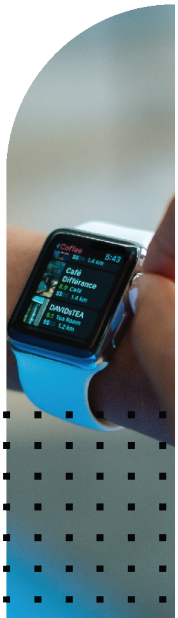
AWS Deployment Guide

FortiDeceptor 4.2.0



FORTINET DOCUMENT LIBRARY
https://docs.fortinet.com

FORTINET VIDEO GUIDE
https://video.fortinet.com

FORTINET BLOG
https://blog.fortinet.com

CUSTOMER SERVICE & SUPPORT
https://support.fortinet.com

FORTINET TRAINING & CERTIFICATION PROGRAM
https://www.fortinet.com/training-certification

FORTINET TRAINING INSTITUTE
https://training.fortinet.com

FORTIGUARD CENTER
https://www.fortiguard.com

END USER LICENSE AGREEMENT
https://www.fortinet.com/doc/legal/EULA.pdf

FEEDBACK
Email: techdoc@fortinet.com

March 14, 2023
FortiDeceptor 4.2.0 AWS Deployment Guide
00-420-809392-20230314

TABLE OF CONTENTS

Change Log

Date	Change Description
2022-05-16	Initial release.
2022-09-15	Added Configure the client on page 35
2023-0-09	Updated Check the imported image in AMIs on page 26.
2023-01-12	Updated Configure the client on page 35.
2023-01-13	Updated FortiDeceptor Cloud topology on page 6.
2023-03-14	Updated Import the FortiDeceptor image to AWS AMI on page 17.

About FortiDeceptor VM on AWS

FortiDeceptor VM is a 64-bit virtual appliance version of FortiDeceptor. It is deployed in a virtual machine environment. Once the virtual appliance is deployed and set up, you can manage FortiDeceptor VM via its GUI in a web browser on your management computer.

This document provides information about deploying a FortiDeceptor VM in the Amazon Web Services (AWS) environment. This includes how to configure the virtual hardware settings of the virtual appliance. This guide presumes that the reader has a thorough understanding of virtualization servers.

This document does not cover configuration and operation of the virtual appliance after it has been successfully installed and started. For that information, see the FortiDeceptor Administration Guide in the Fortinet Document Library.

Licensing

Fortinet offers the FortiDeceptor in a stackable license model. This model allows you to expand your VM solution as your environment expands. For information on purchasing a FortiDeceptor license, contact your Fortinet Authorized Reseller, or visit https://www.fortinet.com/how_to_buy/.

When configuring your FortiDeceptor, ensure that you configure hardware settings as outlined in the following table and consider future expansion. Contact your Fortinet Authorized Reseller for more information.

Technical Specification	Details
AWS support	• t3.medium for 2 NICs • c5.4xlarge for 6 NICs The available EC2-instance type is determined by the zone.
Virtual CPUs (min / max)	4 / Unlimited*
Virtual Network Interfaces	2-6 NICs
Virtual Memory (min / max)	8GB / Unlimited**
Virtual Storage (min / max)	HDD 50G/ 16TB***

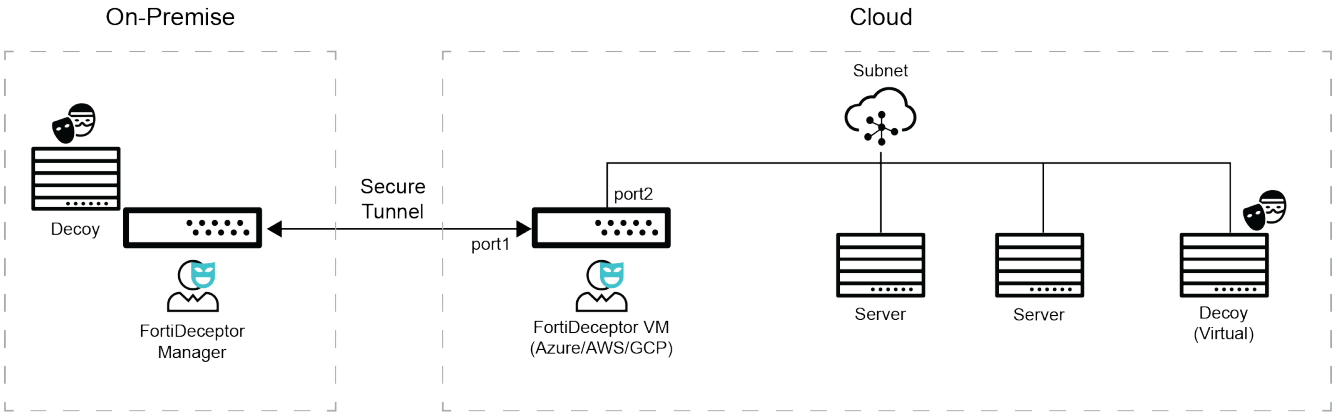
For more information, see the FortiDeceptor product data sheet available on the Fortinet web site, https://www.fortinet.com/content/dam/fortinet/assets/data-sheets/FortiDeceptor.pdf.

After placing an order for FortiDeceptor, a license registration code is sent to the email address used in the order form. Use the license registration code provided to register the FortiDeceptor with Customer Service & Support at https://support.fortinet.com.

Upon registration, you can download the license file. You will need this file to activate your FortiDeceptor. You can configure basic network settings from the CLI to complete the deployment. Once the license file is uploaded and validated, the CLI and GUI will be fully functional.

FortiDeceptor Cloud topology

The cloud appliance is deployed over the public infrastructure but uses a different method for decoy deployment. This new method requires less HW requirements for the cloud appliance itself.



The cloud decoy deployment method is as folows:

- The cloud appliance will be deployed over the cloud infrastructure.
- An on-premise FortiDeceptor Manager will manage the cloud appliance over a propriety network tunnel.
- The propriety network tunnel allows managing the cloud appliance and decoy deployment provisioning over layer2 tunnel communication over layer3.
- The cloud appliance network interfaces will hold IP addresses in the cloud segment. Each IP address represents a network decoy.
- The network decoy will run on the on-premise FortiDeceptor Manager and use the same IP address as the cloud appliance network interfaces.
- The cloud IP address will tunnel over Layer2 to the IP address on the on-premise FortiDeceptor Manager.
- The idea is to run a light appliance in the cloud while running the actual network decoys inside the on-premise FortiDeceptor Manager in a sandbox mode. The cloud network is isolated from the rest of the decoys, the on-premise networks.

While the cloud appliance uses different hardware requirements, the on-premise FortiDeceptor Manager HW requirements that should serve the cloud appliance decoys is the same concept as today.

Preparing FortiDeceptor for deployment

To prepare FortiDeceptor for deployment, download the FortiDeceptor image from FortiCloud. Prepare the AWS network by creating a Virtual Public Cloud, subnets, an Internet gateway, and route table. After the network is prepared you will need to import an AMI image to create a VM instance, then associate the instance with public IP addresses to deploy the decoys.

To prepare for deployment:

1. Prepare the FortiDeceptor image.
2. Prepare the network in AWS.
3. Create a bucket.
4. Import the FortiDeceptor image to AWS AMI.
5. Check the imported image.
6. Create an instance from the AMI image.
7. Verify the instance.
8. Associate a public IP to port1
9. Configure multiple IPs for deployment.

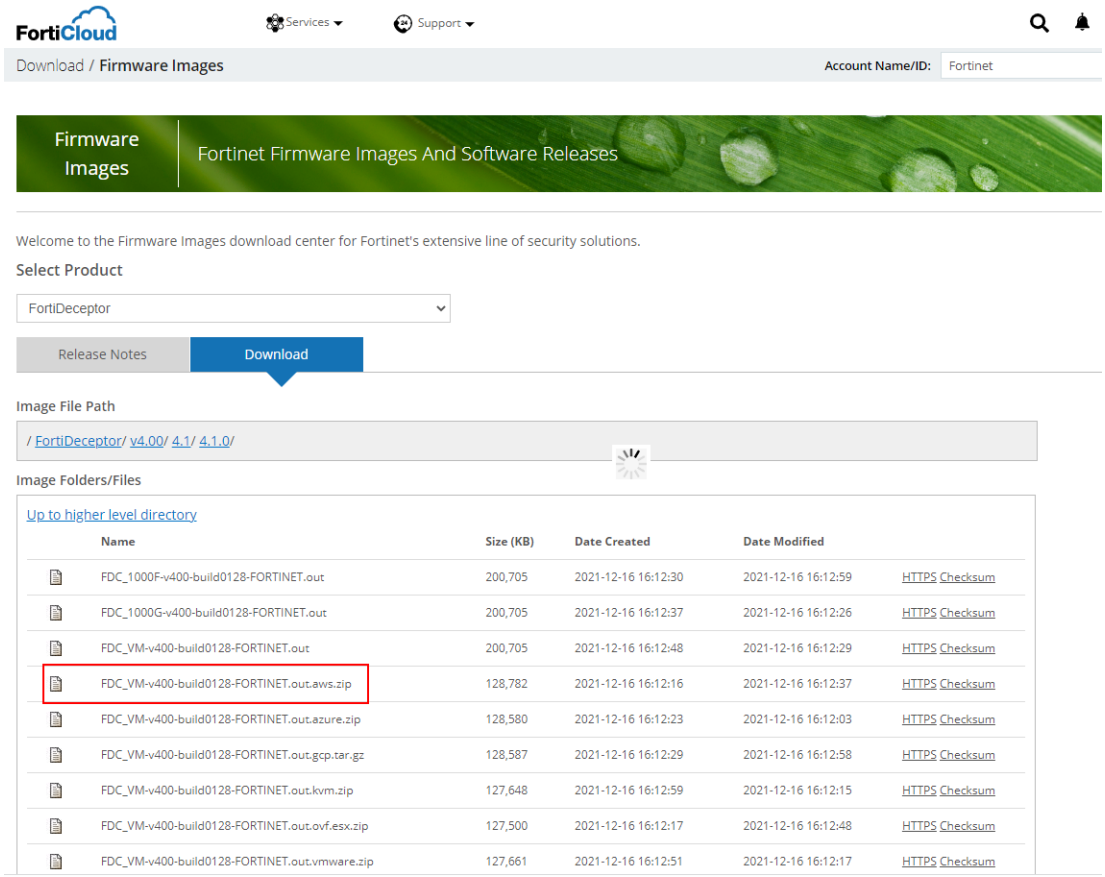
Prepare the FortiDeceptor image for AWS

Download the image archive file for the AWS platform and unzip it to get image file *fdc.aws.vhd*.

To download the FortiDeceptor image:

1. Log in to FortiCloud.
2. In the banner, click *Support* > *Downloads* > *Firmware Download*. The *Download/Firmware Images* page opens.
3. From the *Select Product* dropdown, select *FortiDeceptor*.
4. Click the *Download* tab.
5. In the *Image File Path* section, click the image folder until you reach the image page.

6. Select *FDC_VM-vx.x.x-buildxxxx-FORTINET.out.aws.zip*



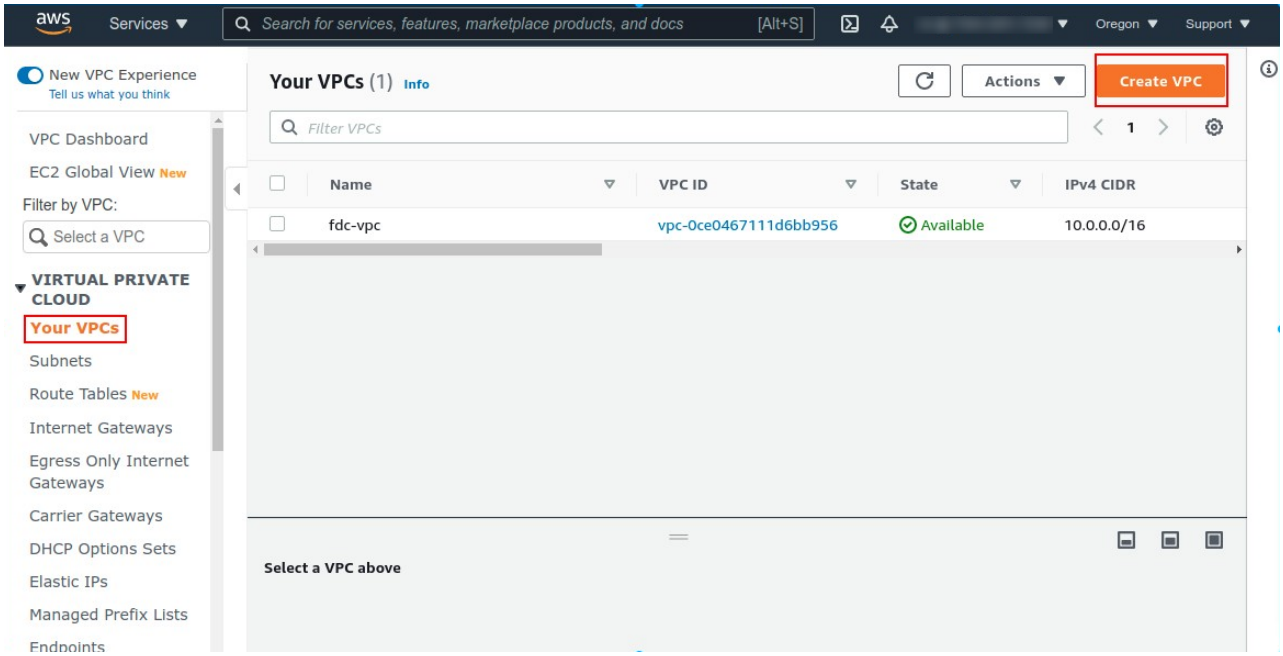
Preparing the network in AWS

To prepare the network, create a Virtual Private Cloud (VPC) and create several subnets. Next you will create an Internet Gateway and route table. Associate the subnets with the route table and then allocate an elastic IP address.

Creating a Virtual Private Cloud (VPC)

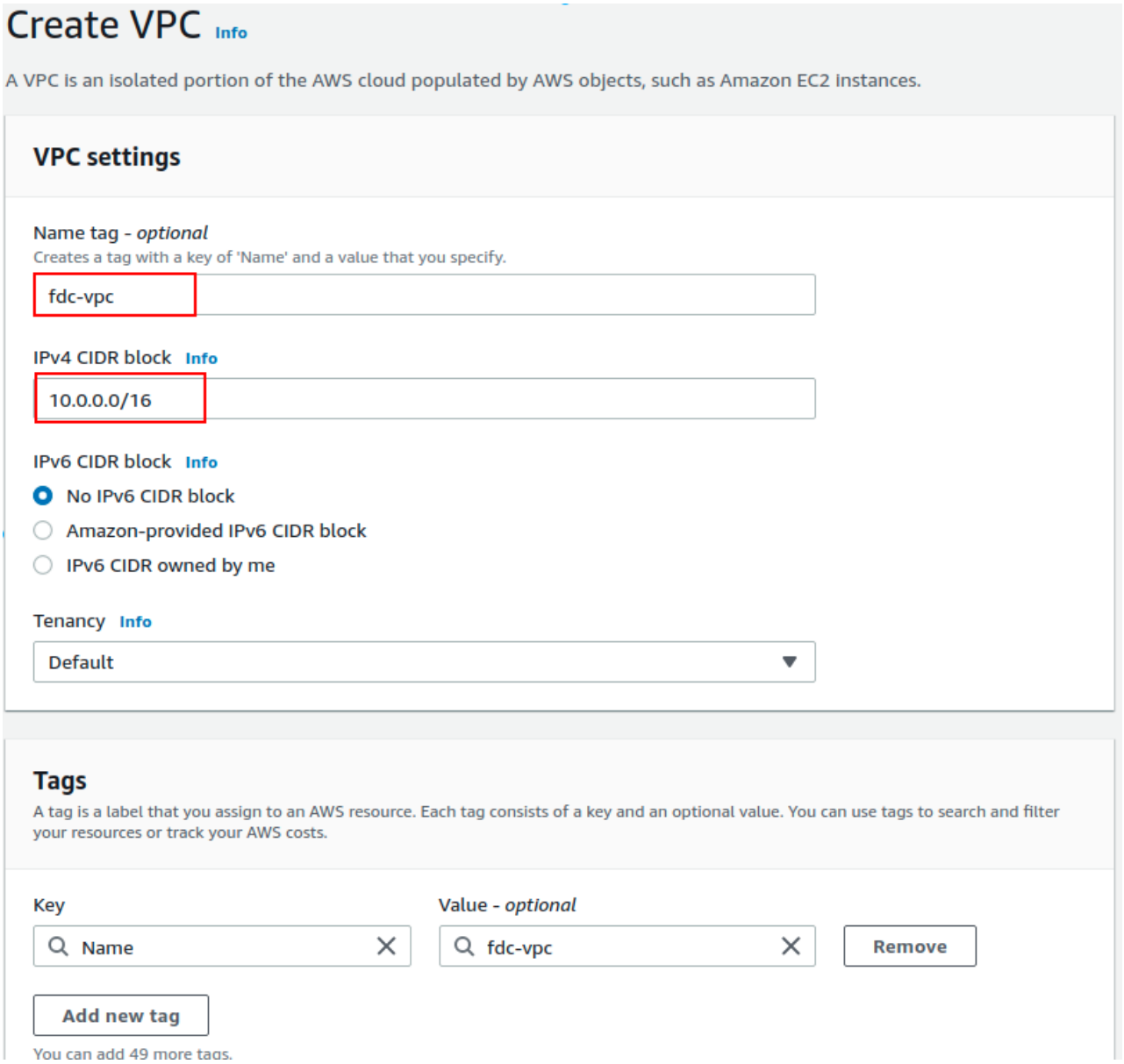
To create a VPC in AWS:

1. In the Services menu, go to *Virtual Private Cloud* > *Your VPCs*.
2. Click *Create VPC*. The *Create VPC* page opens.



