

CLOUD LAB

What is Docker?

Docker is a platform that allows you to develop, ship, and run applications in isolated environments called **containers**. Containers package applications and their dependencies together, ensuring that they run consistently across different environments. It eliminates the "works on my machine" problem.

How Does Docker Work?

Docker uses **containerization** to isolate applications. It works by:

1. Using **Docker Engine**, which runs on the host system to manage containers.
2. Utilizing **Docker Images**, which are pre-packaged applications with all dependencies.
3. Running **Docker Containers**, which are instances of Docker images.
4. Using **Docker Hub**, a repository of pre-built images that you can pull and run.

Steps to Use Docker on Kali Linux

1. Install Docker on Kali Linux

Run the following commands to install Docker:

```
sudo apt update  
sudo apt install docker.io -y
```

Enable and start the Docker service:

```
sudo systemctl enable --now docker
```

Verify installation:

```
docker --version
```

2. Verify Docker Installation

Check if Docker is working properly:

```
docker run hello-world
```

This downloads a small test image and runs it in a container.

Running a Container in Docker

1. Pull an Image from Docker Hub

You can pull images from [Docker Hub](#). For example, to pull an **Ubuntu** image:

```
docker pull ubuntu
```

2. Run a Container

Start a container with an interactive shell:

```
docker run -it ubuntu bash
```

This starts a new Ubuntu container and opens a bash shell inside it.

3. List Running Containers

To see running containers:

```
docker ps
```

To see all containers (including stopped ones):

```
docker ps -a
```

4. Stop and Remove Containers

To stop a running container:

```
docker stop <container_id>
```

To remove a container:

```
docker rm <container_id>
```

5. Remove an Image

To remove a downloaded image:

```
docker rmi <image_name>
```

Example: Running Nginx on Kali Linux

If you want to run a web server using **Nginx**, follow these steps:

1. Pull the Nginx image:

```
docker pull nginx
```

2. Run the Nginx container:

```
docker run -d -p 8080:80 nginx
```

3. Open a browser and visit:

```
http://localhost:8080
```

You should see the default Nginx welcome page.

4. Stop the Nginx container:

```
docker stop <container_id>
```

Running DVWA (Damn Vulnerable Web Application) Using Docker on Kali Linux

DVWA is a deliberately vulnerable web application for security testing. You can quickly set it up using Docker.

Step 1: Pull the DVWA Docker Image

Run the following command to download the DVWA image:

```
docker pull vulnerables/web-dvwa
```

Step 2: Run DVWA Container

Now, start a new DVWA container using the command below:

```
docker run -d -p 8080:80 vulnerables/web-dvwa
```

- `d` : Runs the container in detached mode (in the background).
- `p 8080:80` : Maps port 8080 on your Kali Linux machine to port 80 inside the container.

Step 3: Access DVWA

Open your web browser and go to:

```
http://localhost:8080
```

If running on a remote server, use:

```
http://<your-ip>:8080
```

Step 4: Login to DVWA

Use the default login credentials:

- **Username:** `admin`

- **Password:** `password`

Click **Login**, and you should now be inside DVWA.

Step 5: Configure DVWA

Once logged in:

1. Go to **DVWA Security** settings.
 2. Set **Security Level** to **Low** to enable testing.
 3. Click **Create/Reset Database** under **Database Setup**.
-

