Deploying WCS with CloudFormation

Overview

AWS CloudFormation allows to deploy cloud instances stacks by a certain template. Thus, a simple WCS CDN can be deployed for example. In this case, WCS update to the latest build and instance setup can be made with UserData scripts.

CloudFormation template example to deploy CDN

Below, there is the CloudFormation template example to deploy a simplest CDN of two WCS instances: Origin and Edge. The template alllows:

- to chooseWCS AMI from AWS Marketplace, or Amazon Linux 2, Ubuntu 18.04 andother supported OS AMI as basic image
- to install Java 14 if necessary
- to install or update WCS to the lates build if necessary

CloudFormation example template to deploy WCS CDN of one Origin and one Edge

```
AWSTemplateFormatVersion: "2010-09-09"
Description: "Create WCS CDN stack"
Parameters:
  KeyName:
    Description: "Name of an existing EC2 KeyPair to enable SSH access to the instance"
    Type: AWS::EC2::KeyPair::KeyName
    ConstraintDescription: "must be the name of an existing EC2 KeyPair"
  InstanceName:
    Description: "Name of EC2 instance"
    Type: String
    ConstraintDescription: "must be a valid EC2 instance string name"
  InstanceType:
    Description: "Basic EC2 instance type"
    Type: String
    Default: m5.xlarge
    AllowedValues: [t1.micro, t2.nano, t2.micro, t2.small, t2.medium, t2.large,
      m1.small, m1.medium, m1.large, m1.xlarge,
      m2.xlarge, m2.2xlarge, m2.4xlarge,
      m3.medium, m3.large, m3.xlarge, m3.2xlarge,
      m4.large, m4.xlarge, m4.2xlarge, m4.4xlarge, m4.10xlarge, m5.xlarge,
      c1.medium, c1.xlarge, c3.large, c3.xlarge, c3.2xlarge, c3.4xlarge, c3.8xlarge,
      c4.large, c4.xlarge, c4.2xlarge, c4.4xlarge, c4.8xlarge,
      g2.2xlarge, g2.8xlarge, r3.large, r3.xlarge, r3.2xlarge, r3.4xlarge, r3.8xlarge,
      i2.xlarge, i2.2xlarge, i2.4xlarge, i2.8xlarge,
      d2.xlarge, d2.2xlarge, d2.4xlarge, d2.8xlarge,
      hi1.4xlarge, hs1.8xlarge, cr1.8xlarge, cc2.8xlarge, cg1.4xlarge]
    ConstraintDescription: "must be a valid EC2 instance type"
  ImageId:
    Description: "Basic instance ami (WebCallServer 5.2.944 AMI by default, mapped by region)"
    Type: String
    Default: WCSAMI
    ConstraintDescription: "must be a valid AMI ID"
  VpcId:
    Type: String
    Description: "VpcId of your existing Virtual Private Cloud (VPC)"
  SubnetId:
    Type: String
    Description: "SubnetId of an existing subnet in your Virtual Private Cloud (VPC)"
  SSHLocation:
    Description: "The IP address range that can be used to SSH to the EC2 instances"
    Type: String
    MinLength: 9
    MaxLength: 18
    Default: 0.0.0.0/0
    AllowedPattern: (\d{1,3})\.(\d{1,3})\.(\d{1,3})\.(\d{1,3})/(\d{1,2})
    ConstraintDescription: "must be a valid IP CIDR range of the form x.x.x.x/x"
```

```
  JavaHeapSize:
    Description: "Maximum Java heap size in megabytes (2048m) or gigabytes (2g), 1024m by default"
    Type: String
    Default: 1024m
  UpdateWCS:
    Description: "Update WCS to the latest build"
    Type: String
    Default: true
    ConstraintDescription: "must be true or false"
  WCSLicense:
    Description: "WCS License key to activate (optional, if you do not use Marketplace AMI)"
    Type: String
    Default: ""
  InstallJava:
    Description: "Java installation helper script. Do not change in wizard!"
    Type: String
    Default: |
      JAVA_CMD=`command -v java 2>/dev/null`
      if [[ -z $JAVA_CMD ]]; then
        rm -rf jdk*
        curl -s https://download.java.net/java/GA/jdk14.0.1/664493ef4a6946b186ff29eb326336a2/7/GPL/openjdk-
14.0.1_linux-x64_bin.tar.gz | tar -zx
        if [ -d jdk-14.0.1/bin ]; then
          mkdir -p /usr/java
          [ -d /usr/java/jdk-14.0.1 ] && rm -rf /usr/java/jdk-14.0.1
          mv -f jdk-14.0.1 /usr/java
          if [ -d /usr/java/jdk-14.0.1/bin ]; then
            rm -f /usr/java/default
            ln -sf /usr/java/jdk-14.0.1 /usr/java/default
            update-alternatives --install /usr/bin/java java /usr/java/jdk-14.0.1/bin/java 1
            update-alternatives --install /usr/bin/jstack jstack /usr/java/jdk-14.0.1/bin/jstack 1
            update-alternatives --install /usr/bin/jcmd jcmd /usr/java/jdk-14.0.1/bin/jcmd 1
            update-alternatives --install /usr/bin/jmap jmap /usr/java/jdk-14.0.1/bin/jmap 1
            update-alternatives --set java /usr/java/jdk-14.0.1/bin/java
            update-alternatives --set jstack /usr/java/jdk-14.0.1/bin/jstack
            update-alternatives --set jcmd /usr/java/jdk-14.0.1/bin/jcmd
            update-alternatives --set jmap /usr/java/jdk-14.0.1/bin/jmap
            echo "JDK 14 installed" >> $DELPOY_LOG
          fi
        fi
      fi
  StopPreviousWCS:
    Description: "Stop previosly running WCS helper script. Do not change in wizard!"
    Type: String
    Default: |
      PID=`pgrep -f 'com.flashphoner.server.Server' | grep -v bash`
      if [ -n "$PID" ]; then
        systemctl stop webcallserver
      fi
  UpdateToLatestWCS:
    Description: "WCS update to latest build helper script. Do not change in wizard!"
    Type: String
    Default: |
      # Check if WCS is installed, and install latest build if not
      if [ ! -f /usr/local/FlashphonerWebCallServer/bin/webcallserver ]; then
        echo "No WCS installed, will install latest build" >> $DEPLOY_LOG
        UPDATE=true
      fi
      echo "Update WCS: $UPDATE" >> $DEPLOY_LOG
      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
```

```
          echo "WCS updated to build $(cat /usr/local/FlashphonerWebCallServer/conf/WCS.version)" >> $DEPLOY_LOG
        fi
      fi
  ConfigureWCS:
    Description: "WCS configuration helper script. Do not change in wizard!"
    Type: String
    Default: |
      # Request keyframes from WebRTC publishers every 5 seconds
      echo -e "\n" >> $WCS_CONFIG
      echo -e "periodic_fir_request=true" >> $WCS_CONFIG
      # Disable RTMP keepalives to publish from OBS
      echo -e "keep_alive.enabled=websocket,rtmfp" >> $WCS_CONFIG
      # Configure heap settings
      sed -i -e "s/^\(-Xmx\).*\$/\1$HEAP_SIZE/" $JVM_CONFIG
      sed -i -e "s/^\(-Xms\).*\$/\1$HEAP_SIZE/" $JVM_CONFIG
  ActivateWCS:
    Description: "WCS activation helper script. Do not change in wizard!"
    Type: String
    Default: |
      if [[ ! -z $LICENSE ]]; then
        /usr/local/FlashphonerWebCallServer/bin/activation.sh $LICENSE
      fi
  StartWCS:
    Description: "WCS startup helper script. Do not change in wizard!"
    Type: String
    Default: |
      systemctl restart webcallserver
      # Disable internal firewall, ports are allowed/blocked on security group level
      iptables -F
      chown ec2-user $DEPLOY_LOG
  OriginCDNSetup:
    Description: "WCS Origin intsance setup helper script. Do not change in wizard!"
    Type: String
    Default: |
      echo -e "\n" >> $WCS_CONFIG
      echo -e "cdn_enabled=true" >> $WCS_CONFIG
      echo -e "cdn_ip=0.0.0.0" >> $WCS_CONFIG
      echo -e "cdn_point_of_entry=" >> $WCS_CONFIG
      echo -e "cdn_role=origin" >> $WCS_CONFIG
      echo -e "cdn_nodes_resolve_ip=false" >> $WCS_CONFIG
  EdgeCDNSetup:
    Description: "WCS Edge intsance setup helper script. Do not change in wizard!"
    Type: String
    Default: |
      echo -e "\n" >> $WCS_CONFIG
      echo -e "cdn_enabled=true" >> $WCS_CONFIG
      echo -e "cdn_ip=0.0.0.0" >> $WCS_CONFIG
      echo -e "cdn_point_of_entry=$ORIGIN_IP" >> $WCS_CONFIG
      echo -e "cdn_role=edge" >> $WCS_CONFIG
      echo -e "cdn_nodes_resolve_ip=false" >> $WCS_CONFIG
Mappings:
  WCSAMI:
    eu-north-1:
      AMI: ami-0cd89cf8212fd90b4
    ap-south-1:
      AMI: ami-0861cf9f8d387a5cf
    eu-west-3:
      AMI: ami-0f5d7f6dcaf0910e0
    eu-west-2:
      AMI: ami-0d61a966487038aeb
    eu-west-1:
      AMI: ami-01c249ebee9077dbc
    ap-northeast-2:
      AMI: ami-023e68299437cbf78
    ap-northeast-1:
      AMI: ami-0f01e9f19c3733d99
    sa-east-1:
      AMI: ami-01d3d7a07e6e5beda
    ca-central-1:
      AMI: ami-0aa76aec8c64e3d52
    ap-southeast-1:
```