3. Configure the following settings:

Name Tag	Enter a name for the VPC such as *fdc-vpc*.
IPv4 CIDR block	Enter the IP address for the VPC



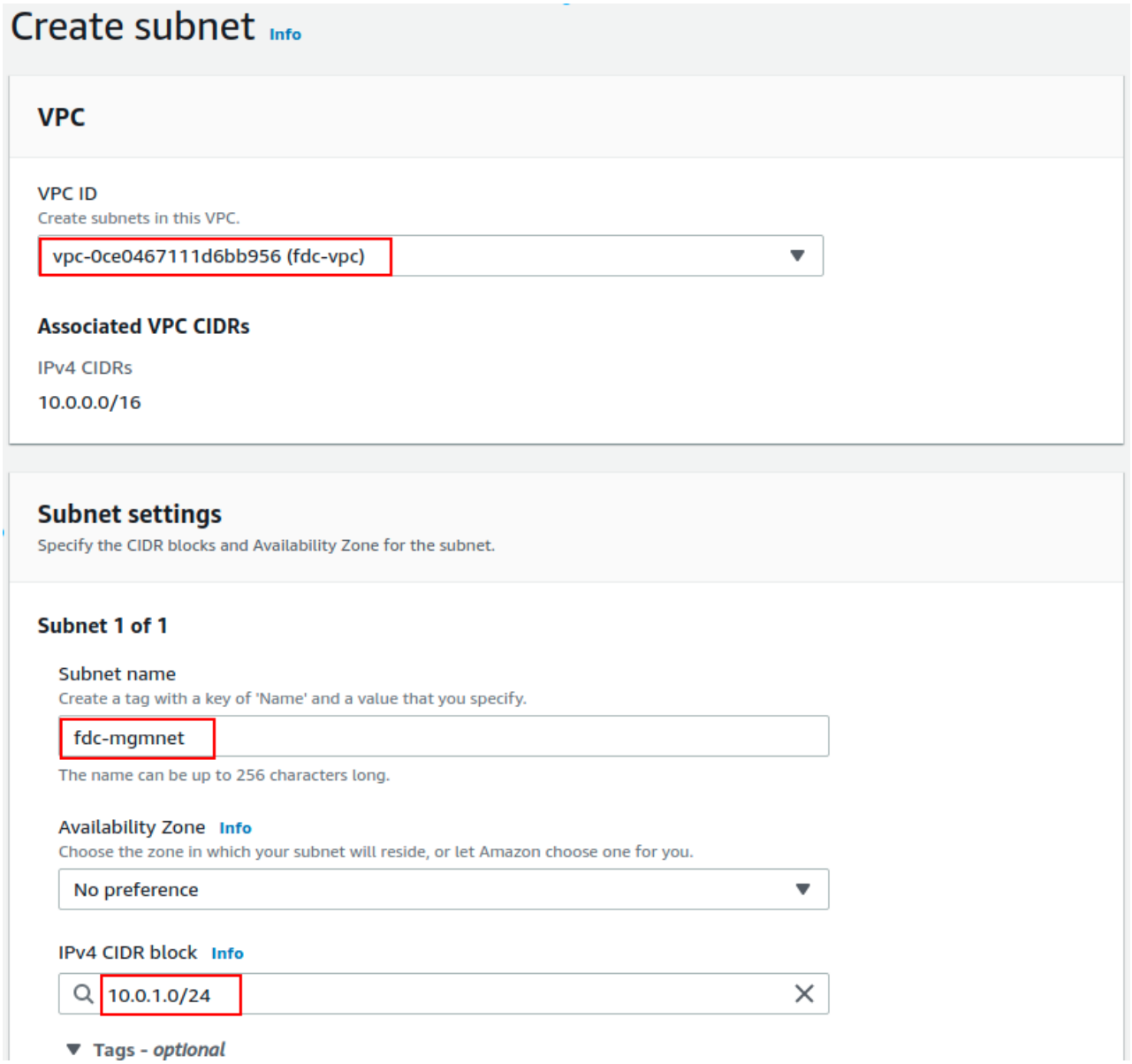
Creating subnets in the VPC

Create several subnets in VPC for FortiDeceptor management and deployment.

To create subnets in the VPC:

1. In the *Services* menu, go to *Virtual Private Cloud* > *Subnets*.
2. Click *Create subnet*. The *Create subnet* page opens.
3. Configure the following settings:

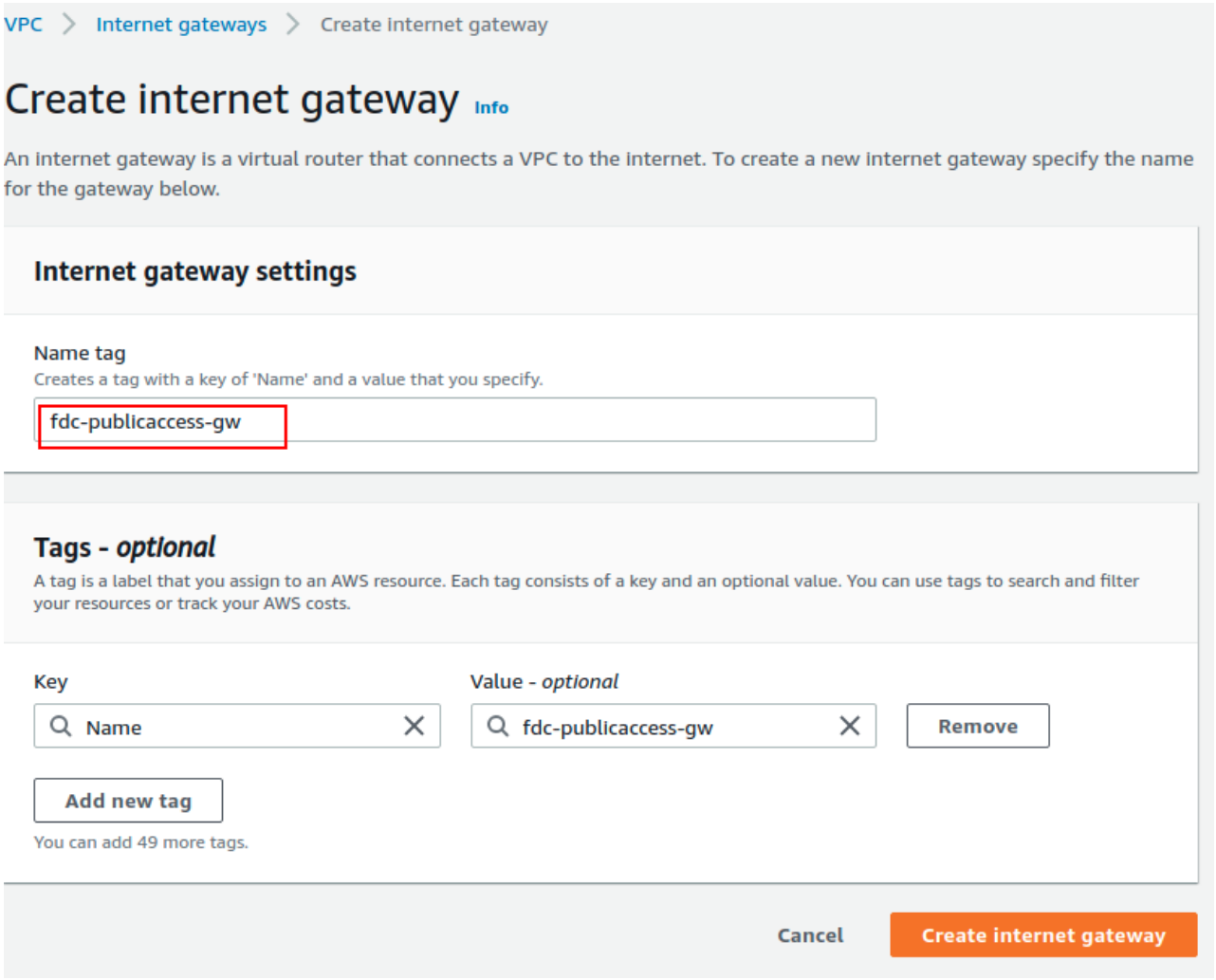
VPC ID	Select an ID from the dropdown.
Subnet name	Enter a name for the subnet such as *fdc-mgmnet*.
IPv4 CIDR block	Enter the IP address for the network.



Creating an internet Gateway

To create an internet Gateway:

1. In the *Services* menu, go to *Virtual Private Cloud* > *Internet Gateways*.
2. Click *Create Internet Gateway*. The *Create Internet Gateway* page opens.
3. In the *Name tag* field, enter a name for the tag such as *fdc-publicaccess-gw*.



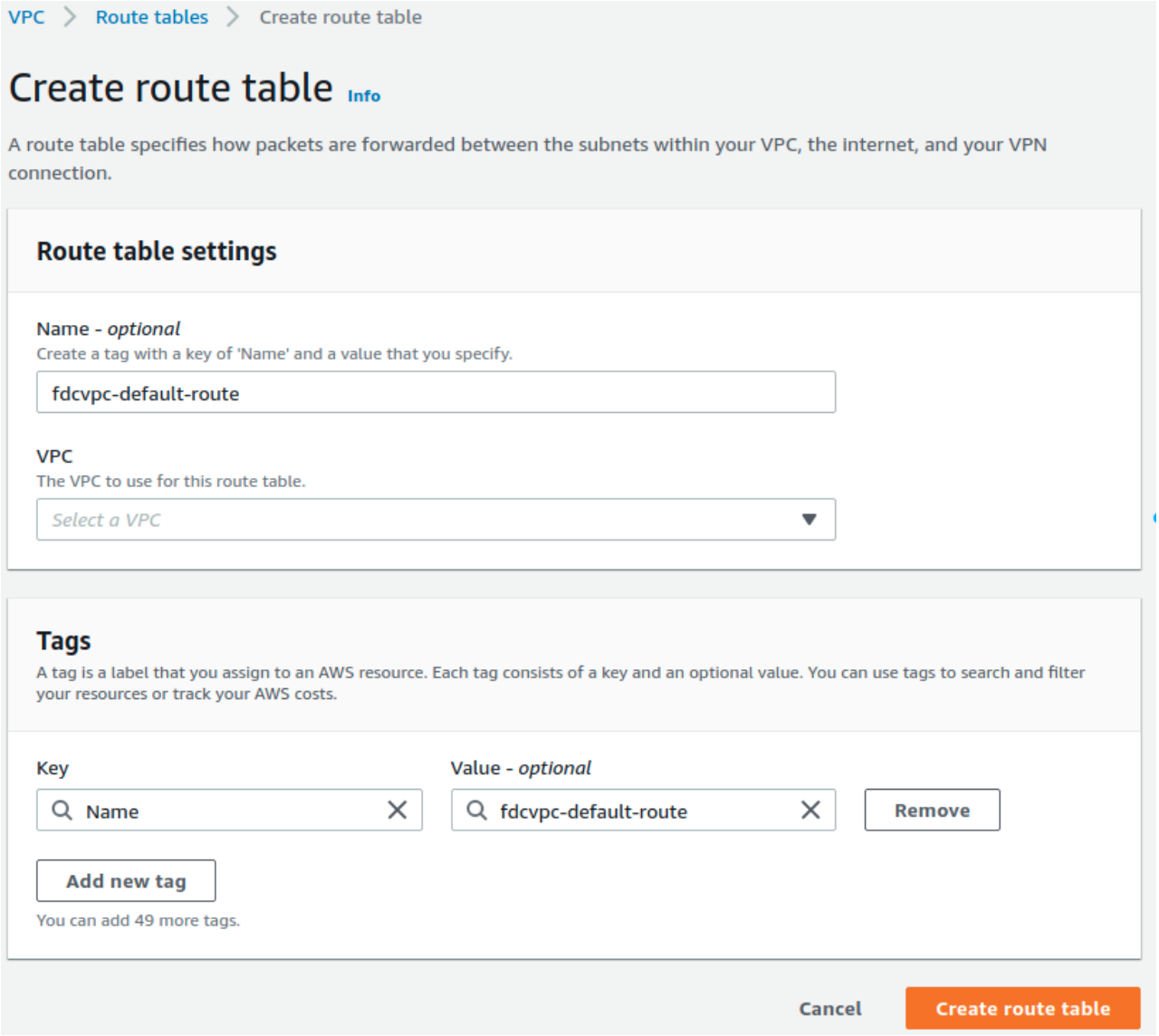
4. Click *Create Internet Gateway*.

Creating a route table

To create a route table:

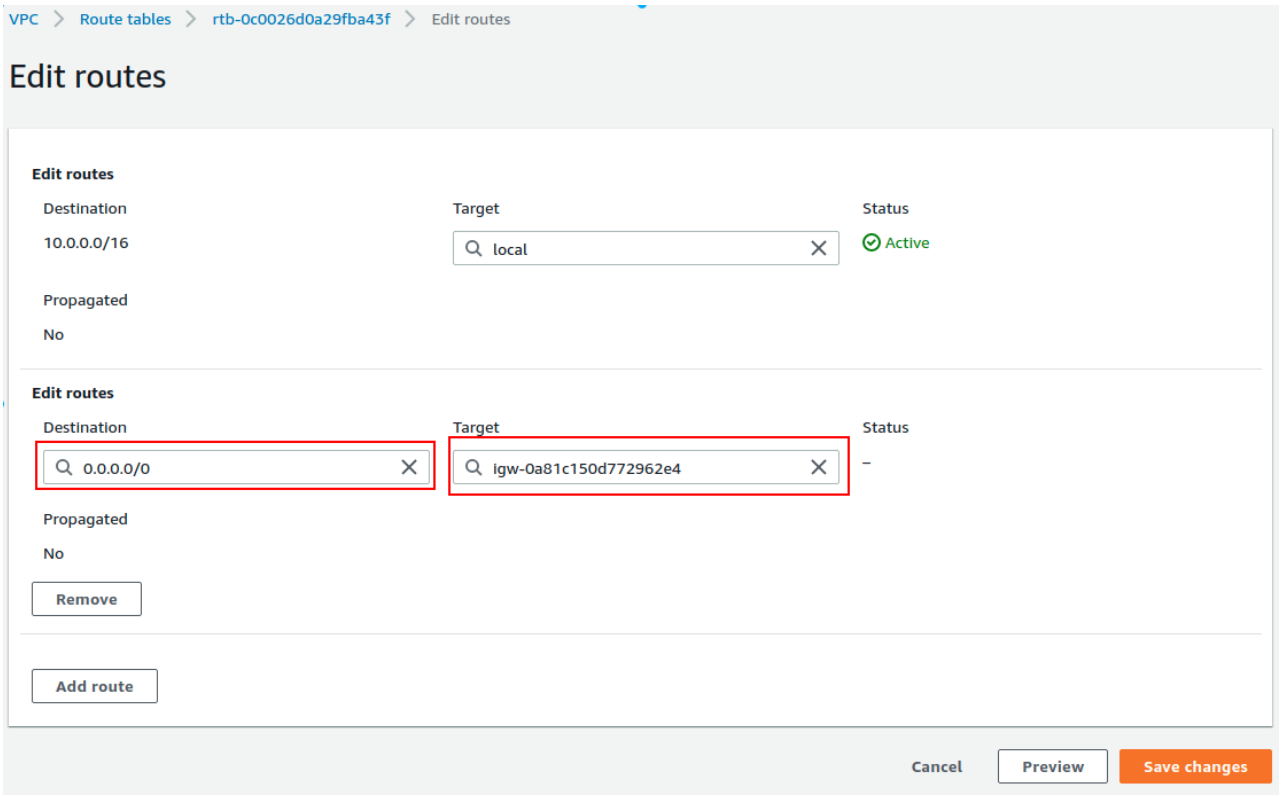
1. In the *Services* menu, go to *Virtual Private Cloud* > *Route Tables*.
2. Click *Create route table*. The *Create route table* page opens.