Step 6: Managing the DVWA Container

Check Running Containers

```
docker ps
```

Stop the Container

```
docker stop <container_id>
```

Restart the Container

```
docker start <container_id>
```

Remove the Container

```
docker rm <container_id>
```

Remove the Image (if needed)

```
docker rmi vulnerables/web-dvwa
```

DC4 using docker

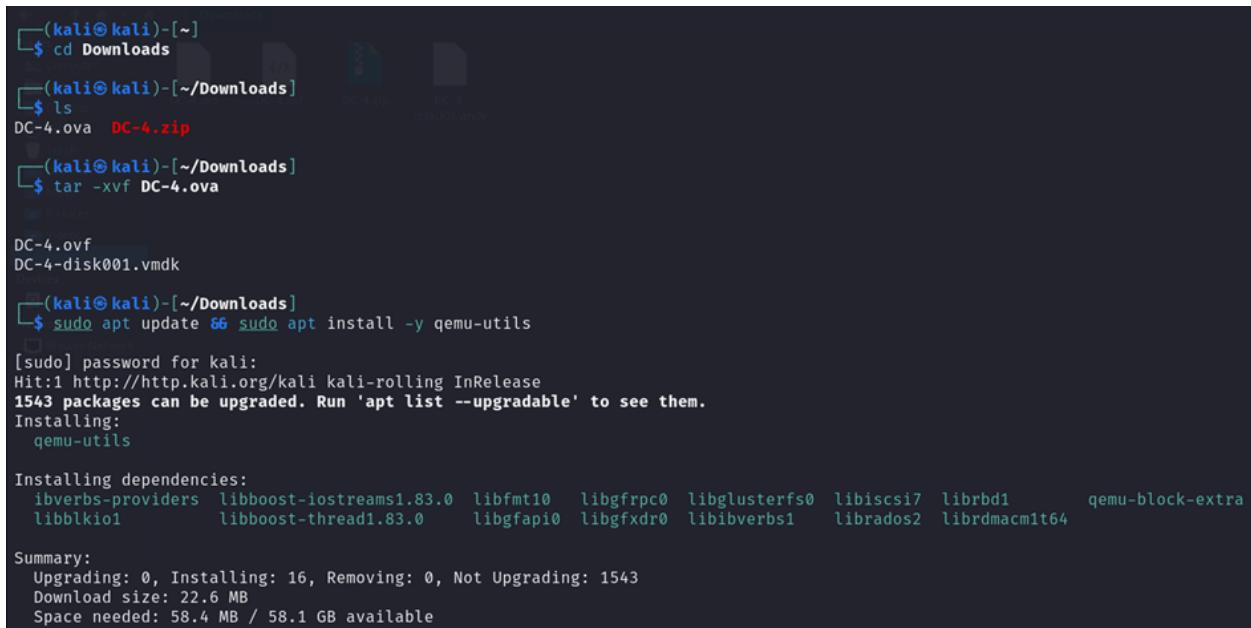
Step 1: Extract the OVA File

First, extract the **DC-4.ova** file to get the virtual disk (.vmdk) inside it.

```
tar -xvf DC-4.ova
```

This will extract files like:

- **DC-4.ovf** (Open Virtualization Format descriptor)
- **DC-4.vmdk** (Virtual disk file)



The screenshot shows a terminal session on a Kali Linux system. The user navigates to the Downloads directory and extracts the contents of a file named 'DC-4.ova'. The extracted files are listed as 'DC-4.ovf' and 'DC-4-disk001.vmdk'. Following this, the user runs an apt update command and installs the 'qemu-utils' package via sudo apt install -y qemu-utils. The terminal output shows the progress of the download and the successful installation of the package.

```
(kali㉿kali)-[~]
$ cd Downloads
(kali㉿kali)-[~/Downloads]
$ ls
DC-4.ova  DC-4.zip

(kali㉿kali)-[~/Downloads]
$ tar -xvf DC-4.ova

DC-4.ovf
DC-4-disk001.vmdk

(kali㉿kali)-[~/Downloads]
$ sudo apt update && sudo apt install -y qemu-utils

[sudo] password for kali:
Hit:1 http://http.kali.org/kali kali-rolling InRelease
1543 packages can be upgraded. Run 'apt list --upgradable' to see them.
Installing:
  qemu-utils

Installing dependencies:
  libverbs-providers  libboost-iostreams1.83.0  libfmt10  libgfrpc0  libglusterfs0  libiscsi7  librbd1      qemu-block-extra
  libblkio1           libboost-thread1.83.0    libgfapi0  libgfxdr0  libibverbs1   librados2  librdmacm1t64

Summary:
  Upgrading: 0, Installing: 16, Removing: 0, Not Upgrading: 1543
  Download size: 22.6 MB
  Space needed: 58.4 MB / 58.1 GB available
```