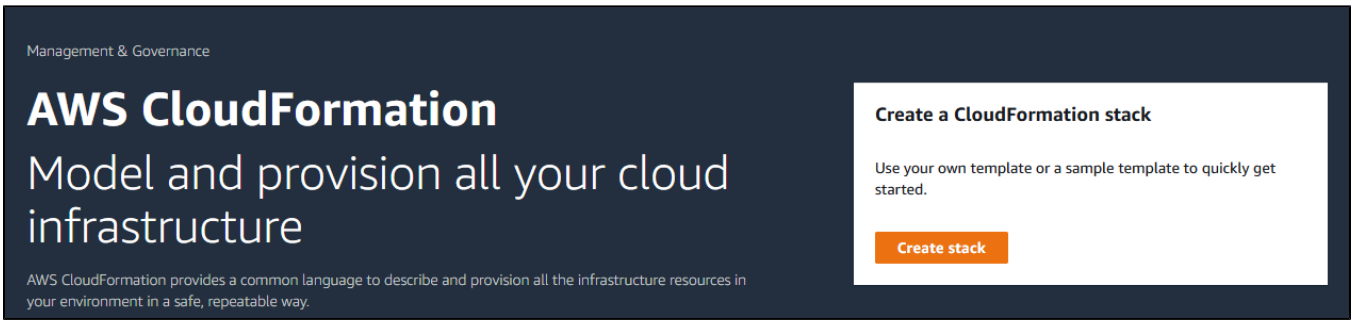
```
      AMI: ami-044fd54e788e44ddc
    ap-southeast-2:
      AMI: ami-0a4f9a18ad123d2ad
    eu-central-1:
      AMI: ami-0f785dd5a9571d373
    us-east-1:
      AMI: ami-038f9ebb3c87f88ac
    us-east-2:
      AMI: ami-0636213ac22f6ef45
    us-west-1:
      AMI: ami-0de64b6cac0f8d81c
    us-west-2:
      AMI: ami-0c8543b7418393ad5
Conditions:
  GetMarketplaceImage:
    Fn::Equals:
      - Ref: 'ImageId'
      - WCSAMI
Resources:
  WCSOriginInstance:
    Type: AWS::EC2::Instance
    Properties:
      Tags:
        - Key: "Name"
          Value:
            Fn::Join:
              - '-'
              - - !Ref 'InstanceName'
                - "edge"
      ImageId: !If [ GetMarketplaceImage,  !FindInMap [ WCSAMI, !Ref 'AWS::Region', AMI ], !Ref 'ImageId' ]
      InstanceType:
        Ref: 'InstanceType'
      SubnetId:
        Ref: 'SubnetId'
      SecurityGroupIds:
        - Ref: 'WCSSecurityGroup'
      KeyName:
        Ref: 'KeyName'
      Monitoring: false
      UserData:
        Fn::Base64:
          Fn::Sub: |
            #!/bin/bash
            # Declare variables
            UPDATE=${UpdateWCS}
            HEAP_SIZE=${JavaHeapSize}
            LICENSE=${WCSLicense}
            # Declare config files to change
            WCS_CONFIG=/usr/local/FlashphonerWebCallServer/conf/flashphoner.properties
            JVM_CONFIG=/usr/local/FlashphonerWebCallServer/conf/wcs-core.properties
            # Declare deployment log
            DEPLOY_LOG=/home/ec2-user/deploy.log
            # Install Java 14 if needed
            ${InstallJava}
            # Stop WCS before reconfiguring
            ${StopPreviousWCS}
            # Update WCS to the latest build
            ${UpdateToLatestWCS}
            # Configuration setup
            ${ConfigureWCS}
            # CDN setup
            ${OriginCDNSetup}
            # Activate WCS license if provided
            ${ActivateWCS}
            # Start WCS after reconfiguring
            ${StartWCS}
  WCSEdgeInstance:
    Type: AWS::EC2::Instance
    DependsOn:
      - WCSOriginInstance
    Properties:
```

```
    Tags:
      - Key: "Name"
        Value:
          Fn::Join:
            - '-'
            - - !Ref 'InstanceName'
              - "edge"
    ImageId: !If [ GetMarketplaceImage,  !FindInMap [ WCSAMI, !Ref 'AWS::Region', AMI ], !Ref 'ImageId' ]
    InstanceType:
      Ref: 'InstanceType'
    SubnetId:
      Ref: 'SubnetId'
    SecurityGroupIds:
      - Ref: 'WCSSecurityGroup'
    KeyName:
      Ref: 'KeyName'
    Monitoring: false
    UserData:
      Fn::Base64:
        Fn::Sub: |
          #!/bin/bash
          # Declare variables
          UPDATE=${UpdateWCS}
          HEAP_SIZE=${JavaHeapSize}
          LICENSE=${WCSLicense}
          ORIGIN_IP=${WCSOriginInstance.PrivateIp}
          # Declare config files to change
          WCS_CONFIG=/usr/local/FlashphonerWebCallServer/conf/flashphoner.properties
          JVM_CONFIG=/usr/local/FlashphonerWebCallServer/conf/wcs-core.properties
          # Declare deployment log
          DEPLOY_LOG=/home/ec2-user/deploy.log
          # Install Java 14 if needed
          ${InstallJava}
          # Stop WCS before reconfiguring
          ${StopPreviousWCS}
          # Update WCS to the latest build
          ${UpdateToLatestWCS}
          # Configuration setup
          ${ConfigureWCS}
          # CDN setup
          ${EdgeCDNSetup}
          # Activate WCS license if provided
          ${ActivateWCS}
          # Start WCS after reconfiguring
          ${StartWCS}
WCSSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    VpcId:
      Ref: 'VpcId'
    GroupDescription: "Enable SSH, websocket, web interface ports and media ports"
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp:
          Ref: 'SSHLocation'
      - IpProtocol: tcp
        FromPort: 554
        ToPort: 554
        CidrIp: 0.0.0.0/0
      - IpProtocol: tcp
        FromPort: 1935
        ToPort: 1935
        CidrIp: 0.0.0.0/0
      - IpProtocol: tcp
        FromPort: 3478
        ToPort: 3478
        CidrIp: 0.0.0.0/0
      - IpProtocol: tcp
        FromPort: 8080
```