3. In the *Name* field, enter a name for the table such as *fdcvpc-defualt-route*.



4. Click *Create route table*.
5. Open the route table you created to edit it.
6. Under *Edit routes*, configure the following settings:

Destination	Enter 0.0.0.0/0.
Target	Enter the Internet gateway you created.



7. Click *Save changes*.

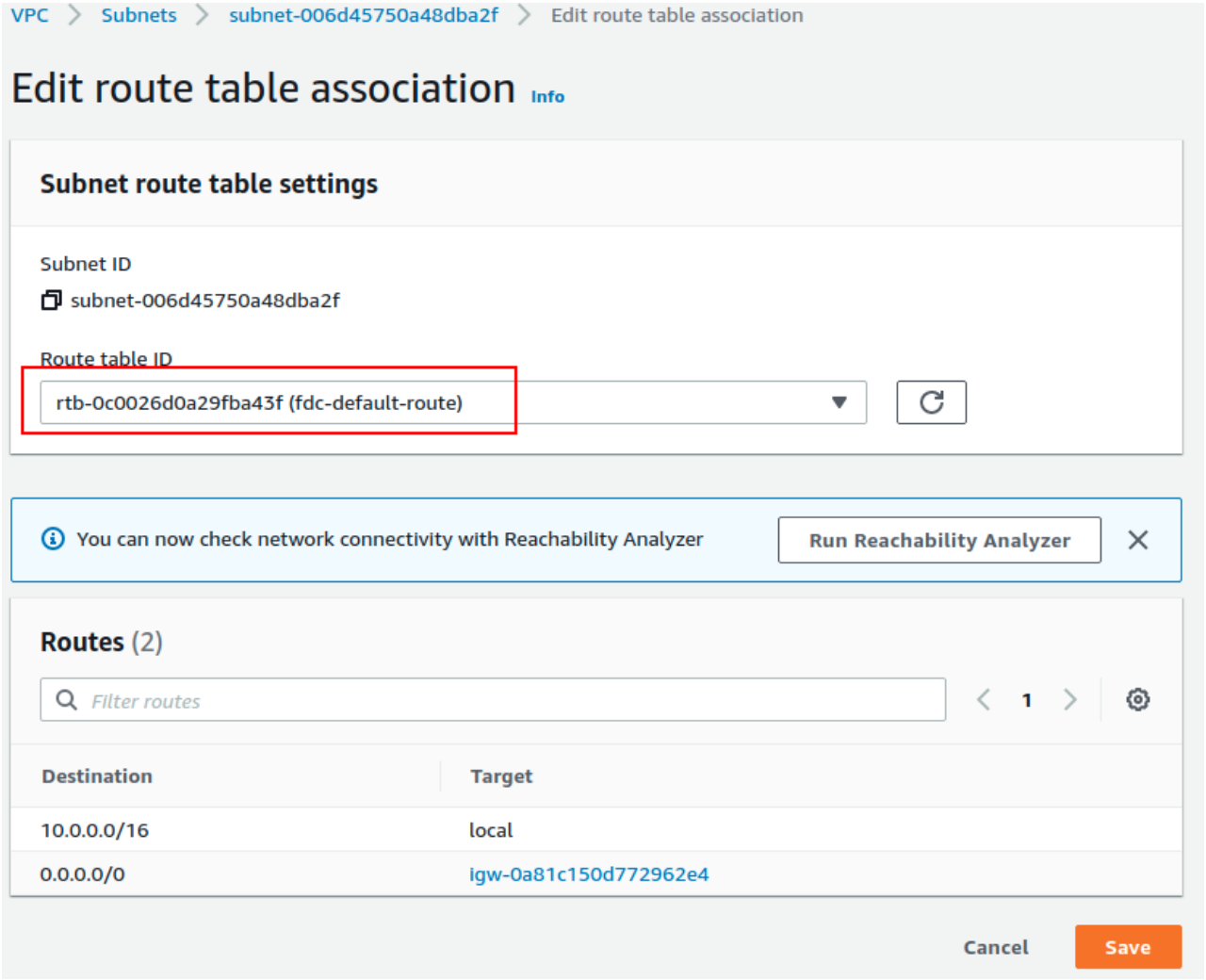
Associating subnets with a route table

Associate a subnet with the route table to apply route rules to that specific subnet.

To associate subnets with Route Table:

1. In the *Services* menu, go to *Virtual Private Cloud* > *Subnets*.
2. Click the subnet you created. The *Edit route table association* page opens.

3. In the *Route table ID* field, select the route table you just created.



4. Click *Save*.

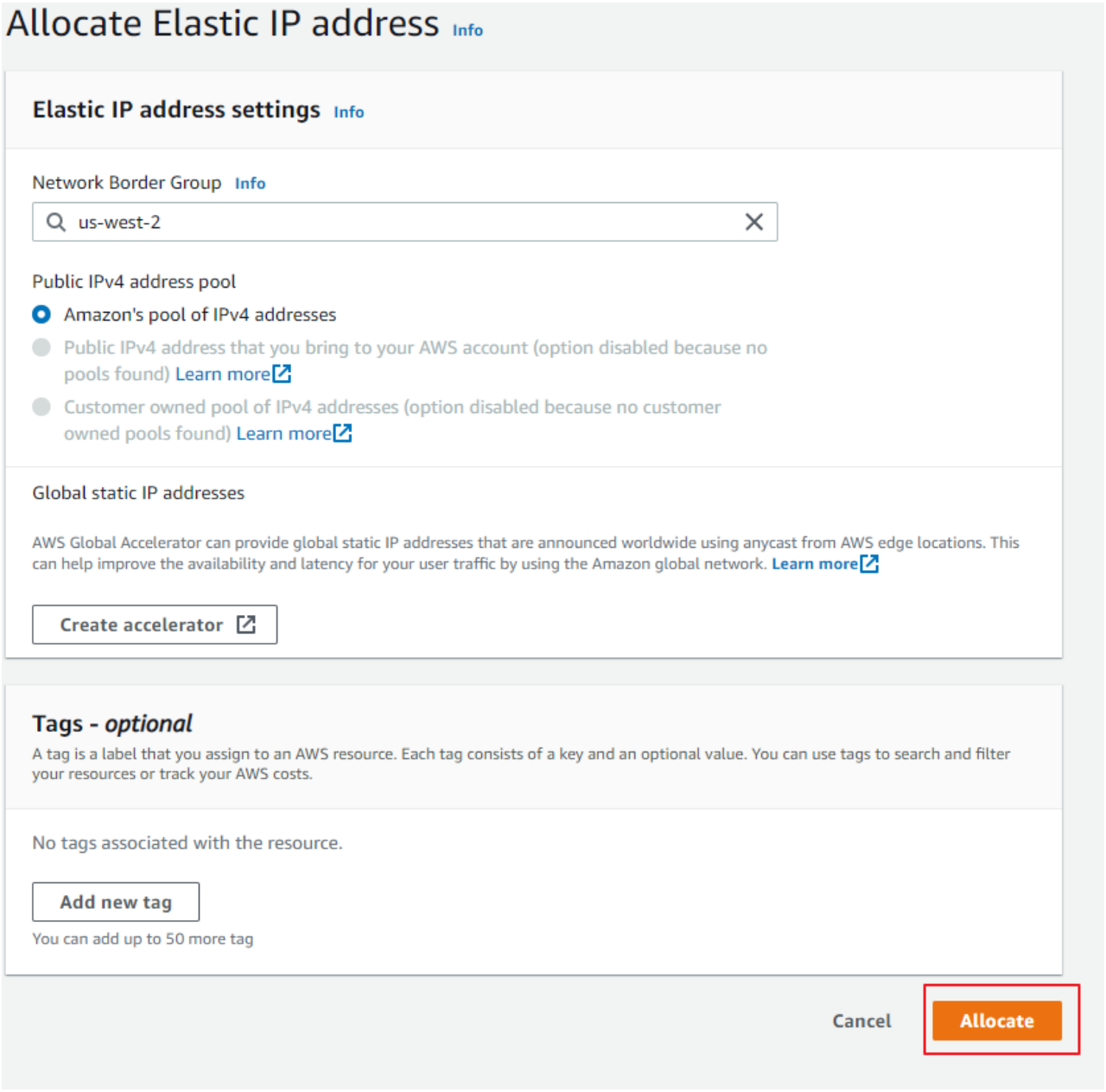
Allocating an elastic IP address

Allocate a public IP for public access to FortiDeceptor management port later. This step is not required for deployment.

To allocate an elastic IP address:

1. In the *Services* menu, go to *Virtual Private Cloud* > *Elastic IPs*.
2. Select an elastic IP. The *Elastic IP address settings* window opens.

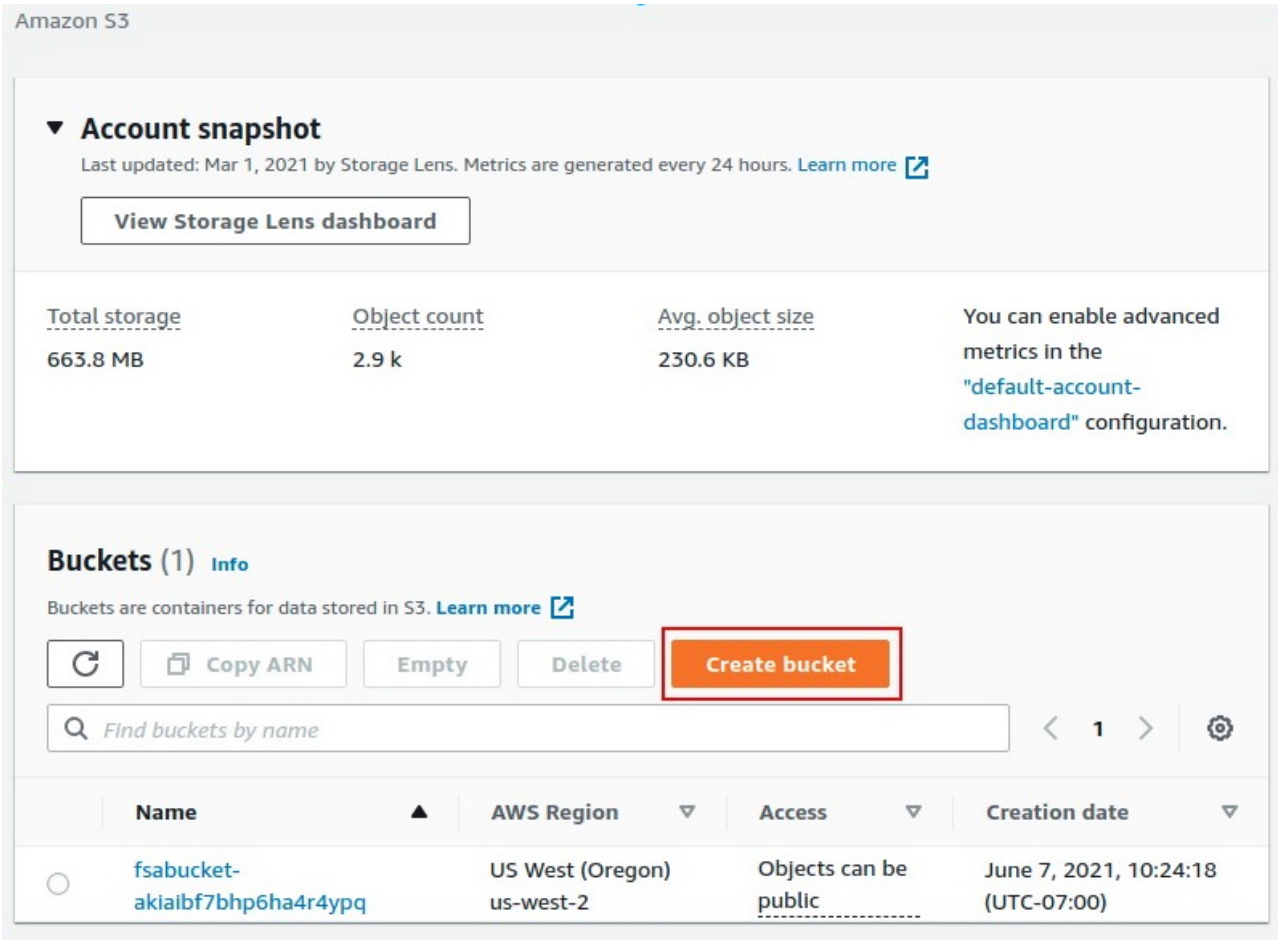
3. Click *Allocate*.



Create a bucket

To create an AWS bucket:

1. In the AWS Management Console, click *Create Bucket*. The *Create bucket* wizard opens.
2. Configure the bucket settings and click *Create bucket*.



Import the FortiDeceptor image to AWS AMI

Go to IAM Service and create users and roles with proper permissions. Then get the *Access Key ID*/*Secret Key* from the *My Security Credentials* menu. You can only get the Secret Key at the time you create the Access Key.

Click this link https://docs.aws.amazon.com/vm-import/latest/userguide/vmie_prereqs.html#vmimport-role for information about creating a *vmimport* role to import/export VM images.

You can import the FortiDeceptor one of two ways:

- With a python script
- With the AWS EC2 toolkit (Recommended)

Importing the FortiDeceptor image with python script

Install Python3, boto3 in Linux, and copy the import script to any work folder. Execute the script to import the FortiDeceptor image into AWS as AMI private image.

To get a copy of the Python script, see Python script for importing the FortiDeceptor image on page 22.

Before you begin, make sure you have copied the *fdc.aws.vhd* file to the current directory. To get a copy of the file, see Python script for importing the FortiDeceptor image on page 22.

