

Lab 3 AWS EC2

Table of Contents

1. Amazon Web Service Portal	2
1.1. Preparations	2
1.2. Launch of an EC2 Instance	3
1.3. Security Groups	5
1.4. Volumes and Snapshots	8
1.5. Load Balancers	12
1.5.1. Classic Load Balancing	12
1.5.2. Path-Based Load Balancing	14
2. AWS CLI	17
2.1. Configuration	17
2.2. AWS CLI with EC2	18
2.2.1. Key-Pairs	18
2.2.2. Virtual Private Cloud - VPC	18
2.2.3. Run instance	19
2.2.4. Querying output	19
2.3. Working with the CLI	21
2.3.1. Instances	21
2.3.2. Security Groups	21
2.3.3. Volumes and Snapshots	22
2.3.4. Load Balancer	24

1. Amazon Web Service Portal

1.1. Preparations

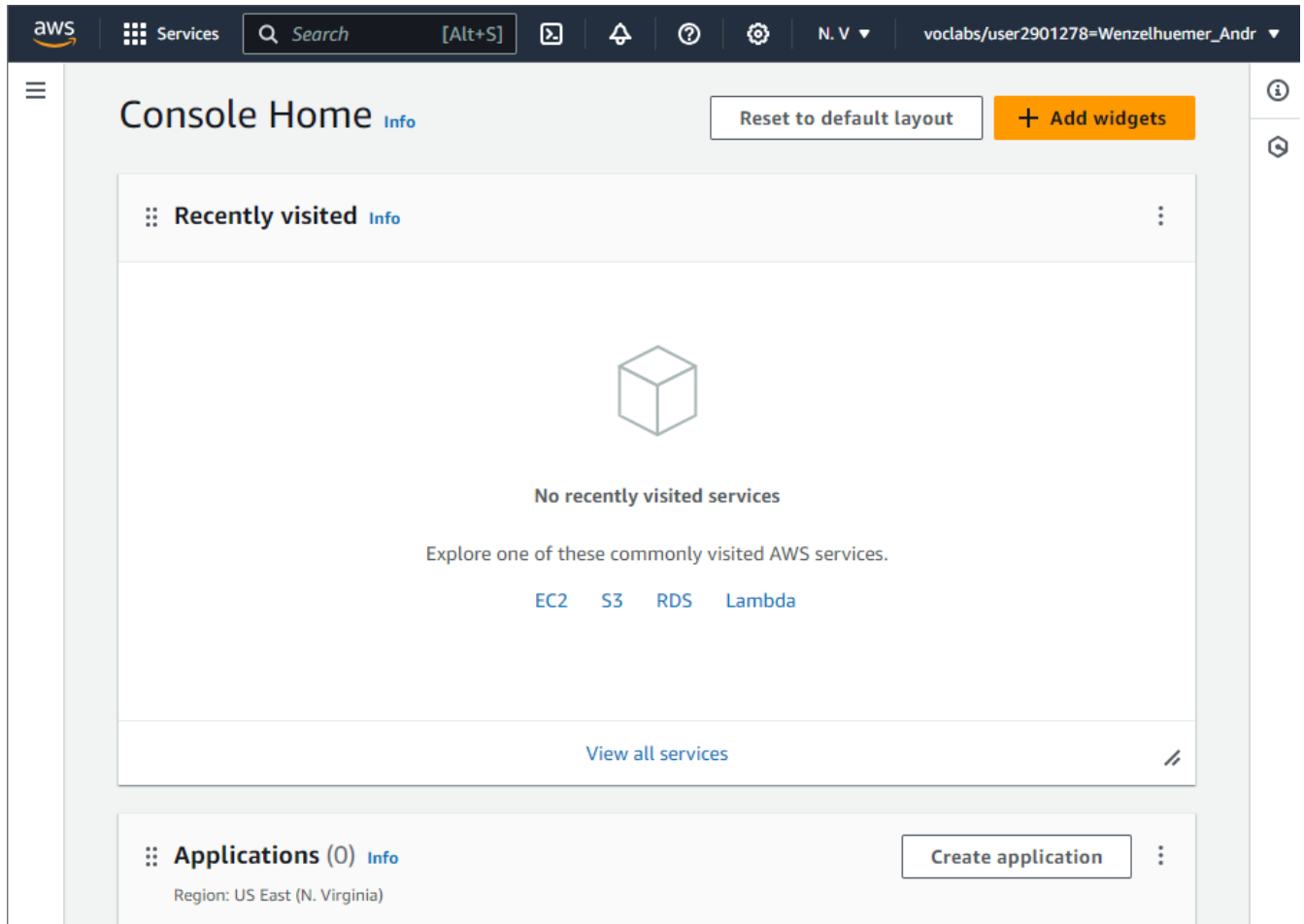


Figure 1. Amazon Web Services Portal

1.2. Launch of an EC2 Instance

▼ Summary

Number of instances

Info

1

Software Image (AMI)

Amazon Linux 2 AMI (HVM) - Ker...read more

ami-00b8917ae86a424c9

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

❗

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

×

Cancel

Launch instance

Review commands

Figure 2. Create new instance

✔ **Success**
Successfully initiated launch of instance ([i-0852c52b47cbac94a](#))

▼ Launch log

Initializing requests	✔ Succeeded
Creating security groups	✔ Succeeded
Creating security group rules	✔ Succeeded
Launch initiation	✔ Succeeded

Figure 3. Instance launched

Info

Connect to your instance `i-0852c52b47cbac94a` (lab3-ec2) using any of these options


EC2 Instance Connect



Session Manager

SSH client


EC2 serial console

Instance ID

 i-0852c52b47cbac94a (lab3-ec2)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is vockey.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
 `chmod 400 "vockey.pem"`
4. Connect to your instance using its Public DNS:
 `ec2-18-207-162-182.compute-1.amazonaws.com`

Example:

 `ssh -i "vockey.pem" ec2-user@ec2-18-207-162-182.compute-1.amazonaws.com`


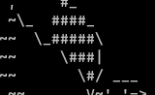
 **Note:** In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

Figure 4. Connect to instance over ssh

```
PS C:\Users\Andi\Documents\Github\fh-mc-cc\exercise\e03> ssh -i "lab3.pem" ec2-user@ec2-54-226-76-194.compute-1.amazonaws.com
The authenticity of host 'ec2-54-226-76-194.compute-1.amazonaws.com (54.226.76.194)' can't be established.
ED25519 key fingerprint is SHA256:kUK7uYm/+UHBGZKPzIWbp1hbVvBhw0/AlIZ+JDLYJGc.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-226-76-194.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
```



```
#_
#####      Amazon Linux 2
### \#####
###  \####|
###   \#/ ---
###    V~!  !->
##### /
### _..
 _/_/_/
_/m/'

AL2 End of Life is 2025-06-30.

A newer version of Amazon Linux is available!

Amazon Linux 2023, GA and supported until 2028-03-15.
https://aws.amazon.com/linux/amazon-linux-2023/
```

```
[ec2-user@ip-172-31-29-67 ~]$ |
```

Figure 5. SSH connection established

1.3. Security Groups

Install Apache Web Server:

```
sudo yum install httpd -y
```

Turn on the Server:

```

Complete!
[ec2-user@ip-172-31-29-67 ~]$ service httpd status
Redirecting to /bin/systemctl status httpd.service
• httpd.service - The Apache HTTP Server
  Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
  Active: inactive (dead)
  Docs: man:httpd.service(8)
[ec2-user@ip-172-31-29-67 ~]$ service httpd start
Redirecting to /bin/systemctl start httpd.service
Failed to start httpd.service: The name org.freedesktop.PolicyKit1 was not provided by any .service files
See system logs and 'systemctl status httpd.service' for details.
[ec2-user@ip-172-31-29-67 ~]$ sudo service httpd start
Redirecting to /bin/systemctl start httpd.service
[ec2-user@ip-172-31-29-67 ~]$ chkconfig httpd on
Note: Forwarding request to 'systemctl enable httpd.service'.
Failed to execute operation: The name org.freedesktop.PolicyKit1 was not provided by any .service files
[ec2-user@ip-172-31-29-67 ~]$ sudo chkconfig httpd on
Note: Forwarding request to 'systemctl enable httpd.service'.
Created symlink from /etc/systemd/system/multi-user.target.wants/httpd.service to /usr/lib/systemd/system/httpd.service.
[ec2-user@ip-172-31-29-67 ~]$ |

```

Figure 6. Turn on Server (Don't forget sudo..)