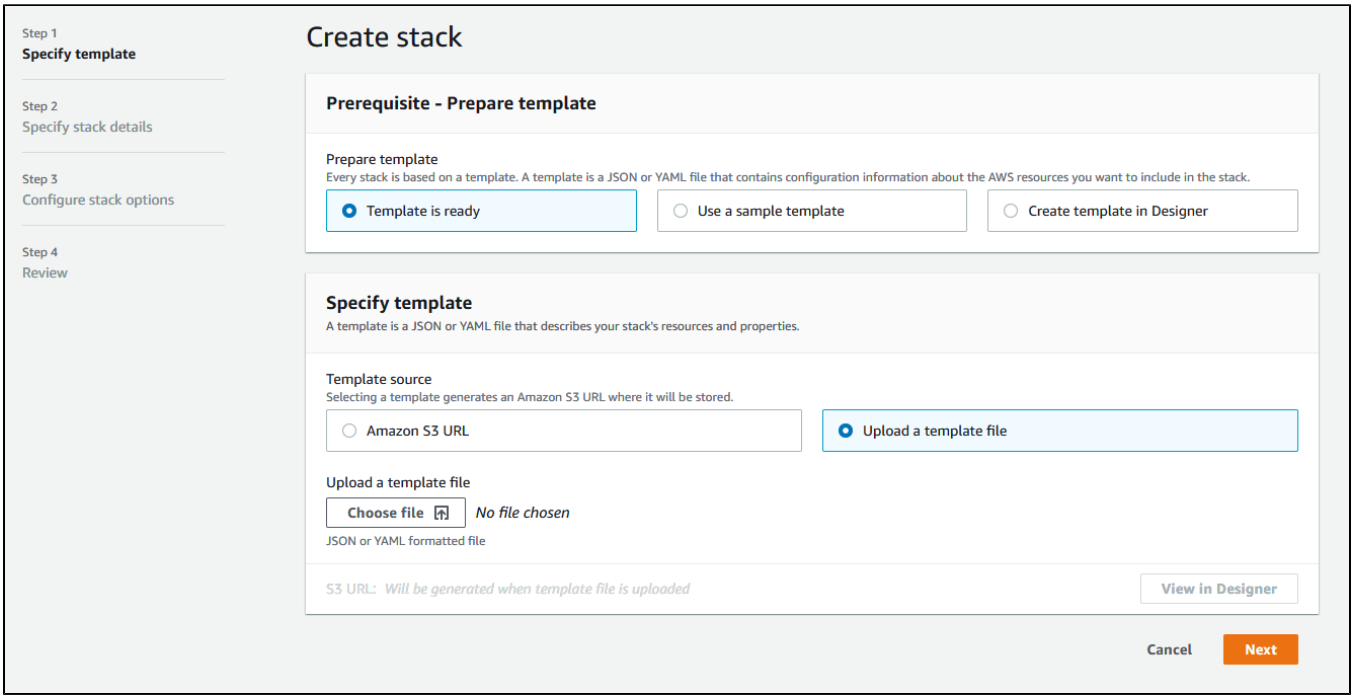
```
          ToPort: 8084
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 8443
          ToPort: 8445
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 8888
          ToPort: 8888
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 9091
          ToPort: 9091
          CidrIp: 0.0.0.0/0
        - IpProtocol: udp
          FromPort: 30000
          ToPort: 33000
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 30000
          ToPort: 33000
          CidrIp: 0.0.0.0/0
Outputs:
  OriginWebsiteURL:
    Description: "URL for newly created WCS Origin instance web interface. Use instance id as admin password"
    Value:
      Fn::Join:
        - ''
        - - "https://"
          - !GetAtt WCSOriginInstance.PublicDnsName
          - ":8444/admin/"
  OriginInstanceId:
    Value:
      Ref: 'WCSOriginInstance'
    Description: "Instance Id of newly created WCS Origin instance"
  OriginPrivateIp:
    Value: !GetAtt WCSOriginInstance.PrivateIp
    Description: "Private IP address of the Origin instance"
  OriginPublicIp:
    Value: !GetAtt WCSOriginInstance.PublicIp
    Description: "Public IP address of the Origin instance"
  EdgeWebsiteURL:
    Description: "URL for newly created WCS Edge instance web interface. Use instance id as admin password"
    Value:
      Fn::Join:
        - ''
        - - "https://"
          - !GetAtt WCSEdgeInstance.PublicDnsName
          - ":8444/admin/"
  EdgeInstanceId:
    Value:
      Ref: 'WCSEdgeInstance'
    Description: "Instance Id of newly created WCS Edge instance"
  EdgePrivateIp:
    Value: !GetAtt WCSEdgeInstance.PrivateIp
    Description: "Private IP address of the Edge instance"
  EdgePublicIp:
    Value: !GetAtt WCSEdgeInstance.PublicIp
    Description: "Public IP address of the Edge instance"
```