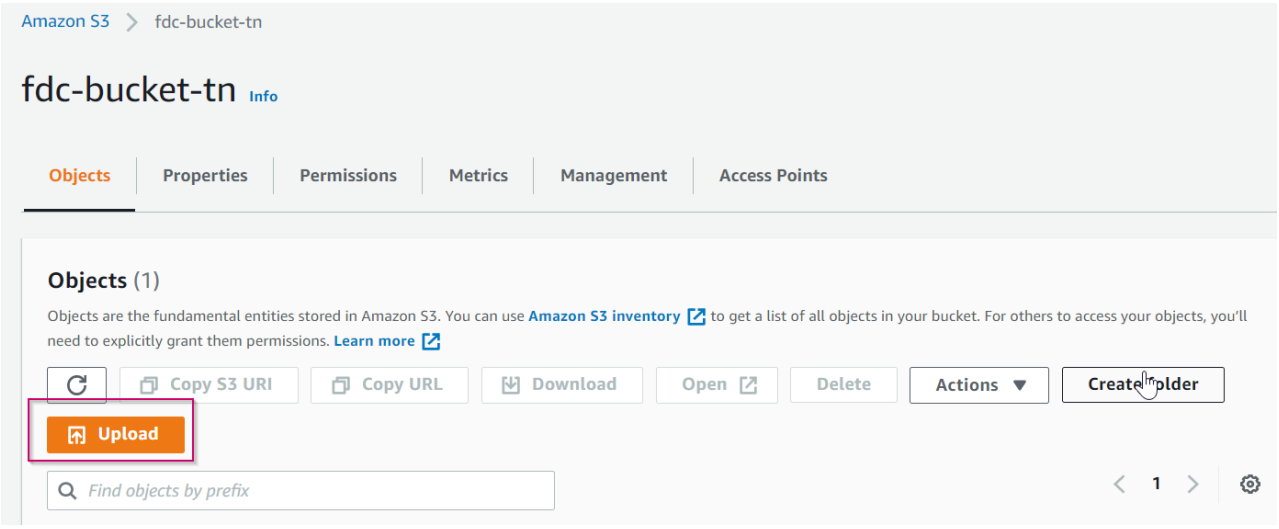
Example command:

```
python3 FDC_import_as_AWS_AMI.py \
-f /fdc.aws.vhd \
-n fdcv4.1.0b0090 \
-a x86_64 \
-s 1 \
-r us-west-2 \
-i AKIA2UEJLWR3DIUPLLF8 \
-k Uj8QO8TKpgHX5krbR88GkWwnQm2Ko4kl4cpUhk99 \
-b fdcbucket-akiaibf7bhp6ha4r4ypq
```

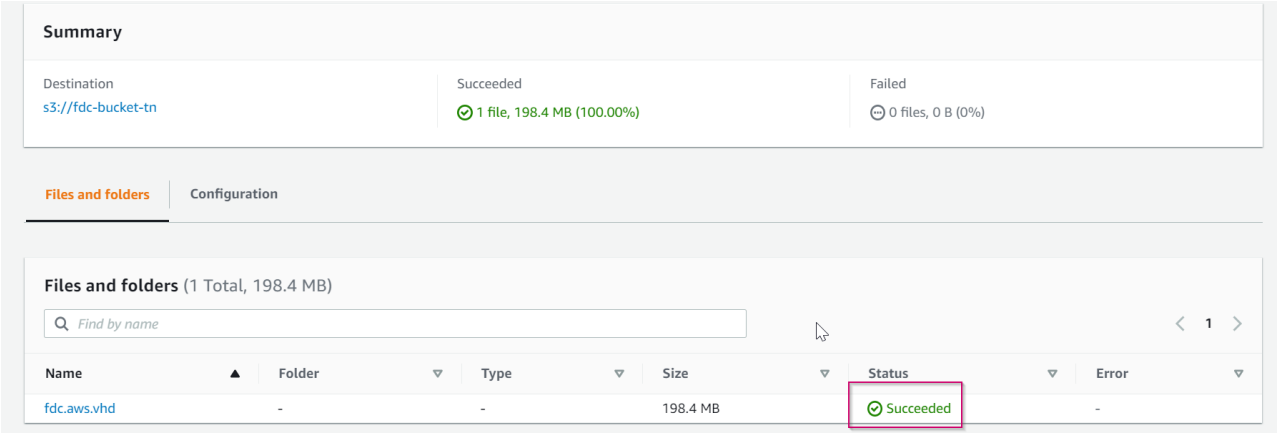
Importing the FortiDeceptor image with AWS EC2 toolkit

To upload the image to a storage bucket:

1. Install the AWS CLI.
2. In the *Buckets* list, open the bucket you created in the previous step.
3. Click *Upload* and navigate to `fdc.aws.vhd` on your device.



4. Click *Upload*. The upload *Status* should display *Succeeded*.



Importing the uploaded VHD file as snapshot

Use the `import-snapshot` command to import a disk.

To import a disk:

1. Run `import-snapshot --description "My FDC VM" --disk-container`.

```
aws ec2 import-snapshot --description "My FDC VM" --disk-container
file://C:\private\aws\containers.json
```

Specify the URL of the S3 bucket, or provide the S3 bucket name and key.

```
{
    "Description": "My FDC VHD",
    "Format": "VHD",
    "UserBucket": {
        "S3Bucket": "fdc-bucket-tn",
        "S3Key": "fdc.aws.vhd"
    }
}
```

The following image shows the response of above command. The status shown is `active`, which means that the import is in progress.

```
C:\Users\nhou>aws ec2 import-snapshot --description "My FDC VM" --disk-container "file://C:\private\aws\containers.json"
{
    "Description": "My FDC VM",
    "ImportTaskId": "import-snap-0aba8b9978bedc8d9",
    "SnapshotTaskDetail": {
        "Description": "My FDC VM",
        "DiskImageSize": 0.0,
        "Progress": "0",
        "Status": "active",
        "StatusMessage": "pending",
        "UserBucket": {
            "S3Bucket": "fdc-bucket-tn",
            "S3Key": "fdc.aws.vhd"
        }
    },
    "Tags": []
}
```

2. Use the `describe-import-snapshot-tasks` command to check the status of an import snapshot task.

```
aws ec2 describe-import-snapshot-tasks --import-task-ids import-snap-
0aba8b9978bedc8d9
```

The snapshot is ready to use when the `status` is `complete`.

```
C:\Users\nhou>aws ec2 describe-import-snapshot-tasks --import-task-ids import-snap-0aba8b9978bedc8d9
{
    "ImportSnapshotTasks": [
        {
            "Description": "My FDC VM",
            "ImportTaskId": "import-snap-0aba8b9978bedc8d9",
            "SnapshotTaskDetail": {
                "Description": "My FDC VM",
                "DiskImageSize": 208028160.0,
                "Format": "VHD",
                "SnapshotId": "snap-083a9220a5876cf77",
                "Status": "completed",
                "UserBucket": {
                    "S3Bucket": "fdc-bucket-tn",
                    "S3Key": "fdc.aws.vhd"
                }
            },
            "Tags": []
        }
    ]
}
```

Creating AMI from a snapshot

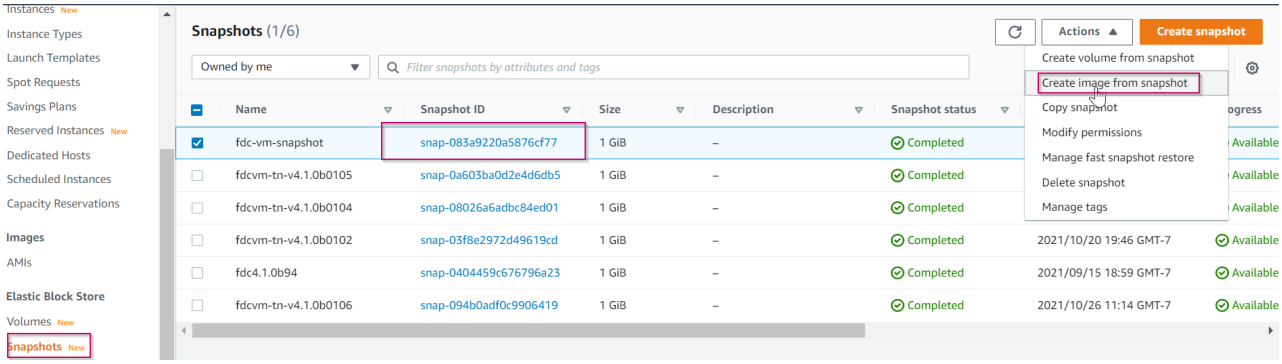
You can create an AMI with either the CLI or the AWS Management Console.

To create the AMI with the CLI:

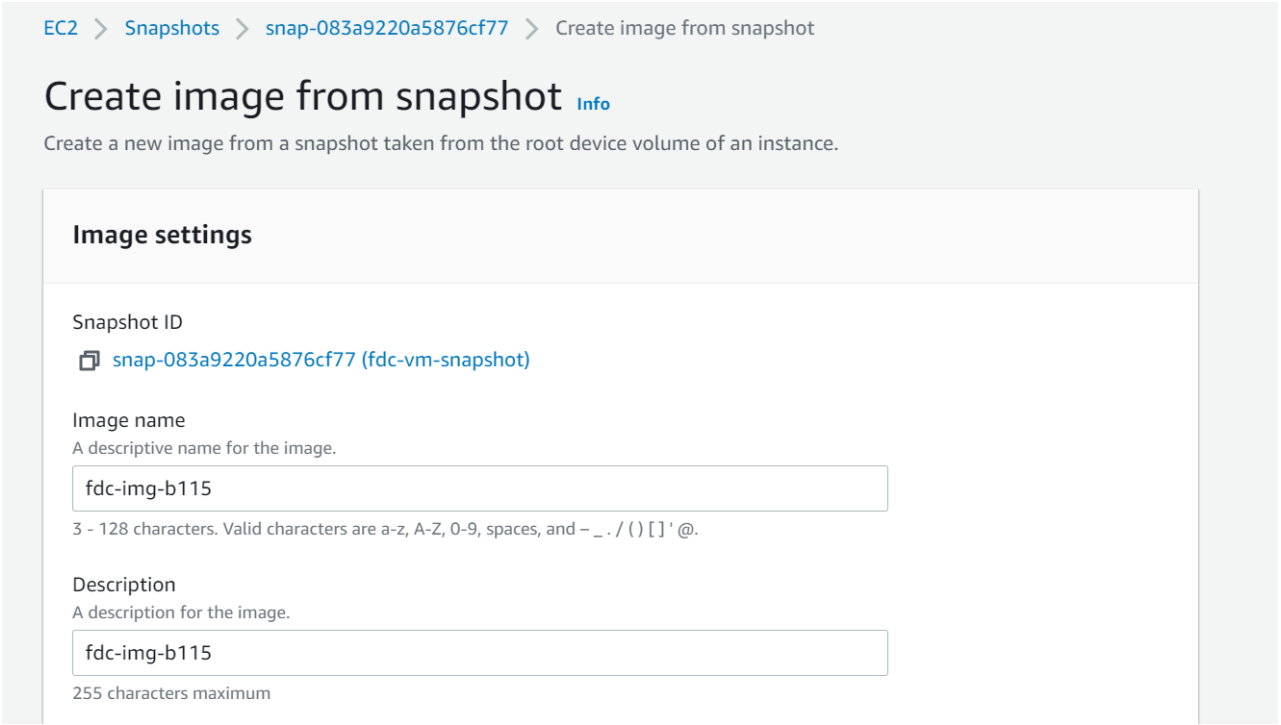
```
aws ec2 register-image --name fdc-img-cm --architecture x86_64 --root-device-name
/dev/sda1 --virtualization-type hvm --ena-support --block-device-mappings
DeviceName=/dev/sda1,Ebs={SnapshotId=snap-
083a9220a5876cf77,VolumeSize=1,VolumeType=gp2,DeleteOnTermination=true}
DeviceName=/dev/sdb,Ebs={VolumeType=gp2,VolumeSize=80,DeleteOnTermination=true}
```

To create the AMI with the AWS web console:

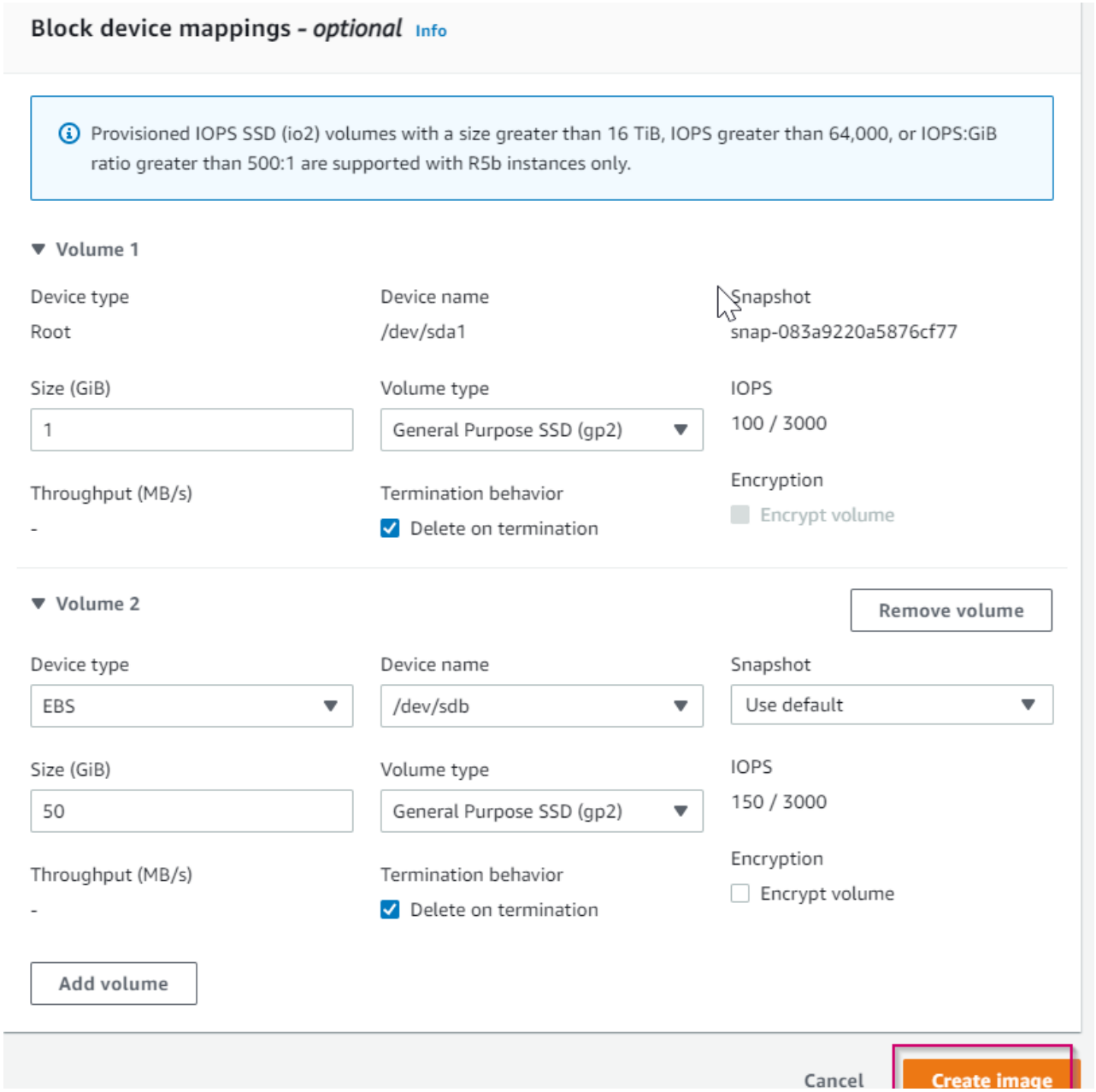
1. Choose *Snapshots* in the navigation pane of EC2.
2. Select the snapshot you imported.
3. Click *Create image from snapshot* in the *Actions* menu.



