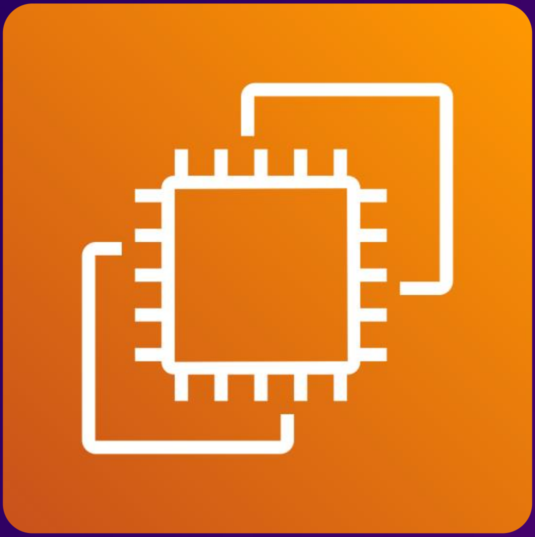
AWS re:Start

LAB

Creating Amazon EC2 Instances



WEEK 8

aws re/start

Overview

Creating Amazon EC2 instances involves using various methods to deploy and manage virtual servers in the cloud. You can use the AWS Management Console for a user-friendly interface to launch and configure instances. Alternatively, you can utilize the AWS CLI for more automated and scriptable deployments. Once the instance is running, EC2 Instance Connect provides a secure and straightforward way to access and manage your server. These tools together offer flexibility and efficiency in setting up and maintaining your cloud infrastructure.

AWS provides multiple ways to launch Amazon Elastic Compute Cloud (Amazon EC2) instance.

In this lab, you use the AWS Management Console to launch an EC2 instance and then use it as a bastion host to launch another EC2 instance, which will be a web server. You use EC2 Instance Connect to securely connect to the bastion host and use the AWS Command Line Interface (AWS CLI) to launch a web server instance.

Topics covered

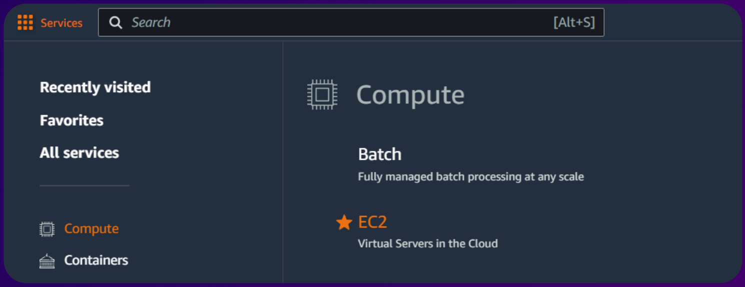
- Launch an EC2 instance by using the AWS Management Console.
- Connect to the EC2 instance by using EC2 Instance Connect.
- Launch an EC2 instance by using the AWS CLI.

Task 1

Launching an EC2 Instance by using the AWS Management Console

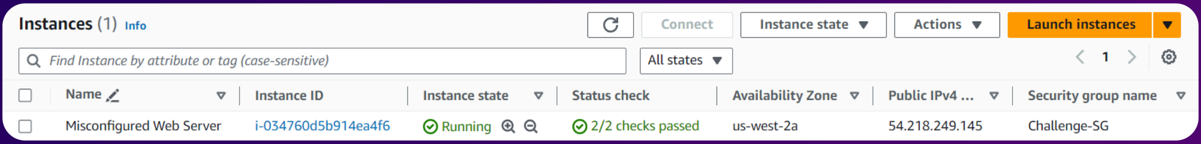
Step 1: Access the AWS Management Console

Open the AWS Management Console, and select EC2.



Step 2: Launch an instance

Navigate to the **Instances** section, and select Launch instances.



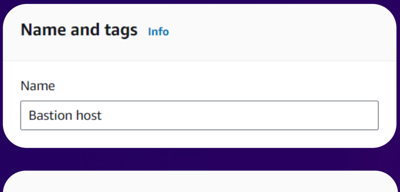
□	Name	Instance ID	Instance state	Status check	Availability Zone	Public IPv4 ...	Security group name
□	Misconfigured Web Server	i-034760d5b914ea4f6	Running	2/2 checks passed	us-west-2a	54.218.249.145	Challenge-SG

Task 1

Launching an EC2 Instance by using the AWS Management Console

Step 3: Set up the instance

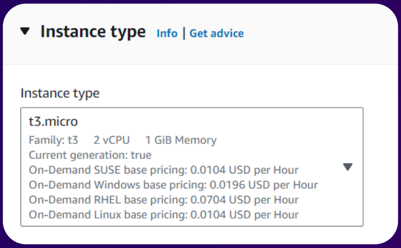
Use the following parameters to configure the instance settings.