Step 2: Convert VMDK to a Raw Image

Use qemu-img to convert the VMDK file to a raw image:

```
qemu-img convert -f vmdk -O raw DC-4.vmdk dc4.raw
```

The screenshot shows a terminal window titled 'kali@kali: ~/Downloads'. It displays the following command and its output:

```
(kali㉿kali)-[~/Downloads]
$ qemu-img convert -f vmdk -O raw DC-4.vmdk dc4.raw
qemu-img: Could not open 'DC-4.vmdk': Could not open 'DC-4.vmdk': No such file or directory
(kali㉿kali)-[~/Downloads]
$ qemu-img convert -f vmdk -O raw DC-4.vmdk dc4.raw

(kali㉿kali)-[~/Downloads]
$ ls
DC-4.ova  DC-4.ovf  dc4.raw  DC-4.vmdk  DC-4.zip
(kali㉿kali)-[~/Downloads]
```

Step 3: Create a Docker Image from the Raw Disk

Now, create a Docker image using the raw disk file.

1. Create a directory for the image: **mkdir dc4-docker && cd dc4-docker**
2. Move the raw image into the directory: **mv .. /dc4.raw .**
3. Create a Dockerfile in the dc4-docker directory: **nano Dockerfile**

The screenshot shows a terminal window titled '(kali㉿kali)-[~/Downloads]'. It displays the following commands and their results:

```
(kali㉿kali)-[~/Downloads]
$ mkdir dc4-docker && cd dc4-docker
Network
(kali㉿kali)-[~/Downloads/dc4-docker]
$ mv .. /dc4.raw .

(kali㉿kali)-[~/Downloads/dc4-docker]
$ ls
dc4.raw
```

##on terminal

```
sudo apt update && sudo apt install -y qemu-system-x86 qemu-utils
```

```
RUN apt update && apt install -y qemu-system-x86 wget
```

```
sudo apt update && sudo apt install -y docker.io qemu-system-x86\n
```

```
sudo systemctl start docker
```

```
sudo systemctl status docker
```

```
sudo systemctl enable --now docker
```

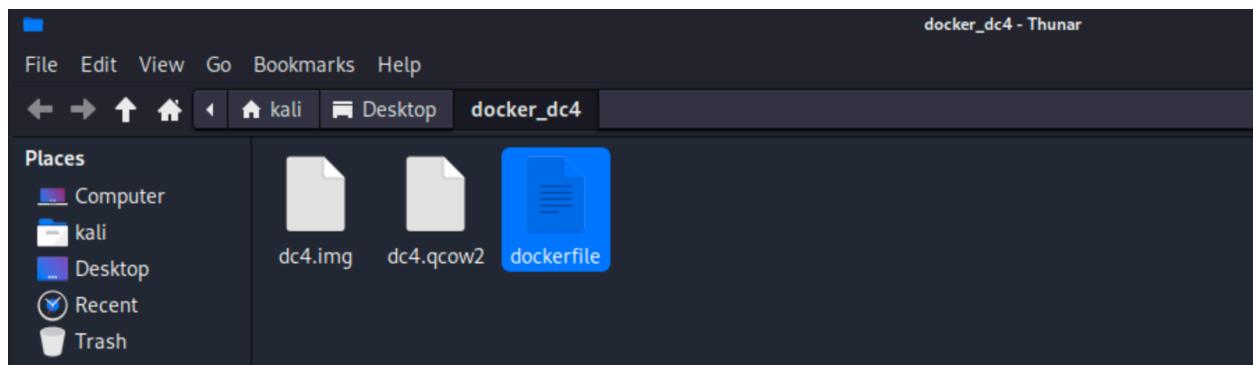
```
mkdir docker_dc4
```

```
#keep a copy of dc4.img and dc4.qcow2
```

```
#steps to extract .img
```

```
qemu-img convert -O raw DC4-disk001.vmdk dc4.img
```

```
sudo qemu-img convert -f raw -O qcow2 dc4.img dc4.qcow2
```



```
touch dockerfile
```

```

##paste this inside dockerfile

FROM ubuntu:latest

RUN apt update && apt install -y qemu-system-x86

COPY dc4.qcow2 /dc4.qcow2

EXPOSE 80

CMD ["qemu-system-x86_64", "-m", "2048", "-hda", "/dc4.qcow2", "-net",
"nic", "-net", "user,hostfwd=tcp::80::80", "-nographic"]