WCS CDN deployment example using CloudFormation web console

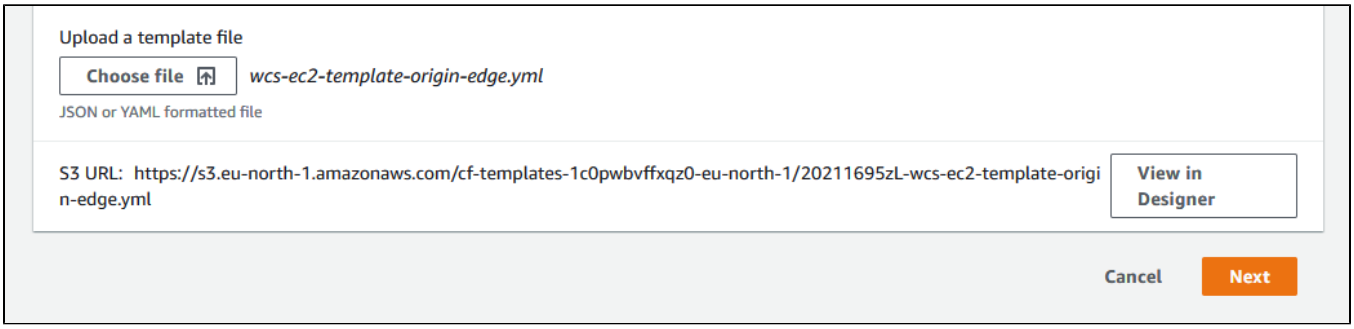
1. Sing in to your AWS account,go to desired region and open CloudFormation in Services menu. Click "Create Stack"



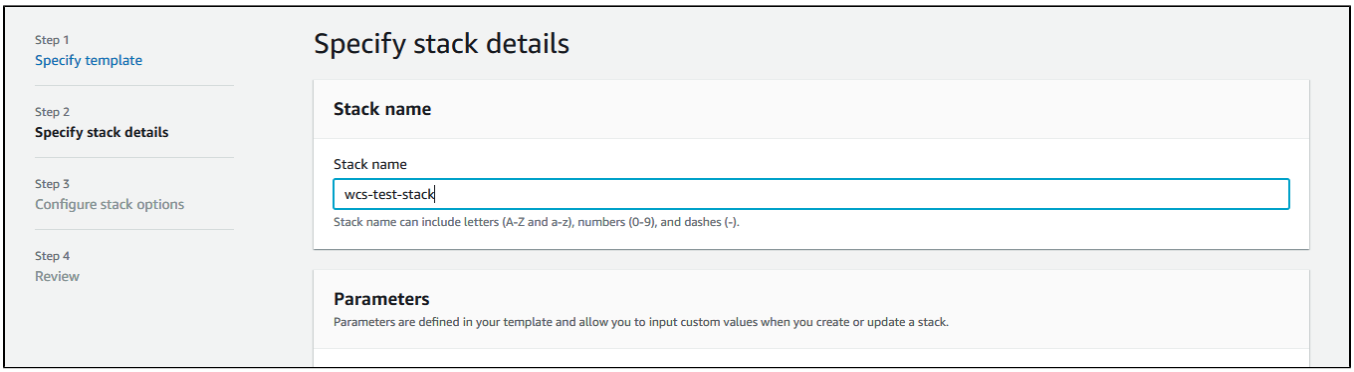
2. Choose "Upload a template file", click "Choose file" and upload the example template



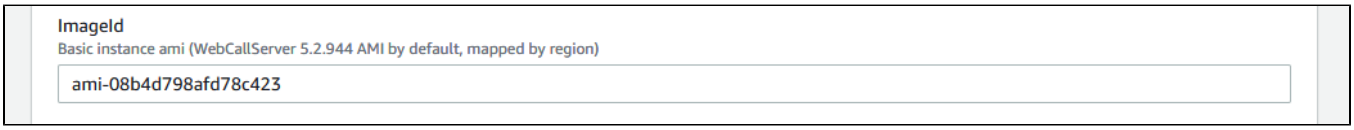
3. When template is uploaded, click Next