Name and tags Info

Name

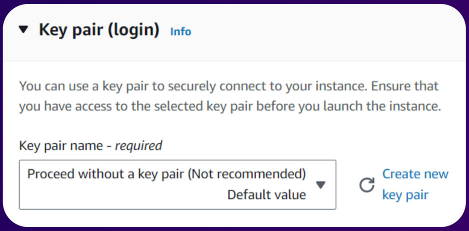
Bastion host



▼ **Instance type** Info | Get advice

Instance type

t3.micro ▼
Family: t3 2 vCPU 1 GiB Memory
Current generation: true
On-Demand SUSE base pricing: 0.0104 USD per Hour
On-Demand Windows base pricing: 0.0196 USD per Hour
On-Demand RHEL base pricing: 0.0704 USD per Hour
On-Demand Linux base pricing: 0.0104 USD per Hour



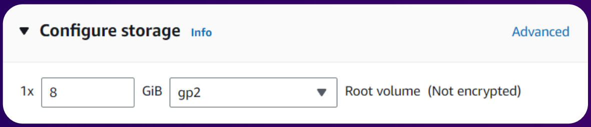
▼ **Key pair (login)** Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

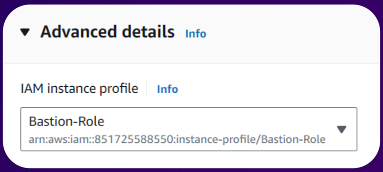
Proceed without a key pair (Not recommended) ▼
Default value

Create new key pair



▼ **Configure storage** Info Advanced

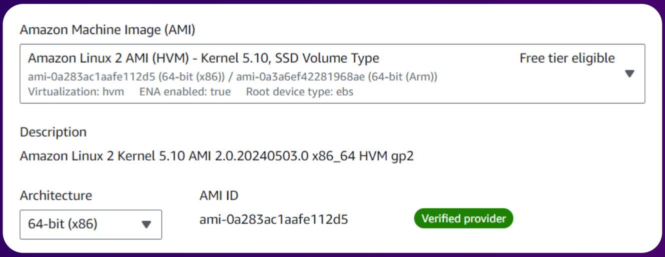
1x 8 GiB gp2 ▼ Root volume (Not encrypted)



▼ **Advanced details** Info

IAM instance profile Info

Bastion-Role ▼
arn:aws:iam::851725588550:instance-profile/Bastion-Role



Amazon Machine Image (AMI)

Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type Free tier eligible ▼
ami-0a283ac1aafe112d5 (64-bit (x86)) / ami-0a3a6ef42281968ae (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Description

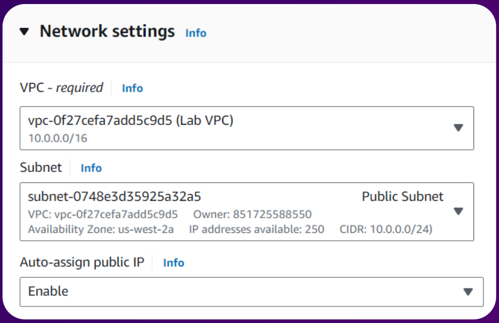
Amazon Linux 2 Kernel 5.10 AMI 2.0.20240503.0 x86_64 HVM gp2

Architecture

64-bit (x86) ▼

AMI ID

ami-0a283ac1aafe112d5 Verified provider



▼ **Network settings** Info

VPC - *required* Info

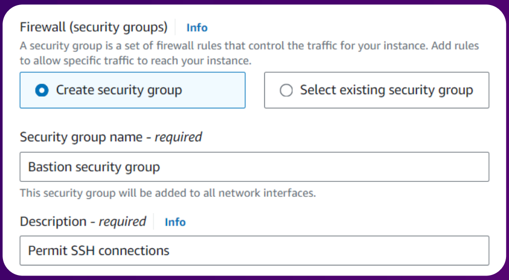
vpc-0f27cefa7add5c9d5 (Lab VPC) ▼
10.0.0.0/16

Subnet Info

subnet-0748e3d35925a32a5 Public Subnet ▼
VPC: vpc-0f27cefa7add5c9d5 Owner: 851725588550
Availability Zone: us-west-2a IP addresses available: 250 CIDR: 10.0.0.0/24)

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

Security group name - *required*

Bastion security group

This security group will be added to all network interfaces.