```

4. Build the Docker image:

```
docker build -t dc4-container .
```

```

(kali㉿kali)-[~/Downloads/dc4-docker]
$ docker build -t dc4-container .
[+] Building 257.1s (8/8) FINISHED
  => [internal] load build definition from Dockerfile
  => [internal] transfering dockerfile: 416B
  => [internal] load metadata for docker.io/library/ubuntu:latest
  => [internal] load .dockerignore
  => [internal] transfering context: 2B
  => CACHED [1/3] FROM docker.io/library/ubuntu:latest@sha256:72297848456d5d37d1262630108ab308d3e9ec7ed1c3286a32fe09856619a782
  => [internal] load build context
  => [internal] transfering context: 34B
  => [2/3] RUN apt update && apt install -y qemu-system-x86
  => [3/3] COPY dc4.qcow2 /dc4.qcow2
  => exporting to image
  => => exporting layers
  => => writing image sha256:cd99c8359d14f7bb40bf881a51354ff4b066ea31ec20f459da289132ee4b8d67
  => => naming to docker.io/library/dc4-container

```

Step 4: Run the DC-4 Container

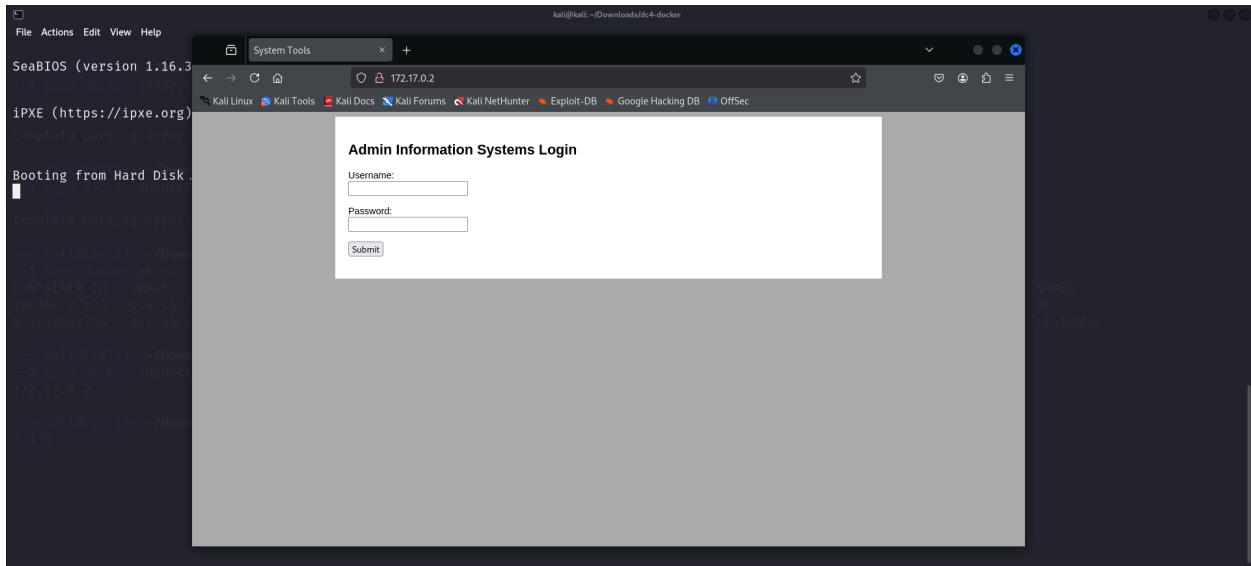
```
sudo docker run --name dc -it dc4-container
```

##it will start to boot

NEW TERMINAL;