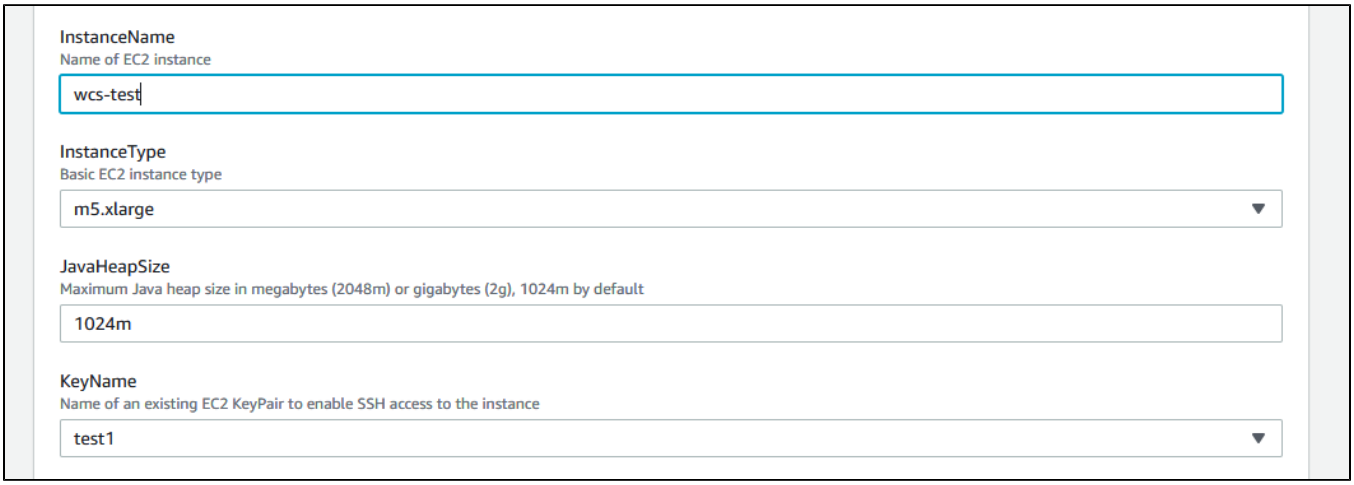
4. Enter stack name



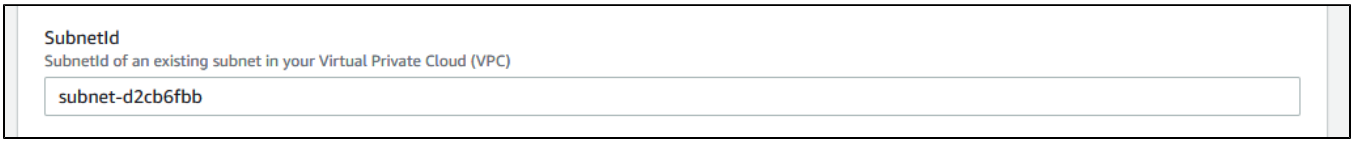
5. Enter Amazon Linux 2 AMI ID for region chosen, or leave WCSAMI (in this case, AWS Marketplace WCS AMI will be used with hourly billing)



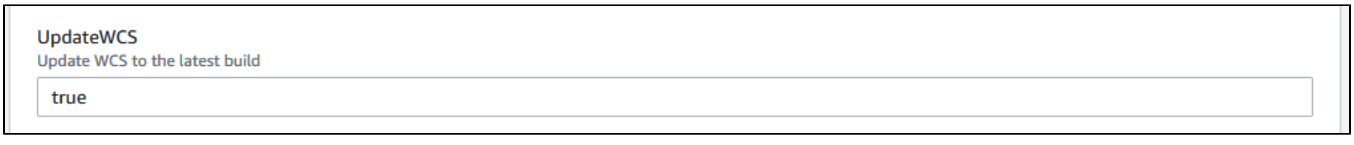
6. Enter basic part of instance name (-origin and -edge will be added respectively), choose instance type, enter Java heap size and choose SSH key to access stack instances