4. Enter the *Image Name* and *Description*.



5. Configure the *Block device mappings information*, and click *Create Image*.



Python script for importing the FortiDeceptor image

To view the help message for the for this script use the command `-h`.

```
import boto3
import time, sys, os,traceback
import json
import pprint
from datetime import datetime
from types import SimpleNamespace
```

```
global_region_name="us-west-2"
global_aws_access_key_id=""
global_aws_secret_access_key=""
global_bucket=""

class DatetimeEncoder(json.JSONEncoder):
    def default(self, obj):
        if isinstance(obj, datetime):
            return obj.strftime('%Y-%m-%dT%H:%M:%SZ')
        elif isinstance(obj, date):
            return obj.strftime('%Y-%m-%d')
    # Let the base class default method raise the TypeError
        return json.JSONEncoder.default(self, obj)

def check_return(resp):
    if resp != None:
        if resp['ResponseMetadata']['HTTPStatusCode'] == 200:
            return 0
    return -1

def list_bucket():
    bna = []
    for bucket in s3.buckets.all():
        bna.append(bucket.name)
    return bna

def resp2obj(resp):
    s = json.dumps(resp, cls=DatetimeEncoder)
    return json.loads(s, object_hook=lambda d: SimpleNamespace(**d))

def bucket_exists(s3s, fk):
    for b in s3s.buckets.all():
        if b.name == fk:
            return True
    return False

def import_as_AMI(filename,imagename, arch, size):
    if filename is None:
        print("Incorrect parameter")
        return

    fn = filename #sys.argv[1]
    fk = imagename #sys.argv[2]
    arch =  arch #sys.argv[3]
    size = size #sys.argv[4]
    s3s = boto3.resource('s3', region_name=global_region_name, aws_access_key_id=global_aws_
access_key_id, aws_secret_access_key=global_aws_secret_access_key)
    s3c = boto3.client('s3', region_name=global_region_name, aws_access_key_id=global_aws_
access_key_id, aws_secret_access_key=global_aws_secret_access_key)
    buck=global_bucket
    if not bucket_exists(s3s, buck):
        bucket = s3s.create_bucket(ACL='private', Bucket=buck, CreateBucketConfiguration=
{'LocationConstraint':global_region_name})
        if bucket != None:
            bucket.wait_until_exists()
        else:
```

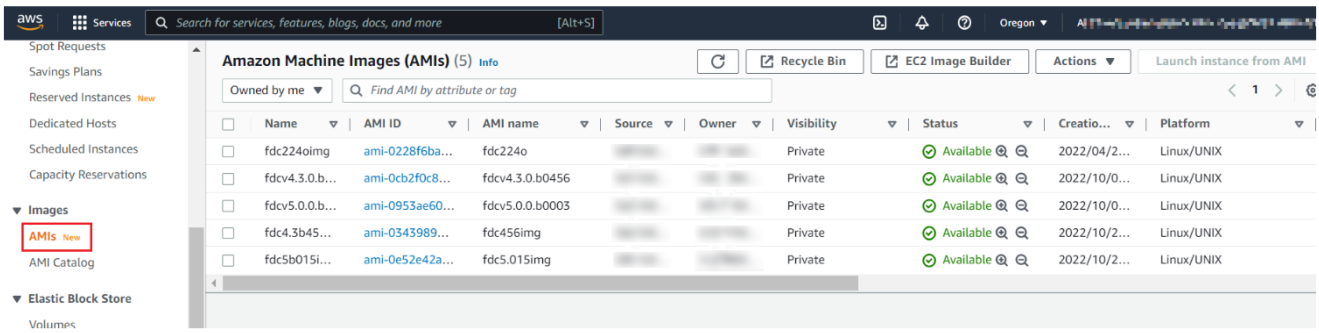
```
            print("Failed to create bucket %s" % (buck))
            return
    else:
        bucket = s3s.Bucket(buck)
    bucket = s3s.Bucket(buck)
    s3c.delete_object(Bucket=buck, Key=fk)
    bucket.upload_file(fn, fk)
    ec2 = boto3.client('ec2', region_name=global_region_name, aws_access_key_id=global_aws_
access_key_id, aws_secret_access_key=global_aws_secret_access_key)
    try:
        resp = ec2.import_snapshot(
            Description='import FDC image snapshot',
            DiskContainer={
                'Format': 'VHD',
                'UserBucket': {
                    'S3Bucket': buck,
                    'S3Key': fk
                }
            })
        r = resp2obj(resp)
    except Exception as e:
        print('''Please make sure you have the service role 'vmimport' with below
permissions:
            -- Resource to s3:your-bucket
            *) s3:ListBucket
            *) s3:GetBucketLocation
            *) s3:GetObject
            -- Resource to ec2:*
            *) ec2:ModifySnapshotAttribute
            *) ec2:CopySnapshot
            *) ec2:RegisterImage
            *) ec2:Describe*

            For more information, please refer to https://docs.aws.amazon.com/vm-
import/latest/userguide/vmie_prereqs.html , section 'Required service role'
        ''')
        print(traceback.format_exc())
        sys.exit(-1)

    print("Importing image: taskid={}".format(r.ImportTaskId))
    while True:
        time.sleep(10)
        resp = ec2.describe_import_snapshot_tasks(ImportTaskIds=[r.ImportTaskId])
        #print(resp)
        if check_return(resp) == 0:
            taskdetail = resp['ImportSnapshotTasks'][0]
            st = taskdetail['SnapshotTaskDetail']['Status']
            print("Importing image: {}".format(st))
            if st == 'completed':
                break
            elif st == "deleted":
                print(taskdetail)
                return

    print("Imported image successfully")
    r = resp2obj(resp)
```

```
    ec2s = boto3.resource('ec2', region_name=global_region_name, aws_access_key_id=global_
aws_access_key_id, aws_secret_access_key=global_aws_secret_access_key)
    snapshot = ec2s.Snapshot(r.ImportSnapshotTasks[0].SnapshotTaskDetail.SnapshotId)
    snapshot.create_tags(Tags=[{'Key':'Name', 'Value':fk}])
    resp = ec2.register_image(Name=fk, Architecture=arch, RootDeviceName='/dev/sda1',
        BlockDeviceMappings=[{'DeviceName':'/dev/sda1',
                              'Ebs':
{'SnapshotId':snapshot.id,'VolumeType':'gp2','VolumeSize':int
(size),'DeleteOnTermination':True}},
                              {'DeviceName':'/dev/sdb',
                              'Ebs':
{'VolumeType':'gp2','VolumeSize':50,'DeleteOnTermination':True}},],
                              VirtualizationType='hvm', EnaSupport=True)
    if check_return(resp) == 0:
        print("Registered image successfully")
    else:
        print("Failed to register image")
        print(resp)
    r = resp2obj(resp)
    image = ec2s.Image(r.ImageId)
    image.create_tags(Tags=[{'Key':'Name', 'Value':fk}])
    s3c.delete_object(Bucket=buck, Key=fk)
    print("Deleted the image file from bucket {}".format(buck))


if __name__ == "__main__":
    import argparse
    parser = argparse.ArgumentParser()

    parser.add_argument("-r", "--region_name", help="region_name")
    parser.add_argument("-i", "--aws_access_key_id", help="aws_access_key_id")
    parser.add_argument("-k", "--aws_secret_access_key", help="aws_secret_access_key")
    parser.add_argument("-b", "--bucket", help="The bucket name")
    parser.add_argument("-f", "--filename", help="The FDC AWS vhd full file name")
    parser.add_argument("-n", "--imagename", help="The AMI image name on AWS")
    parser.add_argument("-a", "--arch", help="Optional: default is 86_64")
    parser.add_argument("-s", "--size", help="Optional: The size of the image file, default
is 1GB. ")
    args = parser.parse_args()

    global_region_name=args.region_name
    global_aws_access_key_id=args.aws_access_key_id
    global_aws_secret_access_key=args.aws_secret_access_key

    global_bucket="fdcbucket".lower()
    if args.bucket:
        global_bucket = args.bucket

    filename=args.filename
    imagename=args.imagename
    arch="x86_64"
    if args.arch:
        arch = args.arch
    size=1
    if args.size:
        size=args.size
```

```
import_as_AMI(filename, imagename, arch, size)
```

Check the imported image in AMIs

In the AWS console go to *Images* > *AMIs*. Verify the AMI you uploaded is displayed.

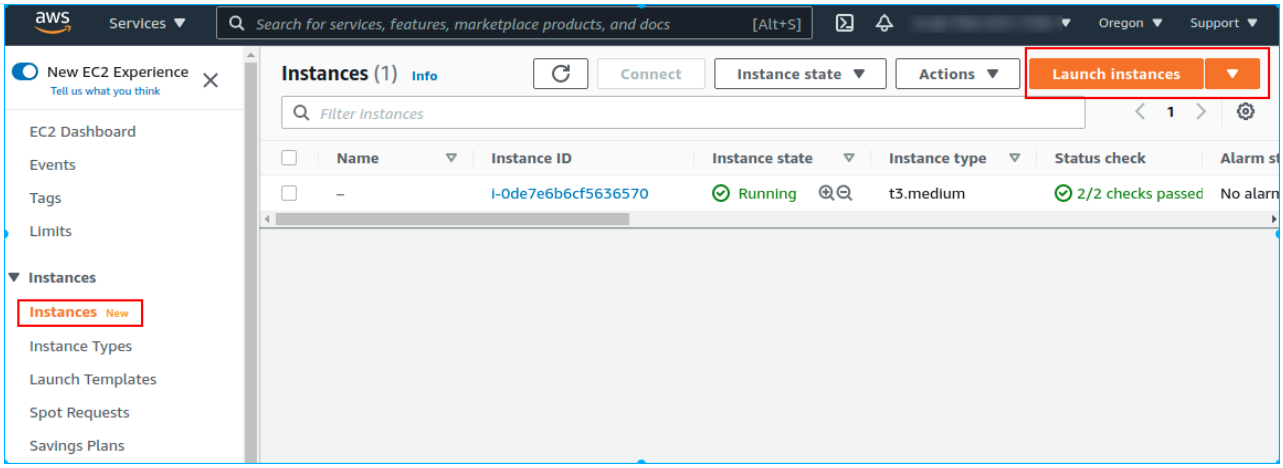


Create an instance with the imported AMI image

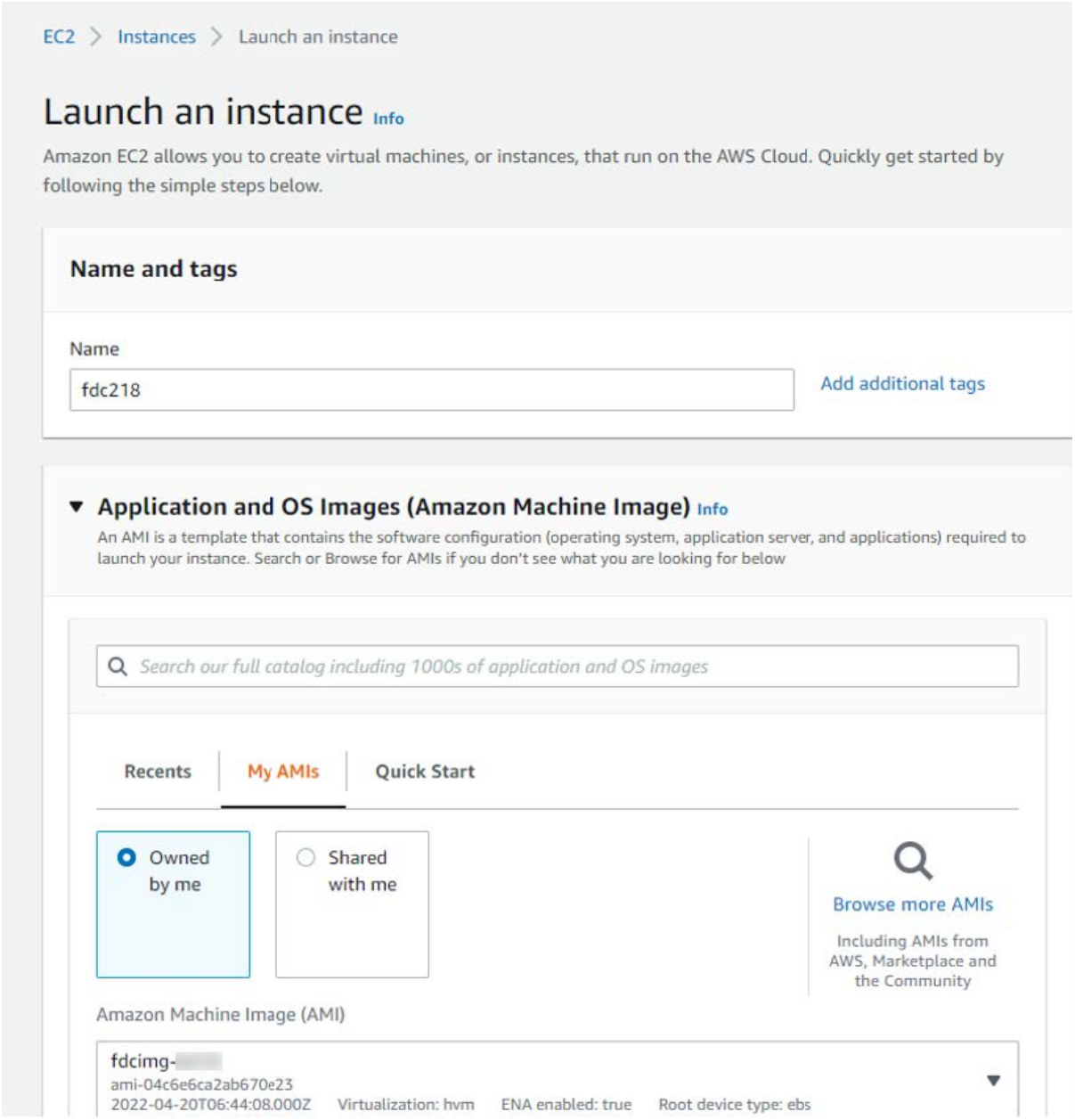
The *Instance Wizard* specifies all the launch parameters required for launching an instance. Where the launch instance wizard provides a default value, you can accept the default or specify your own value, like choosing the AMI you created in the last step, configuring your own network interfaces and specifying the security group.

To create an instance with an imported image:

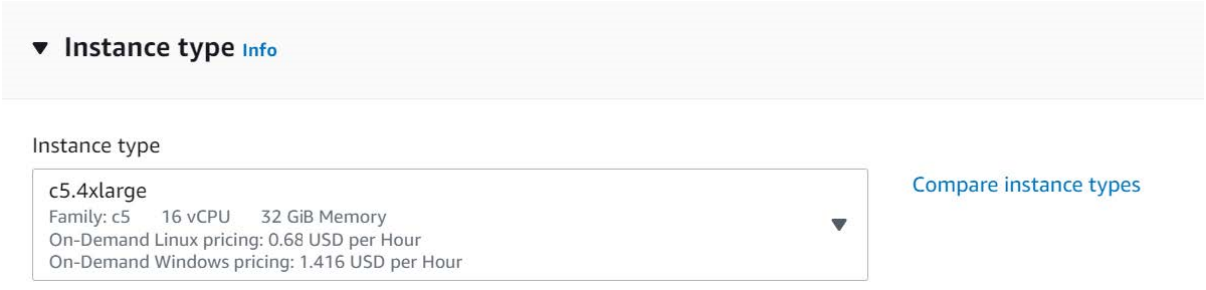
1. In the AWS console go to *Instances* > *Instances*.
2. Click *Launch Instances*. Instance wizard opens.



3. In *Step 1: Choose an Amazon Machine Image (AMI)*, click *My AMIs* and then select the image you just created, then click *Next*.



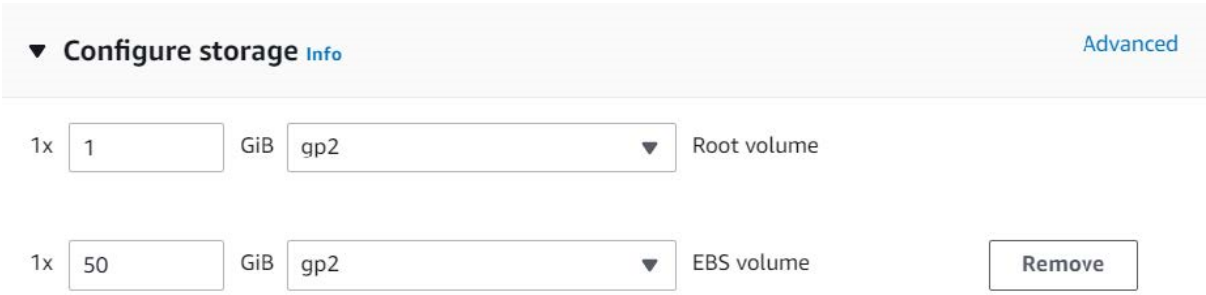
4. In *Step 2: Chose an Instance Type*, select the instance type. For more information, see Elastic network interfaces.



5. In *Step 3: Configure Network settings and Security Group*, click *Edit*.
   a. Select 2 to 6 NICs. You must configure at least two NICs.
   b. Click *Add network interface* to add more network interfaces

Make sure ports 22, 443, 8443 are open in FortiDeceptor port1. This allows the FortiDeceptor Manager to communicate with the cloud clients.