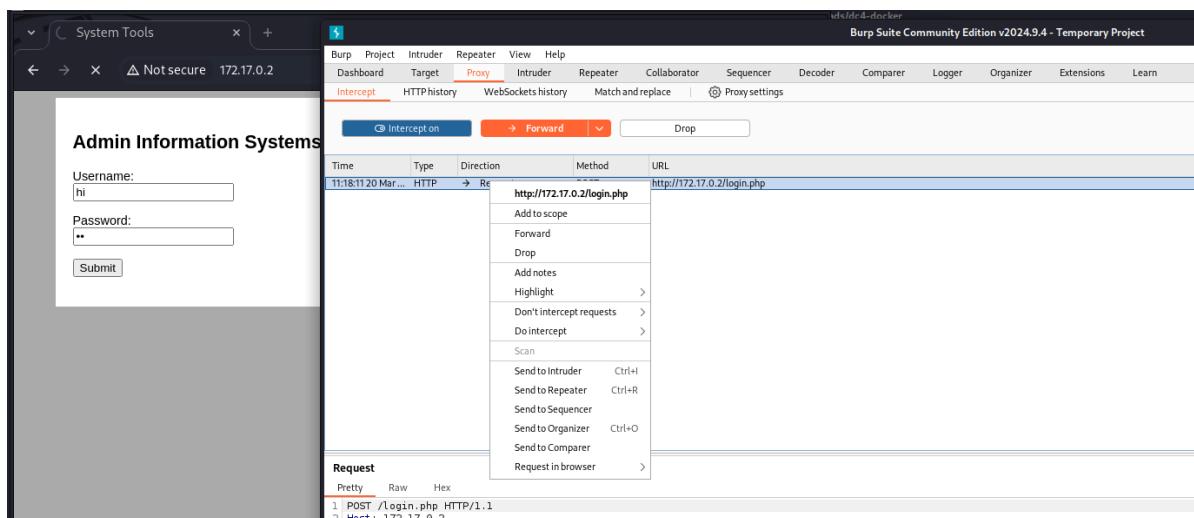
```
sudo docker inspect -f '{{range.NetworkSettings.Networks}}{{.IPAddress}}\n{{end}}' dc4-container
```

copy ip address and paste it in mozilla



Start burpsuit bruteforce attack

1. Send the login request to intruder



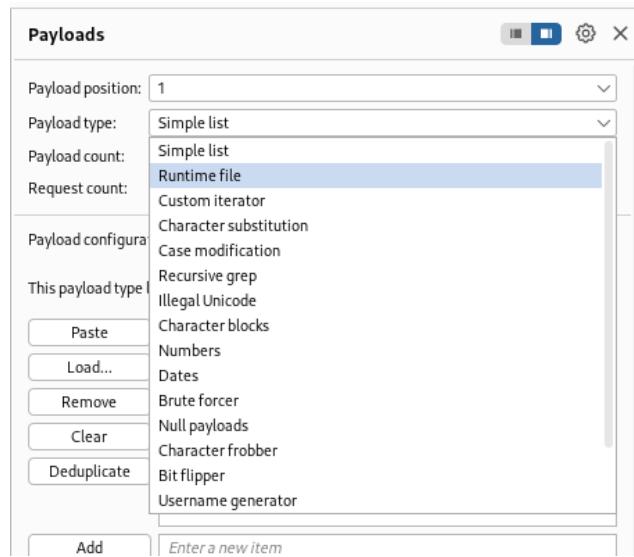
2. Add the section to add payload

Add § Clear § Auto §

```
1 POST /login.php HTTP/1.1
2 Host: 172.17.0.2
3 Content-Length: 23
4 Cache-Control: max-age=0
5 Accept-Language: en-US,en;q=0.9
6 Origin: http://172.17.0.2
7 Content-Type: application/x-www-form-urlencoded
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.127 Safari/537.36
10 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
11 Referer: http://172.17.0.2/
12 Accept-Encoding: gzip, deflate, br
13 Cookie: PHPSESSID=j2blakm4rh5uq22jb5j04
14 Connection: keep-alive
15
16 username=Shi§&password=Sww§
```