7. Set subnet Id

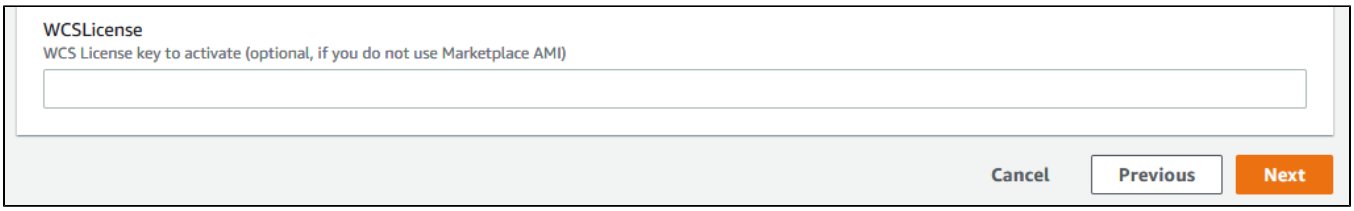


8. Enter "true" to automatically update WCS to the latest build

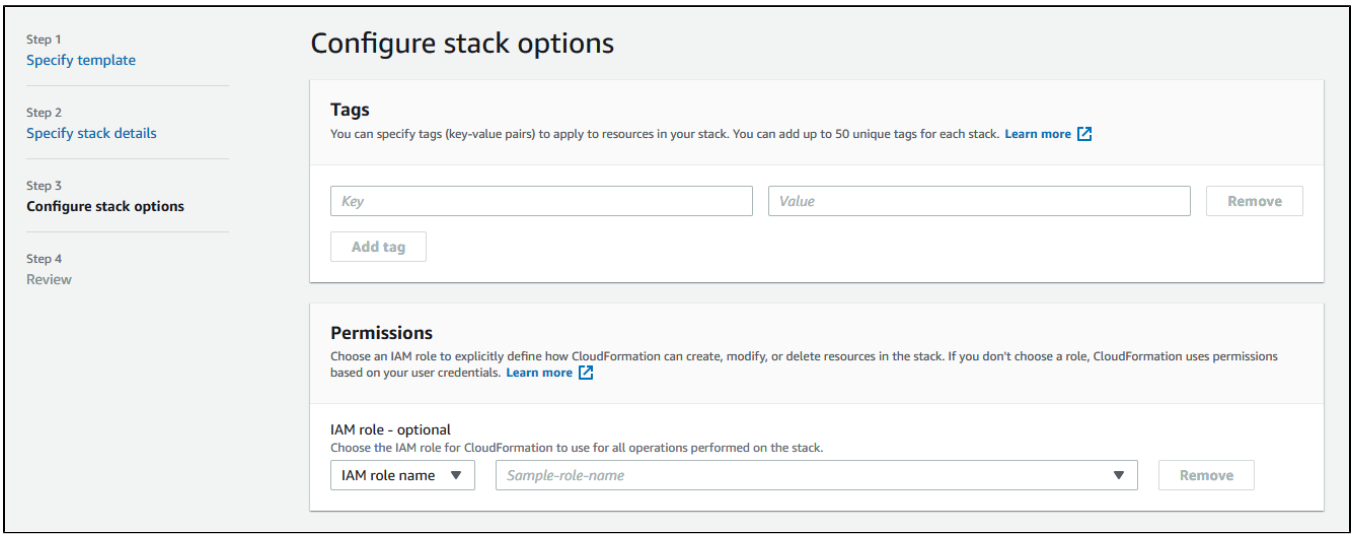


9. Set VPC Id

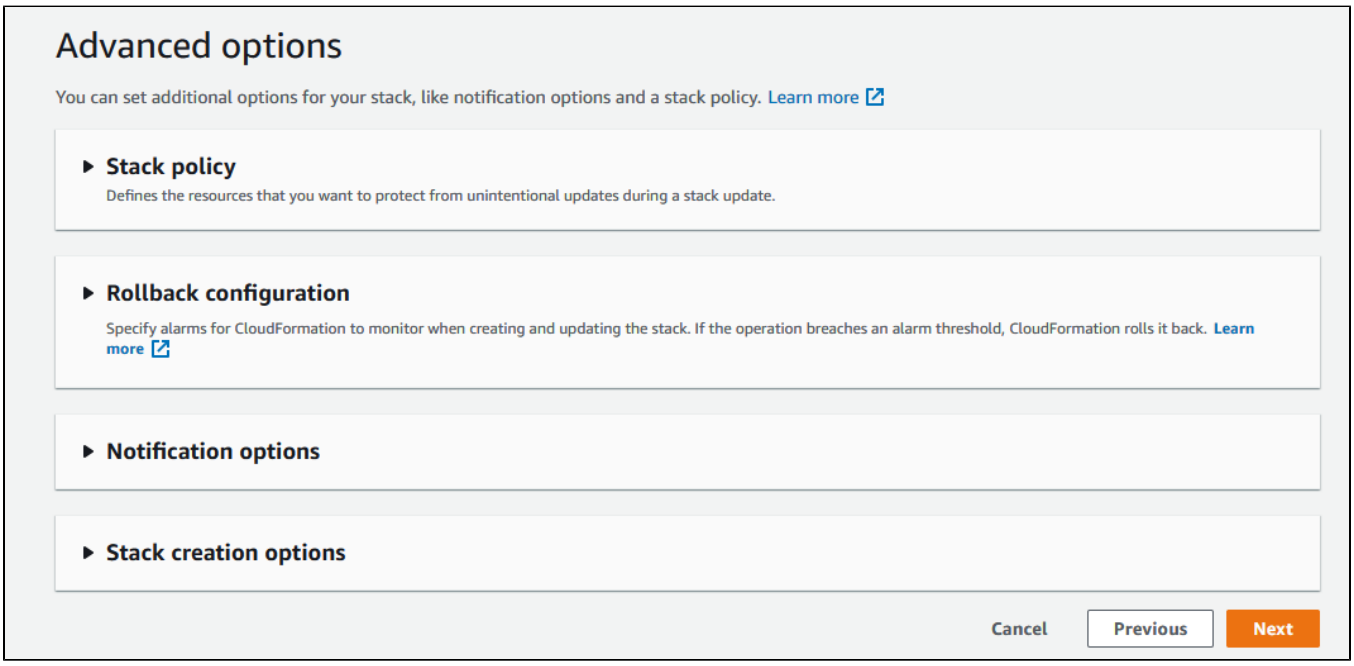
10. If Marketplace WCS AMI is not used, enter the license key to activate on instances and click Next



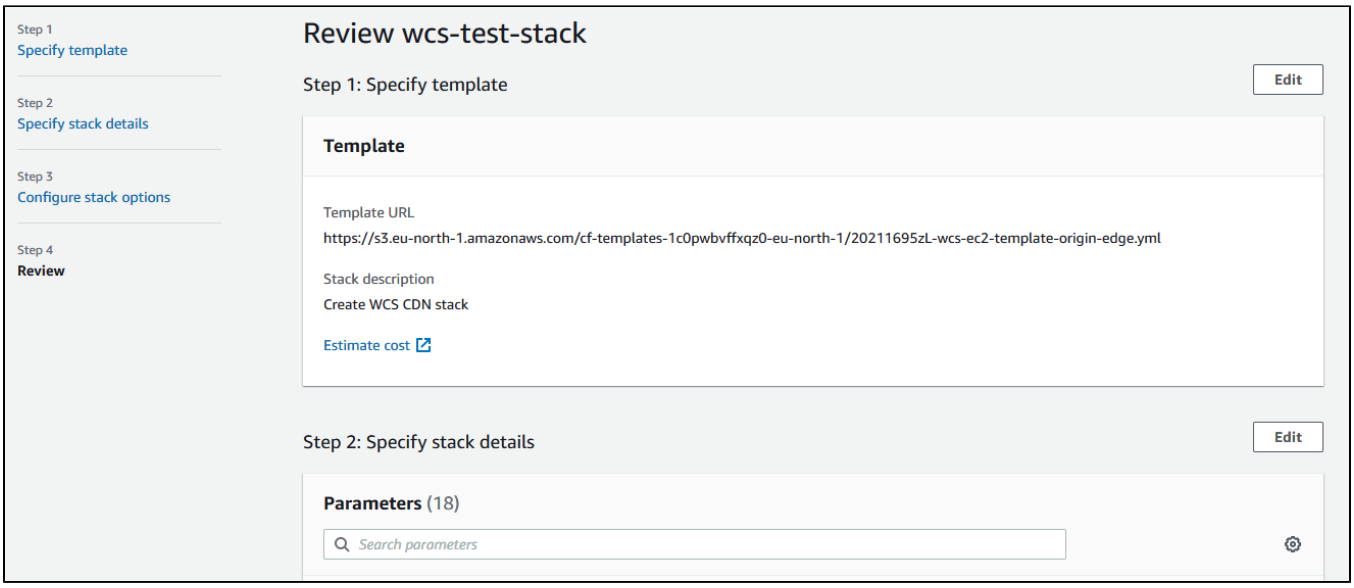
11. Add tags and set permissions if necessary