6. In *Configure storage*, configure the storage settings and click *Launch Instance*.

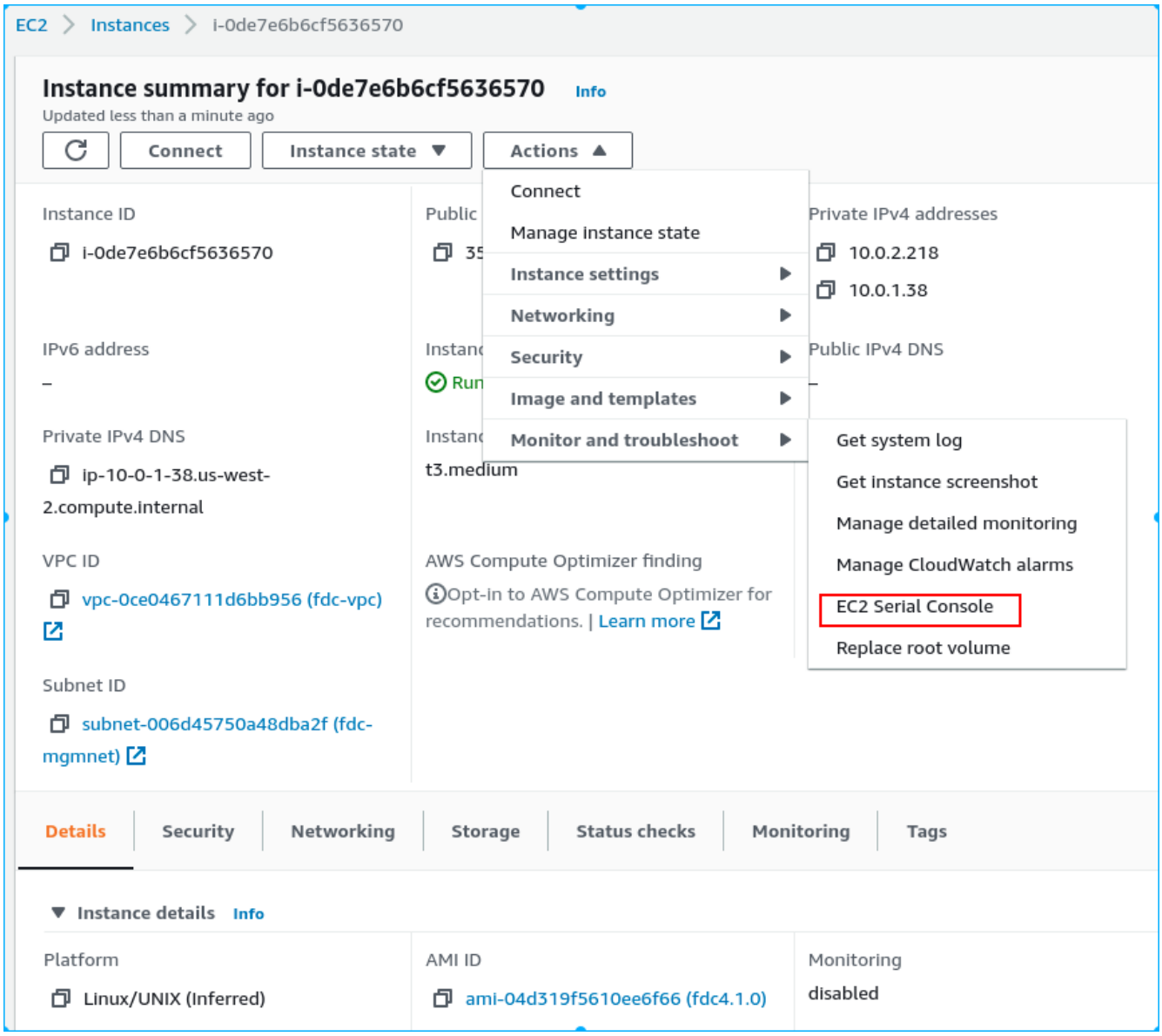


Connect the Instance with the Serial Console

To connect the instance with the Serial Console:

1. In the AWS Management Console, go to *EC2* > *Instances*.
2. Click the instance you created to open it. The *Instance summary for <instance_id>* page opens.
3. Click *Actions* > *Monitor and troubleshoot* > *EC2 Serial Console*.

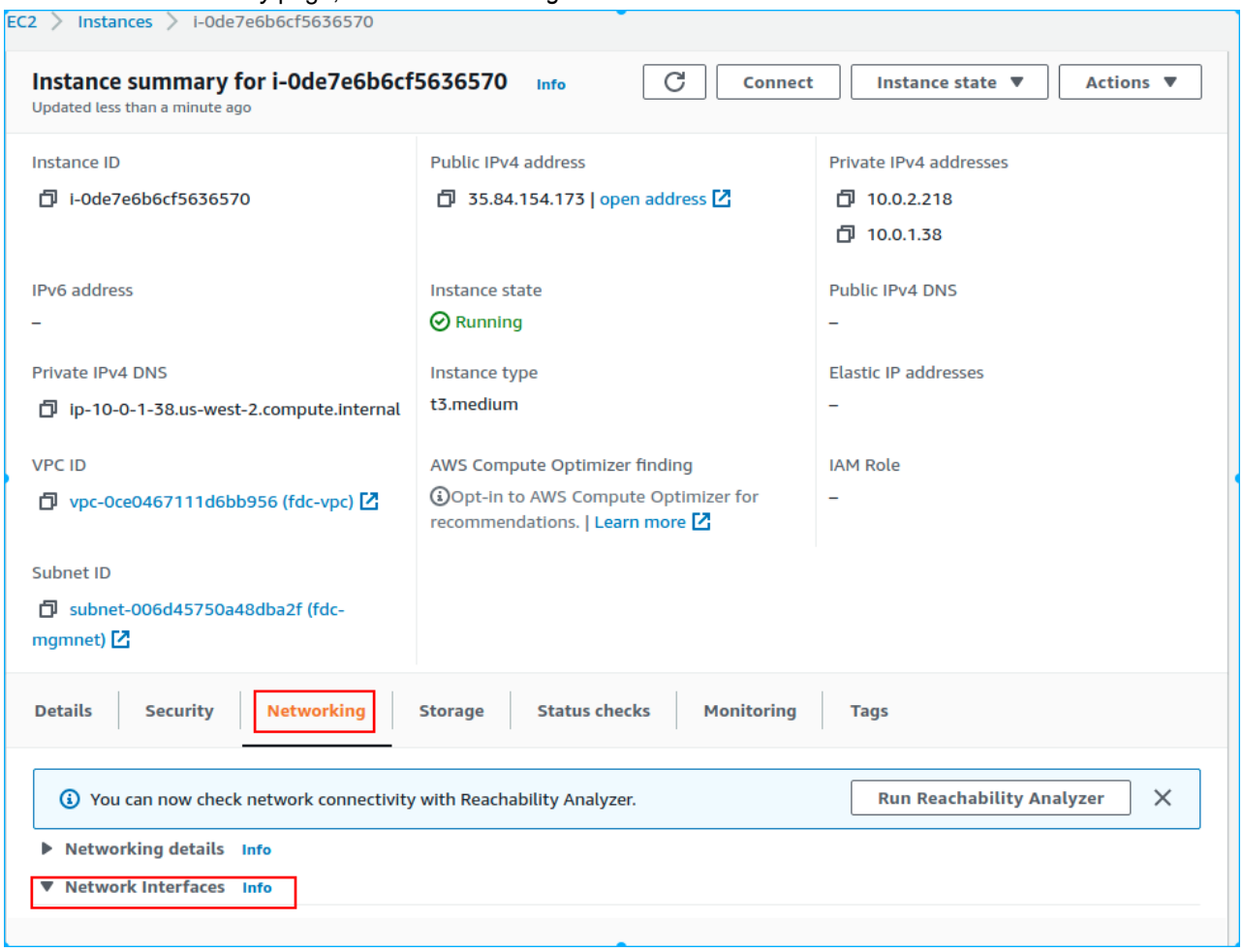
Stay on the Instance Summary page.


EC2 > Instances > i-0de7e6b6cf5636570
Instance summary for i-0de7e6b6cf5636570 Info
Updated less than a minute ago
Connect
Instance state
Actions
Connect
Manage instance state
Instance settings
Networking
Security
Image and templates
Monitor and troubleshoot
Get system log
Get instance screenshot
Manage detailed monitoring
Manage CloudWatch alarms
EC2 Serial Console
Replace root volume
Instance ID
i-0de7e6b6cf5636570
IPv6 address
–
Private IPv4 DNS
ip-10-0-1-38.us-west-2.compute.internal
VPC ID
vpc-0ce0467111d6bb956 (fdc-vpc)
Subnet ID
subnet-006d45750a48dba2f (fdc-mgmnet)
Private IPv4 addresses
10.0.2.218
10.0.1.38
Public IPv4 DNS
–
t3.medium
AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more
Details
Security
Networking
Storage
Status checks
Monitoring
Tags
Instance details Info
Platform
Linux/UNIX (Inferred)
AMI ID
ami-04d319f5610ee6f66 (fdc4.1.0)
Monitoring
disabled

Associate Public IP to instance port1

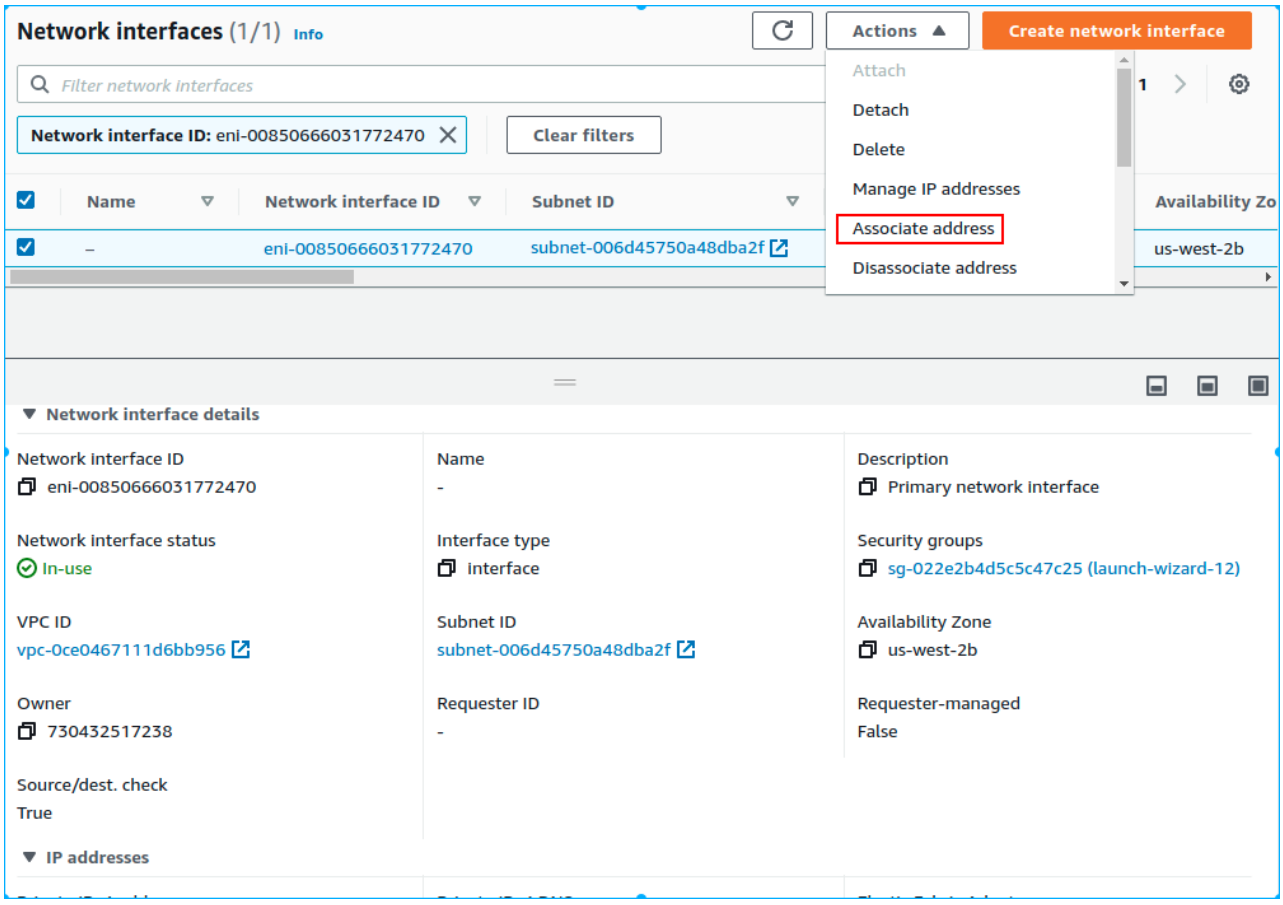
To associate a public IP to an instance:

1. In the *Instance Summary* page, click the *Networking* tab.

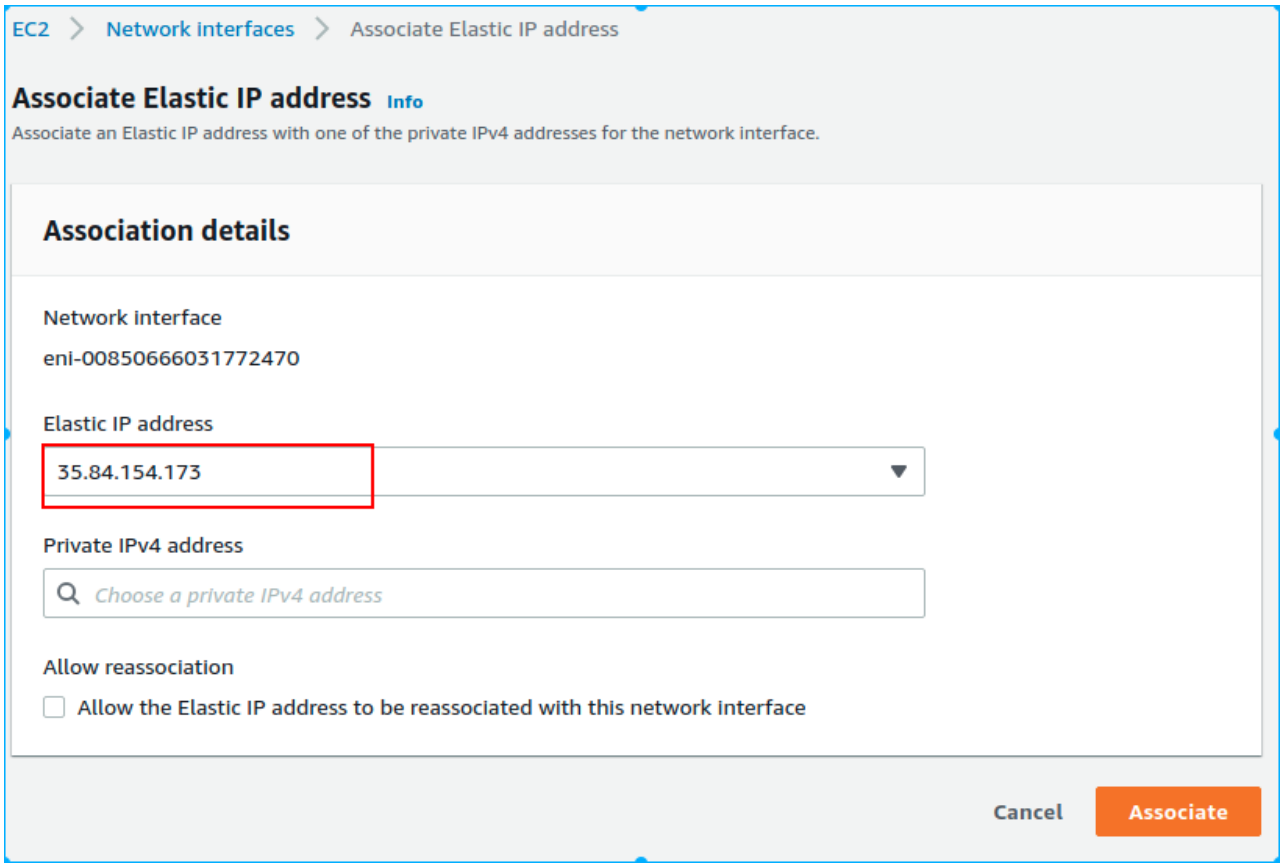


2. Expand the *Network interfaces* section.

3. Click *Actions* > *Associate address*. The *Associate Elastic IP address* page opens.



4. From the *Elastic IP address* dropdown, select the elastic IP you created.



5. Click *Associate*.

Keep the *Network Interfaces* page open.

