

1. Describe Load Balancing and its significance in Cloud Environment.

Load balancing is a crucial concept in cloud computing that involves distributing incoming network traffic or workload across multiple servers to ensure optimal resource utilization, maximize throughput, minimize response time, and avoid overload on any individual server. It plays a significant role in improving the performance, scalability, and reliability of cloud-based applications. Here are some key points about load balancing in a cloud environment:

1. **Enhanced Performance:** By evenly distributing incoming requests across multiple servers, load balancing ensures that no single server is overwhelmed, thereby improving the overall performance and responsiveness of the application.
2. **Redundancy and High Availability:** Load balancers can detect when a server is down or experiencing issues and redirect traffic to healthy servers, ensuring high availability and minimizing downtime.
3. **Scalability:** Load balancing allows for easy scalability by adding or removing servers based on demand. This helps in handling sudden spikes in traffic without affecting the performance of the application.
4. **Fault Tolerance:** Load balancers can route traffic away from failed servers, ensuring that the application remains operational even in the face of hardware or software failures.
5. **Geographical Load Balancing:** In a cloud environment with multiple data centers across different locations, load balancing can distribute traffic based on user location, ensuring better performance and reducing latency.
6. **SSL Offloading and Security:** Load balancers can offload SSL encryption and decryption tasks from servers, improving performance and ensuring secure communication between clients and servers.

Overall, load balancing is essential in a cloud environment to optimize resource utilization, improve performance, ensure high availability, and enhance the overall user experience of cloud-based applications.

2. List the Load Balancing Service available in AWS, Azure and GCP.

Each major cloud provider offers its own load balancing services. Here are the load balancing services available in AWS, Azure, and GCP as of my last knowledge update in December 2023:

1. AWS (Amazon Web Services):

- **Elastic Load Balancing (ELB):** AWS offers Elastic Load Balancing, which includes the following types:
- **Application Load Balancer (ALB):** Best suited for load balancing of HTTP and HTTPS traffic and provides advanced request routing targeted at modern application architectures.
- **Network Load Balancer (NLB):** Designed to handle millions of requests per second while maintaining ultra-low latencies. It is ideal for load balancing of TCP traffic.
- **Classic Load Balancer (CLB):** Provides basic load balancing across multiple Amazon EC2 instances.

2. Azure:

- **Azure Load Balancer:** This is a Layer 4 load balancer that provides high availability by distributing incoming traffic among healthy virtual machines (VMs) within a single data center.
- **Azure Application Gateway:** This is a Layer 7 load balancer that provides advanced traffic distribution features for web traffic, such as URL-based routing and SSL termination.

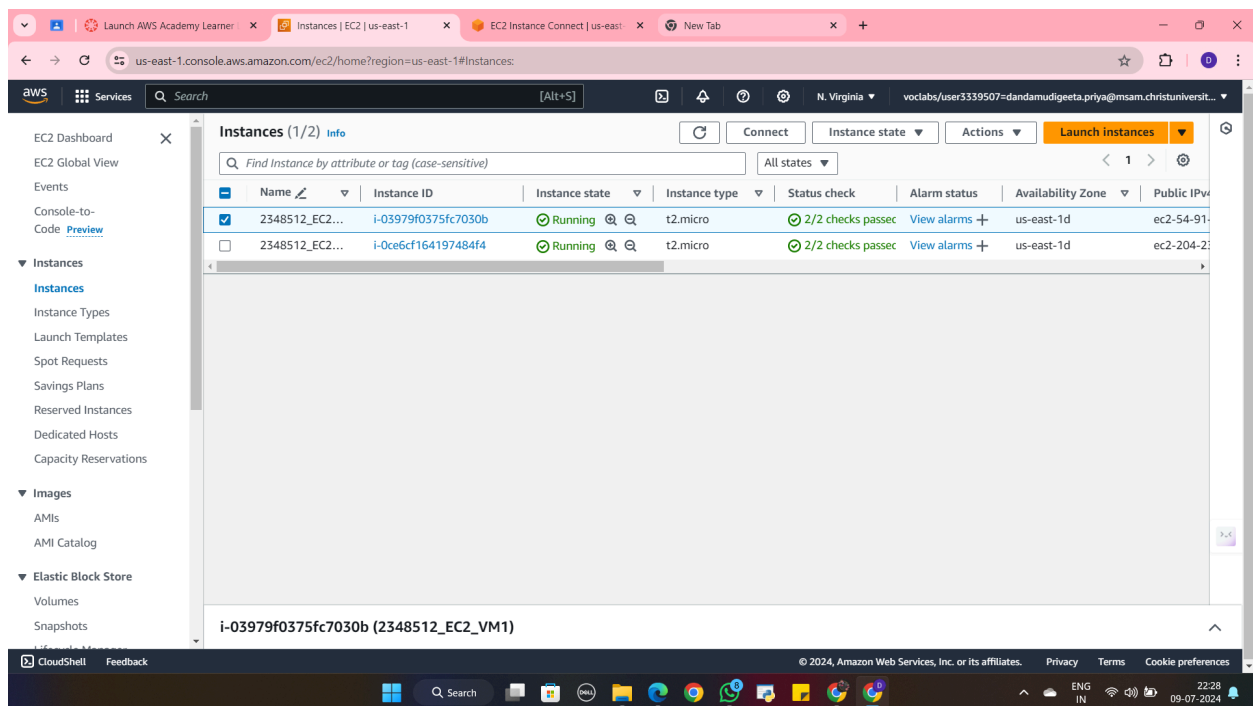
3. GCP (Google Cloud Platform):

- **Google Cloud Load Balancing:** GCP offers a suite of load balancing services, including:
- **HTTP(S) Load Balancing:** Distributes HTTP and HTTPS traffic among instances to ensure that no single instance is overwhelmed.
- **TCP/SSL Proxy Load Balancing:** For non-HTTP(S) traffic, such as SSL or TCP, providing scaling and high availability.
- **Network Load Balancing:** For balancing of non-HTTP(S) traffic at the network level, suitable for extreme performance and scalability requirements.

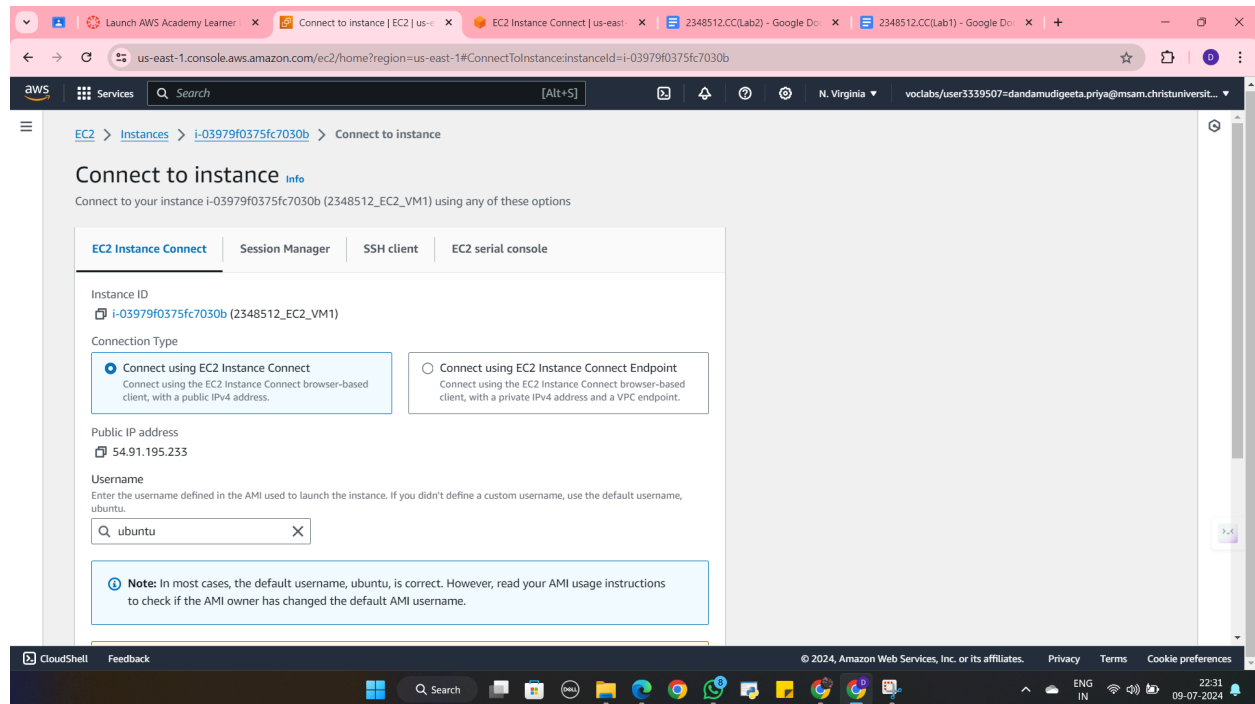
These load balancing services in AWS, Azure, and GCP play a critical role in ensuring the availability, scalability, and performance of applications hosted on their respective cloud platforms. Keep in mind that the services and features offered by cloud providers can change over time, so I recommend checking their official documentation for the latest information.

3. Create AWS EC2 / GCP VM Instances (Instance Name: Regno_EC2_VM1 , Regno_EC2_VM2) and install a webserver of your choice in each of the instances to host the website of your organization globally.

1: Creating the EC2 Instances:



Creating the AWS EC2 instances is a pretty standard procedure. Go to Instances, click on Launch Instance, select your Operating System (I used Ubuntu for this lab assignment), and leave the rest as default since we have a free tier account. And voila, we have our EC2 Instances as you can see in the screenshot below.



Connect to the instance VM1 .After Clicking Connect we will redirect to the console of the instance connect page. Now, how do we connect to our EC2 instance through the Cloud Shell?

Go to Instances, select one of the instances and there will be a big “Connect”button towards the top right, click on it and select Connect with EC2 Connect, and finally click on Connect again. This will redirect to a new page, with the cloud shell.

We are using Ubuntu, but the commands we use in it are very similar to that of Standard Linux.

We will first become the super-user, the equivalent of Admin in windows, by using the following command.

sudo su

We will use the following command to update all the packages.

sudo apt update

We will then install Apache2 using the following command.

sudo apt install apache2

Now, we will go through a set of commands that will allow us to create and deploy a simple static website on our instance.

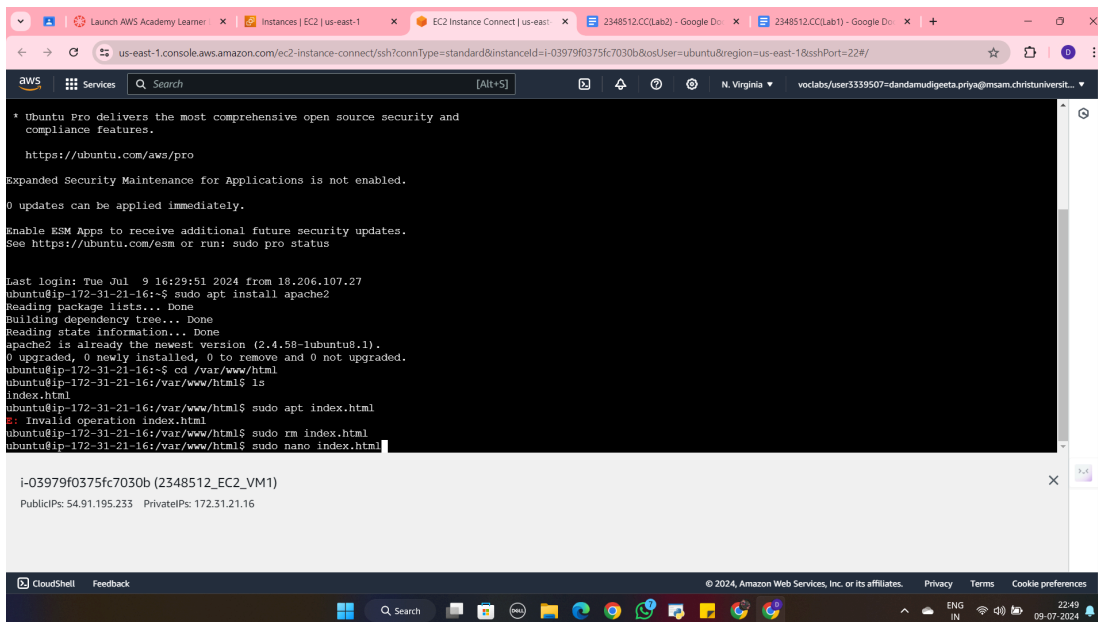
cd /var/www/html

ls

rm -r index.html

nano index.html

We change the root directory to the Apache server directory and delete the default index.html, and create a new index.html of our own with a simple static website. Now, we will go to our Instances page, copy the public IPv4 address of our Instance



```
* Ubuntu Pro delivers the most comprehensive open source security and
compliance features.

https://ubuntu.com/aws/pro

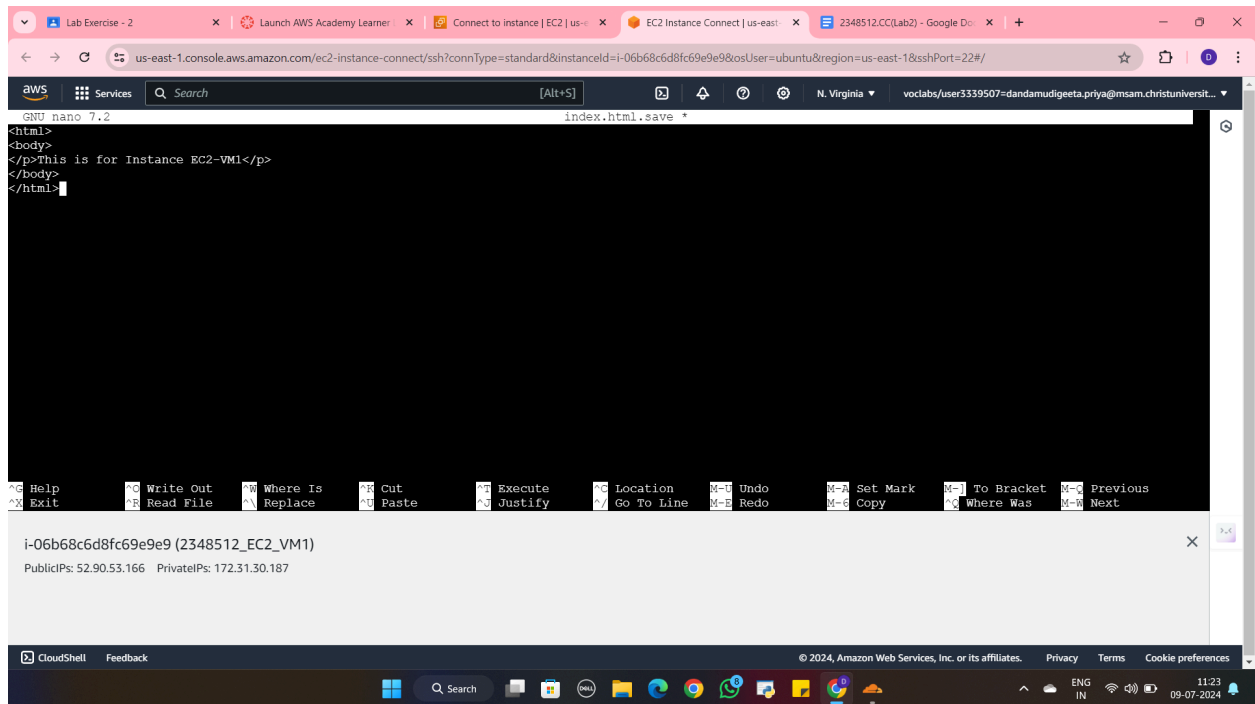
Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Last login: Tue Jul 9 16:29:51 2024 from 18.206.107.27
ubuntu@ip-172-31-21-16:~$ sudo apt install apache2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
apache2 is already the newest version (2.4.58-1ubuntu8.1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
ubuntu@ip-172-31-21-16:~$ cd /var/www/html
ubuntu@ip-172-31-21-16:/var/www/html$ ls
index.html
ubuntu@ip-172-31-21-16:/var/www/html$ sudo apt index.html
E: Invalid operation index.html
ubuntu@ip-172-31-21-16:/var/www/html$ sudo rm index.html
ubuntu@ip-172-31-21-16:/var/www/html$ sudo nano index.html

i-03979f0375fc7030b (2348512_EC2_VM1)
PublicIPs: 54.91.195.233 PrivateIPs: 172.31.21.16
```



← → ↻ ⚠ Not secure 54.242.236.233 ☆ 📄 📄 📄 📄

Hello. This is website number 1

We will follow the same procedure for the second instance. And the end result for that would be the following:

4. Create an Application Load Balancer to ensure the fair allocation of tasks among the web servers deployed on the Virtual machine instances.

Before we create a load balancer, we need to create a Target Group with both of our instances as Targets.

In order to create the Target Groups, scroll down the EC2 sidebar, where you will see a section of Load Balancers, under which there will be a Target Groups. Click on it and go to the Target Groups page.

Click on Create target groups.

- **Target Type: Instances**
- **Protocol: Port – default (IPv4)**

And click on create **target group**.

Now, in the second step, select both the instances that we have created and register them as Targets of the Target Groups.

Target type: Instance

Protocol : Port: HTTP: 80

Protocol version: HTTP1

VPC: [ypc-0af2b550787df7714](#)

IP address type: IPv4

Load balancer: [None associated](#)

2 Total targets

0 Healthy

0 Unhealthy

0 Anomalous

2 Unused

0 Initial

0 Draining

► Distribution of targets by Availability Zone (AZ)

Select values in this table to see corresponding filters applied to the Registered targets table below.

Targets | Monitoring | Health checks | Attributes | Tags

Registered targets (2) [Info](#) [Anomaly mitigation: Not applicable](#) [Refresh](#) [Deregister](#) [Register targets](#)

Target groups route requests to individual registered targets using the protocol and port number specified. Health checks are performed on all registered targets according to the target group's health check settings. Anomaly detection is automatically applied to HTTP/HTTPS target groups with at least 3 healthy targets.

Filter targets

<input type="checkbox"/>	Instance ID	Zone	Health status	Health status details	Launch...	Anomaly detection...
<input type="checkbox"/>	i-0079e758050bcde9e	us-east-1a	Unused	Target group is not co...	July 9, 20...	Normal
<input type="checkbox"/>	i-073a8733a87222039	us-east-1a	Unused	Target group is not co...	July 9, 20...	Normal

Another thing to keep in mind is to check the “launch-wizard” of our instances. Mine were launch-wizard-12 and launch-wizard-13.

Now, onto the Load Balancer creation.

Go to Load balancer [Create Load balancer](#) [Create Application Load balancer](#).

EC2 > Load balancers > Create Application Load Balancer

Create Application Load Balancer [Info](#)

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

► How Application Load Balancers work

Basic configuration

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

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

Scheme [Info](#)
Scheme can't be changed after the load balancer is created.

☒ **Internet-facing**
An internet-facing load balancer routes requests from clients over the internet to targets. Requires a public subnet. [Learn more](#)

☐ **Internal**
An internal load balancer routes requests from clients to targets using private IP addresses. Compatible with the IPv4 and Dualstack IP address types.

Load balancer IP address type [Info](#)
Select the type of IP addresses that your subnets use. Public IPv4 addresses have an additional cost.

☒ **IPv4**

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Give the Load balancer a name. Select the Availability zones, and select the subnets and everything and leave everything as default. And when you go to the Security tab, you should be able to see both our instances launch-wizards there.

EC2 Dashboard

EC2 Global View

Events

Console-to-Code

Preview

▼ Instances

Instances

Instance Types

Launch Templates

Spot Requests

Savings Plans

Reserved Instances

Dedicated Hosts

Capacity Reservations

▼ Images

AMIs

AMI Catalog

▼ Elastic Block Store

Volumes

Snapshots

Lifecycle Manager

EC2 > Load balancers

Load balancers (1/1)

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Filter load balancers

<input checked="" type="checkbox"/>	Name	DNS name	State	VPC ID	Availability Zones	Type
<input checked="" type="checkbox"/>	lb1	lb1-1741407654.us-east-1...	Provisioning...	vpc-0af2b550787df77...	2 Availability Zones	appli

Load balancer: lb1

Details | Listeners and rules | Network mapping | Resource map - new | Security | Monitoring | Integrations | Attributes | Tags

Security groups (2)

A security group is a set of firewall rules that control the traffic to your load balancer.

Security Group ID	Name	Description
sg-02c51f78957260c6a	launch-wizard-...	launch-wizard-12 created 2024-07-09T16:46:24.237Z
sg-08e04177c8bd16178	launch-wizard-...	launch-wizard-13 created 2024-07-09T17:14:01.298Z

CloudShell Feedback

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

EC2 Dashboard

EC2 Global View

Events

Console-to-Code

Preview

▼ Instances

Instances

Instance Types

Launch Templates

Spot Requests

Savings Plans

Reserved Instances

Dedicated Hosts

Capacity Reservations

▼ Images

AMIs

AMI Catalog

▼ Elastic Block Store

Volumes

Snapshots

Lifecycle Manager

EC2 > Load balancers

Load balancers (1/1)

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Filter load balancers

<input checked="" type="checkbox"/>	Name	DNS name	State	VPC ID	Availability Zones	Type
<input checked="" type="checkbox"/>	lb1	lb1-1741407654.us-east-1...	Active	vpc-0af2b550787df77...	2 Availability Zones	appli

Load balancer: lb1

Details | Listeners and rules | Network mapping | Resource map - new | Security | Monitoring | Integrations | Attributes | Tags

Security groups (2)

A security group is a set of firewall rules that control the traffic to your load balancer.

Security Group ID	Name	Description
sg-02c51f78957260c6a	launch-wizard-...	launch-wizard-12 created 2024-07-09T16:46:24.237Z
sg-08e04177c8bd16178	launch-wizard-...	launch-wizard-13 created 2024-07-09T17:14:01.298Z

CloudShell Feedback

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Now that we have configured our Load balancer, we can now go back to our Target Groups and our Load balancer should show up like we see here in the screenshot below. If we go back to the previous screenshot of the Target Groups, we can clearly see that there was no Load balancer associated with the target group.

EC2 Dashboard

EC2 Global View

Events

Console-to-Code

Preview

▼ Instances

Instances

Instance Types

Launch Templates

Spot Requests

Savings Plans

Reserved Instances

Dedicated Hosts

Capacity Reservations

▼ Images

AMIs

AMI Catalog

▼ Elastic Block Store

Volumes

Snapshots

Lifecycle Manager

▼ Network & Security

Security Groups

Elastic IPs

Placement Groups

Key Pairs

Network Interfaces

▼ Load Balancing

Load Balancers

EC2 > Target groups > tg1

tg1

Actions

Details

arn:aws:elasticloadbalancing:us-east-1:244480887546:targetgroup/tg1/1cd1b94a482ac32

Target type	Protocol : Port	Protocol version	VPC
Instance	HTTP: 80	HTTP1	vpc-0a12b550787d77714
IP address type	Load balancer		
IPv4	lb1		

2

Total targets

2 Healthy

0 Unhealthy

0 Unused

0 Initial

0 Draining

0 Anomalous

Distribution of targets by Availability Zone (AZ)

Select values in this table to see corresponding filters applied to the Registered targets table below.

Targets

Monitoring

Health checks

Attributes

Tags

Registered targets (2)

Anomaly mitigation: Not applicable

Deregister

Register targets

Target groups make requests to individual registered targets using the protocol and port number specified. Health checks are performed on all registered targets according to the target group's health check settings. Anomaly detection is automatically applied to HTTP/HTTPS target groups with at least 3 healthy targets.

Filter targets

	Instance ID	Name	Port	Zone	Health status	Health status details	Launch...	Anomaly detection result
<input type="checkbox"/>	i-0079c758050bcde9e	test2	80	us-east-1a	Healthy	-	July 9, 20...	Normal
<input type="checkbox"/>	i-073a8733a87222039	test	80	us-east-1a	Healthy	-	July 9, 20...	Normal

CloudShell

Feedback

© 2024, Amazon Web Services, Inc. or its affiliates

Privacy

Terms

Cookie preferences