Add html and try to access it.

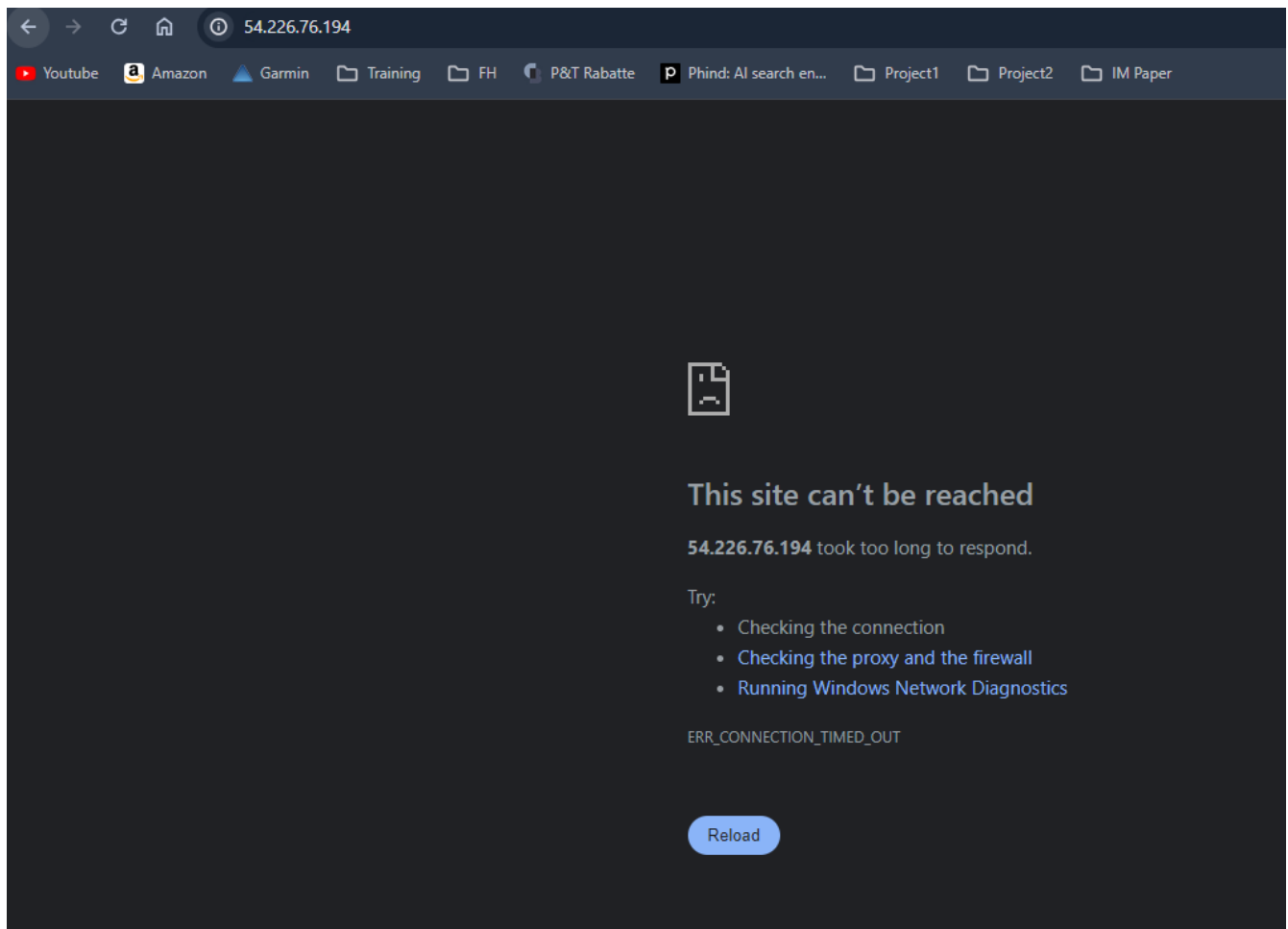


Figure 7. Show html page

Security Groups (2) Info					Refresh	Actions ▼
<input type="text" value="Find resources by attribute or tag"/>						
<input type="checkbox"/>	Name ▼	Security group ID ▼	Security group name ▼	VPC ID ▼		
<input type="checkbox"/>	http	sg-009e740efd6a93830	default	vpc-0492abecb5d55223b 🔗		
<input type="checkbox"/>	ssh 🔗	sg-0ad7b90988d94ad76	launch-wizard-2	vpc-0492abecb5d55223b 🔗		

Figure 8. Add rules for http requests

After adding rules the web server can be accessed.

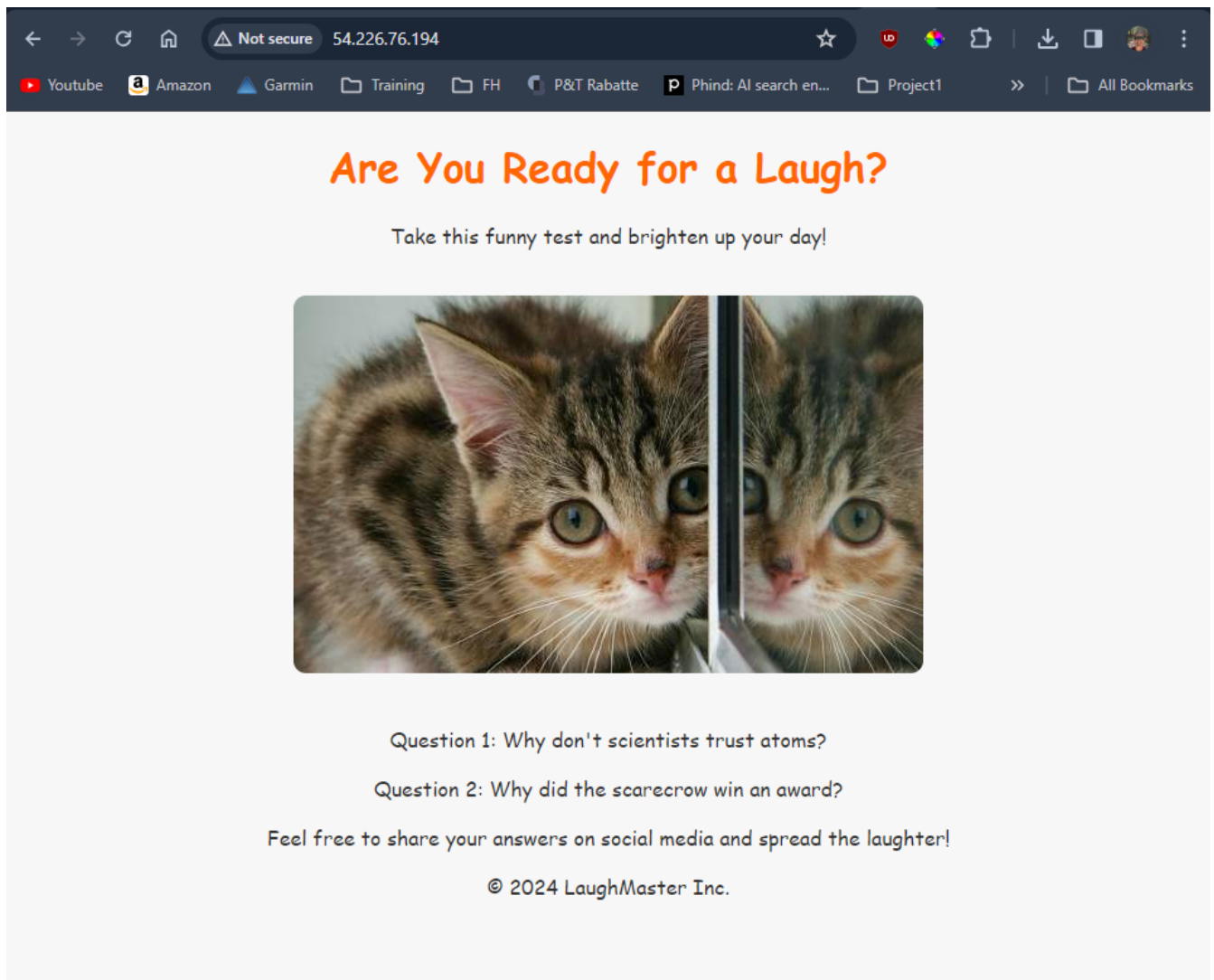


Figure 9. Web page (ChatGPT generated test page)

1.4. Volumes and Snapshots

Create a new volume and attach it to your instance.