Description - *required* Info

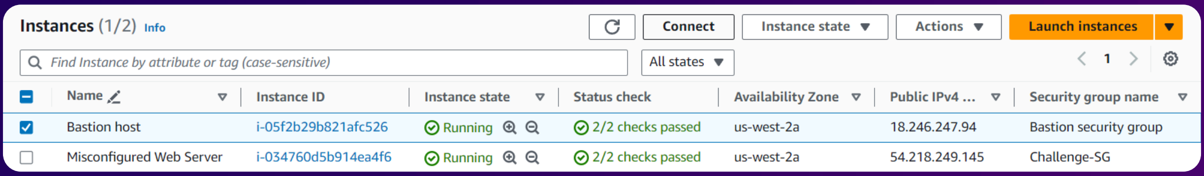
Permit SSH connections

Task 2

Logging in to the bastion host

Step 1: Connect to the Bastion host

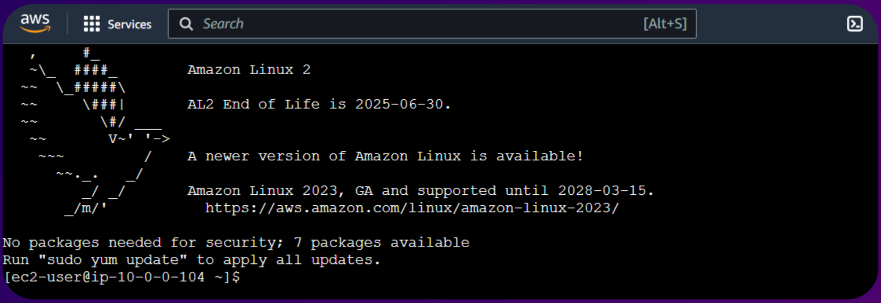
On the EC2 Management Console, in the **Instances** section, choose the **Bastion host** instance, and select Connect.



☐	Name	Instance ID	Instance state	Status check	Availability Zone	Public IPv4 ...	Security group name
☑	Bastion host	i-05f2b29b821afc526	Running	2/2 checks passed	us-west-2a	18.246.247.94	Bastion security group
☐	Misconfigured Web Server	i-034760d5b914ea4f6	Running	2/2 checks passed	us-west-2a	54.218.249.145	Challenge-SG

Step 2: EC2 Instance Connect

Connect to the Bastion host using EC2 Instance Connect. Now you can use the AWS CLI to call AWS services.



```
   ,     #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2025-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

No packages needed for security; 7 packages available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-10-0-0-104 ~]$
```

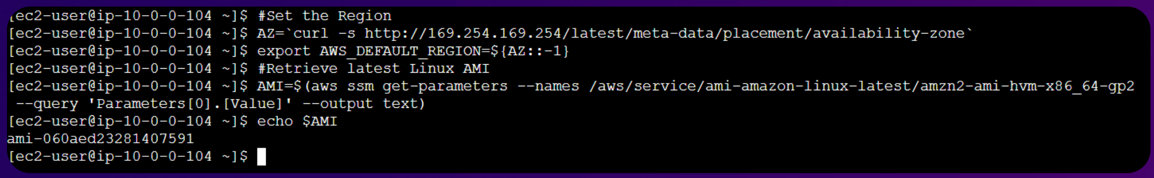
aws re/start

Task 3

Launching an EC2 instance using the AWS CLI

Step 1: Retrieve the AMI to use

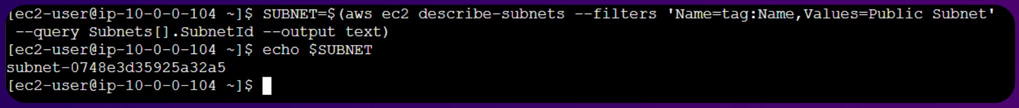
Run the following script in your EC2 Instance Connect session to retrieve the Amazon Linux 2 AMI ID to use.



```
[ec2-user@ip-10-0-0-104 ~]$ #Set the Region
[ec2-user@ip-10-0-0-104 ~]$ AZ=`curl -s http://169.254.169.254/latest/meta-data/placement/availability-zone`
[ec2-user@ip-10-0-0-104 ~]$ export AWS_DEFAULT_REGION=${AZ::-1}
[ec2-user@ip-10-0-0-104 ~]$ #Retrieve latest Linux AMI
[ec2-user@ip-10-0-0-104 ~]$ AMI=$(aws ssm get-parameters --names /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2
 --query 'Parameters[0].[Value]' --output text)
[ec2-user@ip-10-0-0-104 ~]$ echo $AMI
ami-060aed23281407591
[ec2-user@ip-10-0-0-104 ~]$
```

Step 2: Retrieve the subnet to use

To retrieve the subnet ID for the public subnet, run the following command.



```
[ec2-user@ip-10-0-0-104 ~]$ SUBNET=$(aws ec2 describe-subnets --filters 'Name=tag:Name,Values=Public Subnet'
 --query Subnets[].SubnetId --output text)
[ec2-user@ip-10-0-0-104 ~]$ echo $SUBNET
subnet-0748e3d35925a32a5
[ec2-user@ip-10-0-0-104 ~]$
```