3. Use the required type of brute force attack and payload type

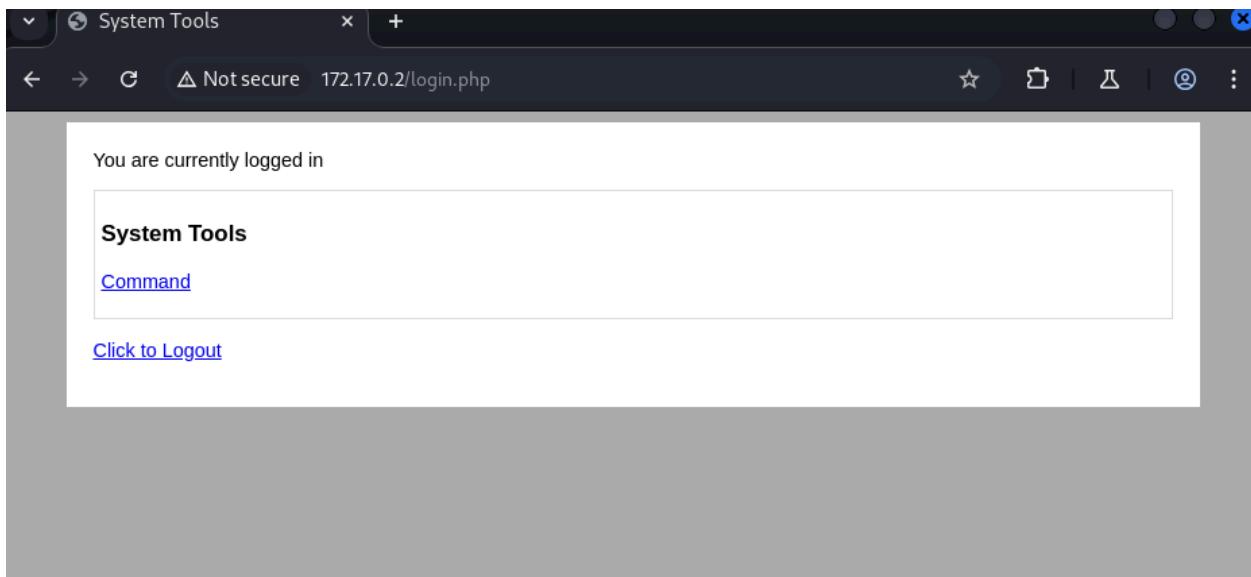
The screenshot shows the Burp Suite Community Edition interface. The 'Intruder' tab is active. On the left, a list of attack types is shown, with 'Cluster bomb attack' selected. The right side displays the 'Payloads' configuration panel, which includes fields for 'Payload position' (set to 'All payload positions'), 'Payload type' (set to 'Simple list'), and a list of payloads. The payloads list contains the string 'Shi§'. Below the payloads list, there are buttons for Paste, Load..., Remove, Clear, and Deduplicate, along with an 'Add' button and an input field for 'Enter a new item'.



4. Add wordlist

5. Start attack check the payload for where status code is 200 you have the password Congrats 😊

Request	Payload 1	Payload 2	Status code	Response received	Error	Timeout	Length	Comment
1. POST	72	happy	302	4			504	
2. Host	73	hello	302	5			504	
3. Content-Type	74	hi	302	12			504	
4. User-Agent	75	password	302	8			504	
5. Accept	76	admin	302	7			665	
6. Origin	77	root	200	6			641	
7. Content-Security-Policy	78	happy	200	6			641	
8. User-Agent	79	pass	200	8			641	
9. User-Agent	80	toor	200	8			641	
10. User-Agent	81	happy	200	5			641	



How to push image to docker Hub?