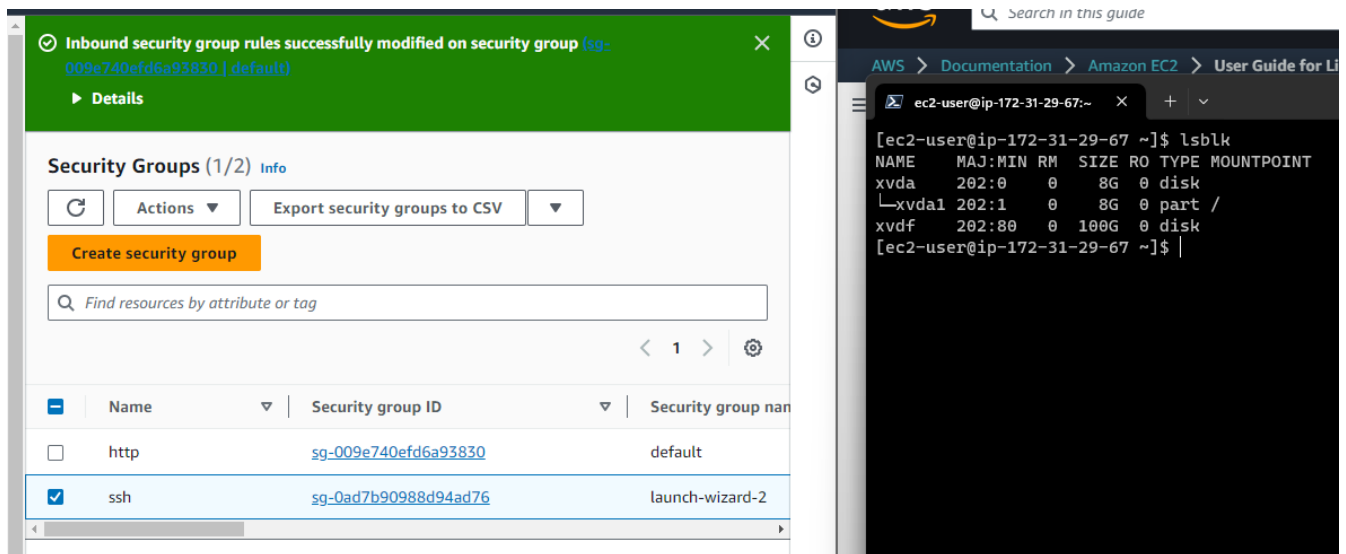


Figure 10. New volume

Check type of the volume.

```
sudo file -s /dev/xvdf
```

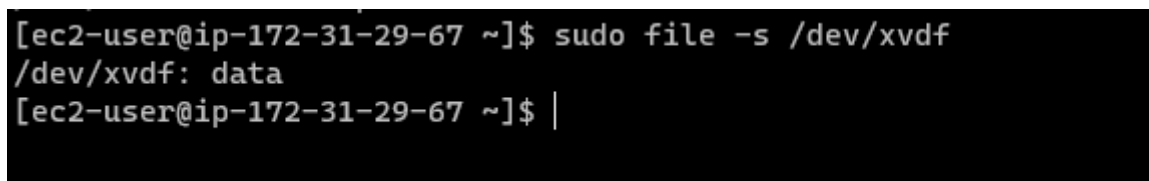


Figure 11. Type


```
[ec2-user@ip-172-31-29-67 ~]$ sudo mkfs -t ext4 /dev/xvdf
mke2fs 1.42.9 (28-Dec-2013)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
6553600 inodes, 26214400 blocks
1310720 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=2174746624
800 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
    4096000, 7962624, 11239424, 20480000, 23887872

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

[ec2-user@ip-172-31-29-67 ~]$ mkdir /fileserver
mkdir: cannot create directory '/fileserver': Permission denied
[ec2-user@ip-172-31-29-67 ~]$ sudo mkdir /fileserver
[ec2-user@ip-172-31-29-67 ~]$ mount /dev/xvdf /fileserver
mount: only root can do that
[ec2-user@ip-172-31-29-67 ~]$ sudo mount /dev/xvdf /fileserver
[ec2-user@ip-172-31-29-67 ~]$ |
```

Figure 12. Create filesystem and mount

```
[ec2-user@ip-172-31-29-67 fileserver]$ sudo touch i-was-here
[ec2-user@ip-172-31-29-67 fileserver]$ ls -l
total 16
-rw-r--r-- 1 root root    0 Jan  8 15:21 i-was-here
drwx----- 2 root root 16384 Jan  8 15:19 lost+found
```

Figure 13. Create file on new partition

```
[ec2-user@ip-172-31-29-67 ~]$ ls -l
total 0
-rw-rw-r-- 1 ec2-user ec2-user 0 Jan  8 15:21 i-was-here
[ec2-user@ip-172-31-29-67 ~]$ cd /
[ec2-user@ip-172-31-29-67 /]$ ls
bin boot dev etc fileserver home lib lib64 local m
[ec2-user@ip-172-31-29-67 /]$ sudo umount /dev/xvdf
[ec2-user@ip-172-31-29-67 /]$ cd /fileserver
[ec2-user@ip-172-31-29-67 fileserver]$ ls
[ec2-user@ip-172-31-29-67 fileserver]$ ls -l
total 0
[ec2-user@ip-172-31-29-67 fileserver]$ |
```

Figure 14. Unmount

Create snapshot [Info](#)

Create a point-in-time snapshot of an EBS volume and use it as a baseline for new volumes or for data backup. You can create snapshots from an individual volume, or you can create multi-volume snapshots from all of the volumes attached to an instance.

Snapshot settings

Resource type [Info](#)



Volume

Create a snapshot from a specific volume.



Instance

Create multi-volume snapshots from an instance.

Volume ID

The volume from which to create the snapshot.

vol-05ff315174425a542



Description

Add a description for your snapshot.

255 characters maximum

Encryption [Info](#)

Not encrypted

Tags [Info](#)

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

No tags associated with the resource.

Add tag

You can add 50 more tags.

Cancel

Create snapshot

Figure 15. Create snapshot of volume

Successfully deleted volume vol-05ff315174425a542.

Volumes (1) [Info](#)

☐

Name	Volume ID	Type	Size	IOPS	Throughput
-	vol-0f7be7717f9fb0b5e	gp2	8 GiB	100	-

Create volume

Figure 16. Volume deleted

Create volume [Info](#)

Create an Amazon EBS volume to attach to any EC2 instance in the same Availability Zone.

Volume settings

Volume type [Info](#)

General Purpose SSD (gp2) ▼

Size (GiB) [Info](#)

100

Min: 1 GiB, Max: 16384 GiB. The value must be an integer.

IOPS [Info](#)

300 / 3000

Baseline of 3 IOPS per GiB with a minimum of 100 IOPS, burstable to 3000 IOPS.

Throughput (MiB/s) [Info](#)

Not applicable

Availability Zone [Info](#)

us-east-1a ▼

Snapshot ID - optional [Info](#)

snap-08621d977f6442e37 ▼

Fast snapshot restore [Info](#)

☐ Not enabled for selected snapshot

Encryption [Info](#)

Use Amazon EBS encryption as an encryption solution for your EBS resources associated with your EC2 instances.

☐ Encrypt this volume

Figure 17. Create from snapshot

```
[ec2-user@ip-172-31-29-67 /]$ sudo mount /dev/xvdf /fileserver
[ec2-user@ip-172-31-29-67 /]$ cd fileserver/
[ec2-user@ip-172-31-29-67 fileserver]$ ls
i-was-here  lost+found
[ec2-user@ip-172-31-29-67 fileserver]$ |
```

Figure 18. Mount and check if file exists

1.5. Load Balancers

1.5.1. Classic Load Balancing

▼ Network settings Info

VPC - required Info

vpc-0492abecb5d55223b (default) ↕
172.31.0.0/16

Subnet Info

subnet-0882fd8526c00f350 ↕
VPC: vpc-0492abecb5d55223b Owner: 677263074526
Availability Zone: us-east-1d IP addresses available: 4091 CIDR: 172.31.80.0/20

↕ Create new subnet

Auto-assign public IP Info

Enable ↕

Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☐ Create security group

☒ Select existing security group

Common security groups Info

Select security groups ↕

launch-wizard-2 sg-0ad7b90988d94ad76 ✕
VPC: vpc-0492abecb5d55223b

↕ Compare security group rules

Security groups that you add or remove here will be added to or removed from all your network interfaces.

► Advanced network configuration

Figure 19. Create instance with different availability zone

Create new load balancer and new instance target group for routing.

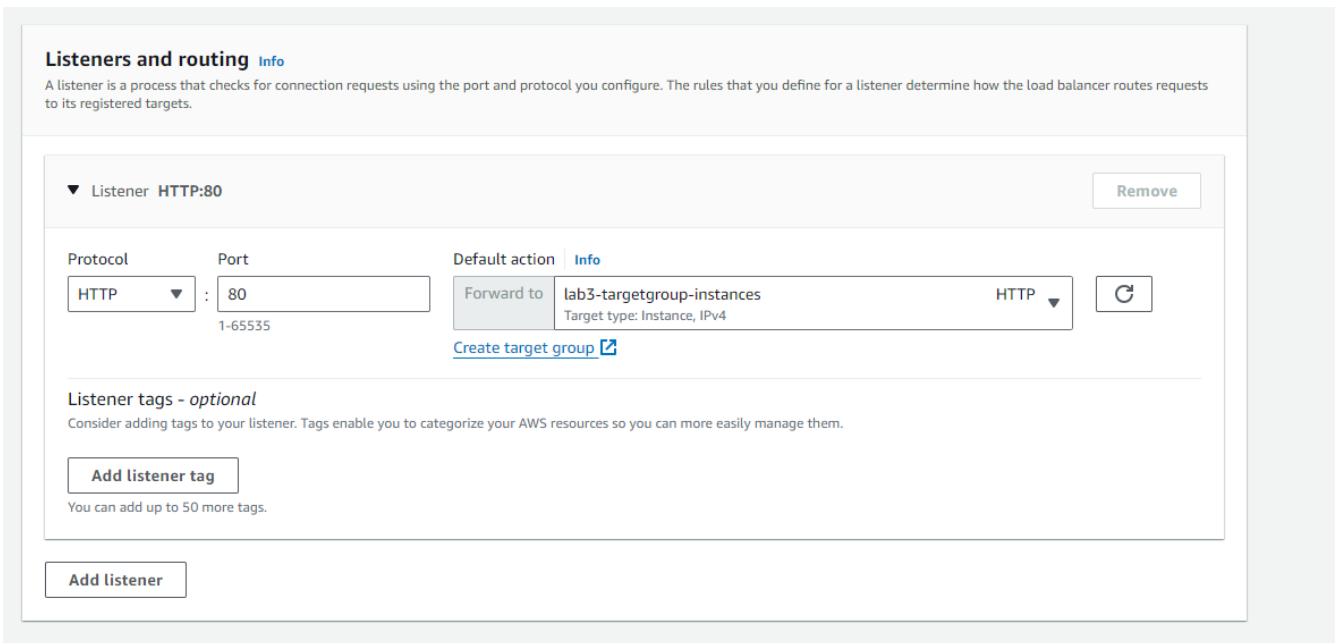


Figure 20. Target group with instances

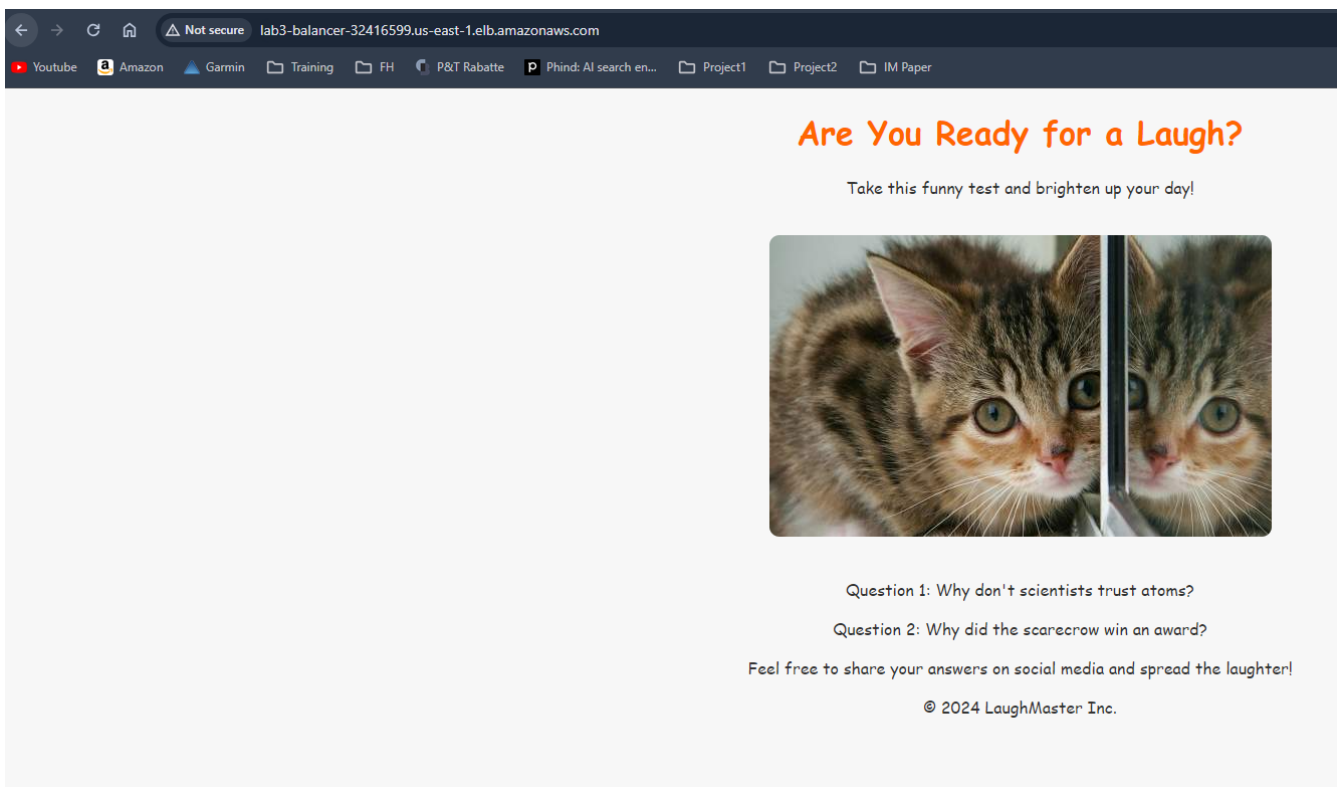


Figure 21. Access through load balancer (Instance 1 reached)

Second instance is missing the http security group that's why its unhealthy.

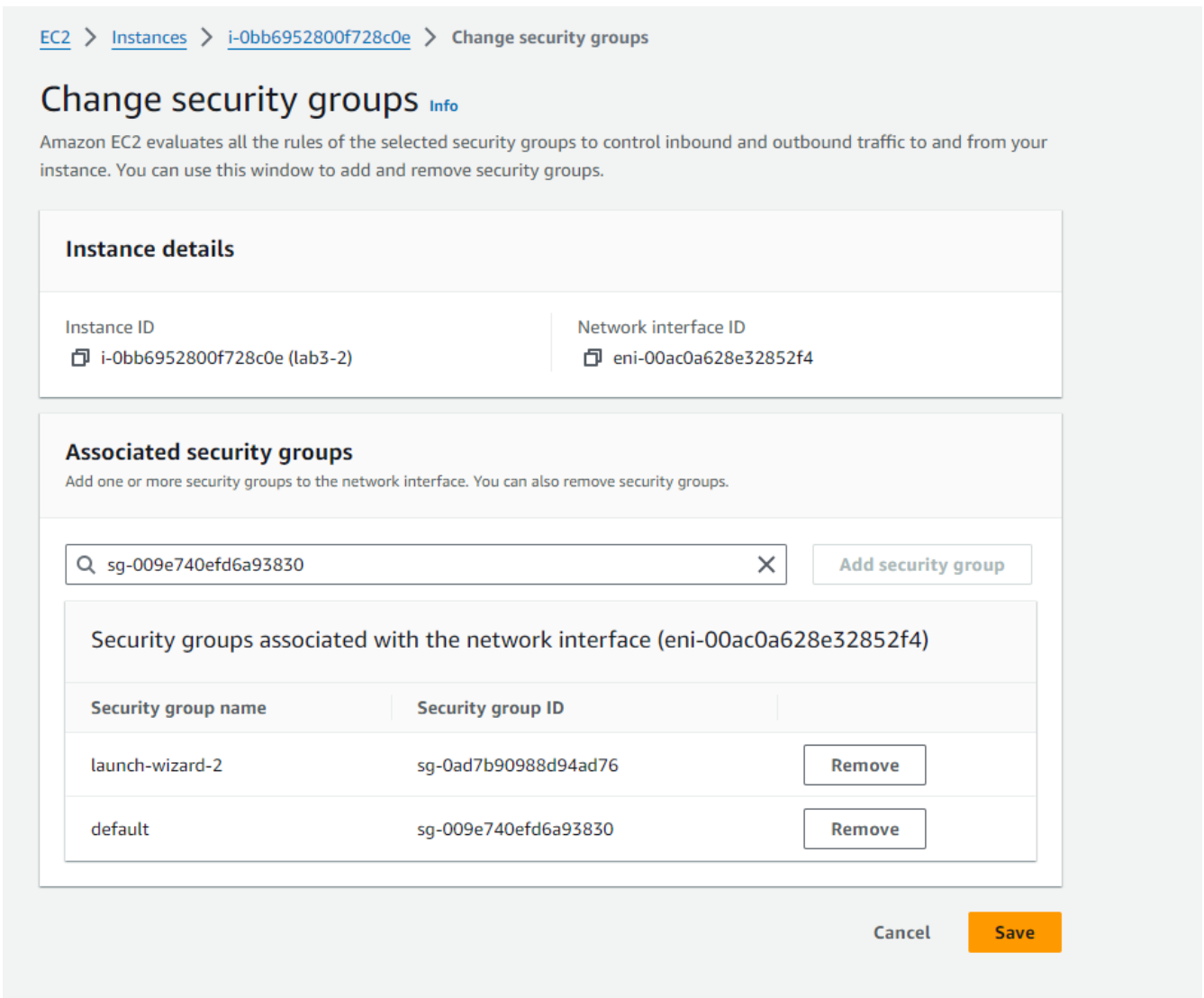


Figure 22. Add second security group

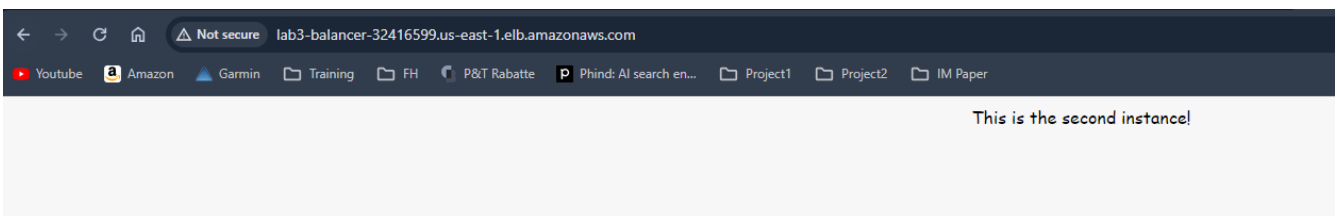


Figure 23. Both instances are healthy

The load balancer is now routing to both instances, that's why the website changes from one instance to the other.

1.5.2. Path-Based Load Balancing

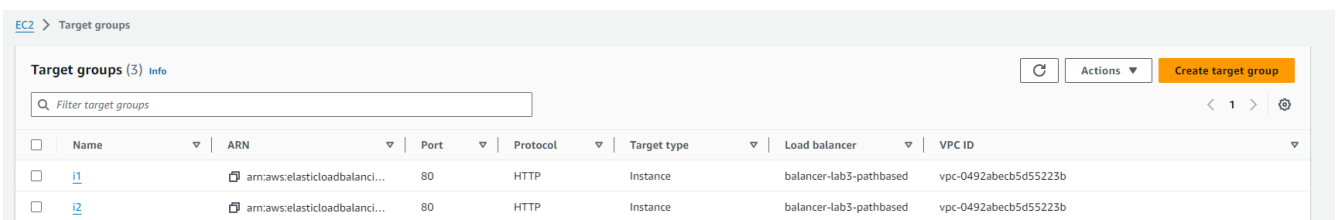


Figure 24. Add two target groups i1 and i2

Add folders i1 and i2 and add html file there.

After that rules for the different paths have to be added.

HTTP:80

info

Details

A listener checks for connection requests using the protocol and port that you configure. The default action and any additional rules that you create determine how the Application Load Balancer routes requests to its registered targets.

Protocol:Port

HTTP:80

Load balancer

[balancer-lab3-pathbased](#)

Default actions

Forward to target group

i1

1

(100%)

Group-level stickiness: Off

Listener ARN

arn:aws:elasticloadbalancing:us-east-1:677263074526:listener/app/balancer-lab3-pathbased/9ed2326bab5f0d7c/2e58a4b4b9d972f6

Rules

Tags

Introducing the new Application Load Balancer listener rules experience

We've redesigned Application Load Balancer listener rules to be easier to use. The changes include:

Find rules quickly by giving them a name tag

Set rule priority with gaps in numerical sequence to accommodate future changes

Or you can [use the old manage rules experience](#).

Listener rules (3)

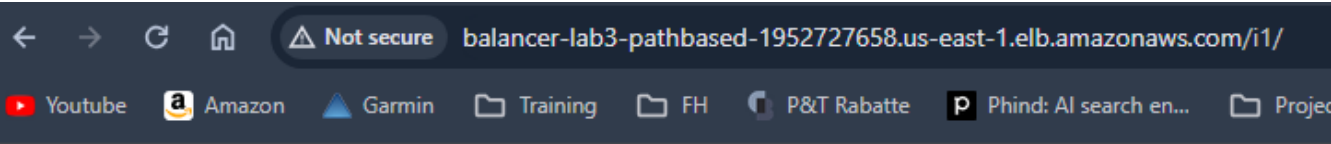
info

Traffic received by the listener is routed according to the default action and any additional rules. Rules are evaluated in priority order from the lowest value to the highest value.

Filter rules

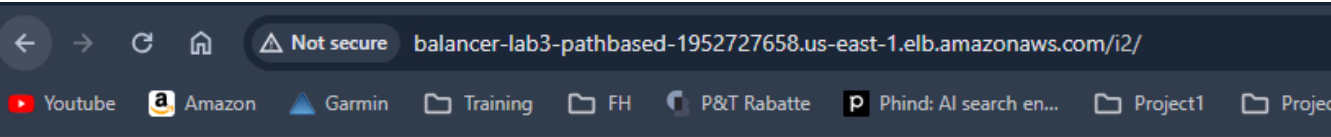
	Name tag	Priority	Conditions (If)	Actions (Then)	ARN	Tags
<input type="checkbox"/>	-	1	Path Pattern is /i2/*	<div>Forward to target group</div> <div><div>i2</div><div>1</div><div>(100%)</div></div> <div>Group-level stickiness: Off</div>	<div>ARN</div>	<div>0 tags</div>
<input type="checkbox"/>	-	2	Path Pattern is /i1/*	<div>Forward to target group</div> <div><div>i1</div><div>1</div><div>(100%)</div></div> <div>Group-level stickiness: Off</div>	<div>ARN</div>	<div>0 tags</div>
<input type="checkbox"/>	Default	Last (default)	If no other rule applies	<div>Forward to target group</div> <div><div>i1</div><div>1</div><div>(100%)</div></div> <div>Group-level stickiness: Off</div>	<div>ARN</div>	<div>0 tags</div>

Figure 25. Set rules



Instance 1

Figure 26. Verify routing (Instance 1)



Instance 2

Figure 27. Verify routing (Instance 2)

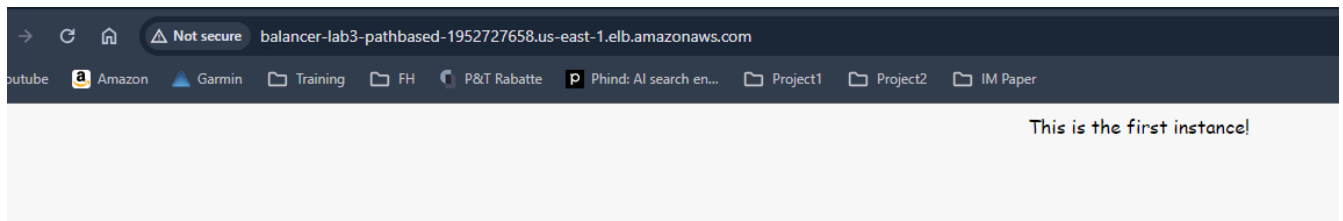


Figure 28. Fallback (Root index.html from instance 1)

2. AWS CLI

2.1. Configuration

Install the aws shell:

```
pip install aws-shell
```

```
PS C:\Users\Andi\Documents\Github\fh-mc-cc\exercise\e03> aws-shell
First run, creating autocomplete index...
Creating doc index in the background. It will be a few minutes before all documentation is available
aws> |
```

Add credentials file from aws portal to ~/.aws/credentials.

Configure aws shell:

```
aws> configure
AWS Access Key ID [*****5CEY]:
AWS Secret Access Key [*****j/7B]:
Default region name [us-east-1]:
Default output format [json]:
aws> |
```

Figure 29. aws-shell configure

```
aws> ec2 describe-instances
{
  "Reservations": [
    {
      "Groups": [],
      "Instances": [
        {
          "AmiLaunchIndex": 0,
          "ImageId": "ami-00b8917ae86a424c9",
          "InstanceId": "i-0c65ff9bcbf527ea3",
          "InstanceType": "t2.micro",
          "KeyName": "ssh-key",
          "LaunchTime": "2024-01-08T17:17:32.000Z",
          "Monitoring": {
            "State": "disabled"
          },
          "Placement": {
            "AvailabilityZone": "us-east-1a",
            "GroupName": ""
          }
        }
      ]
    }
  ]
}
```

Figure 30. Test command after credential configuration

2.2. AWS CLI with EC2

2.2.1. Key-Pairs

Default output format [json]:

```
aws> ec2 describe-key-pairs
{
  "KeyPairs": [
    {
      "KeyPairId": "key-038c4273dd1a6d524",
      "KeyFingerprint": "bb:20:e0:ad:d1:20:17:a0:fa:4d:0d:0d:16:b4:7c:c0:f8:07:1d:9a",
      "KeyName": "ssh-key",
      "KeyType": "rsa",
      "Tags": [],
      "CreateTime": "2024-01-08T13:21:00.674Z"
    },
    {
      "KeyPairId": "key-0ee589ac2e54abf5a",
      "KeyFingerprint": "a4:95:6f:e1:f6:94:e3:35:84:bf:47:a1:49:23:08:b0:90:17:54:24",
      "KeyName": "vockey",
      "KeyType": "rsa",
      "Tags": [],
      "CreateTime": "2024-01-08T17:17:38.778Z"
    }
  ]
}
aws> |
```

2.2.2. Virtual Private Cloud - VPC

```
aws> ec2 describe-vpcs
{
  "Vpcs": [
    {
      "CidrBlock": "172.31.0.0/16",
      "DhcpOptionsId": "dopt-0c61c4fcf2427827b",
      "State": "available",
      "VpcId": "vpc-0492abecb5d55223b",
      "OwnerId": "677263074526",
      "InstanceTenancy": "default",
      "CidrBlockAssociationSet": [
        {
          "AssociationId": "vpc-cidr-assoc-03d668ab340b533b1",
          "CidrBlock": "172.31.0.0/16",
          "CidrBlockState": {
            "State": "associated"
          }
        }
      ],
      "IsDefault": true
    }
  ]
}
aws> |
```

Figure 31. Describe the VPCs

```
aws> ec2 describe-subnets --filters Name=vpc-id,Values=vpc-0492abecb5d55223b
{
  "Subnets": [
    {
      "AvailabilityZone": "us-east-1e",
      "AvailabilityZoneId": "use1-az3",
      "AvailableIpAddressCount": 4091,
      "CidrBlock": "172.31.48.0/20",
      "DefaultForAz": true,
      "MapPublicIpOnLaunch": true,
      "MapCustomerOwnedIpOnLaunch": false,
      "State": "available",
      "SubnetId": "subnet-09fec55be67f9079a",
      "VpcId": "vpc-0492abecb5d55223b",
      "OwnerId": "677263074526",
      "AssignIpv6AddressOnCreation": false,
      "Ipv6CidrBlockAssociationSet": [],
      "SubnetArn": "arn:aws:ec2:us-east-1:677263074526:subnet/subnet-09fec55be67f9079a",
      "EnableDns64": false,
      "Ipv6Native": false,
      "PrivateDnsNameOptionsOnLaunch": {

```

Figure 32. Filter by id

2.2.3. Run instance

```
aws ec2 run-instances --image-id ami-00b8917ae86a424c9 --instance-type t2.micro --key-name ssh-key --security-group-ids sg-0ad7b90988d94ad76 --subnet-id subnet-0882fd8526c00f350 --tags "Key=Name,Value=i1"
```

```

}
instances --image-id ami-0d5eff06f840b45e9 --instance-type t2.micro --key-name ssh-key --security-group-ids sg-0ad7b90988d94ad76 --subnet-id subnet-0882fd8526c00f350
{
  "Groups": [],
  "Instances": [
    {
      "AmiLaunchIndex": 0,
      "ImageId": "ami-0d5eff06f840b45e9",
      "InstanceId": "i-004b3e568e5442c96",
      "InstanceType": "t2.micro",
      "KeyName": "ssh-key",
      "LaunchTime": "2024-01-09T07:25:10.000Z",

```

Figure 33. Using the aws shell

<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
<input type="checkbox"/>	lab3	i-0c65ff9bcbf527ea3	Running	t2.micro	2/2 checks passed	View alarms	us-east-1a	ec2-52-207-187-131.co...
<input type="checkbox"/>	lab3-2	i-0bb6952800f728c0e	Running	t2.micro	2/2 checks passed	View alarms	us-east-1d	ec2-34-207-128-71.co...
<input type="checkbox"/>		i-004b3e568e5442c96	Pending	t2.micro	-	View alarms	us-east-1d	ec2-3-86-208-219.com...

Figure 34. New instance

2.2.4. Querying output

Command for querying all instances by instance id and state:

```
aws ec2 describe-instances --query "Reservations[].Instances[].{InstanceId:InstanceId,State:State.Name}"
```

```
aws> ec2 describe-instances --query "Reservations[].Instances[].{InstanceId:InstanceId,State:State.Name}"
[
  {
    "InstanceId": "i-0c65ff9bcbf527ea3",
    "State": "running"
  },
  {
    "InstanceId": "i-0bb6952800f728c0e",
    "State": "running"
  },
  {
    "InstanceId": "i-004b3e568e5442c96",
    "State": "running"
  }
]
```

Figure 35. Querying all instances

2.3. Working with the CLI

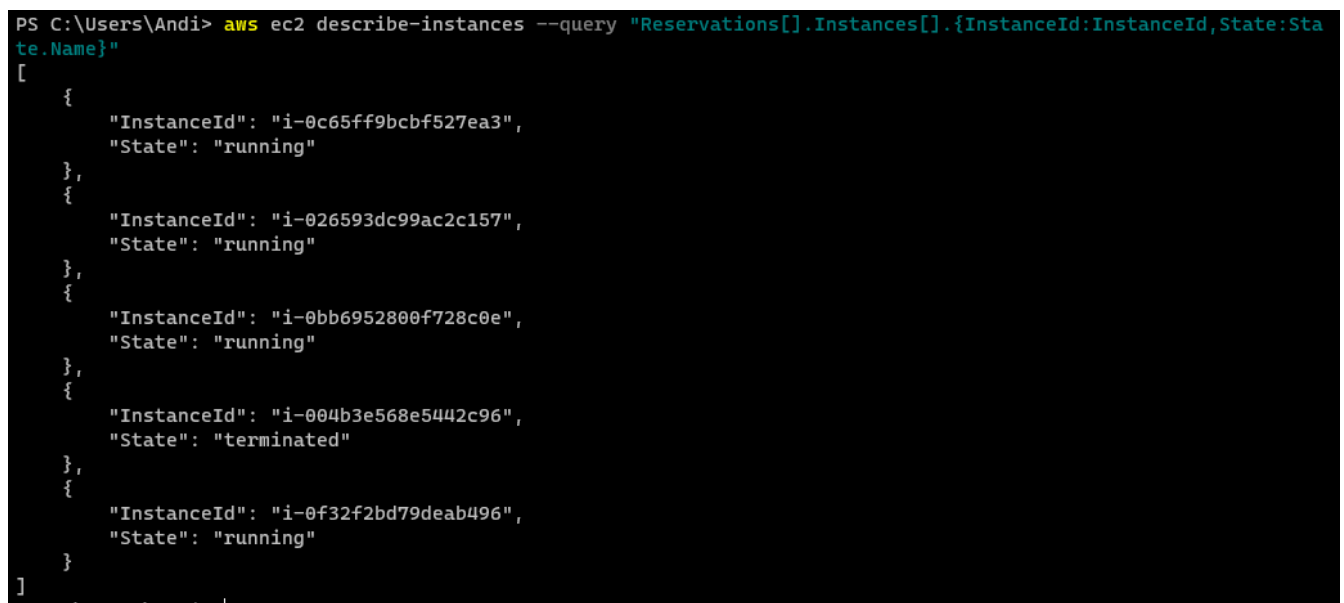
2.3.1. Instances

Create second instance with different availability zone

```
aws ec2 run-instances --image-id ami-00b8917ae86a424c9 --instance-type t2.micro --key-name ssh-key --security-group-ids sg-0ad7b90988d94ad76 --subnet-id subnet-0882fd8526c00f350 --placement AvailabilityZone=us-east-1d
```

Querying all instances

```
aws ec2 describe-instances --query "Reservations[].Instances[].{InstanceId:InstanceId,State:State.Name}"
```



```
PS C:\Users\Andi> aws ec2 describe-instances --query "Reservations[].Instances[].{InstanceId:InstanceId,State:State.Name}"
[
  {
    "InstanceId": "i-0c65ff9bcbf527ea3",
    "State": "running"
  },
  {
    "InstanceId": "i-026593dc99ac2c157",
    "State": "running"
  },
  {
    "InstanceId": "i-0bb6952800f728c0e",
    "State": "running"
  },
  {
    "InstanceId": "i-004b3e568e5442c96",
    "State": "terminated"
  },
  {
    "InstanceId": "i-0f32f2bd79deab496",
    "State": "running"
  }
]
```

Figure 36. 4 Instances and 1 test instance which got terminated.

2.3.2. Security Groups

Create security group

```
aws ec2 create-security-group --group-name httpsg --description "HTTP Security Group"
```

Allow http traffic from my ip only

```
aws ec2 authorize-security-group-ingress --group-name MyHTTPSG --protocol tcp --port 80 --cidr 77.220.105.192/32
```

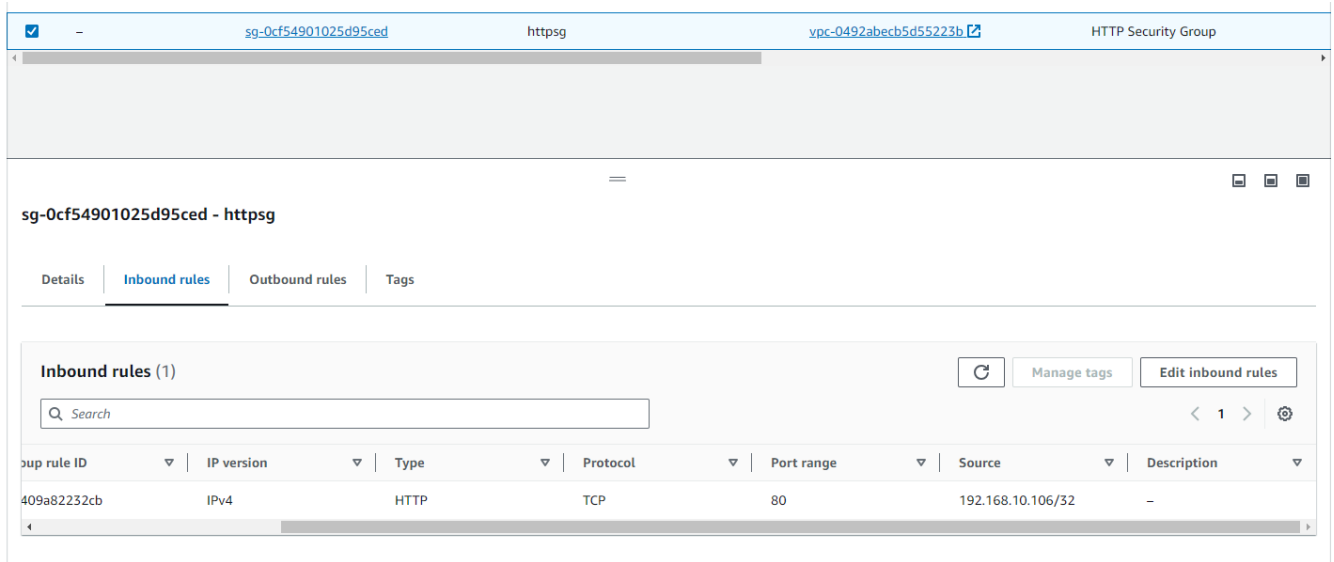
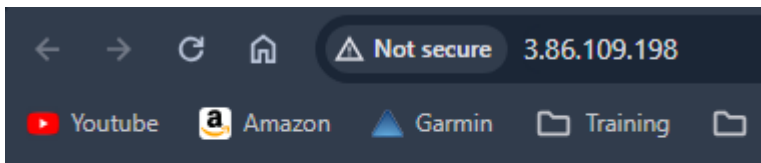


Figure 37. Create Security Group with inbound rule

Change security groups of the instances:

```
aws ec2 modify-instance-attribute --instance-id i-026593dc99ac2c157 --groups sg-0cf54901025d95ced sg-0ad7b90988d94ad76
```

```
aws ec2 modify-instance-attribute --instance-id i-0f32f2bd79deab496 --groups sg-0cf54901025d95ced sg-0ad7b90988d94ad76
```



This is instance 1!

Figure 38. Check if access is possible

2.3.3. Volumes and Snapshots

Create volume

```
aws ec2 create-volume --availability-zone us-east-1d --size 8
```

Created volume

EC2 > Volumes > vol-02b9e17994a2175b5

vol-02b9e17994a2175b5				Refresh Actions Delete Modify
Volume ID vol-02b9e17994a2175b5	Size 8 GiB	Type gp2	Volume status ✔ Okay	
AWS Compute Optimizer finding ⚠ This user is not authorized to call AWS Compute Optimizer. Retry	Volume state ✔ Available	IOPS 100	Throughput -	
Encryption Not encrypted	KMS key ID -	KMS key alias -	KMS key ARN -	
Fast snapshot restored No	Snapshot -	Availability Zone us-east-1a	Created Tue Jan 09 2024 09:13:01 GMT+0100 (Central European Standard Time)	
Multi-Attach enabled No	Attached Instances -	Outposts ARN -		

Attach volume to instance

```
aws ec2 attach-volume --volume-id vol-0d509bb887609d1c8 --instance-id i-026593dc99ac2c157 --device /dev/sdg
```

Attached volume

```
PS C:\Users\Andi\Documents\Github\fh-mc-cc\exercise\03> aws ec2 attach-volume --volume-id vol-0d509bb887609d1c8 --instance-id i-026593dc99ac2c157 --device /dev/sdg
{
  "AttachTime": "2024-01-09T08:18:54.309Z",
  "Device": "/dev/sdg",
  "InstanceId": "i-026593dc99ac2c157",
  "State": "attaching",
  "VolumeId": "vol-0d509bb887609d1c8"
}
```

Create snapshot

```
aws ec2 create-snapshot --volume-id vol-0d509bb887609d1c8 --description "Test snapshot"
```

Detach volume

```
aws ec2 detach-volume --volume-id vol-0d509bb887609d1c8
```

Delete volume

```
aws ec2 delete-volume --volume-id vol-0d509bb887609d1c8
```

Create volume from snapshot

```
aws ec2 create-volume --snapshot-id snap-0007714f04824ed3d --availability-zone us-east-1d
```

Check if file exists

i-was-here still exists

```
[ec2-user@ip-172-31-80-122 ~]$ sudo mount /dev/xvdcg /fileserver
[ec2-user@ip-172-31-80-122 ~]$ cd /fileserver/
[ec2-user@ip-172-31-80-122 fileserver]$ ls
i-was-here  lost+found
[ec2-user@ip-172-31-80-122 fileserver]$ |
```

2.3.4. Load Balancer

Create load balancer

```
aws elbv2 create-load-balancer --name my-load-balancer --subnets subnet-0882fd8526c00f350 subnet-08e8a61e2b32bc655 --security-groups sg-0cf54901025d95ced
```

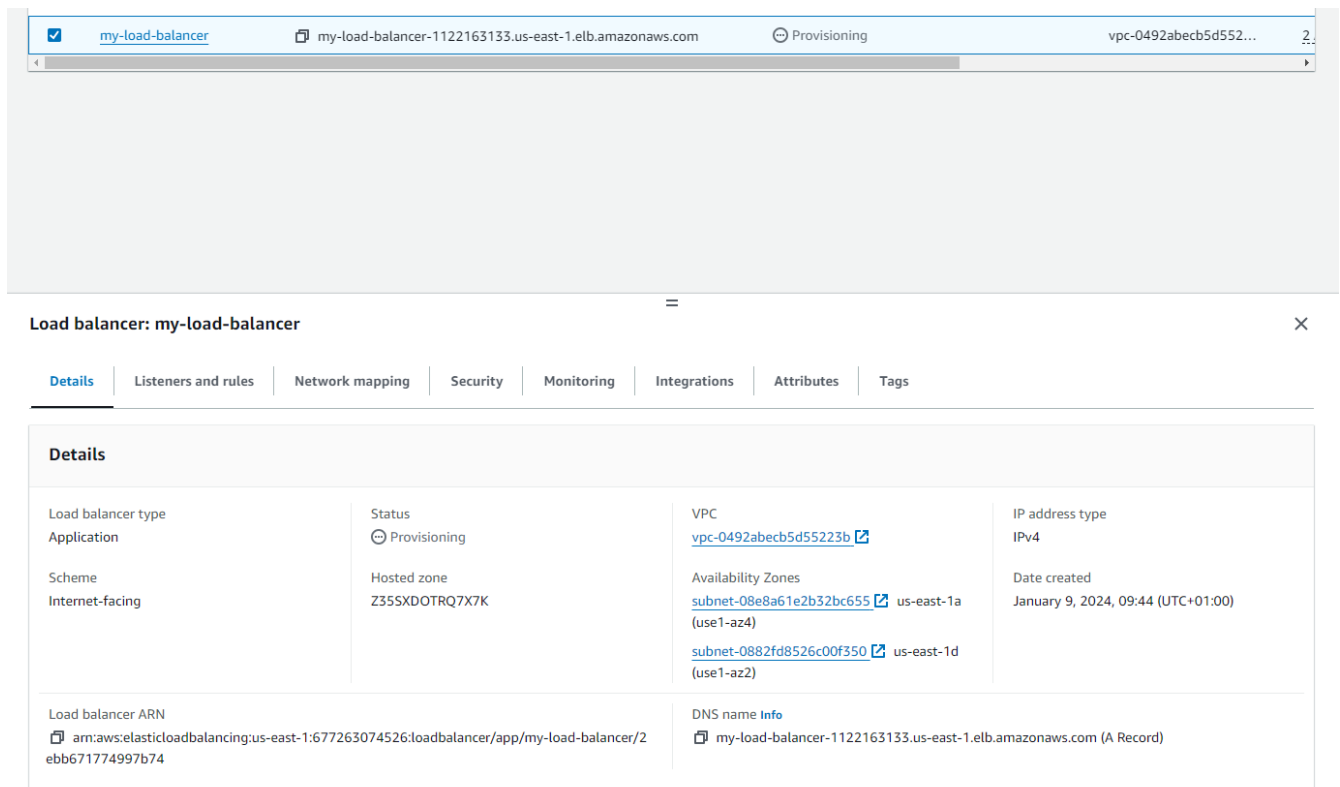


Figure 39. Created load balancer "my-load-balancer"

Create new target group

```
aws elbv2 create-target-group --name MyTargetGroup --protocol HTTP --port 80 --vpc-id vpc-0492abecb5d55223
```

Create new listener for load balancer with target group

```
aws elbv2 create-listener --load-balancer-arn arn:aws:elasticloadbalancing:us-east-1:677263074526:loadbalancer/app/my-load-balancer/2ebb671774997b74 --protocol HTTP --port 80 --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-east-1:677263074526:targetgroup/MyTargetGroup/0d709139aa1c35bb
```

Register instances as targets to the target group

```
aws elbv2 register-targets --target-group-arn arn:aws:elasticloadbalancing:us-east-1:677263074526:targetgroup/MyTargetGroup/0d709139aa1c35bb --targets "Id=i-026593dc99ac2c157" "Id=i-0bb2c09b1f9587e15"
```

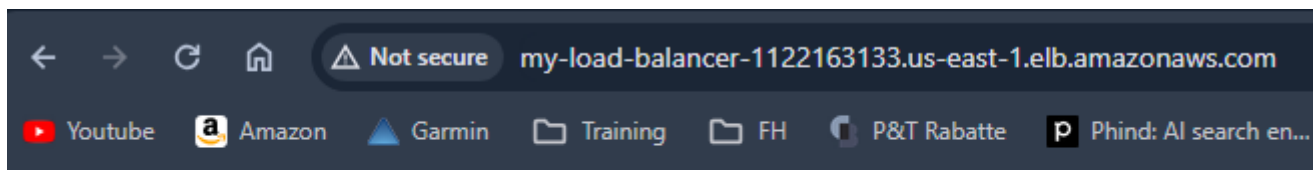
Remove existing security group rule

```
aws ec2 revoke-security-group-ingress --group-id sg-0cf54901025d95ced --protocol tcp --port 80 --cidr 77.220.105.192/32
```

Add new security group rule


```
aws ec2 authorize-security-group-ingress --group-id sg-0cf54901025d95ced --protocol tcp --port 80 --cidr 0.0.0.0/0
```

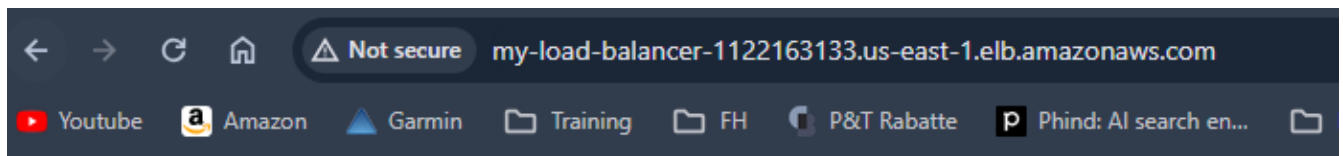
Test load balancer



This is instance 1!

Figure 40. Instance 1

Instance 2



This is instance 2