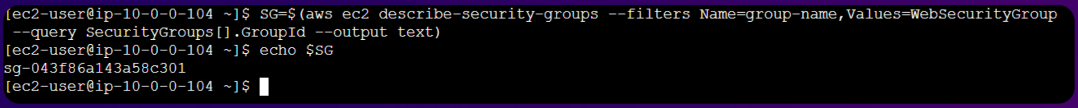
aws re/start

Task 3

Launching an EC2 instance using the AWS CLI

Step 3: Retrieve the security group to use

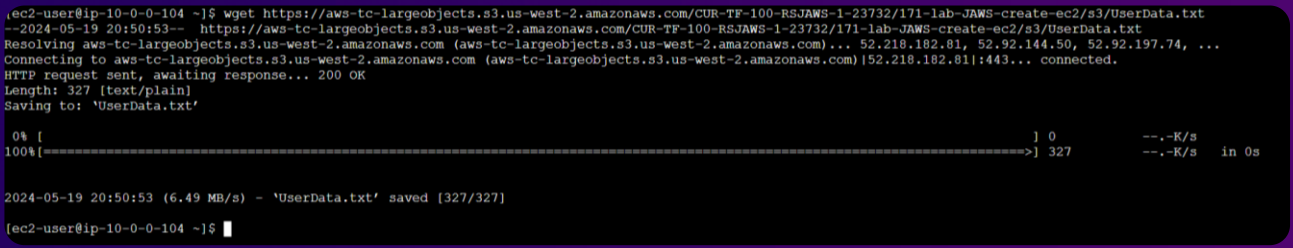
Run the following command to retrieve the security group ID of the web security group.



```
[ec2-user@ip-10-0-0-104 ~]$ SG=$(aws ec2 describe-security-groups --filters Name=group-name,Values=WebSecurityGroup
 --query SecurityGroups[].GroupId --output text)
[ec2-user@ip-10-0-0-104 ~]$ echo $SG
sg-043f86a143a58c301
[ec2-user@ip-10-0-0-104 ~]$ █
```

Step 4: Download a user data script

To download the user data script, run the following command.



```
[ec2-user@ip-10-0-0-104 ~]$ wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-100-RSJAWS-1-23732/171-lab-JAWS-create-ec2/s3/UserData.txt
--2024-05-19 20:50:53--  https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-100-RSJAWS-1-23732/171-lab-JAWS-create-ec2/s3/UserData.txt
Resolving aws-tc-largeobjects.s3.us-west-2.amazonaws.com (aws-tc-largeobjects.s3.us-west-2.amazonaws.com)... 52.218.182.81, 52.92.144.50, 52.92.197.74, ...
Connecting to aws-tc-largeobjects.s3.us-west-2.amazonaws.com (aws-tc-largeobjects.s3.us-west-2.amazonaws.com)|52.218.182.81|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 327 [text/plain]
Saving to: 'UserData.txt'

 0% [                                                                                                                             ] 0             --.-K/s
100%[============================================================================================================================>] 327           --.-K/s    in 0s


2024-05-19 20:50:53 (6.49 MB/s) - 'UserData.txt' saved [327/327]

[ec2-user@ip-10-0-0-104 ~]$ █
```

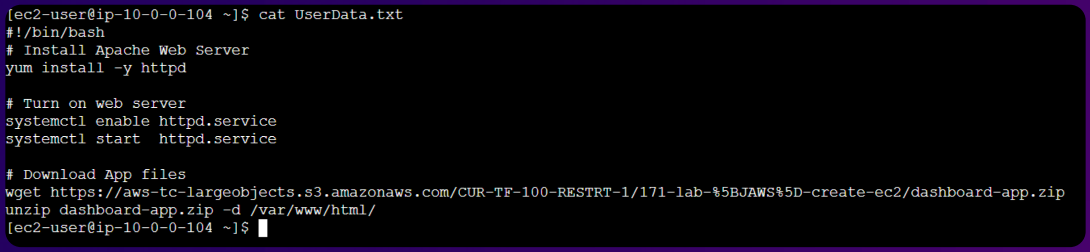
aws re/start

Task 3

Launching an EC2 instance using the AWS CLI

Step 5: Review the user data script

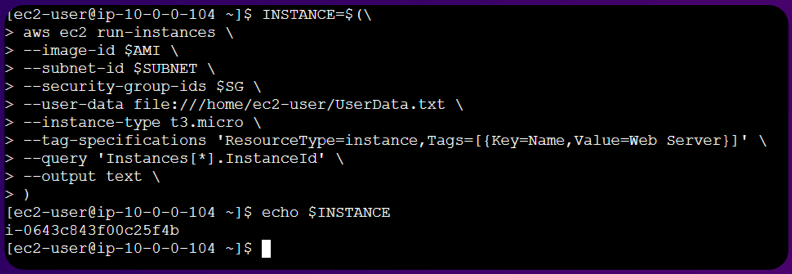
Review the contents of the user data script. The script installs a web server, downloads a .zip file containing the web application, and installs the web application.



```
[ec2-user@ip-10-0-0-104 ~]$ cat UserData.txt
#!/bin/bash
# Install Apache Web Server
yum install -y httpd

# Turn on web server
systemctl enable httpd.service
systemctl start  httpd.service

# Download App files
wget https://aws-tc-largeobjects.s3.amazonaws.com/CUR-TF-100-RESTRT-1/171-lab-%5BJAWS%5D-create-ec2/dashboard-app.zip
unzip dashboard-app.zip -d /var/www/html/
[ec2-user@ip-10-0-0-104 ~]$ █
```

Step 6: Launch the instance

You now have all the necessary information required to launch the web server instance. Run the following aws ec2 run-instances command to launch the instance.



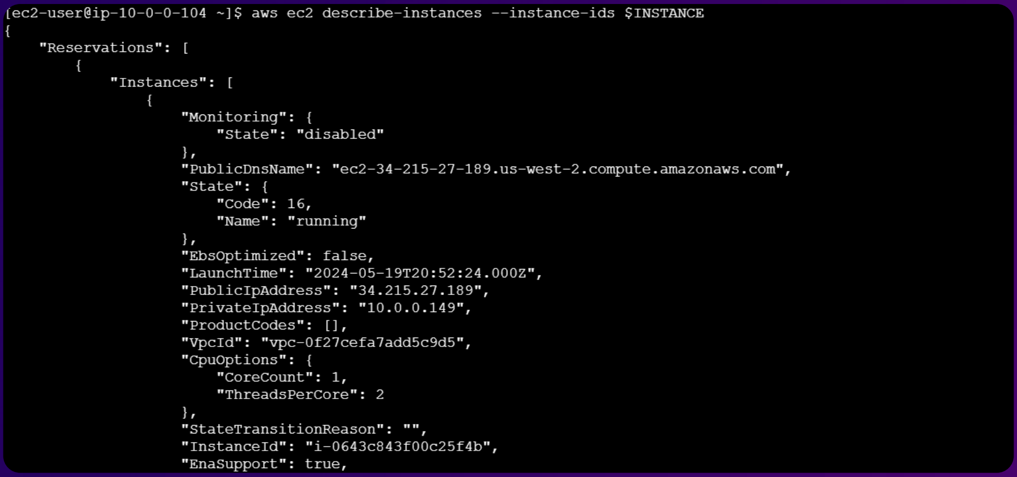
```
[ec2-user@ip-10-0-0-104 ~]$ INSTANCE=$(\
> aws ec2 run-instances \
> --image-id $AMI \
> --subnet-id $SUBNET \
> --security-group-ids $SG \
> --user-data file:///home/ec2-user/UserData.txt \
> --instance-type t3.micro \
> --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=Web Server}]' \
> --query 'Instances[*].InstanceId' \
> --output text \
> )
[ec2-user@ip-10-0-0-104 ~]$ echo $INSTANCE
i-0643c843f00c25f4b
[ec2-user@ip-10-0-0-104 ~]$ █
```

Task 3

Launching an EC2 instance using the AWS CLI

Step 7: Display instance information

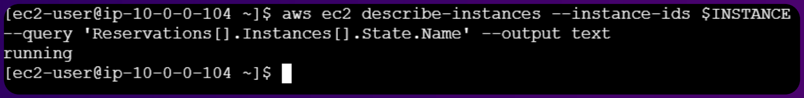
Run the following aws ec2 describe-instances command to display all information related to the instance in JSON format.



```
[ec2-user@ip-10-0-0-104 ~]$ aws ec2 describe-instances --instance-ids $INSTANCE
{
    "Reservations": [
        {
            "Instances": [
                {
                    "Monitoring": {
                        "State": "disabled"
                    },
                    "PublicDnsName": "ec2-34-215-27-189.us-west-2.compute.amazonaws.com",
                    "State": {
                        "Code": 16,
                        "Name": "running"
                    },
                    "EbsOptimized": false,
                    "LaunchTime": "2024-05-19T20:52:24.000Z",
                    "PublicIpAddress": "34.215.27.189",
                    "PrivateIpAddress": "10.0.0.149",
                    "ProductCodes": [],
                    "VpcId": "vpc-0f27cefa7add5c9d5",
                    "CpuOptions": {
                        "CoreCount": 1,
                        "ThreadsPerCore": 2
                    },
                    "StateTransitionReason": "",
                    "InstanceId": "i-0643c843f00c25f4b",
                    "EnaSupport": true,
```

Step 8: Display instance state

Run the previous command using the query parameter to display only the name of the instance state.



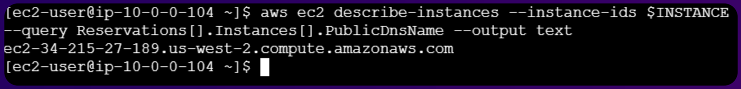
```
[ec2-user@ip-10-0-0-104 ~]$ aws ec2 describe-instances --instance-ids $INSTANCE
--query 'Reservations[].Instances[].State.Name' --output text
running
[ec2-user@ip-10-0-0-104 ~]$ █
```

Task 3

Launching an EC2 instance using the AWS CLI

Step 9: Retrieve the public IPv4 DNS name

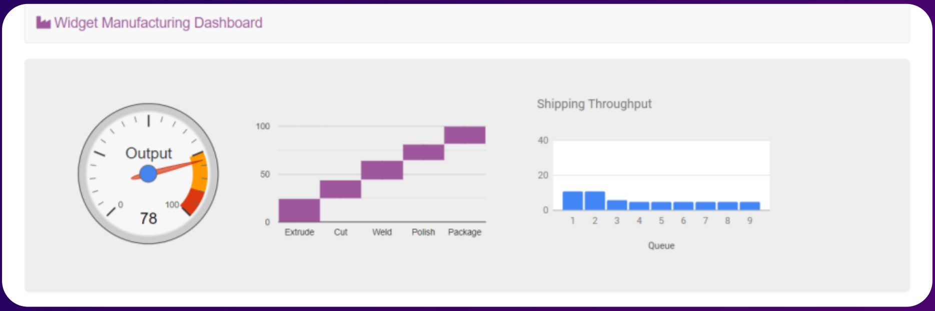
Run the following aws ec2 describe-instances command to return the public IPv4 DNS name of the instance.

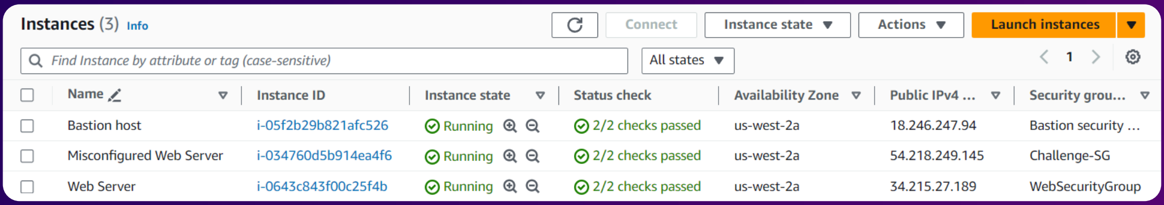


```
[ec2-user@ip-10-0-0-104 ~]$ aws ec2 describe-instances --instance-ids $INSTANCE
--query Reservations[].Instances[].PublicDnsName --output text
ec2-34-215-27-189.us-west-2.compute.amazonaws.com
[ec2-user@ip-10-0-0-104 ~]$ 
```

Step 10: Test the web server

The web server page is displayed, which demonstrates that the web server was successfully launched and configured. You can also see the instance on the Amazon EC2 management console.





□	Name	Instance ID	Instance state	Status check	Availability Zone	Public IPv4 ...	Security grou...
□	Bastion host	i-05f2b29b821afc526	Running	2/2 checks passed	us-west-2a	18.246.247.94	Bastion security ...
□	Misconfigured Web Server	i-034760d5b914ea4f6	Running	2/2 checks passed	us-west-2a	54.218.249.145	Challenge-SG
□	Web Server	i-0643c843f00c25f4b	Running	2/2 checks passed	us-west-2a	34.215.27.189	WebSecurityGroup

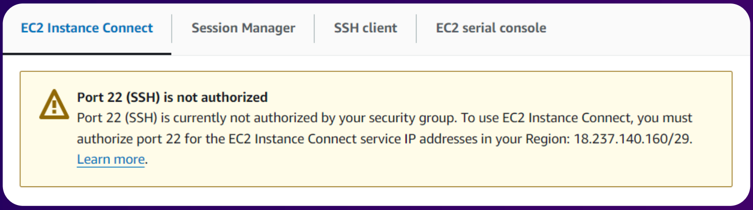
aws re/start

Task 4

Optional challenge 1: Connect to an EC2 instance

Step 1: Try to connect to the instance

Try to connect to the **Misconfigured Web Server** instance using EC2 Instance Connect. A Port 22 (SSH) is not authorized message appears.

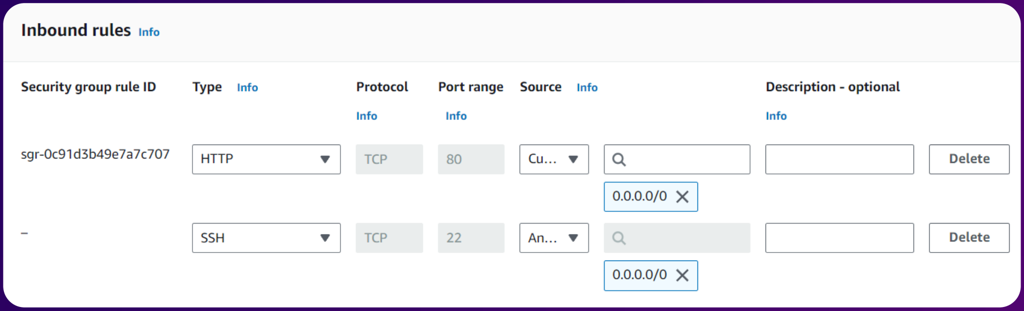


EC2 Instance Connect | Session Manager | SSH client | EC2 serial console

⚠ **Port 22 (SSH) is not authorized**
Port 22 (SSH) is currently not authorized by your security group. To use EC2 Instance Connect, you must authorize port 22 for the EC2 Instance Connect service IP addresses in your Region: 18.237.140.160/29. Learn more.

Step 2: Add a security group rule

Add an inbound rule to the associated Security Group that allows SSH traffic into the **Misconfigured Web Server** instance.



Inbound rules Info

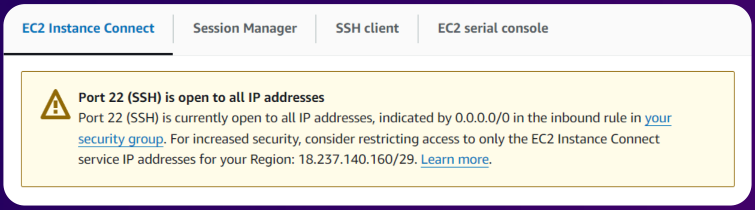
Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-0c91d3b49e7a7c707	HTTP	TCP	80	Cu...	0.0.0.0/0 ✕		Delete
–	SSH	TCP	22	An...	0.0.0.0/0 ✕		Delete

Task 4

Optional challenge 1: Connect to an EC2 instance

Step 3: EC2 Instance Connect

Try to connect to the **Misconfigured Web Server** instance. The EC2 Instance Connect tab now shows a Port 22 (SSH) is open to all IP addresses message.

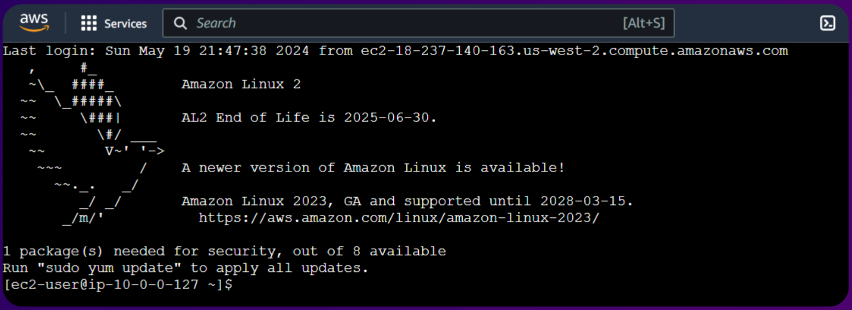


Port 22 (SSH) is open to all IP addresses

Port 22 (SSH) is currently open to all IP addresses, indicated by 0.0.0.0/0 in the inbound rule in your security group. For increased security, consider restricting access to only the EC2 Instance Connect service IP addresses for your Region: 18.237.140.160/29. Learn more.

Step 4: Connect to the instance

Establish a connection to the **Misconfigured Web Server** instance using EC2 Instance Connect.



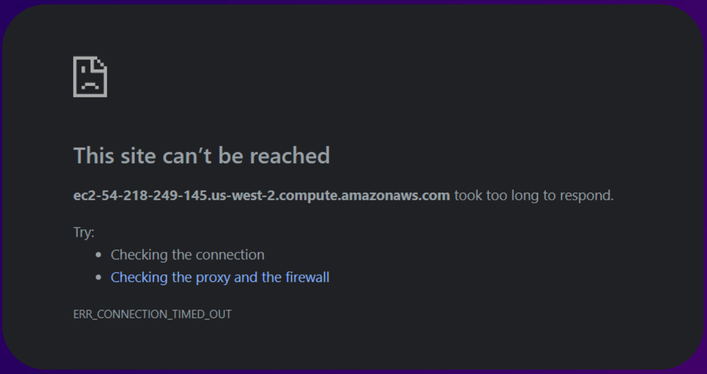
```
Last login: Sun May 19 21:47:38 2024 from ec2-18-237-140-163.us-west-2.compute.amazonaws.com
   ,     #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2025-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

1 package(s) needed for security, out of 8 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-10-0-0-127 ~]$
```

Task 5

Optional challenge 2: Fix the web server installation

Step 1: Visit the web server page

Retrieve the public IPv4 DNS name of the **Misconfigured Web Server** instance, and visit the web server page. It doesn’t work.



This site can’t be reached

ec2-54-218-249-145.us-west-2.compute.amazonaws.com took too long to respond.

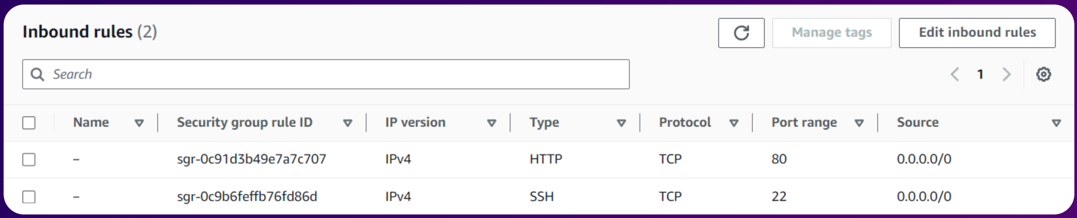
Try:

- Checking the connection
- Checking the proxy and the firewall

ERR_CONNECTION_TIMED_OUT

Step 2: Review the Security Group

The associated Security Group does allow incoming HTTP traffic in port 80.



Inbound rules (2)

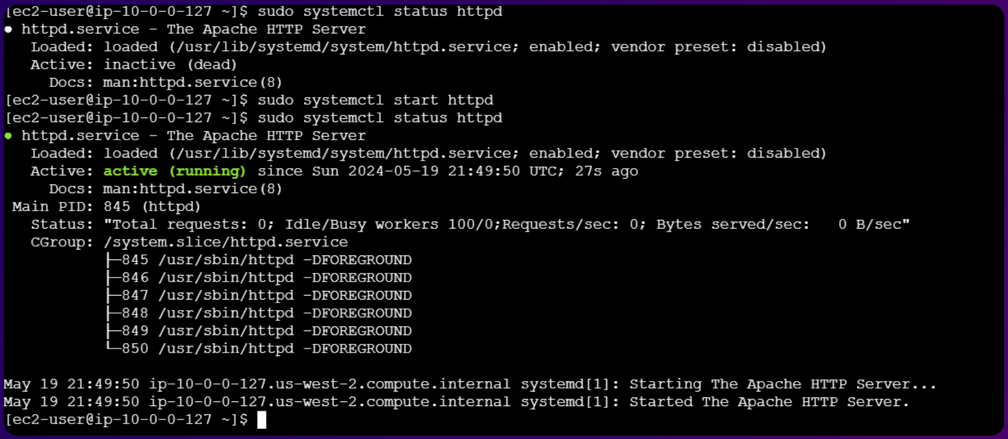
□	Name	Security group rule ID	IP version	Type	Protocol	Port range	Source
□	–	sgr-0c91d3b49e7a7c707	IPv4	HTTP	TCP	80	0.0.0.0/0
□	–	sgr-0c9b6feffb76fd86d	IPv4	SSH	TCP	22	0.0.0.0/0

Task 5

Optional challenge 2: Fix the web server installation

Step 3: Start the httpd service

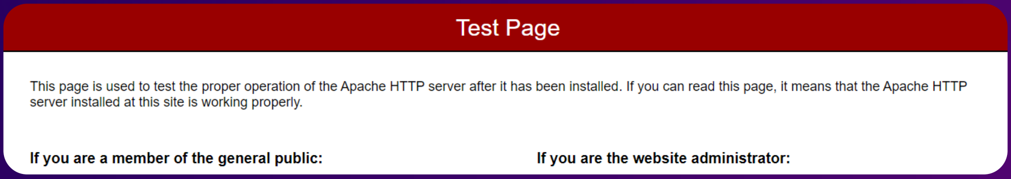
Review the status of the httpd service, and start the httpd service if necessary.



```
[ec2-user@ip-10-0-0-127 ~]$ sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
   Active: inactive (dead)
     Docs: man:httpd.service(8)
[ec2-user@ip-10-0-0-127 ~]$ sudo systemctl start httpd
[ec2-user@ip-10-0-0-127 ~]$ sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
   Active: active (running) since Sun 2024-05-19 21:49:50 UTC; 27s ago
     Docs: man:httpd.service(8)
 Main PID: 845 (httpd)
   Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes served/sec:   0 B/sec"
   CGroup: /system.slice/httpd.service
           ├─845 /usr/sbin/httpd -DFOREGROUND
           ├─846 /usr/sbin/httpd -DFOREGROUND
           ├─847 /usr/sbin/httpd -DFOREGROUND
           ├─848 /usr/sbin/httpd -DFOREGROUND
           ├─849 /usr/sbin/httpd -DFOREGROUND
           └─850 /usr/sbin/httpd -DFOREGROUND

May 19 21:49:50 ip-10-0-0-127.us-west-2.compute.internal systemd[1]: Starting The Apache HTTP Server...
May 19 21:49:50 ip-10-0-0-127.us-west-2.compute.internal systemd[1]: Started The Apache HTTP Server.
[ec2-user@ip-10-0-0-127 ~]$ 
```

Step 4: Validate the solution

Visit the web server page to validate the solution.



Test Page

This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you can read this page, it means that the Apache HTTP server installed at this site is working properly.

If you are a member of the general public:

If you are the website administrator:

Conclusions

Launching EC2 instances

Launching EC2 instances provides scalable and resizable compute capacity in the cloud, enabling flexible and cost-effective infrastructure management.

The AWS Management Console

The AWS Management Console offers a user-friendly interface for launching EC2 instances, simplifying the process for users of all skill levels.

The AWS Command Line Interface

The AWS Command Line Interface allows for automated and scriptable instance launches, enhancing efficiency and consistency in deployment processes.

EC2 Instance Connect

EC2 Instance Connect provides secure and easy access to instances without needing traditional SSH keys, improving security and convenience.

The aws ec2 commands

The aws ec2 commands enable powerful and flexible instance management, allowing detailed control over instance creation, configuration, and monitoring through the command line.

aws re/start

aws re/start

Cristhian Becerra

cristhian-becerra-espinoza

+51 951 634 354

cristhianbecerra99@gmail.com

Lima, Peru

aws re/start