1. Create an account in docker hub
2. Create a repository
3. Login to the docker hub account from local using **docker login**
4. If your image is not already tagged correctly, tag it using: **docker tag <local-image-id> kanusharao/cloud-lab:tagname**
5. Find your **local image ID** using: **docker images**
6. Run the following command to push the image: **docker push kanusharao/cloud-lab:tagname**
7. Repo Link: <https://hub.docker.com/repository/docker/kanusharao/cloud-lab/tags>

```
(kali㉿kali)-[~/Downloads/dc4-docker]
└─$ docker login
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com/ to create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants better security and is required for organizations using SSO. Learn more at https://docs.docker.com/go/access-tokens/

Username: anusharao4520@gmail.com
Password:
WARNING! Your password will be stored unencrypted in /home/kali/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded

(kali㉿kali)-[~/Downloads/dc4-docker]
└─$ docker images
REPOSITORY          TAG      IMAGE ID      CREATED     SIZE
<none>            <none>   48a0db00d404  2 hours ago  2.42GB
dc4-container       latest   c5256f14b4ce  2 hours ago  2.42GB
<none>            <none>   cd99c8359d14  2 hours ago  2.42GB
<none>            <none>   147a9431e4e5  2 hours ago  2.42GB
dc4                latest   e050eb2e5992  22 hours ago 5.44GB
<none>            <none>   2a3e68654697  22 hours ago 5.37GB
gcr.io/k8s-minikube/kicbase v0.0.46  e72c4cbe9b29  2 months ago 1.31GB
```

```
(kali㉿kali)-[~/Downloads/dc4-docker]
└─$ docker tag c5256f14b4ce kanusharao/cloud-lab:dc4

(kali㉿kali)-[~/Downloads/dc4-docker]
└─$ docker images
REPOSITORY          TAG      IMAGE ID      CREATED     SIZE
<none>            <none>   147a9431e4e5  2 hours ago  2.42GB
<none>            <none>   48a0db00d404  2 hours ago  2.42GB
kanusharao/cloud-lab        dc4    c5256f14b4ce  2 hours ago  2.42GB
dc4-container       latest   c5256f14b4ce  2 hours ago  2.42GB
<none>            <none>   cd99c8359d14  2 hours ago  2.42GB
dc4                latest   e050eb2e5992  22 hours ago 5.44GB
<none>            <none>   2a3e68654697  22 hours ago 5.37GB
gcr.io/k8s-minikube/kicbase v0.0.46  e72c4cbe9b29  2 months ago 1.31GB

(kali㉿kali)-[~/Downloads/dc4-docker]
└─$ docker push kanusharao/cloud-lab:dc4

The push refers to repository [docker.io/kanusharao/cloud-lab]
058270631f00: Pushed
738005c3537e: Pushed
4b7c01ed0534: Mounted from library/ubuntu
dc4: digest: sha256:9dd245707e91835fa2b36d1a043b77d83479fc16c9ebd3a0bacd9b6668b518e3 size: 955
```

kanusharao Docker Personal

Repositories / [cloud-lab](#) / Tags

Using 0 of 1 private repositories. [Get more](#)

kanusharao/cloud-lab ⓘ

Last pushed 1 minute ago • Repository size: 884.9 MB

Docker commands

To push a new tag to this repository.

```
docker push kanusharao/cloud-lab:  
tagname
```

Add a description ⓘ ⓘ

Add a category ⓘ ⓘ

General Tags Image Management BETA Collaborators Webhooks Settings

Sort by Newest Filter tags Delete

TAG	Digest	OS/ARCH	Last pull	Compressed size
dc4	9dd245707e91	linux/amd64	less than 1 day	884.9 MB

AWS

After signup console:

eu-north-1.console.aws.amazon.com/console/home?region=eu-north-1

Gmail Inbox (46) - 20ai010... Learn Python - Free... Cyber Career Pathw... Python Tutorial | Pr... GitHub PyFlo Learn to become a... All Bookmarks

aws Search [Alt+S]

Console Home Info Reset to default layout + Add widgets

Recently visited Info

- Billing and Cost Management
- Service Quotas
- IAM
- IAM Identity Center

View all services

Applications (0) Info Create application

Region: Europe (Stockholm)

Select Region eu-north-1 (Current Region) