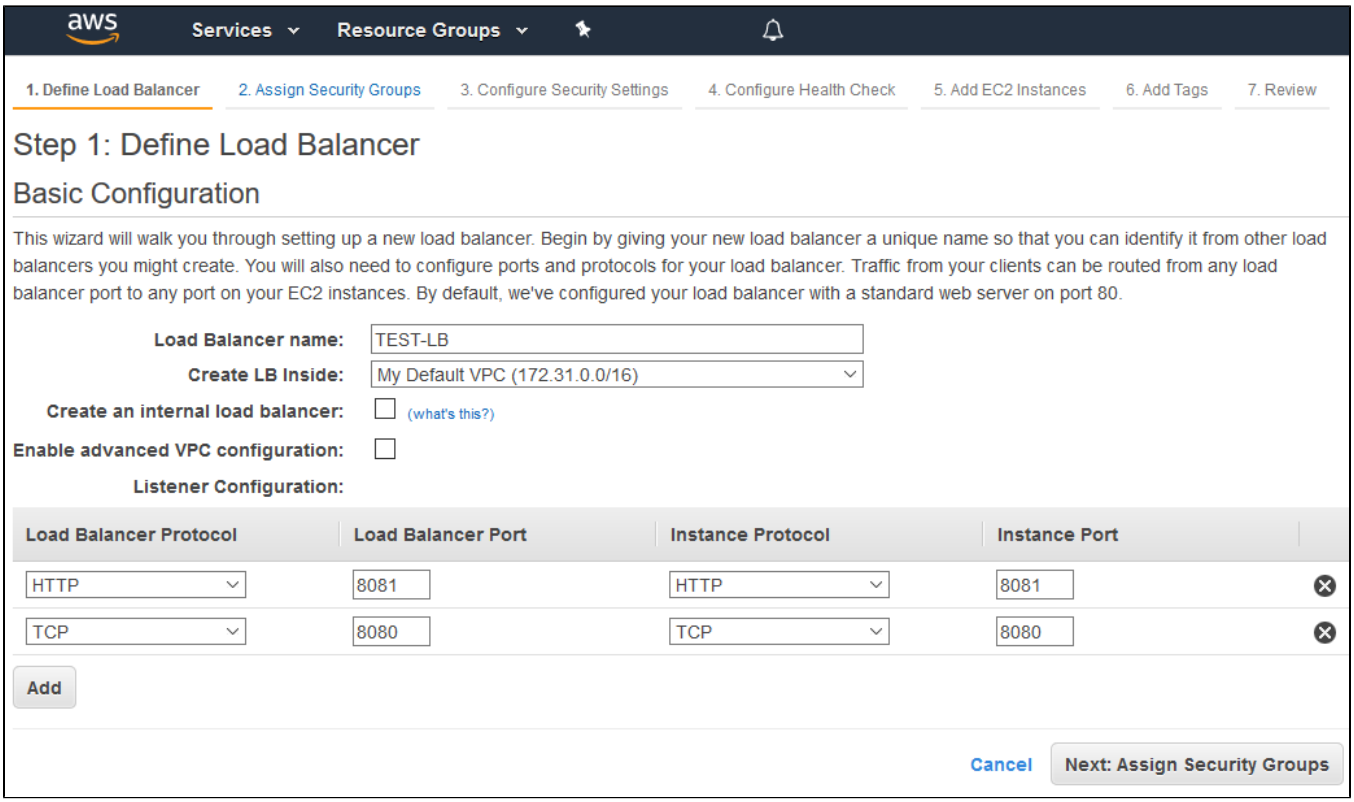
2.1. In AWS console, go to “EC2” | “Load Balancers” and click “Create Load Balancer”



2.2 Select “Classic Load Balancer” type (This type allows specifying port for health check.)



2.3. When defining load balancer, add required protocols. For exampleTCP, port 8080 for WebSocket connections (ws:<host>:8080).



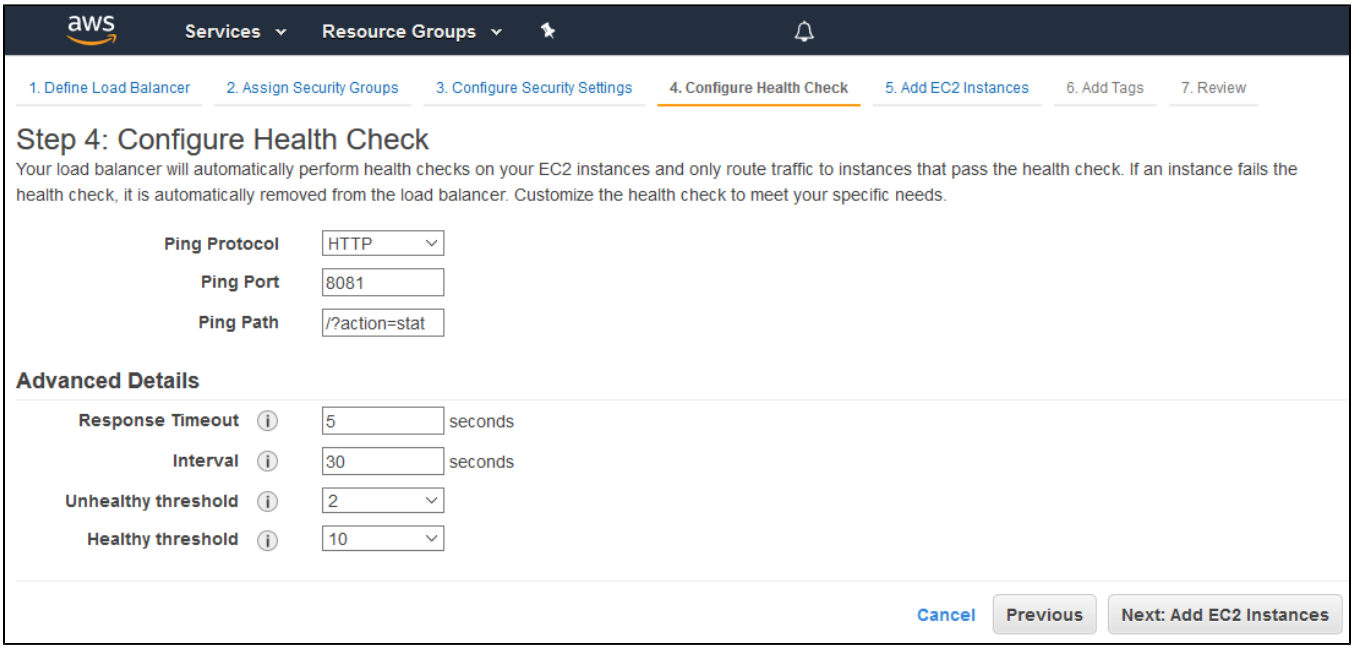
2.4. Assign a security group.

2.5. Configure health check

The URL for health check is

- for HTTP: http://WCS_ADDRESS:8081/?action=stat

- for HTTPS:https://WCS_ADDRESS:8444/?action=stat

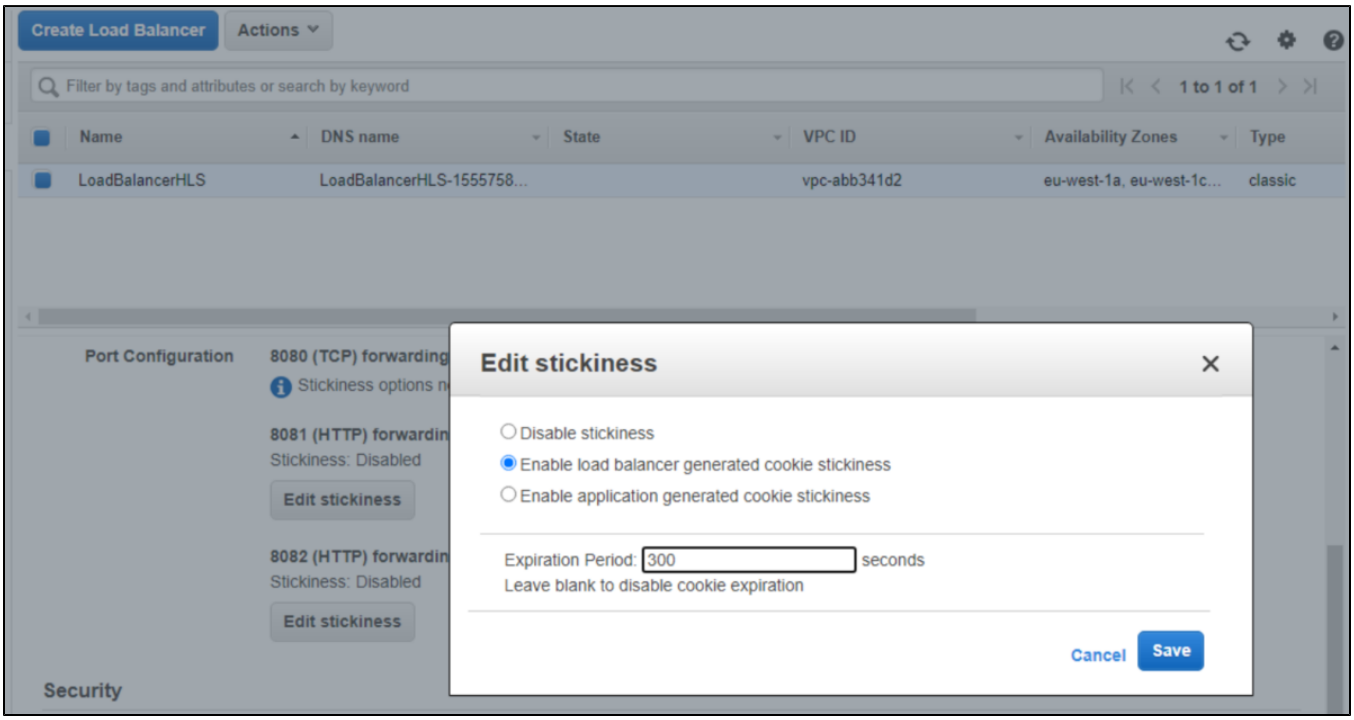


2.6. Add existing EC2 instances as required

By default, cross-zone load balancing is enabled to distribute traffic between all available availability zones in your region.

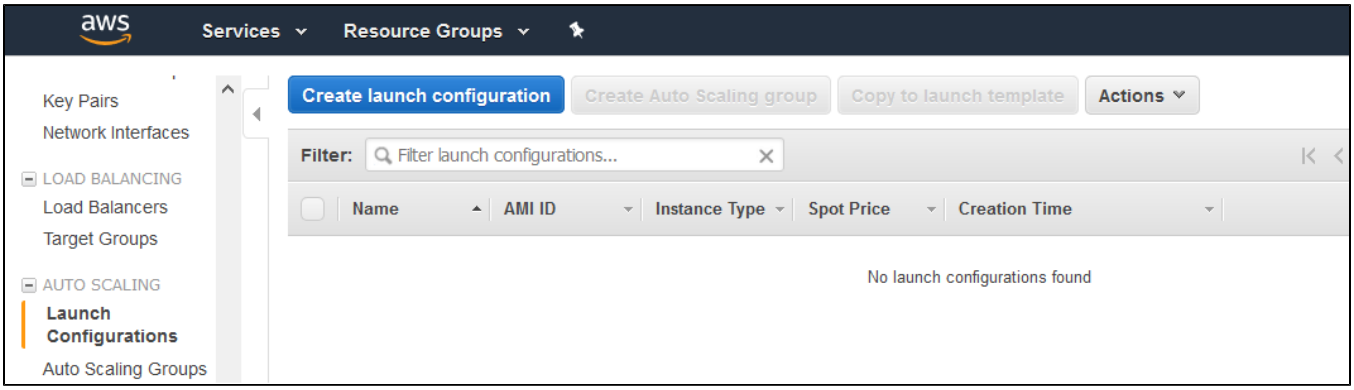
2.7. Complete the wizard to create the load balancer

2.8. Enable stickiness for HTTP/HTTPS LB ports

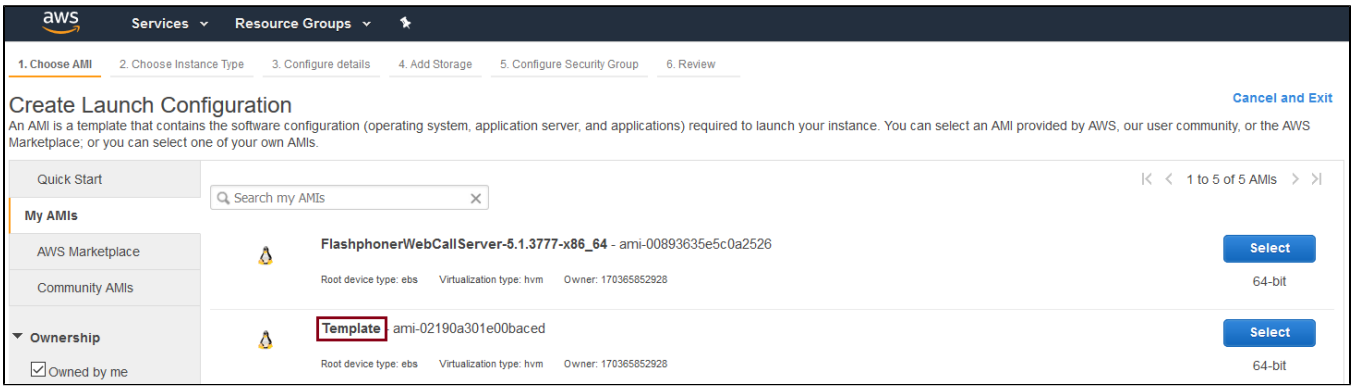


3. Create new Launch Configuration

3.1. In AWS console, go to “EC2” | “Launch Configurations” and click “Create launch configuration”

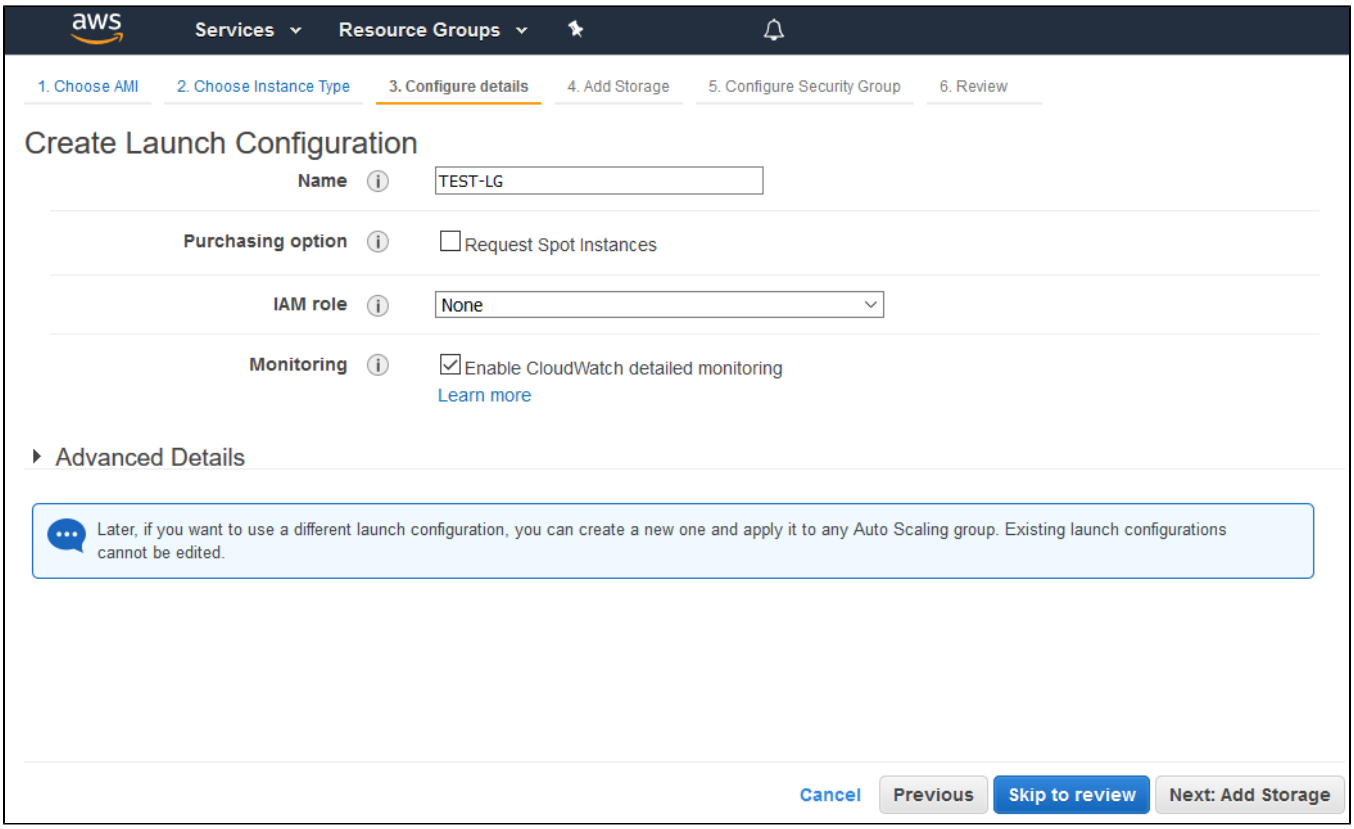


3.2. When choosing AMI, select the AMI previously created from an instance with required WCS configuration



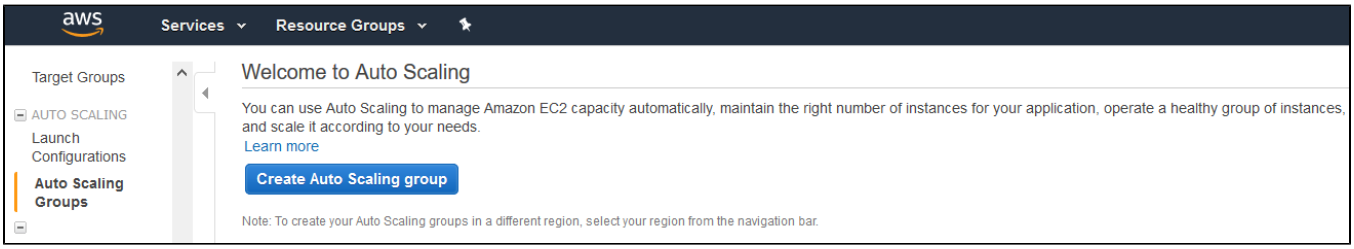
3.3. Complete the wizard to create the configuration

Detailed monitoring, where data is available in 1-minute periods, can be enabled when configuring details.



4. Create new Auto Scaling group

4.1. In AWS console, go to “EC2” | “Auto Scaling Groups” and click “Create Auto Scaling group”



4.2.Select the required launch configuration or template, or select to create a new one

aws Services Resource Groups

Create Auto Scaling Group

Cancel and Exit

Complete this wizard to create your Auto Scaling group. First, choose either a launch configuration or a launch template to specify the parameters that your Auto Scaling group uses to launch instances.

Launch Configuration

You can continue to use your launch configurations if they support the Amazon EC2 features you need. Learn more

Launch Template New

Launch templates give you the option of launching one type of instance, or a combination of instance types and purchase options. Launch templates include the latest Amazon EC2 features and can be updated and versioned.

Learn more

Create new launch template

Create a new launch configuration

Use an existing launch configuration

Filter launch configurations...

1 to 1 of 1 Launch Configurations

Name	AMI ID	Instance Type	Spot Price	Security Groups
TEST-LG	ami-0cf30f44be371890d	t2.small		sg-03b1bc951522cb94a

Cancel Next Step

4.3. Configure Auto Scaling group details

- add required subnets
- add required load balancer

aws Services Resource Groups

1. Configure Auto Scaling group details | 2. Configure scaling policies | 3. Configure Notifications | 4. Configure Tags | 5. Review

Cancel and Exit

Create Auto Scaling Group

Group name: TEST-ASG

Launch Configuration: TEST-LG

Group size: Start with 1 instances

Network: vpc-714e7e15 (172.31.0.0/16) (default) — Create new VPC

Subnet:
- subnet-d3d17ff9(172.31.48.0/20) | Default in us-east-1d
- subnet-861169e3(172.31.64.0/20) | Default in us-east-1c

Create new subnet

Each instance in this Auto Scaling group will be assigned a public IP address.

Advanced Details

Load Balancing: ☑ Receive traffic from one or more load balancers — Learn about Elastic Load Balancing

Classic Load Balancers: TEST-LB

Target Groups:

Health Check Type: ◉ ELB ○ EC2

Health Check Grace Period: 300 seconds

Monitoring: ☑ Enable CloudWatch detailed monitoring

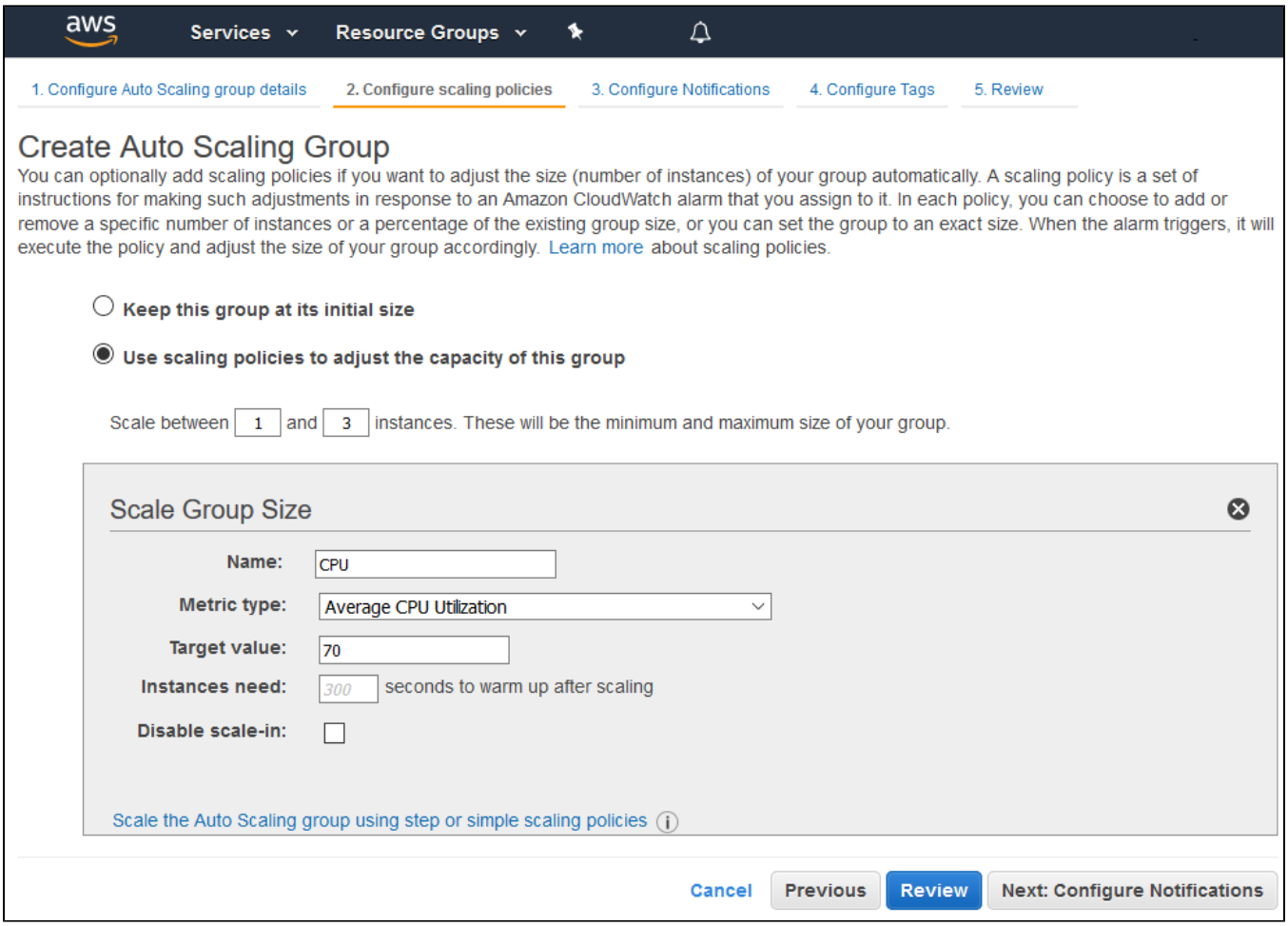
Learn more

Instance Protection:

Service-Linked Role: AWSServiceRoleForAutoScaling — View Role in IAM

Cancel | Next: Configure scaling policies

4.4. Configure scaling policies



4.5. Complete the wizard to create the auto scaling group

Launching Application Load Balancer using existing instances

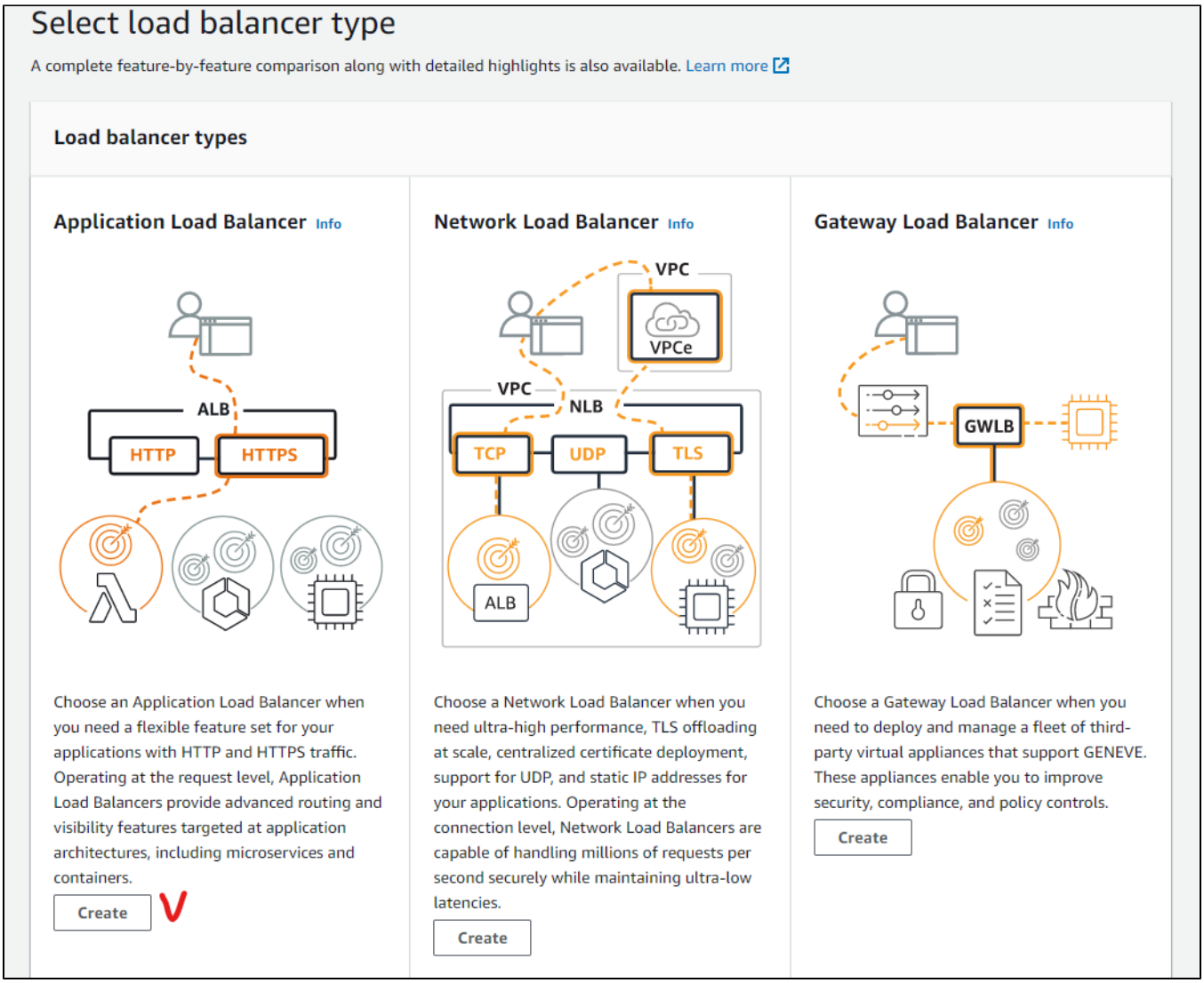
Sometimes, a certain set of instances is already launched and configured (Origin servers group in CDN, for example), and load balancing between those servers should be set up. Use Application Load Balancer to do this.

1. Instances launching

Launch and configure server instances as needed by this manual.

2. Application Load Balancer creation

2.1. In EC2 Console menu, go to "Load balancers - Load balancers" section and click "Create load balancer". Click Create for Application Load Balancer



2.2. Enter the balancer name, choose Internet-facing type (supposed by default)

Create Application Load Balancer Info

The Application Load Balancer distributes incoming HTTP and HTTPS traffic across multiple targets such as Amazon EC2 instances, microservices, and containers, based on request attributes. When the load balancer receives a connection request, it evaluates the listener rules in priority order to determine which rule to apply, and if applicable, it selects a target from the target group for the rule action.

▶ **How Application Load Balancers work**

Basic configuration

Load balancer name
Name must be unique within your AWS account and cannot be changed after the load balancer is created.

TEST-APP-LB

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Scheme Info
Scheme cannot be changed after the load balancer is created.

- (●) Internet-facing
  An internet-facing load balancer routes requests from clients over the internet to targets. Requires a public subnet. Learn more
- ( ) Internal
  An internal load balancer routes requests from clients to targets using private IP addresses.

IP address type Info
Select the type of IP addresses that your subnets use.

- (●) IPv4
  Recommended for internal load balancers.
- ( ) Dualstack
  Includes IPv4 and IPv6 addresses.

2.3. In "Network mapping" section choose a subnets needed

Network mapping Info

The load balancer routes traffic to targets in the selected subnets, and in accordance with your IP address settings.

VPC Info
Select the virtual private cloud (VPC) for your targets. Only VPCs with an internet gateway are enabled for selection. The selected VPC cannot be changed after the load balancer is created. To confirm the VPC for your targets, view your target groups.

-
vpc-e305fc9a
IPv4: 172.31.0.0/16

Mappings Info
Select at least two Availability Zones and one subnet per zone. The load balancer routes traffic to targets in these Availability Zones only. Availability Zones that are not supported by the load balancer or the VPC are not available for selection. Subnets cannot be removed after the load balancer is created, but additional subnets can be added.

- [x] **eu-west-1a**

  Subnet

  subnet-003b4c5a

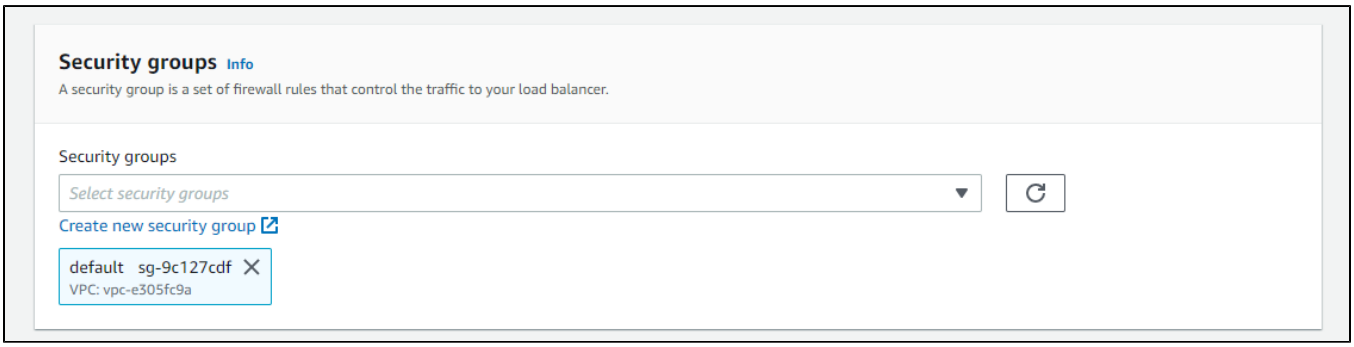
  IPv4 settings

  Assigned by AWS

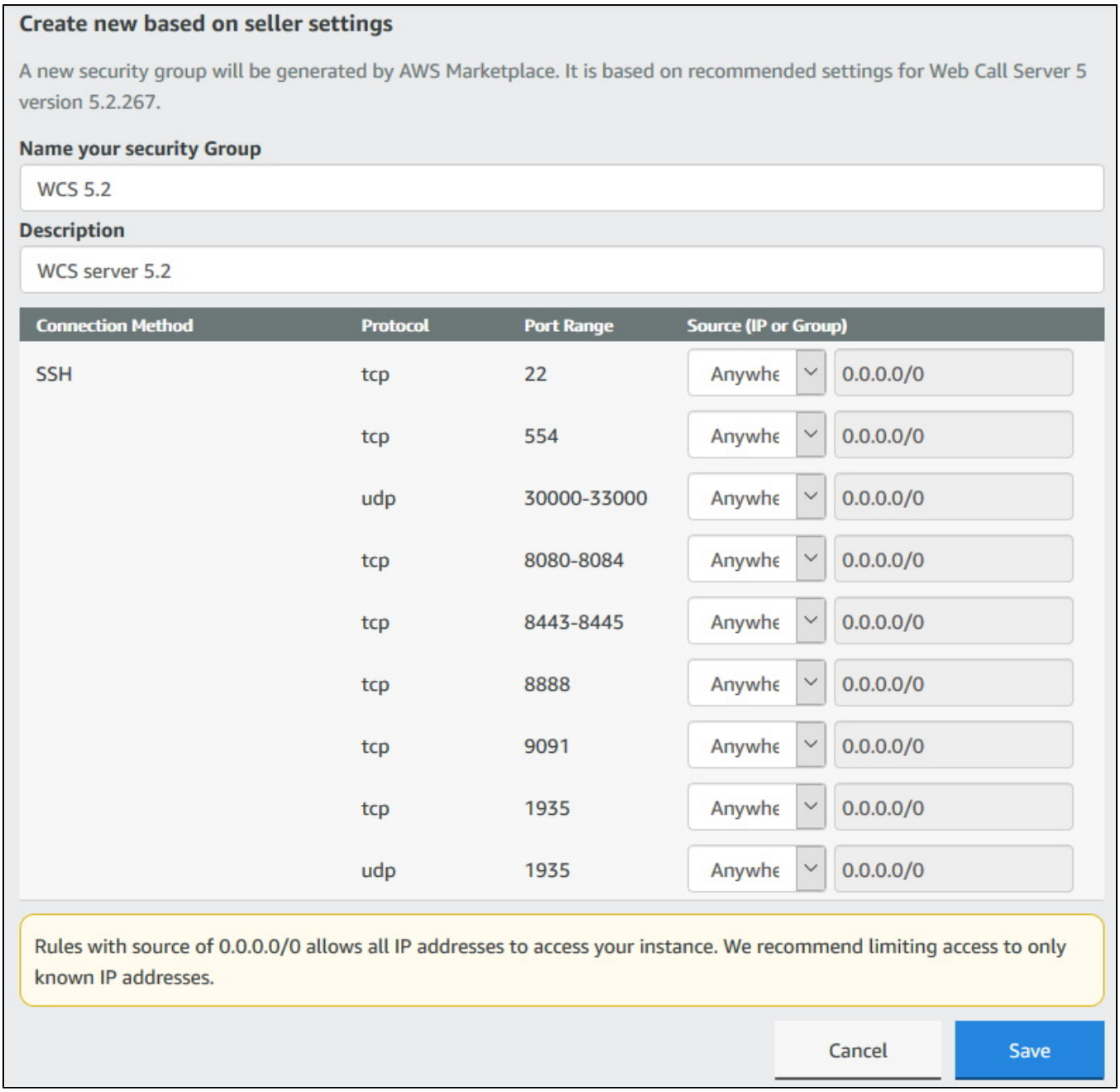
- [ ] **eu-west-1b**

- [ ] **eu-west-1c**

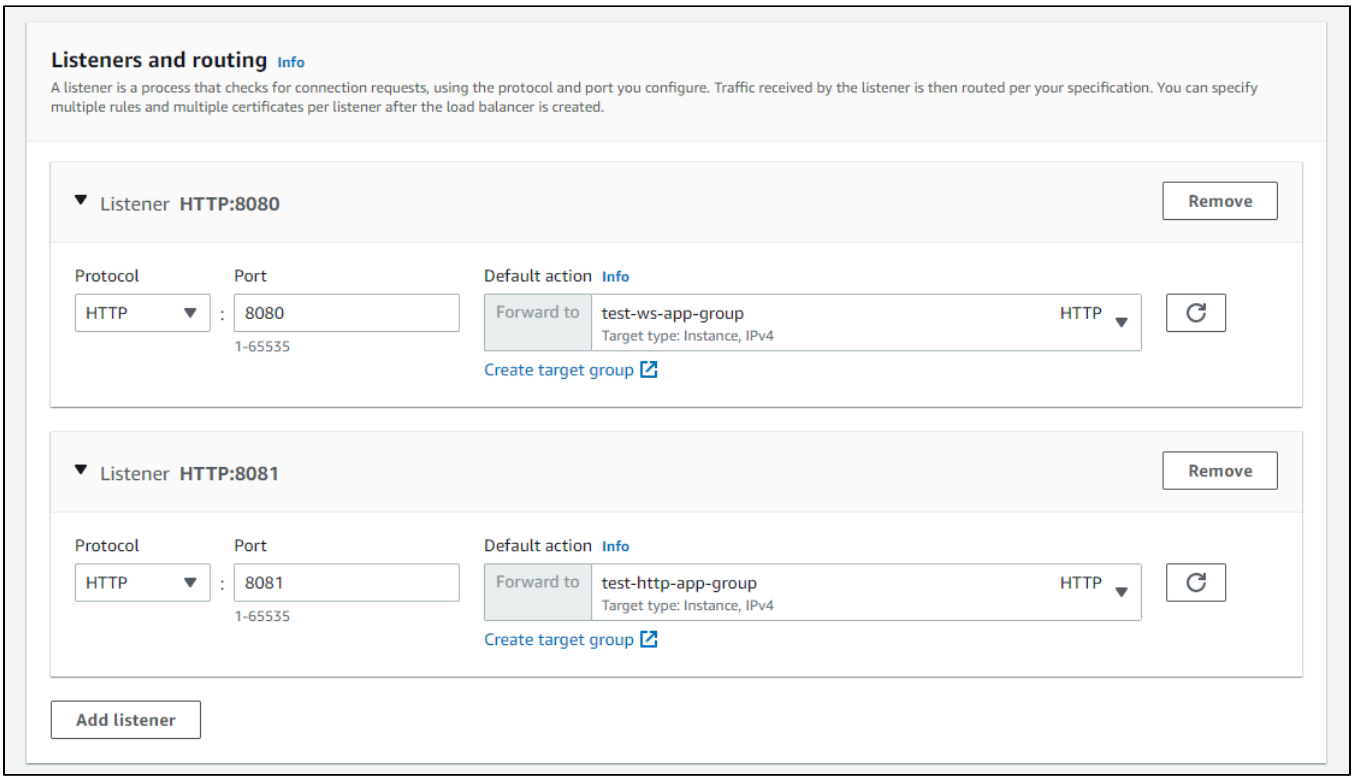
2.4. Choose or create security groups as needed



Note that a standard WCS ports set should be opened in load balancer security group

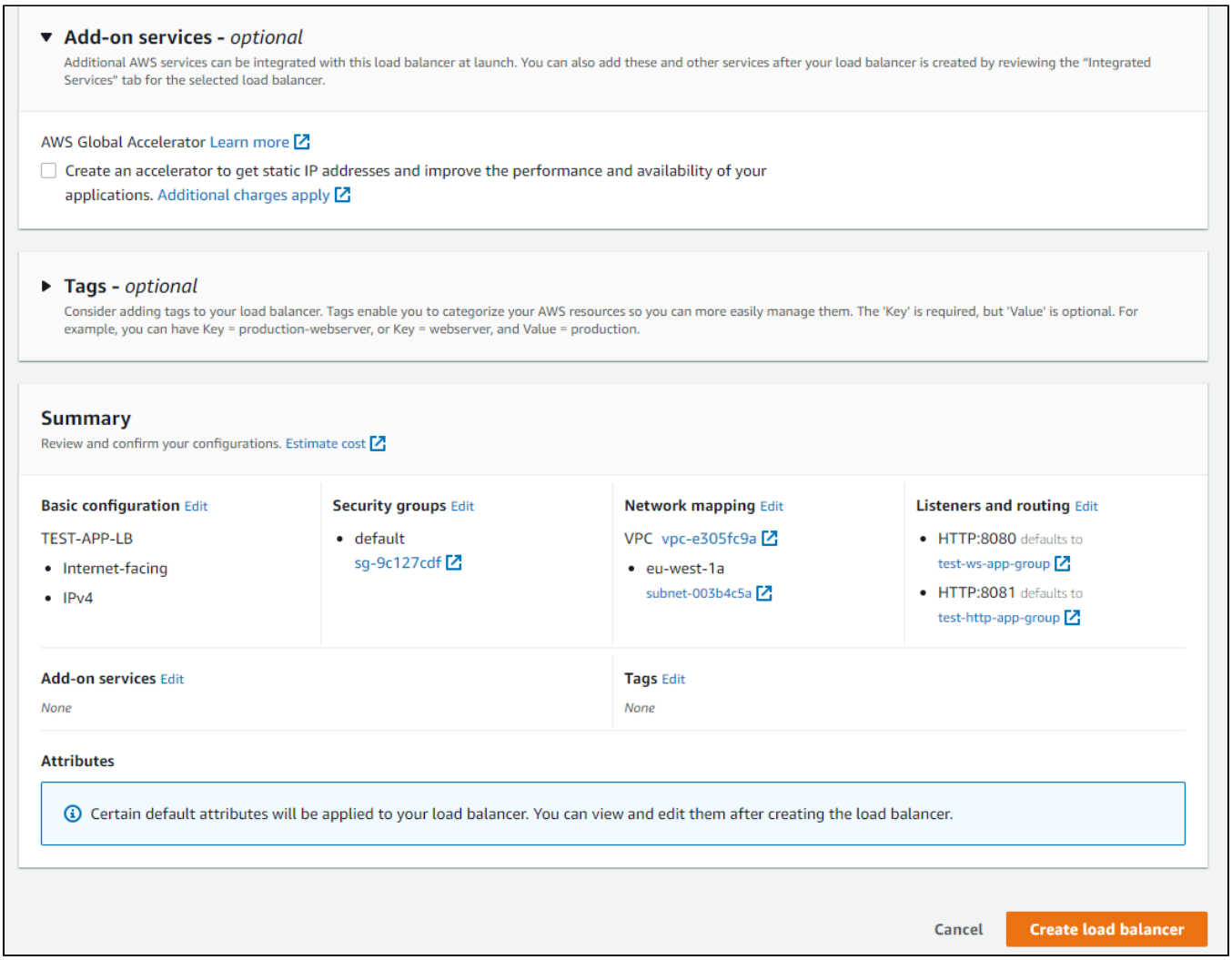


2.5. In "Listeners and routing" section add Websocket port listener (mandatory) and HTTP port listener (if needed)

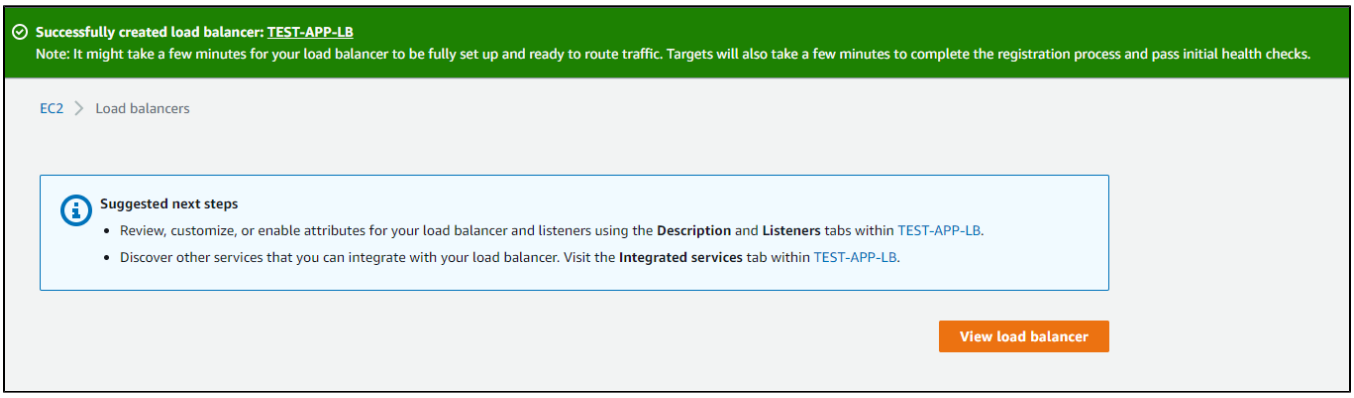


A target group must be created for every listener, see below.

2.6. Click Create load balancer



Load balancer is created



3. Websocket listener target group creation

3.1. Choose target type Instances (supposed by default), set group name

Step 1
Specify group details

Step 2
Register targets

Specify group details

Your load balancer routes requests to the targets in a target group and performs health checks on the targets.

Basic configuration

Settings in this section cannot be changed after the target group is created.

Choose a target type

- (●) Instances
  - Supports load balancing to instances within a specific VPC.
- ( ) IP addresses
  - Supports load balancing to VPC and on-premises resources.
  - Facilitates routing to multiple IP addresses and network interfaces on the same instance.
  - Offers flexibility with microservice based architectures, simplifying inter-application communication.
- ( ) Lambda function
  - Facilitates routing to a single Lambda function.
  - Accessible to Application Load Balancers only.
- ( ) Application Load Balancer
  - Offers the flexibility for a Network Load Balancer to accept and route TCP requests within a specific VPC.
  - Facilitates using static IP addresses and PrivateLink with an Application Load Balancer.

Target group name

test-ws-app-group

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Protocol: HTTP : Port: 8080

3.2. Set WCS instance Websocket port (8080), choose subnet and protocol version (HTTP1)

Protocol: HTTP : Port: 8080

VPC

Select the VPC with the instances that you want to include in the target group.

-
vpc-e305fc9a
IPv4: 172.31.0.0/16

Protocol version

- (●) HTTP1
  Send requests to targets using HTTP/1.1. Supported when the request protocol is HTTP/1.1 or HTTP/2.
- ( ) HTTP2
  Send requests to targets using HTTP/2. Supported when the request protocol is HTTP/2 or gRPC, but gRPC-specific features are not available.
- ( ) gRPC
  Send requests to targets using gRPC. Supported when the request protocol is gRPC.

3.3. In "Health check" configigure instance health check using HTTP port (8081) and statistics page query /?action=stat

Health checks

The associated load balancer periodically sends requests, per the settings below, to the registered targets to test their status.

Health check protocol

HTTP

Health check path

Use the default path of "/" to ping the root, or specify a custom path if preferred.

/?action=stat

Up to 1024 characters allowed.

Advanced health check settings

Restore defaults

Port

The port the load balancer uses when performing health checks on targets. The default is the port on which each target receives traffic from the load balancer, but you can specify a different port.

- Traffic port
- Override

8081

1-65535

Healthy threshold

The number of consecutive health checks successes required before considering an unhealthy target healthy.

5

2-10

Unhealthy threshold

The number of consecutive health check failures required before considering a target unhealthy.

2

2-10

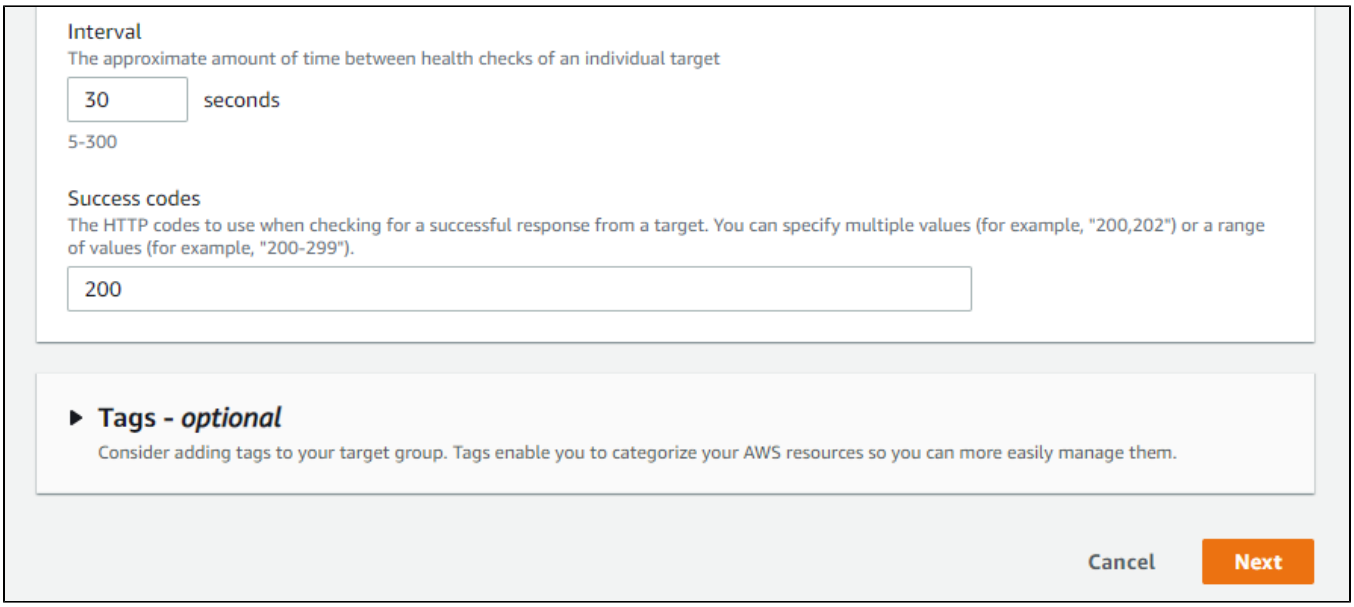
Timeout

The amount of time, in seconds, during which no response means a failed health check.

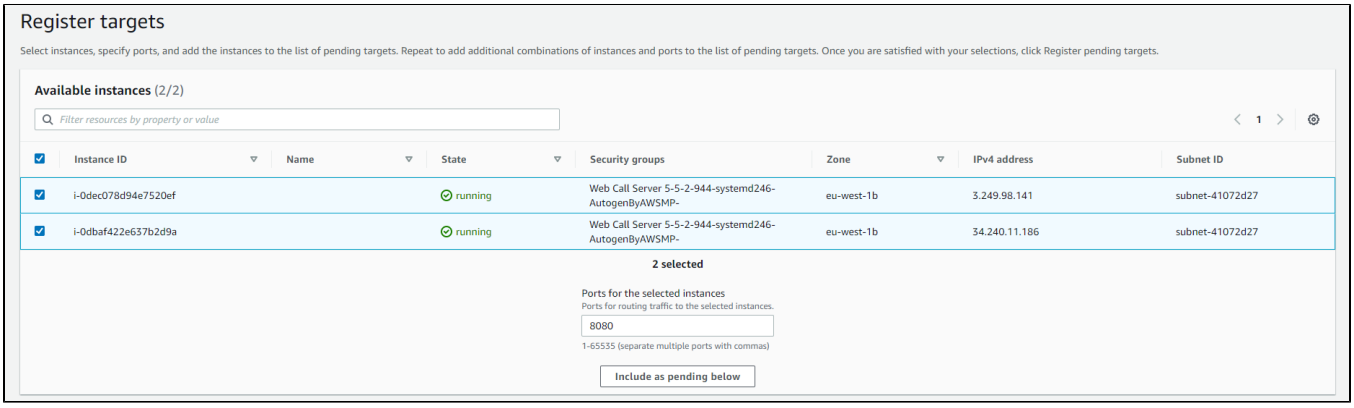
5 seconds

2-120

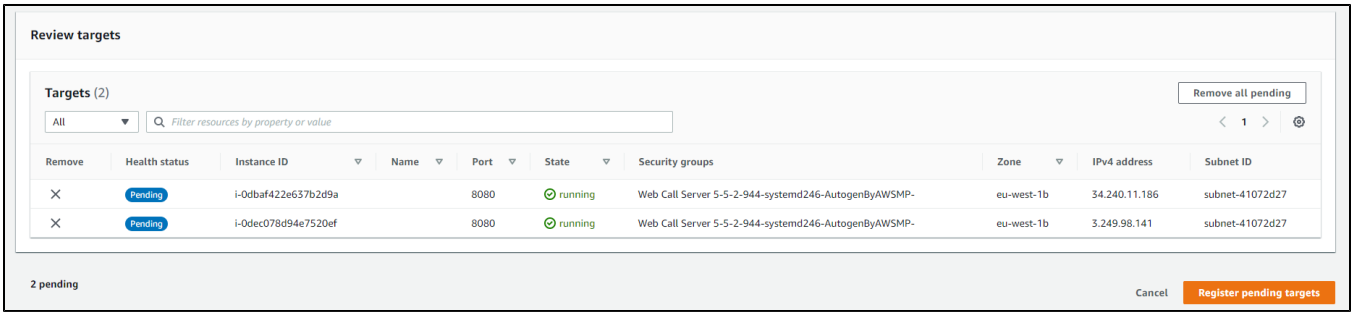
Then click Next



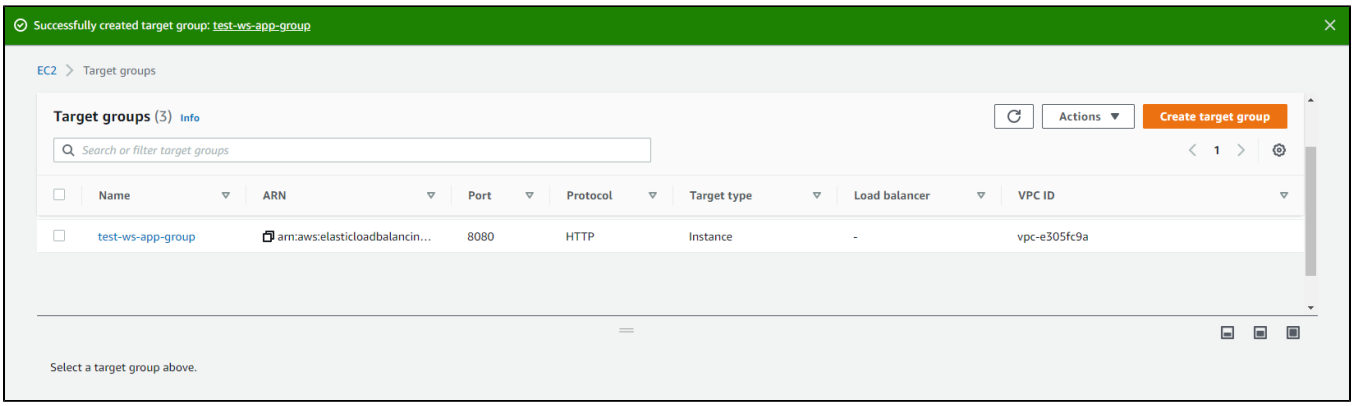
3.4. At "Register targets" page select instances as needed and click "Include as pending below"



Then click "Register pending targets"



Target group is created

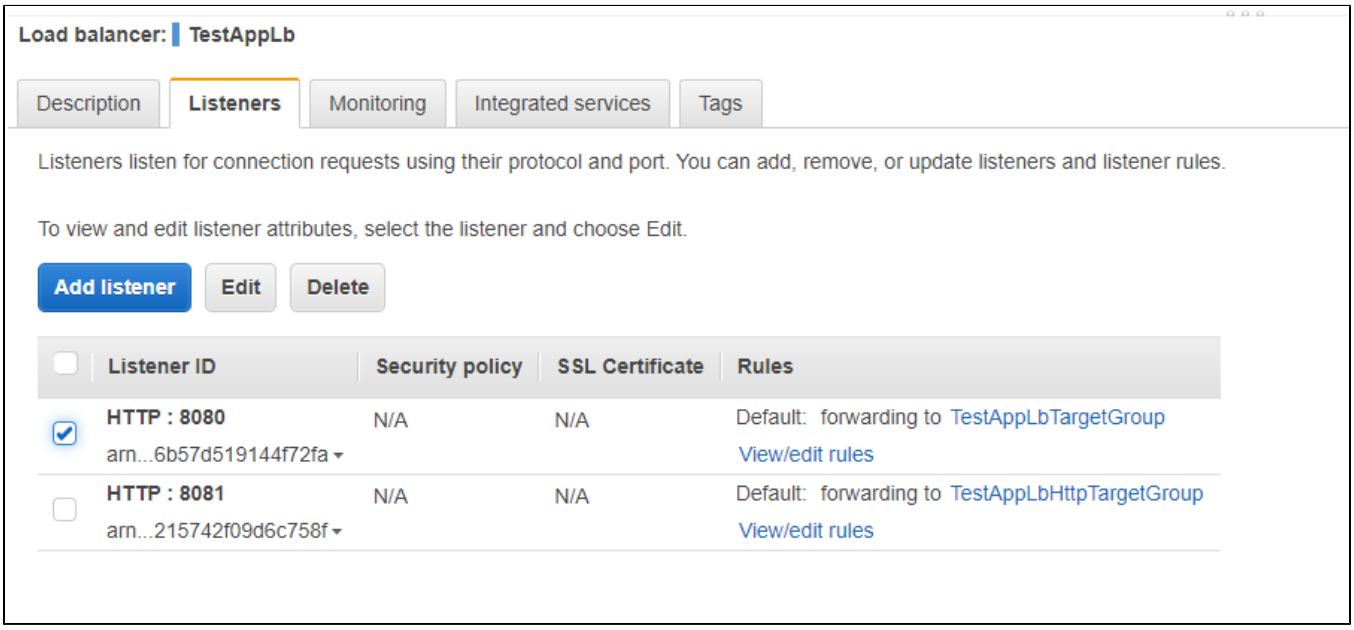


Application load balancer using this group will forward requests to it after at least one of the group instances passes health check.

4. Listener parameters configuration

If Application Load Balancer is created to use in Autoscaling group (see below), HTTPS listener cannot be configured on creation, only HTTP. In this case, listener parameters shoul be changed.

4.1. In EC2 Console section "Load balancers - Load balancers" choose "Listeners" tab for load balancer to configure. Choose Websocket listener and click Edit



4.2. Choose HTTPS protocol and set Secure Websocket port (8443 for example)

Edit listener

arn:aws:elasticloadbalancing:eu-north-1:170365852928:listener/app/TestAppLb/8e650022f70c3d2f/6b57d519144f72fa

Listener details

A listener is a process that checks for connection requests, using the protocol and port you configure. Traffic received by the listener is then routed per your specification. You can specify multiple rules and multiple certificates per listener after the load balancer is created.

Protocol: HTTPS

Port: 8443

1-65535

Default actions Info

Specify the default actions for traffic on this listener. Default actions apply to traffic that does not meet the conditions of rules on your listener. Rules can be configured after the listener is created.

1. Forward to Info

Remove

Target group

Weight (0-999)

TestAppLbTargetGroup HTTP

Target type: Instance, IPv4

1

Traffic distribution: 100%

Select a target group

0

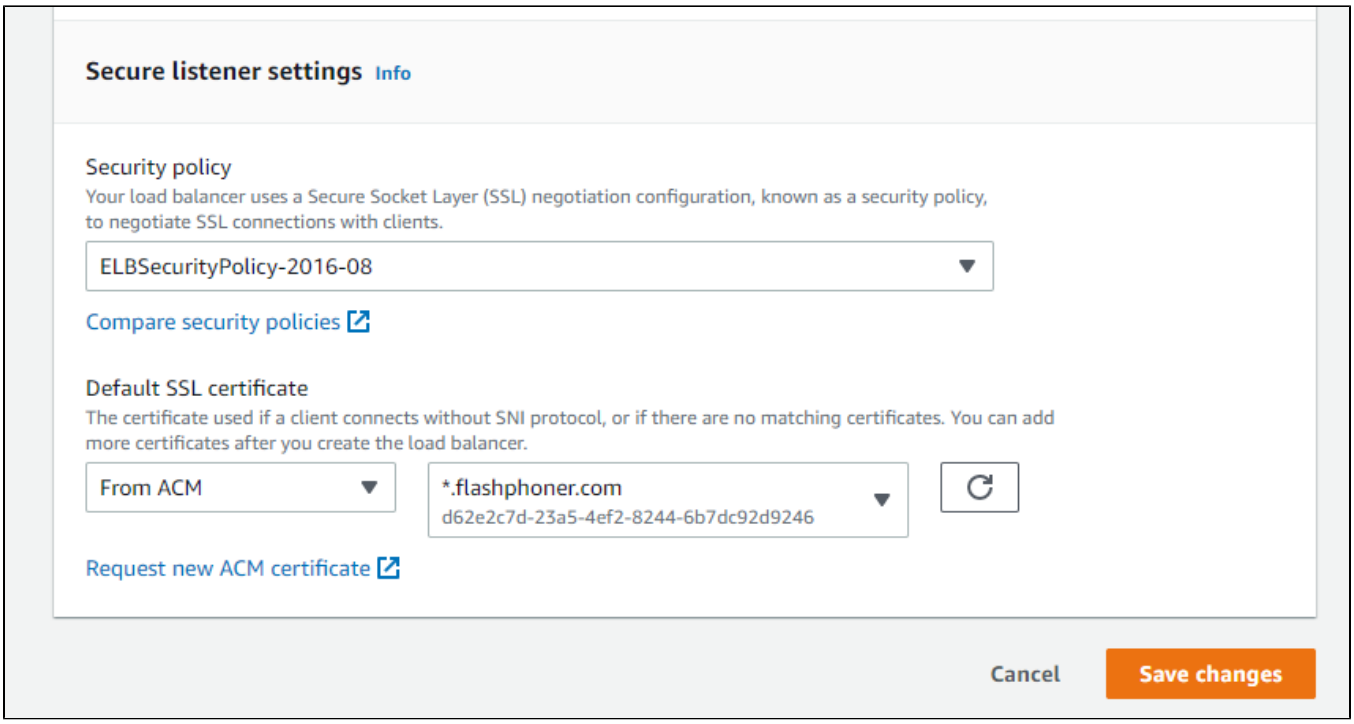
Create target group

Enable group-level stickiness Info

If you enable stickiness for your target group, requests routed to it remain in the same group for the duration you specify.

Add action

4.3. In "Secure listener settings" section choose SSL certificate to use with domain asigned to load balancer entry point or create a new one. Then click "Save changes"



Load balancer listener parameters are changed and will be applied immediately

Launching AWS Auto Scaling group with load balancing from AWS Marketplace AMI

Load balancer with autoscaling deployment from AWS Marketplace AMI can be useful for periodic servers group launching, for example, during the event (lasting for hours, days, weeks). In this case, WCS monthly subscription may be more expensive then AWS hourly payment, therefore it is recommeded to use AWS Marketplace AMI.

1. Create launch template

1.1. In EC2 Console go to "Instances - Launch Templates" section and click "Create launch template". Launch template creation wizard will open. Enter template name and description

EC2 > Launch templates > Create launch template

Create launch template

Creating a launch template allows you to create a saved instance configuration that can be reused, shared and launched at a later time. Templates can have multiple versions.

Launch template name and description

Launch template name - *required*

TestTemplate

Must be unique to this account. Max 128 chars. No spaces or special characters like '&', '*', '@'.

Template version description

Test autoscaling launch template

Max 255 chars

Auto Scaling guidance Info

Select this if you intend to use this template with EC2 Auto Scaling

☐ Provide guidance to help me set up a template that I can use with EC2 Auto Scaling

▶ **Template tags**

▶ **Source template**

1.2. Choose latest FlashphonerWebCallServer image

Launch template contents

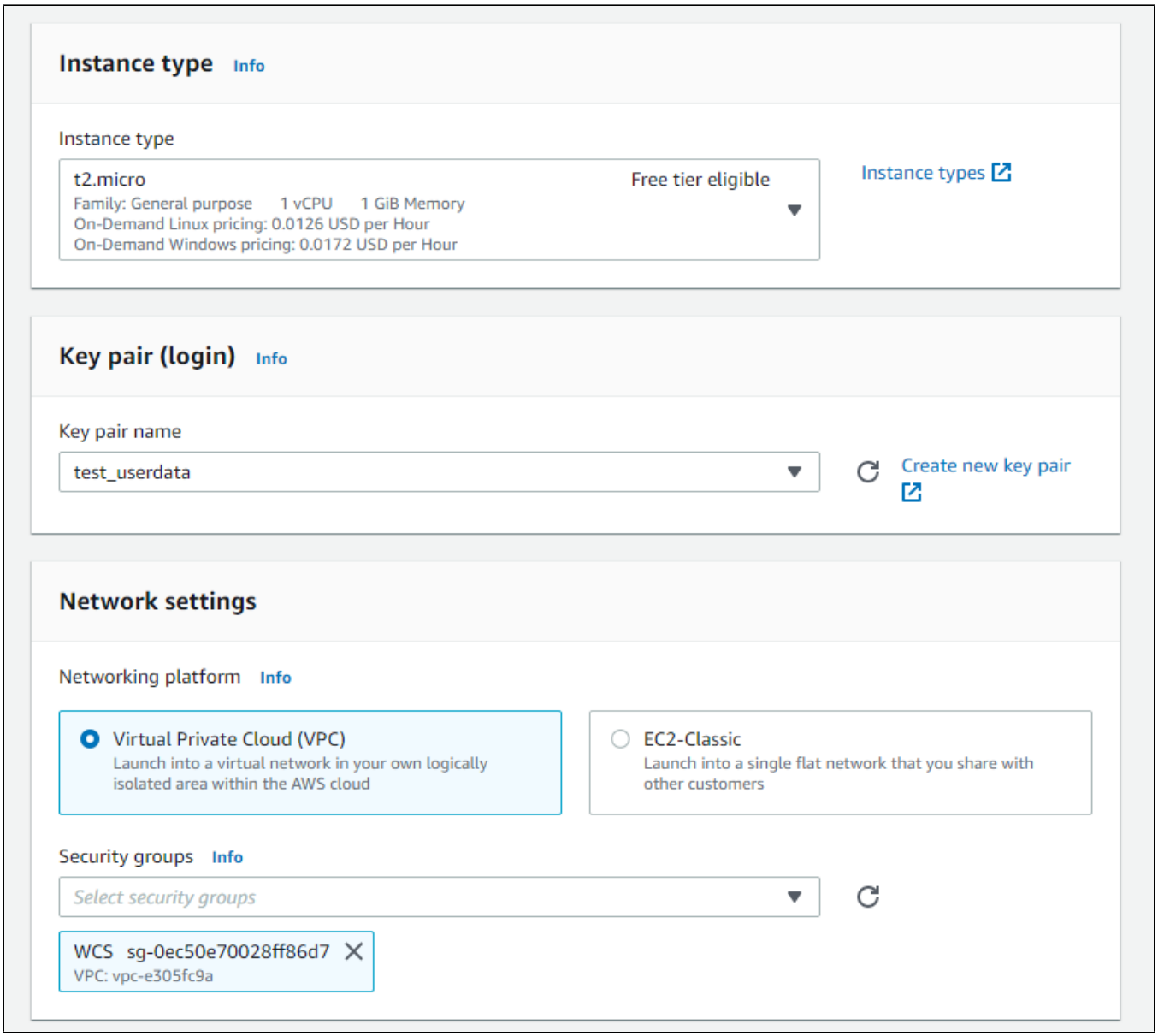
Specify the details of your launch template below. Leaving a field blank will result in the field not being included in the launch template.

Amazon machine image (AMI) Info

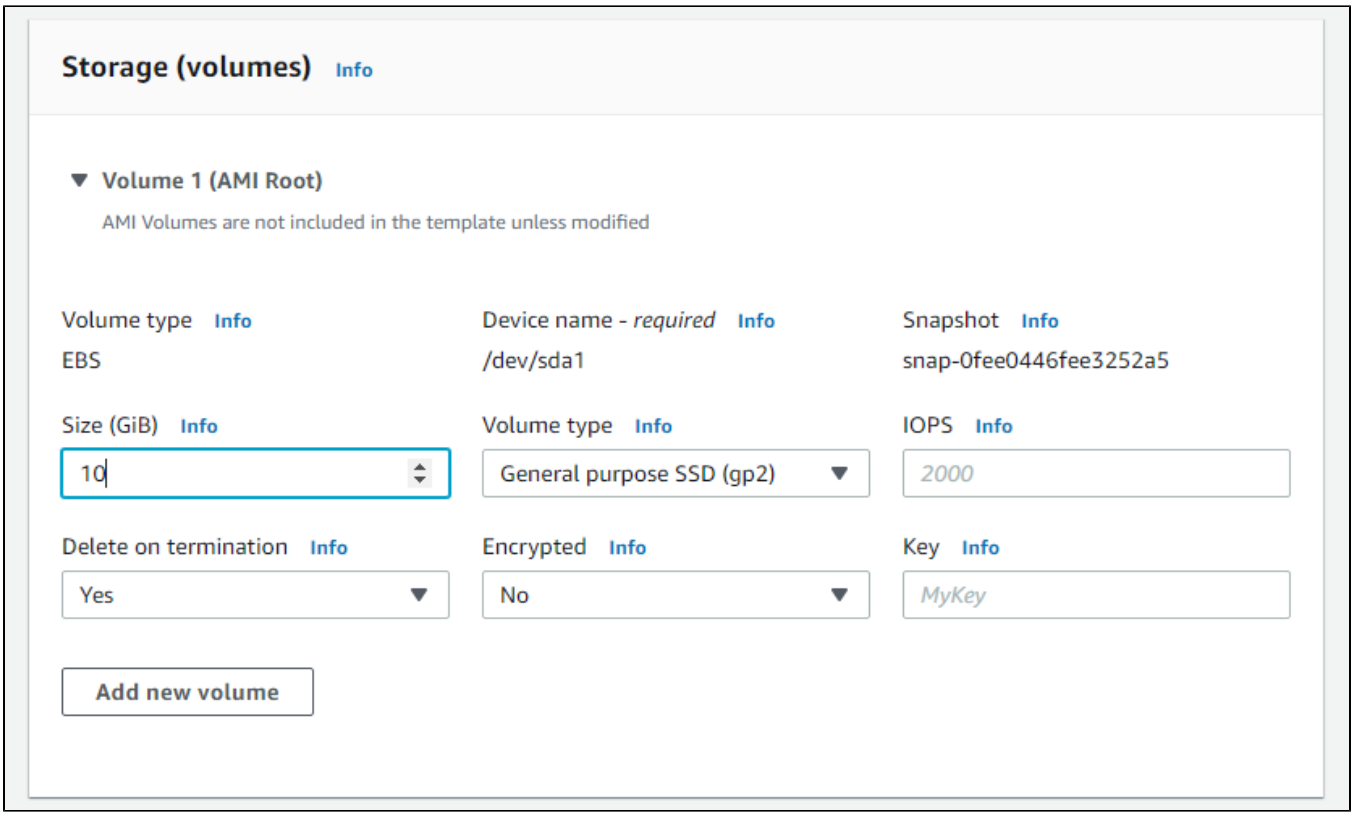
AMI

FlashphonerWebCallServer-5.2.629-x86_64-hourly-01e37234-6170-4b8d-98b...
ami-035ddfe555ad2e6f8
Catalog: AWS Marketplace architecture: 64-bit (x86) virtualization: hvm

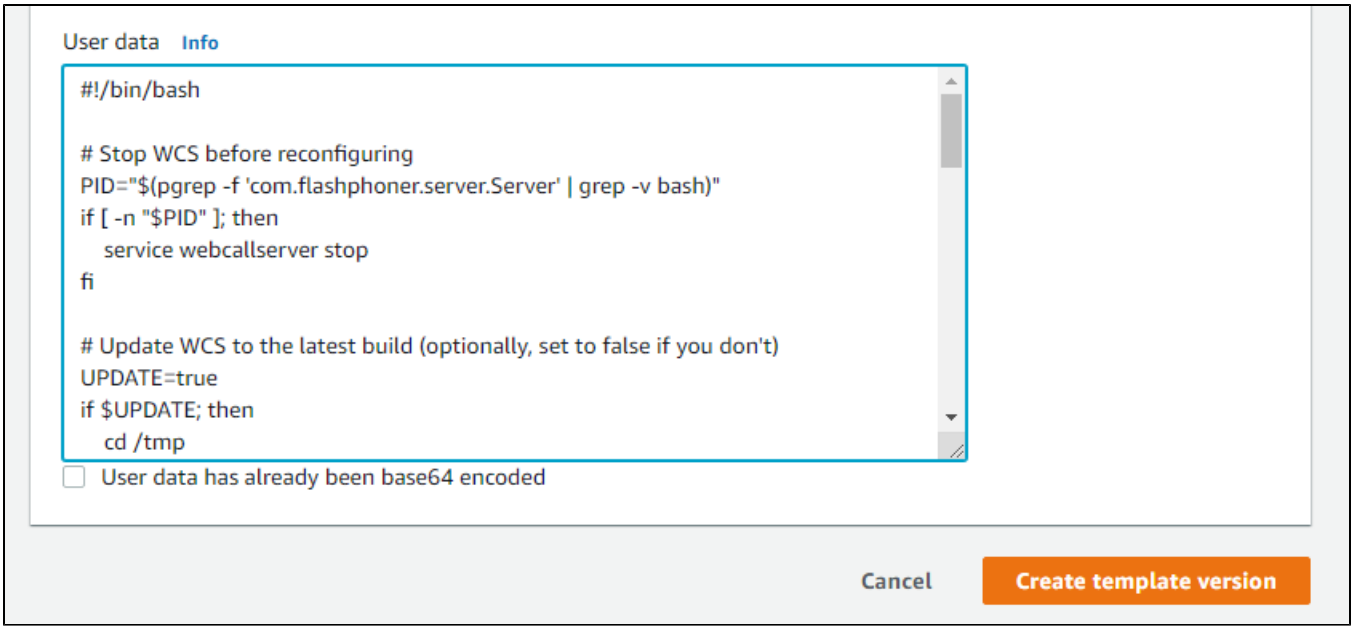
1.3. Choose instance type, key pair for SSH access to an instance, security group



1.4. Set disk size and parameters for instances



1.5. Expand "Advanced details" section. Insert custom update and setup script to "User data" text box

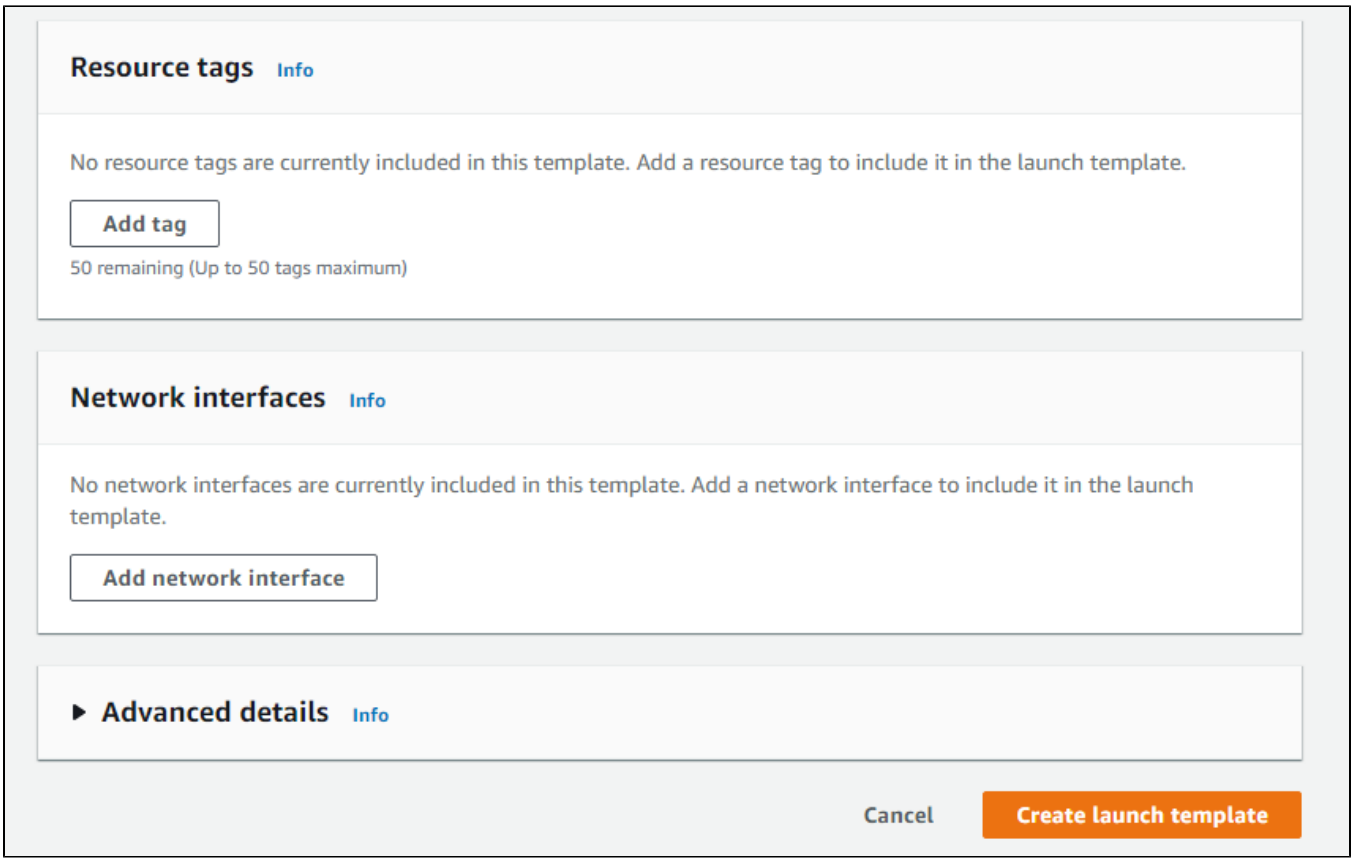


The setup script example to update WCS to latest build and to configure CDN Edge server for WebRTC playback

Edge setup script

```
#!/bin/bash

# Stop WCS before reconfiguring
PID="$(pgrep -f 'com.flashphoner.server.Server' | grep -v bash)"
if [ -n "$PID" ]; then
    service webcallserver stop
fi

# Update WCS to the latest build (optionally, set to false if you don't)
UPDATE=true
if $UPDATE; then
    cd /tmp
    wget --timeout=10 --no-check-certificate https://flashphoner.com/download-wcs5.2-server.tar.gz -O wcs5-
server.tar.gz
    if [ $? -eq 0 ]; then
        mkdir -p FlashphonerWebCallServer-5.2-latest && tar xzf wcs5-server.tar.gz -C FlashphonerWebCallServer-
5.2-latest --strip-components 1
        cd FlashphonerWebCallServer-5.2-latest
        chmod +x install.sh
        ./install.sh -silent
        cd ..
        rm -rf FlashphonerWebCallServer-5.2-latest wcs5-server.tar.gz
    fi
fi

# Configuration setup
WCS_CONFIG=/usr/local/FlashphonerWebCallServer/conf/flashphoner.properties
JVM_CONFIG=/usr/local/FlashphonerWebCallServer/conf/wcs-core.properties
USERS_CONFIG=/usr/local/FlashphonerWebCallServer/conf/database.yml

#CDN settings
CDN_ROLE=edge
CDN_IP=0.0.0.0
CDN_POINT_OF_ENTRY=172.31.43.82
echo -e "\ncdn_enabled=true" >> $WCS_CONFIG
echo -e "\ncdn_ip=$CDN_IP" >> $WCS_CONFIG
echo -e "\ncdn_role=$CDN_ROLE" >> $WCS_CONFIG
echo -e "\ncdn_point_of_entry=$CDN_POINT_OF_ENTRY" >> $WCS_CONFIG
echo -e "\ncdn_nodes_resolve_ip=false" >> $WCS_CONFIG

# Configure heap settings
HEAP_SIZE=512m
sed -i -e "s/^\(-Xmx\).*\$/\1$HEAP_SIZE/" $JVM_CONFIG

# Disable demo user (optionally, set to true if you want to disable)
DISABLE_DEMO=false
if $DISABLE_DEMO; then
 if grep "demo:" $USERS_CONFIG > /dev/null 2>&1; then
  sed -i -e "/demo:/s/active:\ true/active:\ false/" $USERS_CONFIG
 fi
fi

# Start WCS after reconfiguring
PID="$(pgrep -f 'com.flashphoner.server.Server' | grep -v bash)"
if [ -n "$PID" ]; then
    service webcallserver restart
else
    service webcallserver start
fi

# Disable internal firewall, ports are allowed/blocked on security group level
iptables -F
```

1.6. Click "Create launch template"



Launch template will be created

2. Create Auto scaling group

2.1. In EC2 Console go to "Instances - Auto Scaling Groups" section and click "Create an Auto Scaling Group". Autoscaling group creation wizard will open. Enter group name

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1
Choose launch template or configuration

Step 2
Configure settings

Step 3 *(optional)*
Configure advanced options

Step 4 *(optional)*
Configure group size and scaling policies

Step 5 *(optional)*
Add notifications

Step 6 *(optional)*
Add tags

Step 7
Review

Choose launch template or configuration Info

Specify a launch template that contains settings common to all EC2 instances that are launched by this Auto Scaling group. If you currently use launch configurations, you might consider migrating to launch templates.

Name

Auto Scaling group name
Enter a name to identify the group.

TestAutoscalingGroup

Must be unique to this account in the current Region and no more than 255 characters.

Launch template Info

Switch to launch configuration

Launch template
Choose a launch template that contains the instance-level settings, such as the Amazon Machine Image (AMI), instance type, key pair, and security groups.

TestTemplate

Create a launch template

Version

Default (1)

Create a launch template version

Description	Launch template	Instance type
Test autoscaling launch template	TestTemplate lt-04d44d426947bae18	-
AMI ID	**Security groups**	**Security group IDs**
ami-035ddfe555ad2e6f8	-	-
Key pair name		

2.2 Choose launch template, set "Latest" version

Launch template Info

Switch to launch configuration

Launch template

Choose a launch template that contains the instance-level settings, such as the Amazon Machine Image (AMI), instance type, key pair, and security groups.

TestTemplate

Create a launch template

Version

Latest (2)

Create a launch template version

Description	Launch template	Instance type
-	TestTemplate lt-04d44d426947bae18	t2.micro
AMI ID	**Security groups**	**Security group IDs**
ami-035ddfe555ad2e6f8	-	sg-0ec50e70028ff86d7
Key pair name		
test_userdata		

Additional details

Storage (volumes)	Date created
/dev/sda1	Tue Jul 14 2020 14:02:03 GMT+0700 (Novosibirsk Standard Time)

2.3. Set instances distributon percentage (on demand/spot). By default 70 % on demand will be set, it is recommended to raise this value to 100 %

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1
Choose launch template or configuration

Step 2
Configure settings

Step 3 *(optional)*
Configure advanced options

Step 4 *(optional)*
Configure group size and scaling policies

Step 5 *(optional)*
Add notifications

Step 6 *(optional)*
Add tags

Step 7
Review

Configure settings Info

Configure the settings below. Depending on whether you chose a launch template, these settings may include options to help you make optimal use of EC2 resources.

Purchase options and instance types Info

Adhere to launch template
The launch template determines the purchase option (On-Demand or Spot) and instance type.

Combine purchase options and instance types
Specify how much On-Demand and Spot capacity to launch and multiple instance types (optional). This choice is most helpful for optimizing the scale and cost for a fleet of instances.

Instances distribution

Optional On-Demand base
Specify how much On-Demand capacity the Auto Scaling group should have for its base portion. The maximum group size will be increased (but not decreased) to this value.

0 On-Demand Instances

On-Demand percentage above base
Define the percentage split of On-Demand Instances and Spot Instances for your additional capacity beyond the base portion.

70 % On-Demand

30 % Spot

Spot allocation strategy per Availability Zone

- Capacity optimized - Launch Spot Instances optimally based on the available Spot capacity (recommended)
- Lowest price - Launch Spot Instances from the lowest priced instance pools

Number of lowest priced Spot Instance pools to diversify across

2

Value must be between 1 and 20

2.4 Choose instance types

Instance types Info

Choose the instance types that best suit the needs of your application.

Primary instance type | Weight Info

1. t2.micro
1vCPU 1 Gib Memory

Your launch template does not specify an instance type. As a result, Adhere to launch template cannot be chosen. You can continue by adding an instance type above.

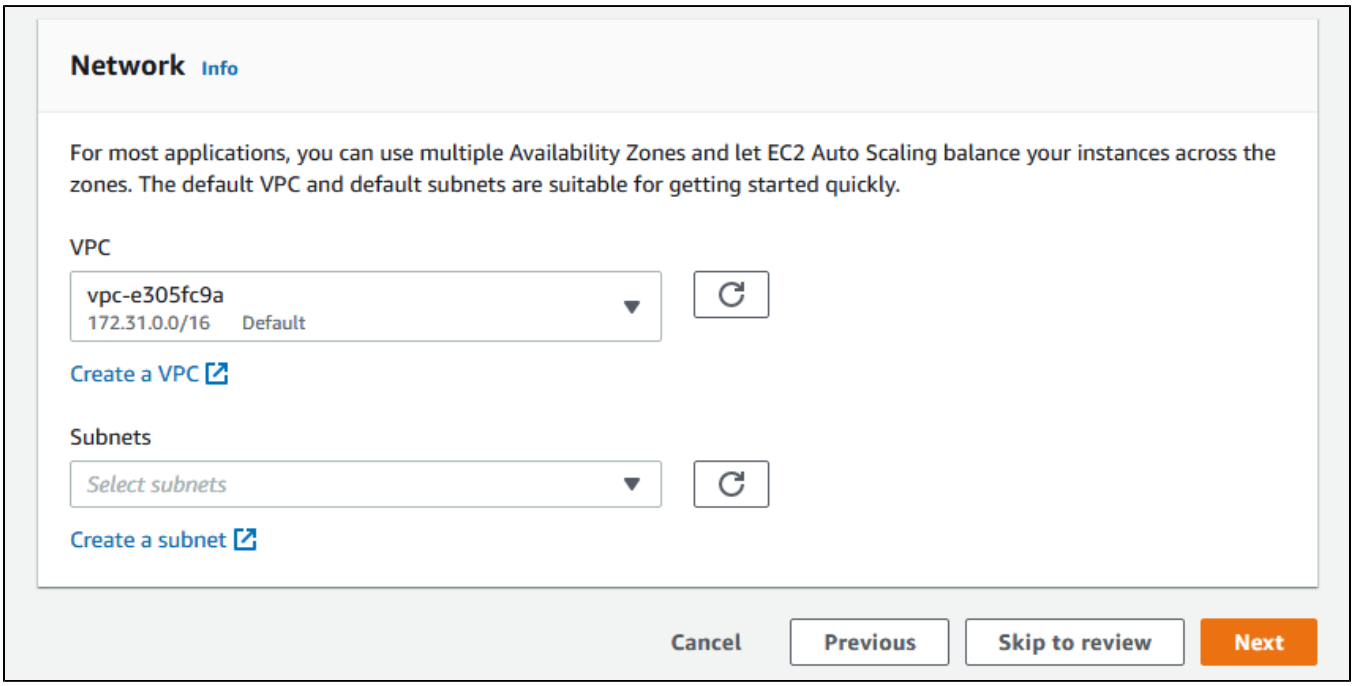
Additional instance types

Redo recommendations

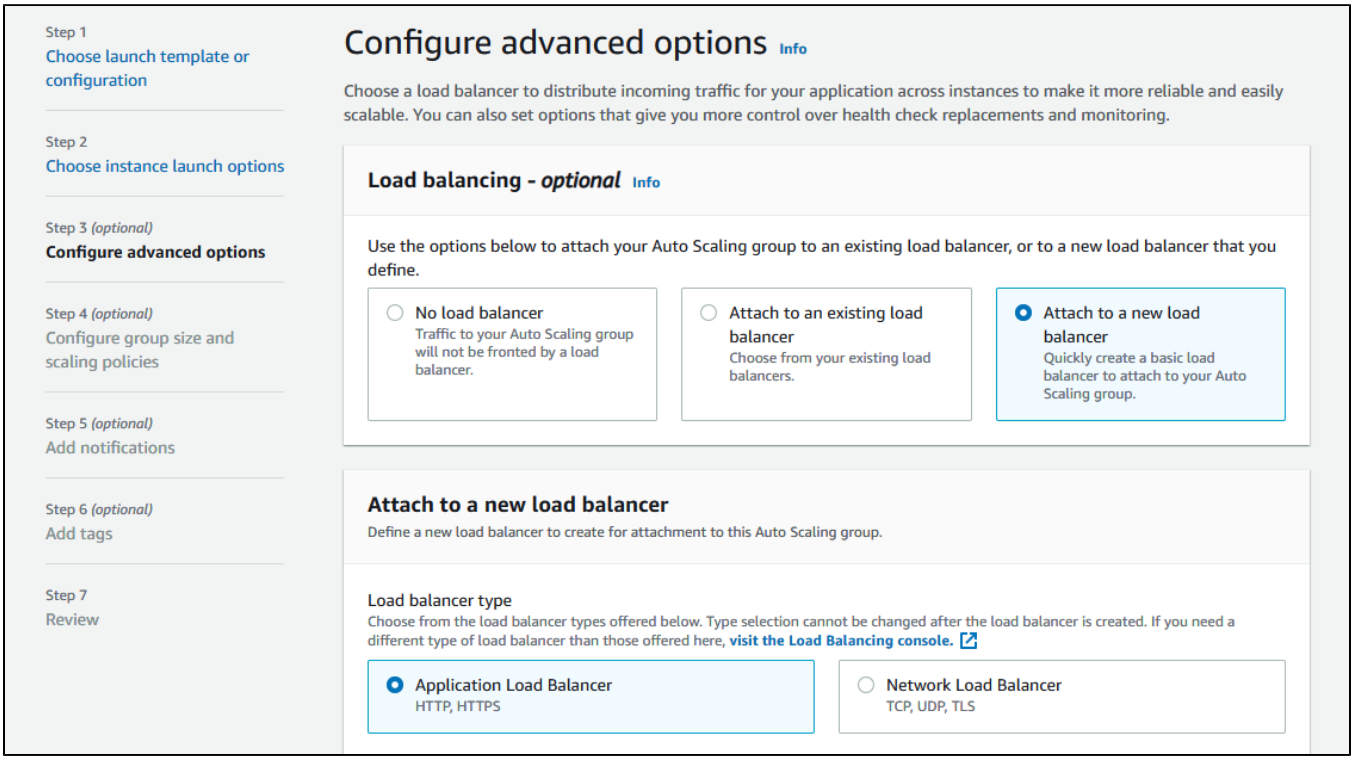
2. t2.micro
1vCPU 1 Gib Memory

Add instance type

2,5. Choose VPC and subnets for instances



2.6.Choose "Attach to a new load balancer"



2.7. Choose "Application Load balancer" type, set the name, choose Internet-facing, set availability zones and corresponding subnets

Attach to a new load balancer

Define a new load balancer to create for attachment to this Auto Scaling group.

Load balancer type

Choose from the load balancer types offered below. Type selection cannot be changed after the load balancer is created. If you need a different type of load balancer than those offered here, **visit the Load Balancing console.**

- (●) Application Load Balancer — HTTP, HTTPS
- ( ) Network Load Balancer — TCP, UDP, TLS

Load balancer name

Name cannot be changed after the load balancer is created.

TestAppLb

Load balancer scheme

Scheme cannot be changed after the load balancer is created.

- ( ) Internal
- (●) Internet-facing

Network mapping

Your new load balancer will be created using the same VPC and Availability Zone selections as your Auto Scaling group. You can select different subnets and add subnets from additional Availability Zones.

VPC

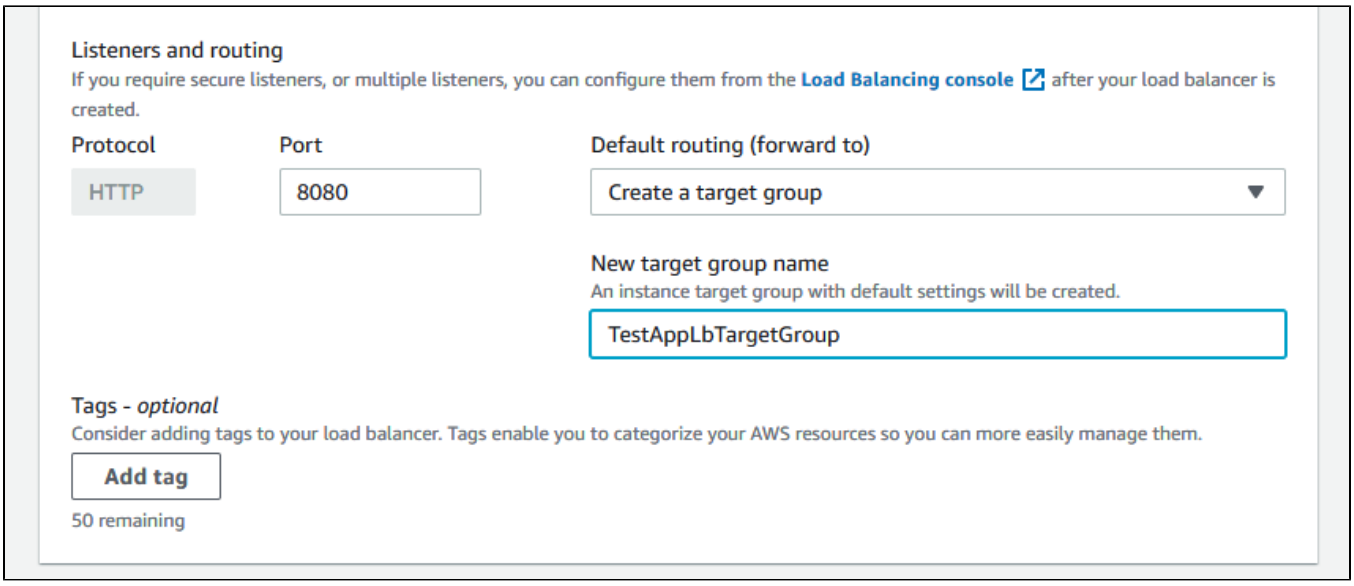
vpc-5e65c237

Availability Zones and subnets

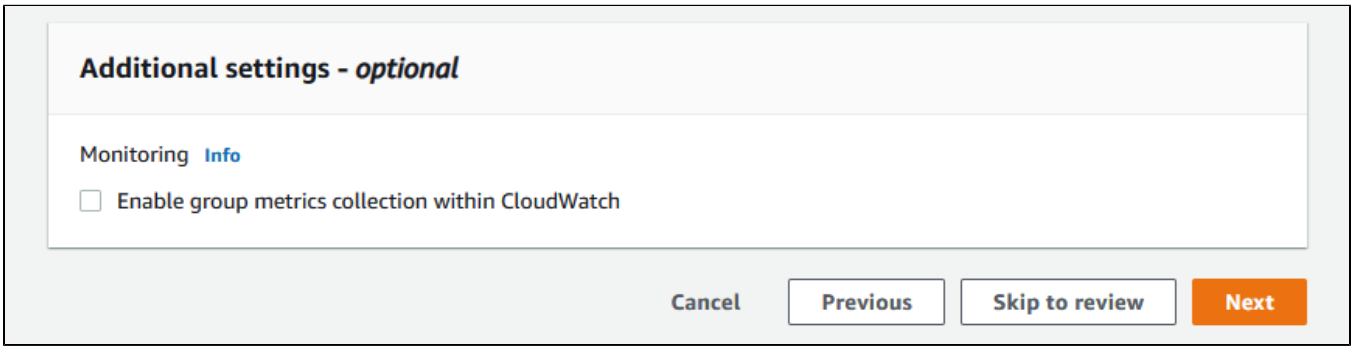
You must select a single subnet for each Availability Zone enabled. Only public subnets are available for selection to support DNS resolution.

[x] eu-north-1a	subnet-d2cb6fbb
[ ] eu-north-1c	*Select a subnet*
[ ] eu-north-1b	*Select a subnet*

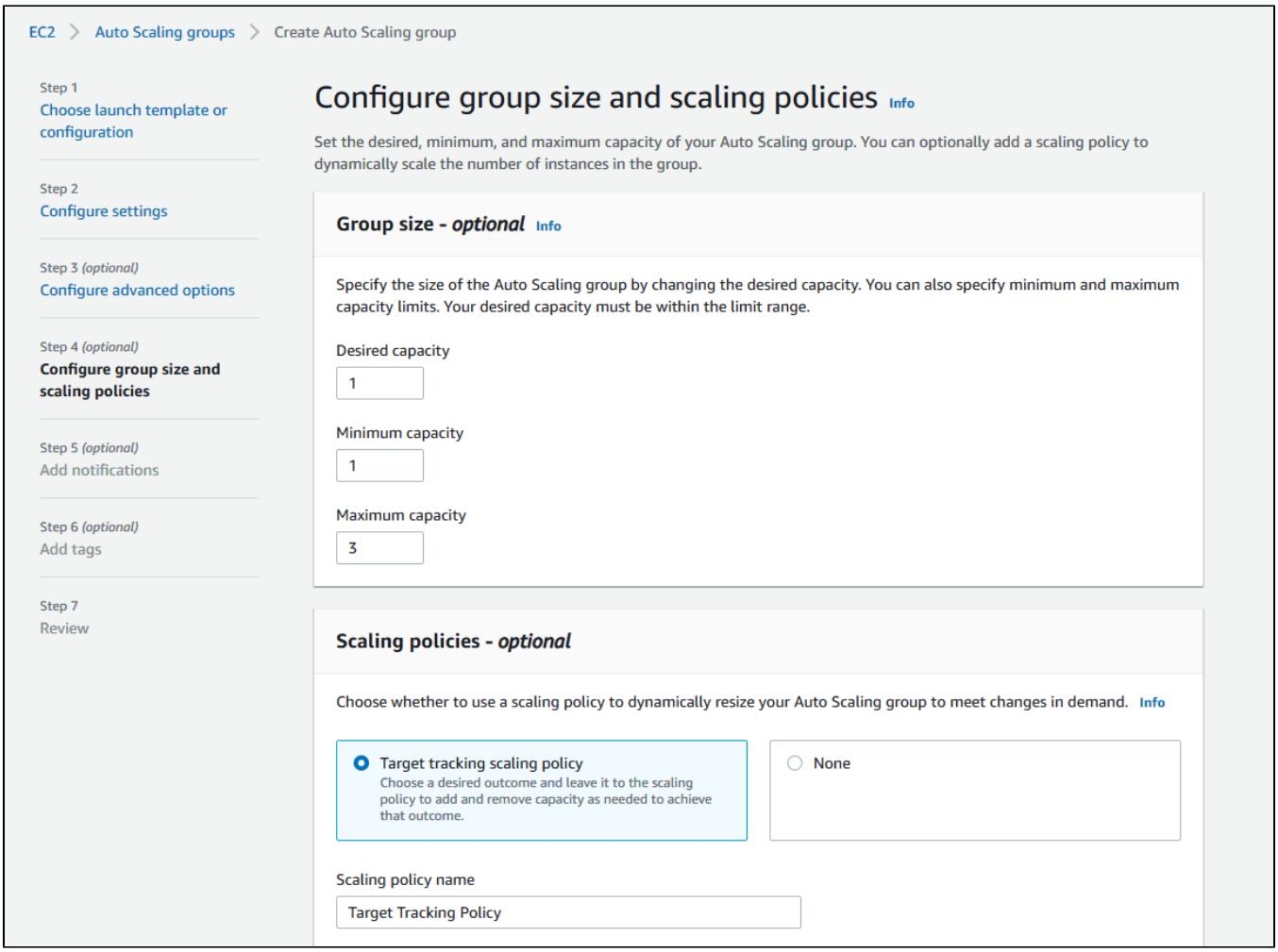
2.8. In "Listeners and routing" section set Websocket port (8080), choose "Create a target group" and set the target group name to be created



Then click Next



2.9. Set the maximum group size



2.10. Select scaling policy by CPU utilization, set target value and instance warming time

Scaling policies - *optional*

Choose whether to use a scaling policy to dynamically resize your Auto Scaling group to meet changes in demand. Info

- (selected) **Target tracking scaling policy**
  Choose a desired outcome and leave it to the scaling policy to add and remove capacity as needed to achieve that outcome.
- (not selected) **None**

Scaling policy name

Target Tracking Policy

Metric type

Average CPU utilization

Target value

80

Instances need

60 seconds warm up before including in metric

☐ Disable scale in to create only a scale-out policy

Instance scale-in protection - *optional*

Instance scale-in protection

If protect from scale in is enabled, newly launched instances will be protected from scale in by default.

☐ Enable instance scale-in protection

Cancel | Previous | Skip to review | Next

2.11. Review group parameters

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1
Choose launch template or configuration

Step 2
Configure settings

Step 3 *(optional)*
Configure advanced options

Step 4 *(optional)*
Configure group size and scaling policies

Step 5 *(optional)*
Add notifications

Step 6 *(optional)*
Add tags

Step 7
Review

Review Info

Step 1: Choose launch template or configuration

Edit

Group details

Auto Scaling group name
TestAutoscalingGroup

Launch template

Launch template	Version	Description
TestTemplate lt-04d44d426947bae18	Latest	

Step 2: Configure settings

Edit

Purchase options and instance types

Instances distribution

On-Demand base	On-Demand and Spot percentages	Spot allocation strategy
Designate the first 0 instances as On-Demand	100 % On-Demand 0 % Spot	Capacity optimized

Instance types

Instance type	vCPUs	Memory	Network performance
1. t2.micro	1 vCPU	1 GiB	Low to Moderate

2.12. Click "Create Auto Scaling group"

Step 5: Add notifications

Edit

Notifications

No notifications

Step 6: Add tags

Edit

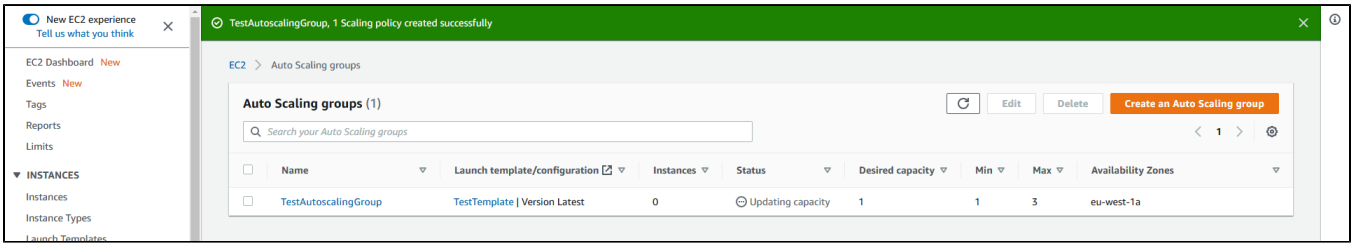
Tags (0)

Key	Value	Tag new instances
	No tags	

Cancel

Create Auto Scaling group

Autoscaling group will be created, and one instnce will be launched



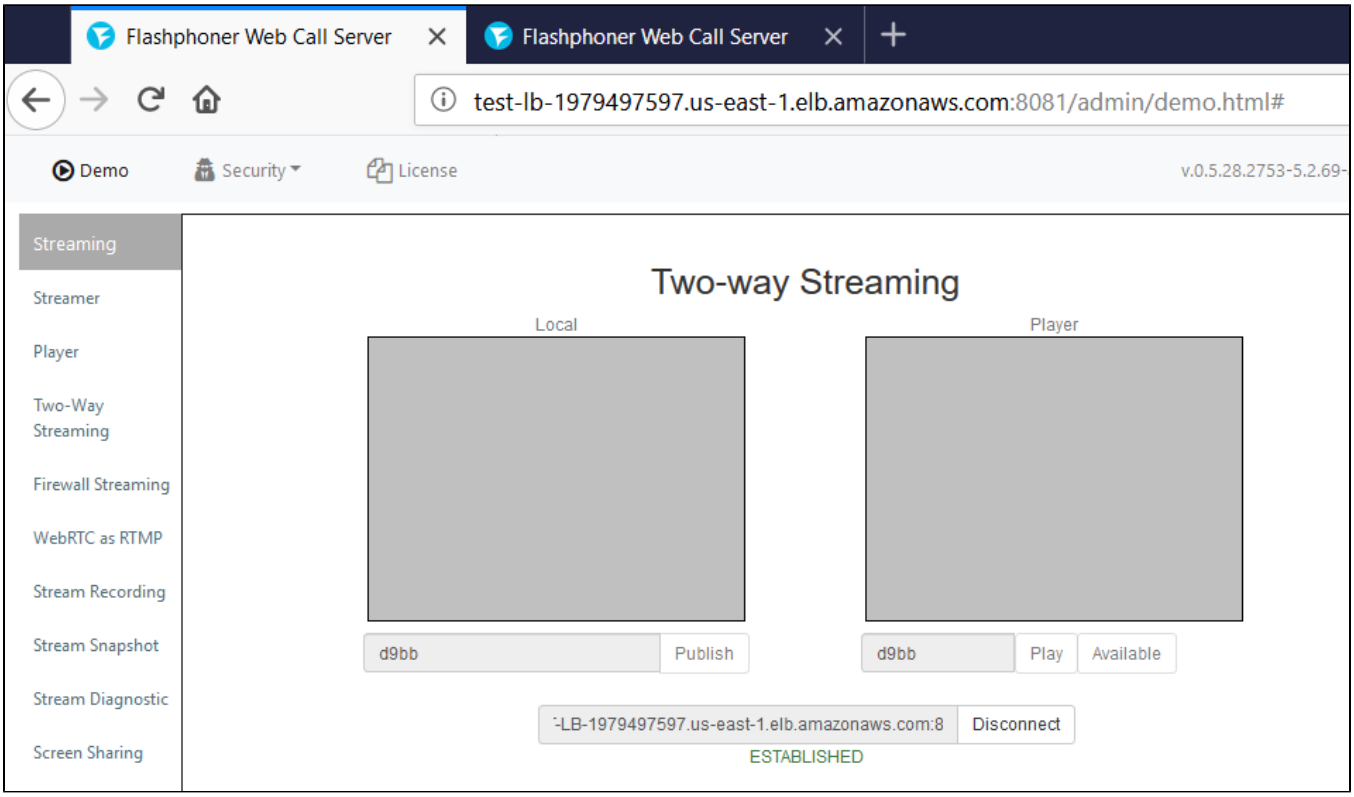
2.13. Configure load balancer listener as described above

Testing

If load balancer has no running instances, then a new instance will be started when an auto scaling group receiving traffic from the load balancer is created. More instances will be started in case scaling is triggered. (For testing purposes, streaming with transcoding – e.g., streaming RTMP to auto created mixer – can be used to load server CPU.) All the started instances will be auto added to the corresponding load balancer.

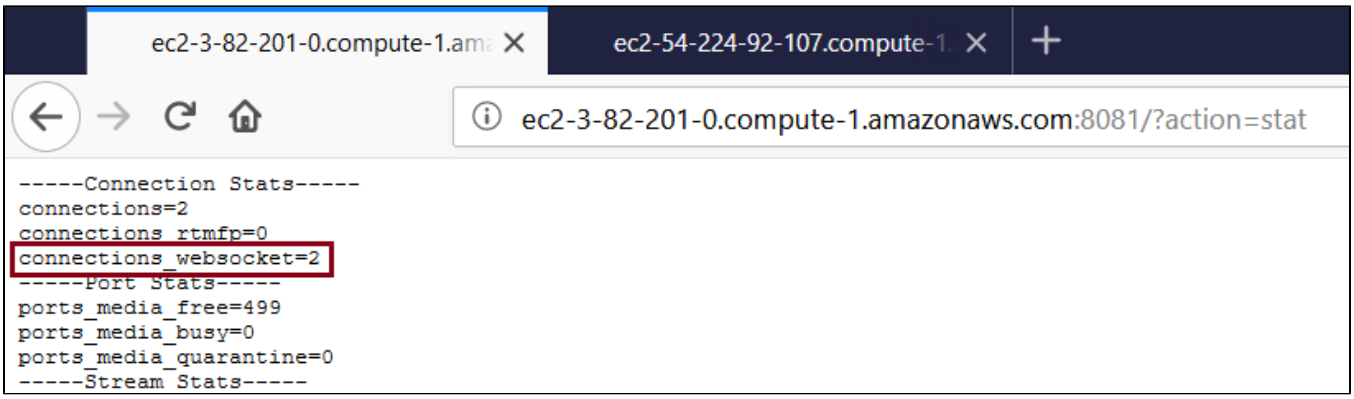
When an instance (one or more of the added to the balancer) is in service, ws-connection can be done to, e.g., ws://<Load balancer DNS name>:8080.

A demo – e.g., Two-way Streaming - example (opened either by the balancer or an instance address) can be used to establish ws-connection:



To verify that the connections are distributed between active load balancer instances, use the stats page: http://WCS_ADDRESS:8081/?action=stat

Open the page for each of the instances to see the connection_websocket number:


ec2-3-82-201-0.compute-1.am
ec2-54-224-92-107.compute-1.
ec2-3-82-201-0.compute-1.amazonaws.com:8081/?action=stat
-----Connection Stats-----
connections=2
connections_rtmfp=0
connections_websocket=2
-----Port Stats-----
ports_media_free=499
ports_media_busy=0
ports_media_quarantine=0
-----Stream Stats-----