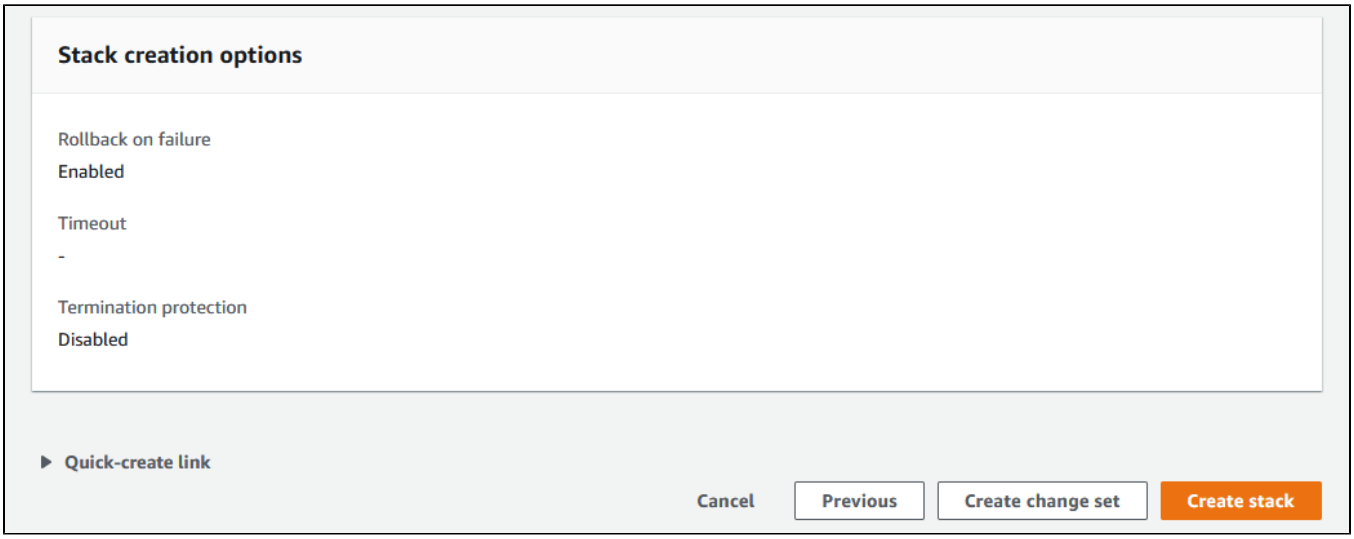
12. Set advanced stack options if necessary, then click Next



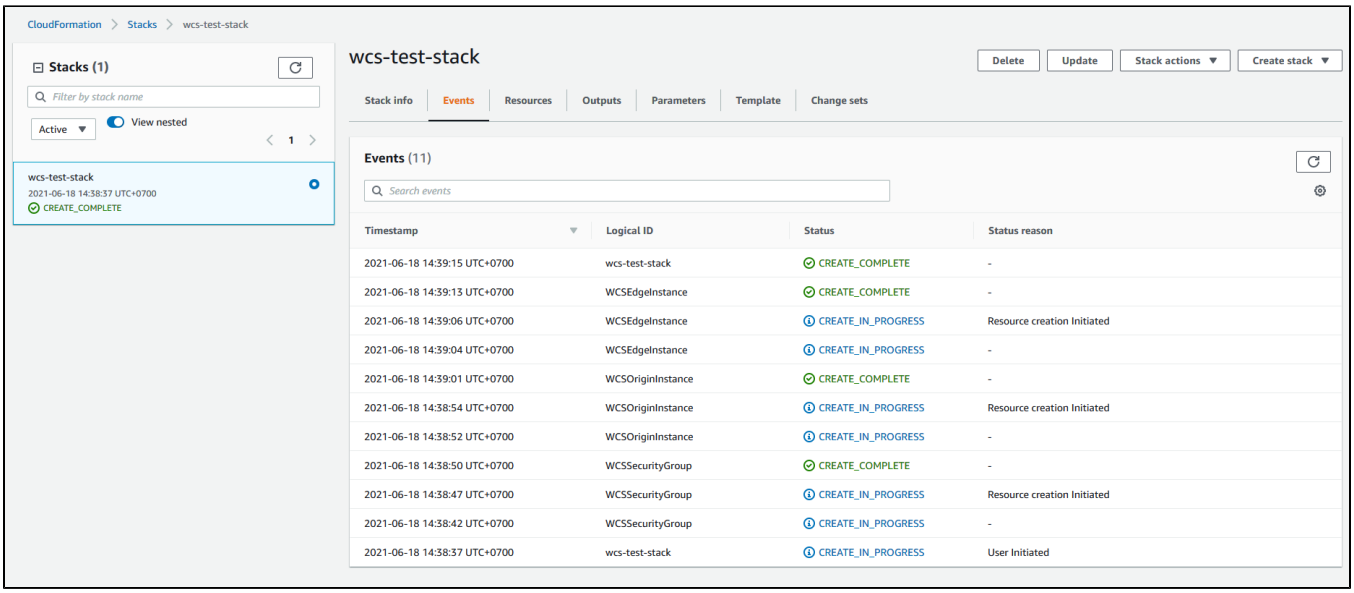
13. Review stack parameters



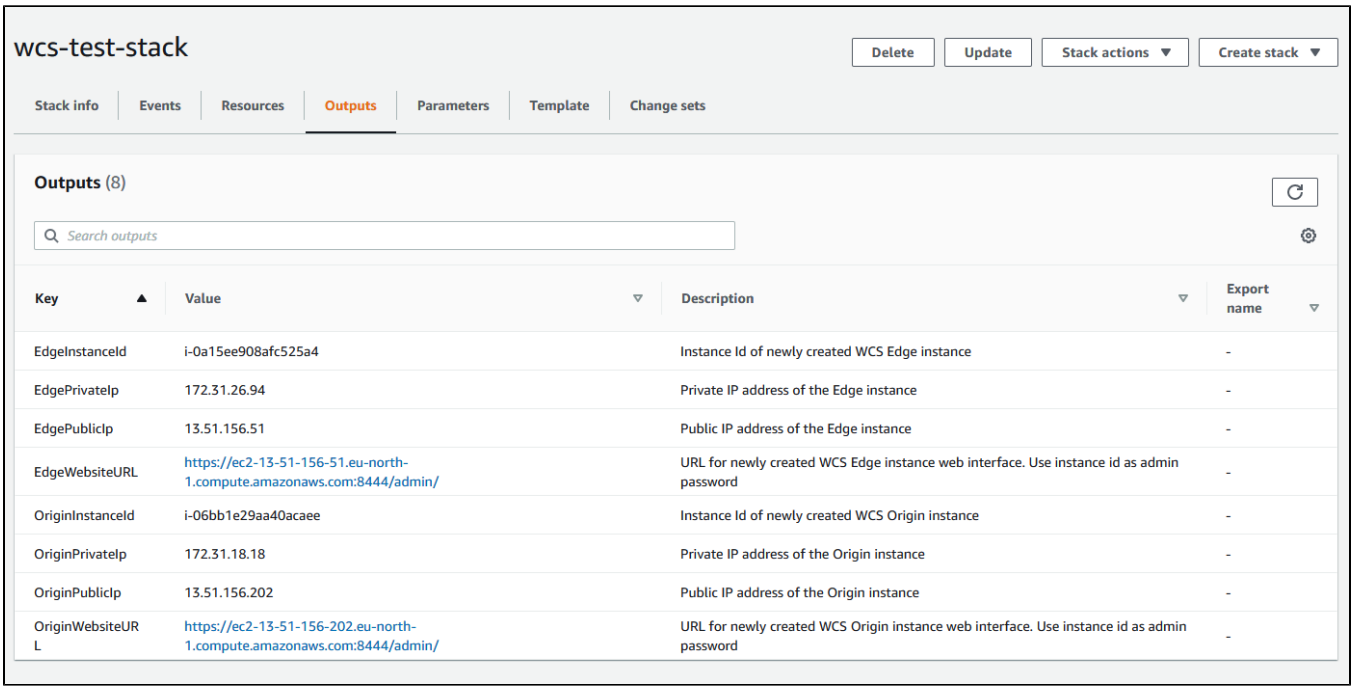
14. Click "Create stack"



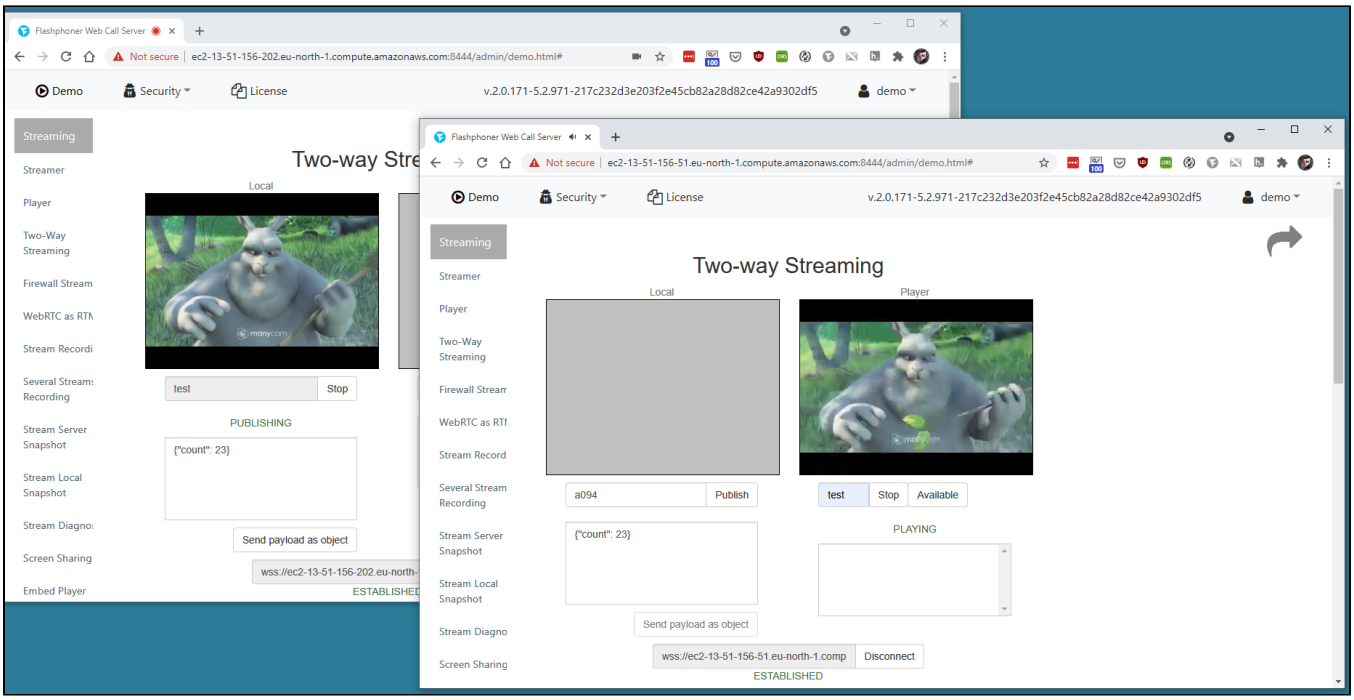
15. Wait for stack creation completion



16. Go to Outputs tab



17. Open Origin and Edge web interfaces,publish the stream test to Origin using Two Way Streaming example, then play the stream on Edge


Flashphoner Web Call Server
Not secure | ec2-13-51-156-202.eu-north-1.compute.amazonaws.com:8444/admin/demo.html#
Demo
Security
License
v.2.0.171-5.2.971-217c232d3e203f2e45cb82a28d82ce42a9302df5
demo
Streaming
Streamer
Player
Two-Way Streaming
Firewall Stream
WebRTC as RTM
Stream Recordi
Several Stream: Recording
Stream Server Snapshot
Stream Local Snapshot
Stream Diagno:
Screen Sharing
Embed Player
Two-way Stre
Local
test
Stop
PUBLISHING
{"count": 23}
Send payload as object
wss://ec2-13-51-156-202.eu-north-
ESTABLISHED
Flashphoner Web Call Server
Not secure | ec2-13-51-156-51.eu-north-1.compute.amazonaws.com:8444/admin/demo.html#
Demo
Security
License
v.2.0.171-5.2.971-217c232d3e203f2e45cb82a28d82ce42a9302df5
demo
Streaming
Streamer
Player
Two-Way Streaming
Firewall Stream
WebRTC as RTI
Stream Record
Several Stream Recording
Stream Server Snapshot
Stream Local Snapshot
Stream Diagno
Screen Sharing
Two-way Streaming
Local
Player
a094
Publish
test
Stop
Available
PLAYING
{"count": 23}
Send payload as object
wss://ec2-13-51-156-51.eu-north-1.comp
Disconnect
ESTABLISHED