Configure secondary IPs

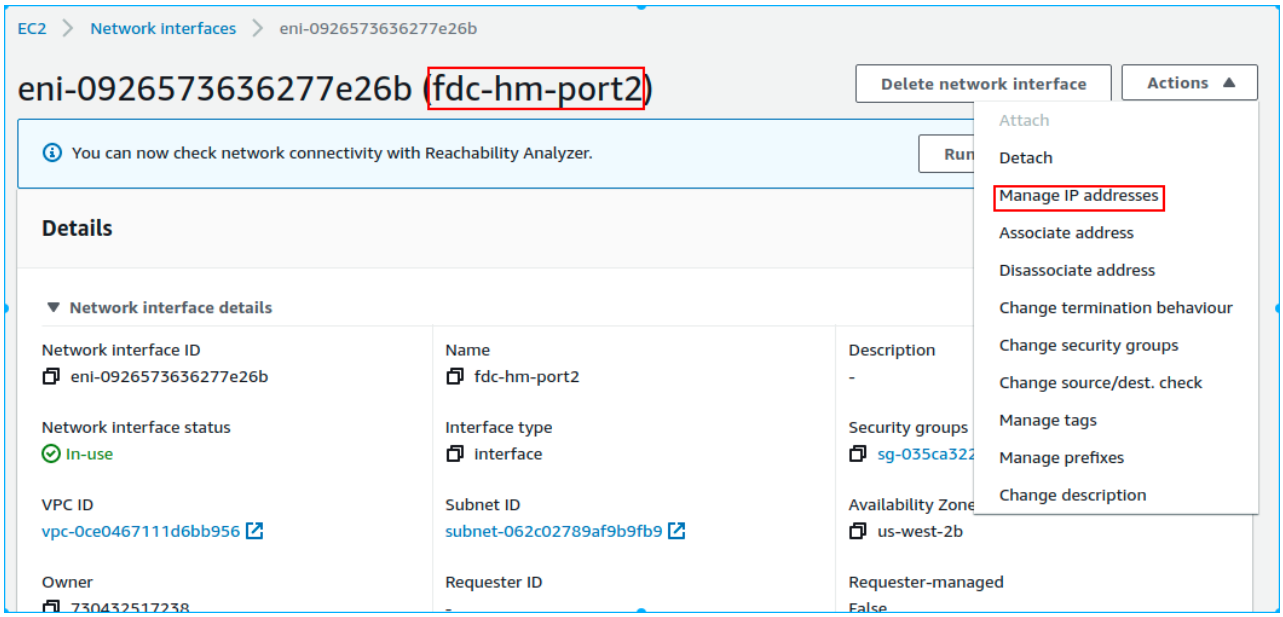
Add more Interfaces and then configure multiple secondary IPs for FDC port2 for decoy deployment. This required when adding more decoy IPs in the future.

To add more interfaces:

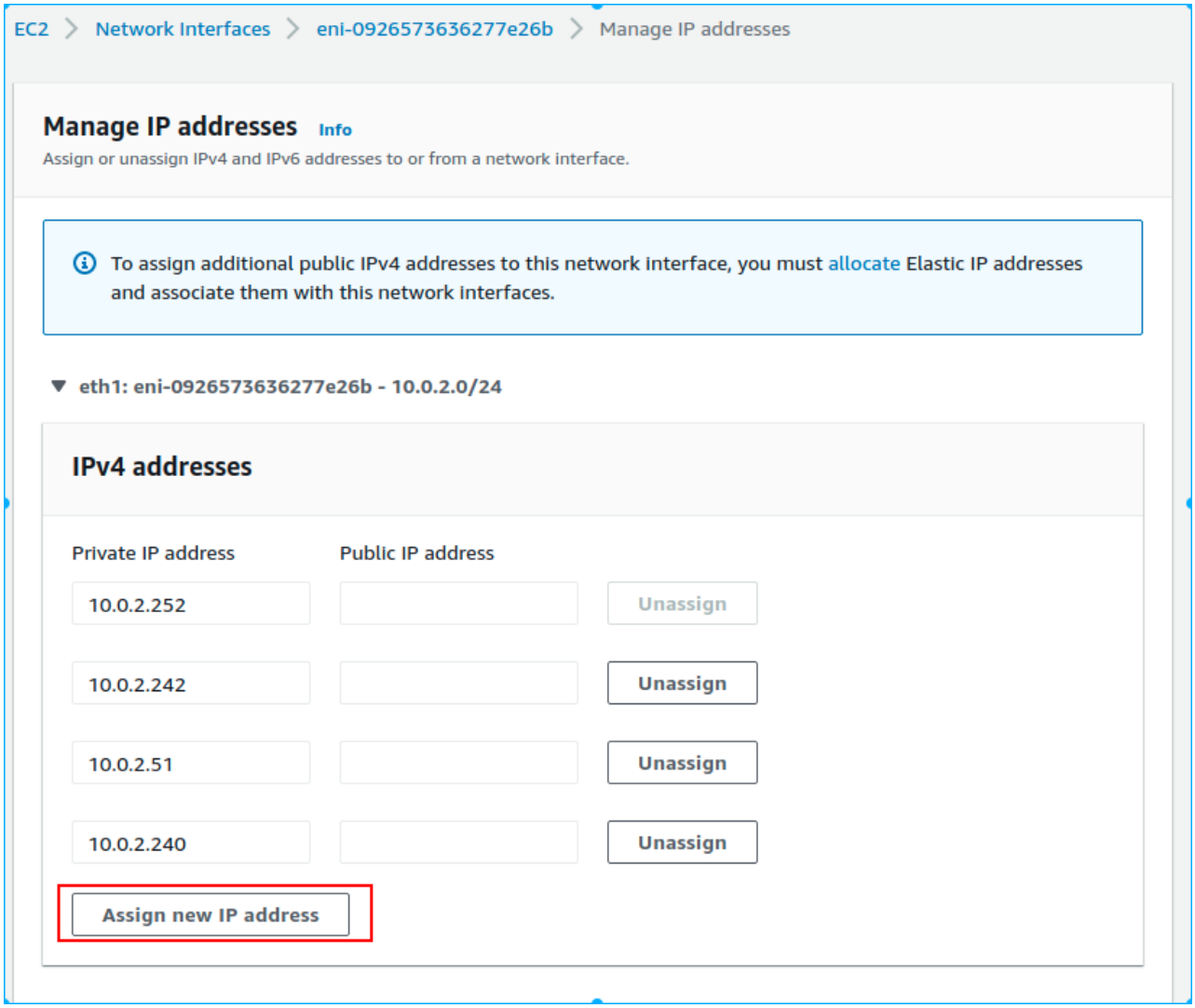
1. Go to *EC2* > *Instance* and click the *Instance ID*.
2. Go to *Action* > *Networking* > *Attach network interface*.

To configure secondary IPs:

1. Click *Actions* > *Manage IP address*. The *Manage IP addresses* page opens.



2. In the *IPv4 addresses* section, click *Assign new IP address*.



Configuring the FortiDeceptor Manager and AWS Client

To configure FortiDeceptor manager, get the appliance authentication key and use it to add a cloud appliance. Next you will add subnets to the deployment network and deploy the decoys.

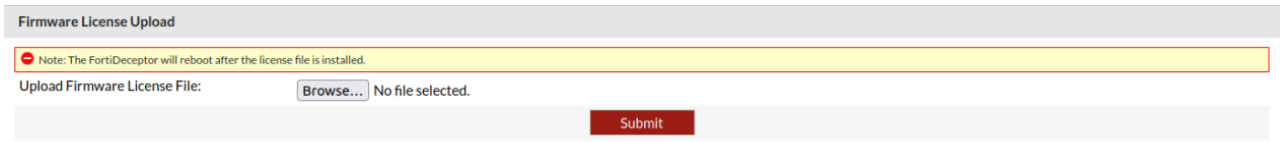
To configure FortiDeceptor and AWS Client:

1. Configure the client on page 35.
2. Configure FortiDeceptor manager.
3. Configure the deployment network.
4. Deploy the decoys.

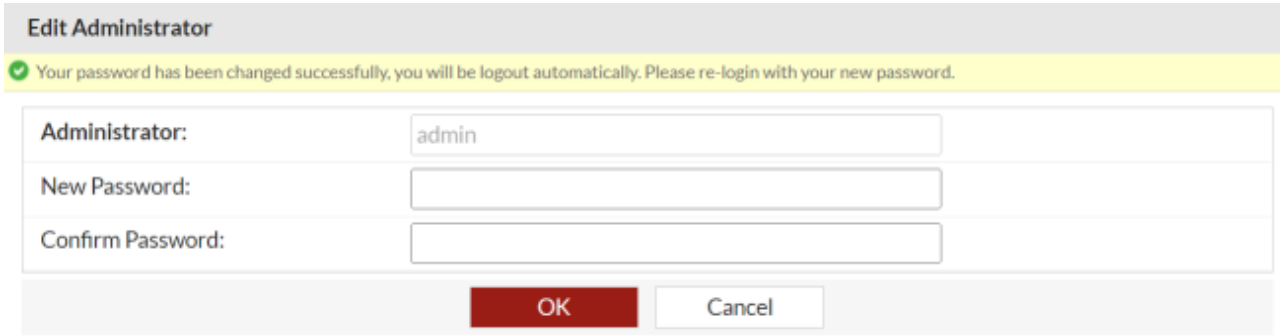
Configure the client

To configure the AWS client:

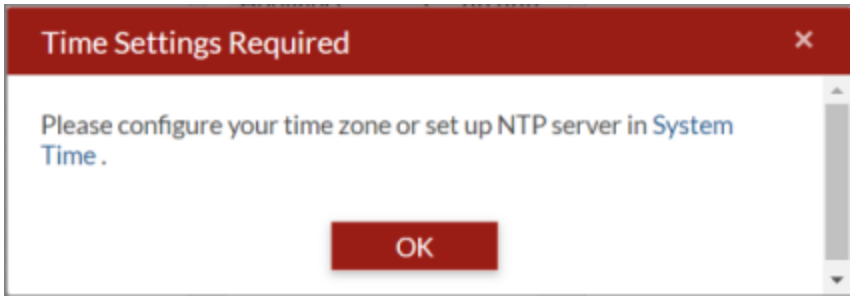
1. Log in to the AWS client with the public IP address. By default, the *admin* user account has no password.
2. After logging in, the FortiDeceptor instance prompts you to upload the license file. Click *Choose File* to navigate to the file and click *Submit*. After the file submitted, FortiDeceptor will reboot.



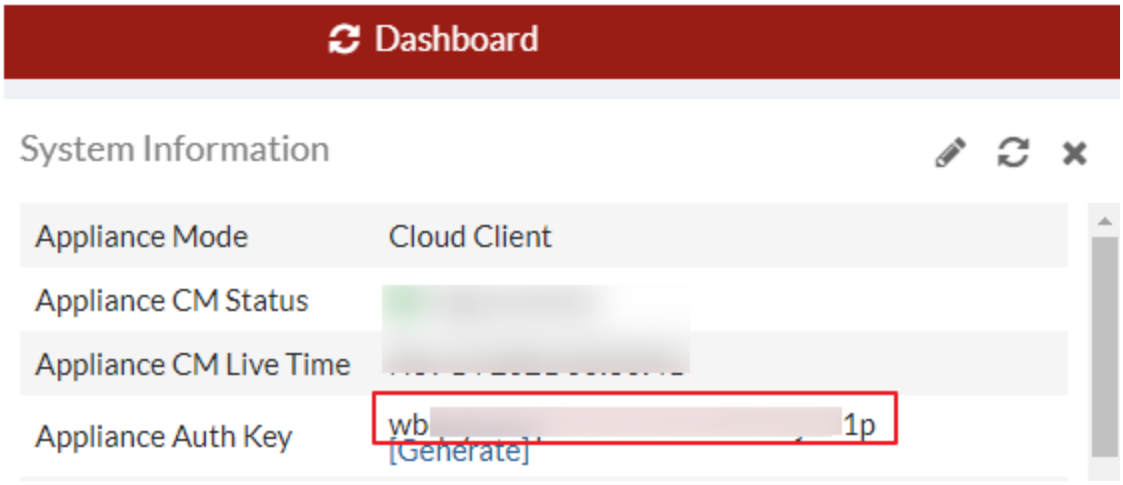
3. After the instance reboots, you are prompted to change the password and log in again.



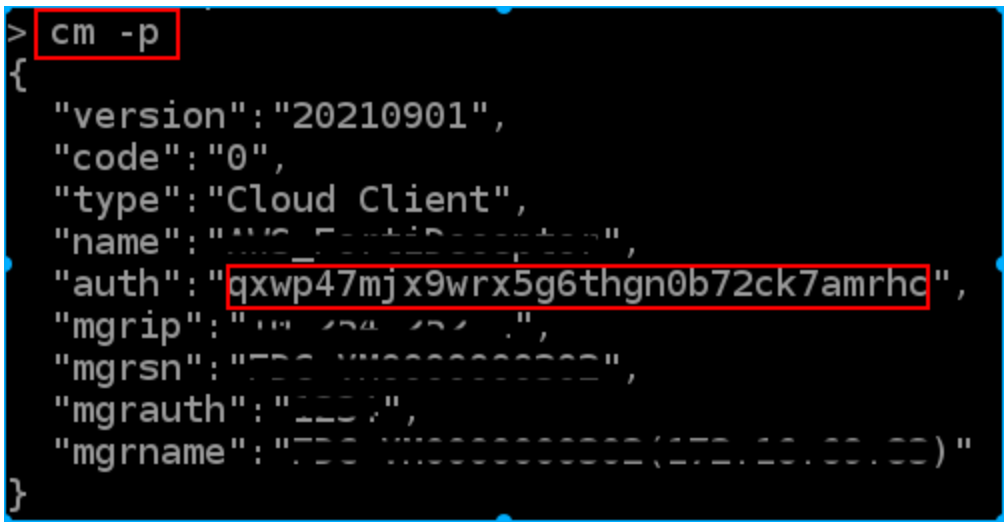
4. After you log in, you are prompted to configure the timezone and time.



5. Change the Host Name.
   a. Go to *Dashboard* > *System information* > *Host Name* and click *Change*. The *Edit Host Name* field opens.
   b. In the *New Name* field, enter a the new Host Name.
6. Get the appliance key with the GUI or CLI.
   - GUI: Go to *Dashboard* > *System Information* widget and locate the *Appliance Auth Key*.



   - CLI: `cm -p`

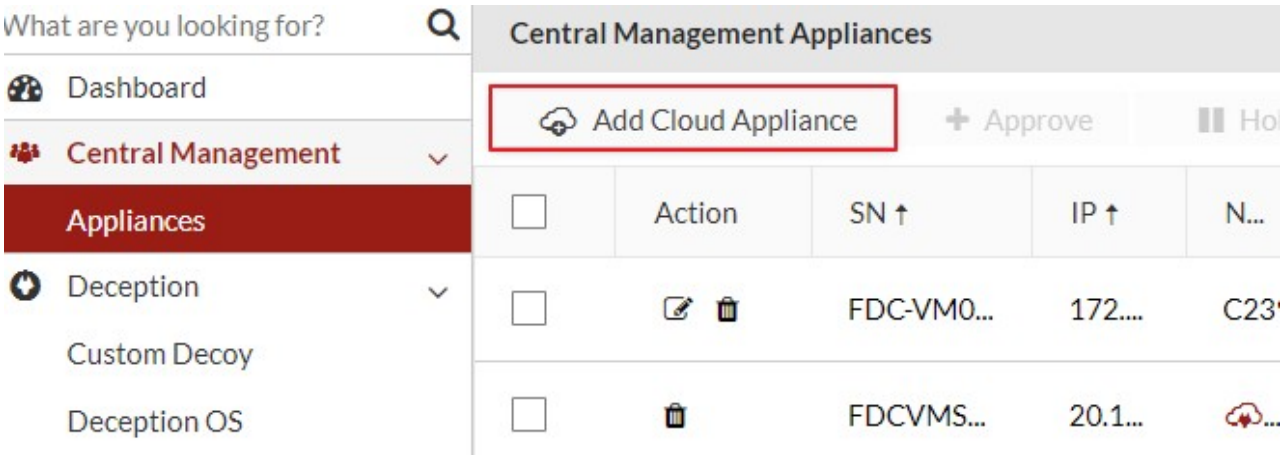


Configuring FortiDeceptor Manager

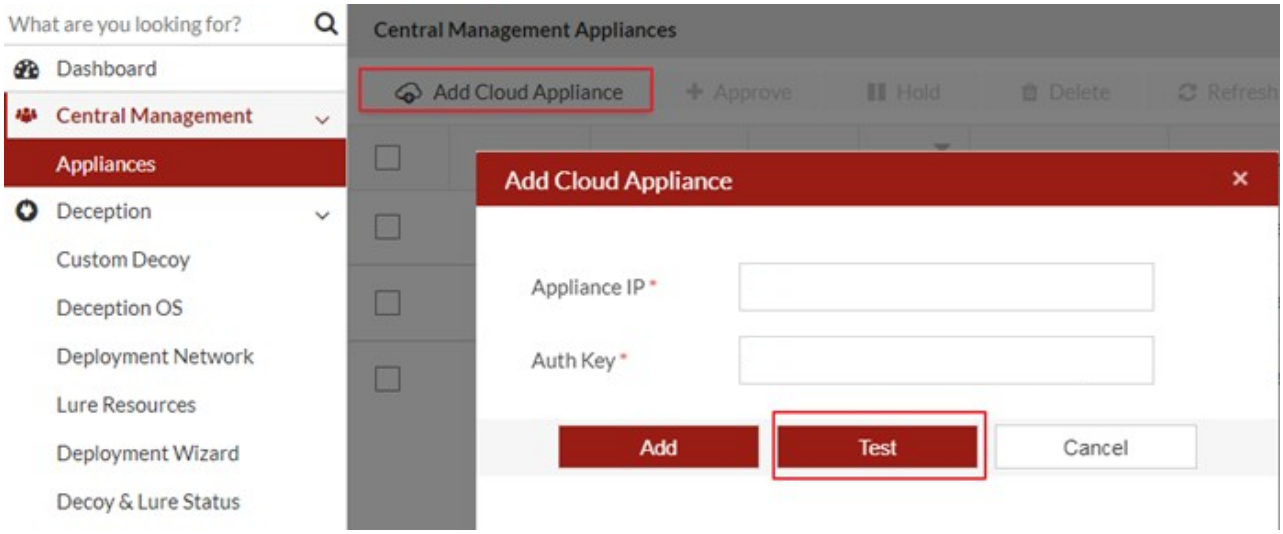
Add the AWS FortiDeceptor as a client appliance.

To configure FortiDeceptor manager:

1. In FortiDeceptor, go to *Central Management* > *Appliances*.
2. Click *Add Cloud Appliance*. The *Add Cloud Appliance* dialog opens.



3. Configure the following settings:



Appliance IP	Enter the cloud client's public IP address.
Auth Key	Enter Appliance Authorization Key you generated in Step 8.

4. Click *Test*. You should see the message, *Successfully communicated with the cloud appliance*.

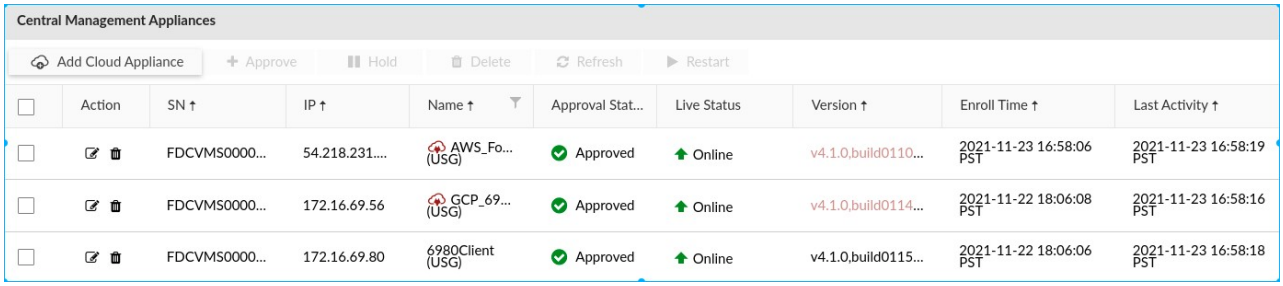
5. Click *Add* to add this cloud appliance.

Delete the previous client and add the client with new public IP once the public IP is changed.

Manage Cloud Clients

To delete a cloud appliances:

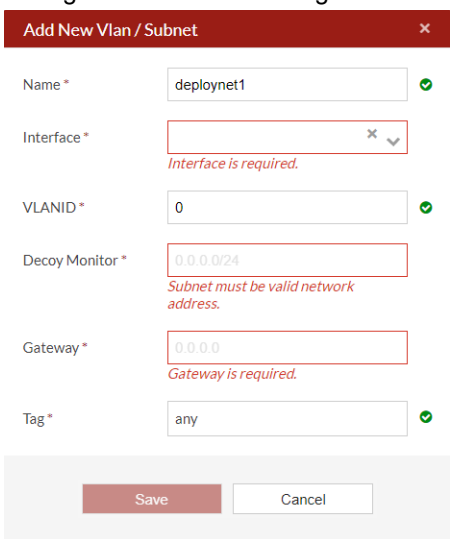
1. Go to *Central Management* > *Appliances*.
2. In the *Action* column, click the *Trash* icon.



Configure the deployment network

To configure the deployment network:

1. Go to *Deception* > *Deployment Network*.
2. Click *Add New Vlan/Subnet*. The *Add New Vlan/Subnet* dialog opens.
3. Configure the network settings and click *Save*.

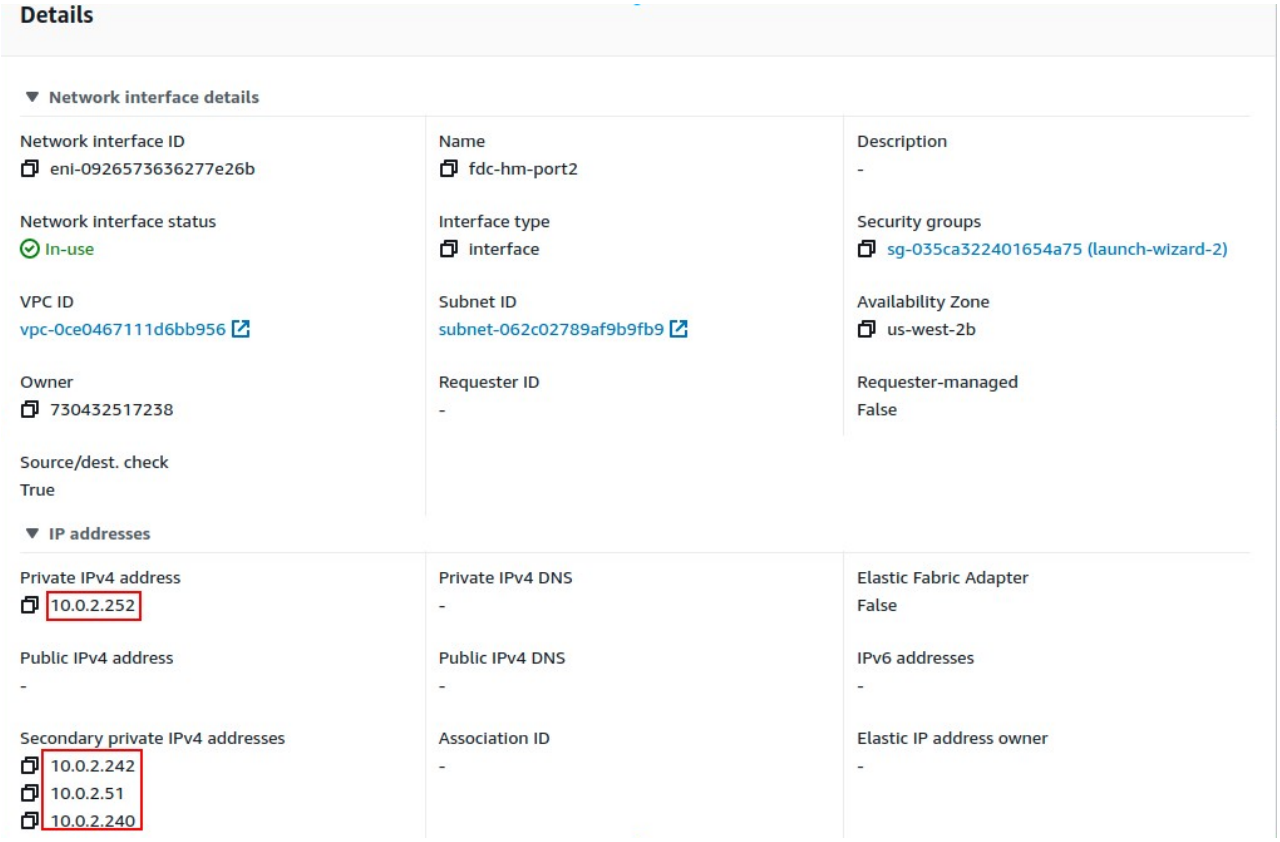


Deploy the decoys

Checking for multiple IPs

To check the multiple IPs on AWS platform:

1. In the AWS Management Console, go to *Network Interfaces*.

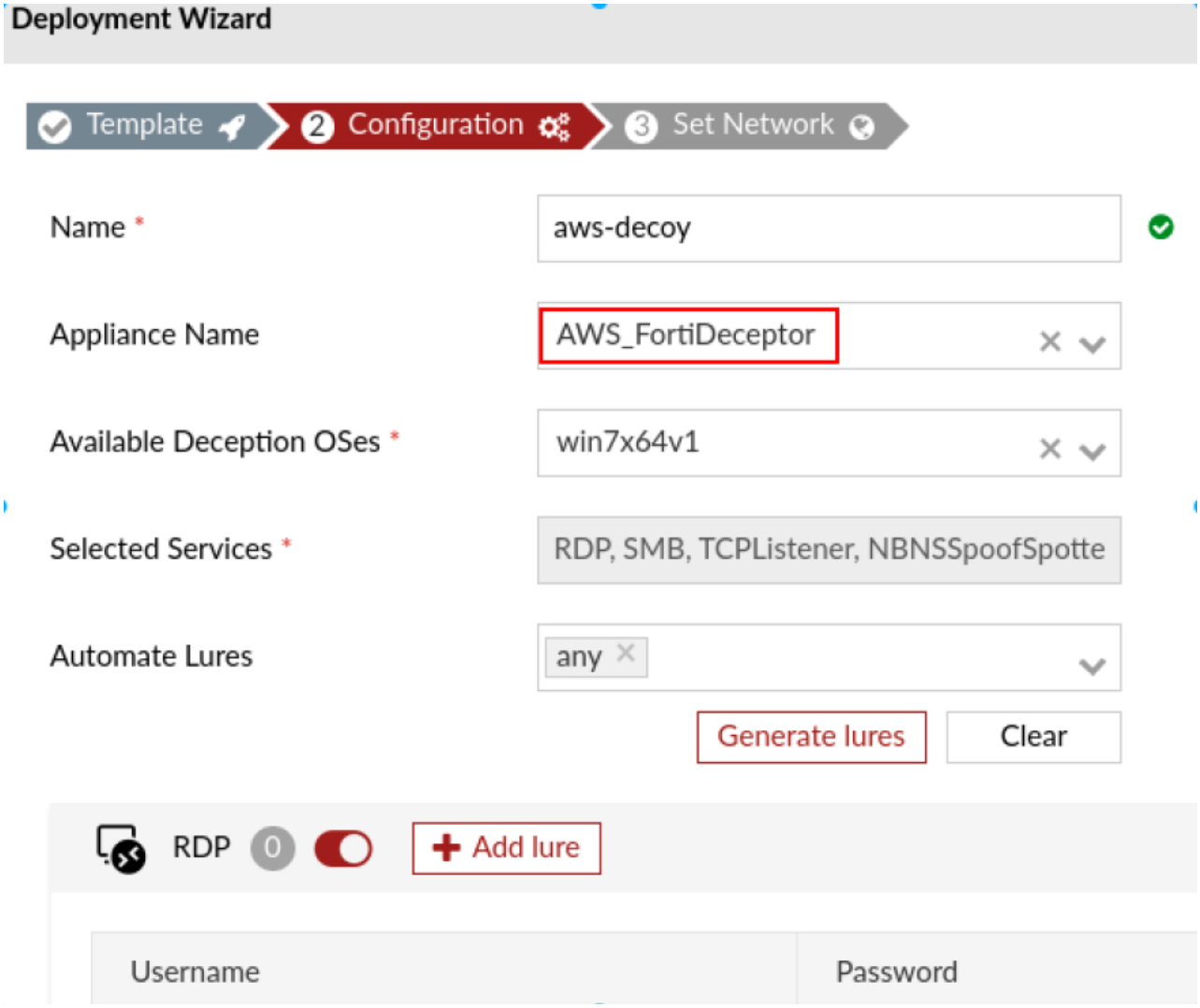


2. Open the Interface you created and verify the values in *Private IPv4 address* and *Secondary private IPv4 Addresses*.

Configuring decoys on FortiDeceptor manager

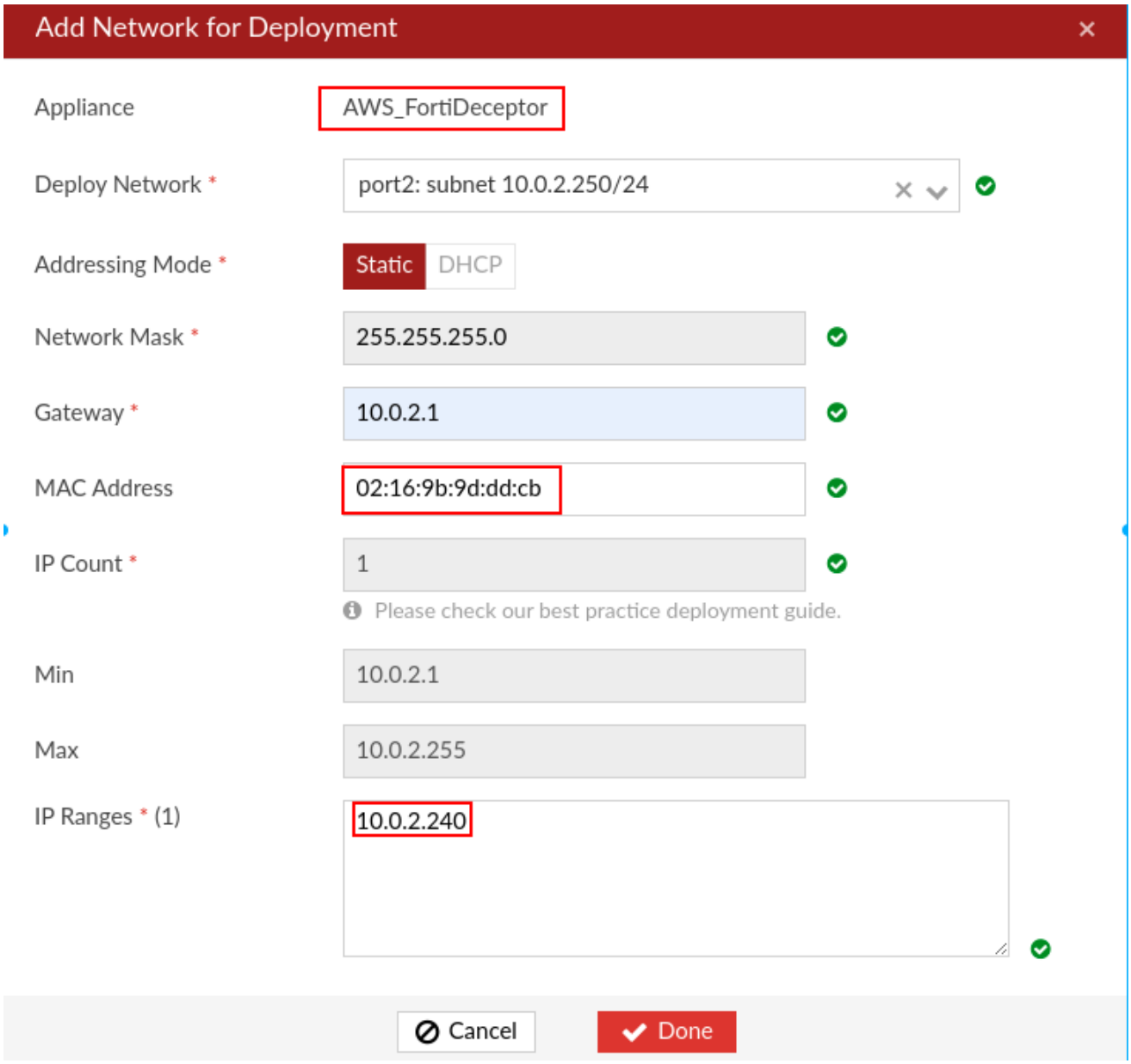
To choose a cloud appliance in deployment wizard.

1. In FortiDeceptor, go to *Deception Deployment Wizard* and create a new template.
2. In the *Configuration* tab, in *Appliance Name* choose the AWS cloud device.



3. In the *Set Network* tab, click *Add network for Deployment* and configure the following settings:

Deploy Network	Select one of the multiple IPs.
Mac Address	Enter the MAC address for the cloud device.
IP Ranges	Enter the IP range.



4. Click *Done* to deploy the decoy.
5. (Optional) Deploy more decoys.
   - To deploy decoys for different interfaces, repeat Checking for multiple IPs on page 39
   - To deploy more decoys for the same interface, repeat Configuring decoys on FortiDeceptor manager on page 40.
6. Attack this decoy IP via the endpoint in the cloud and check the incidents as regular deployment.

www.fortinet.com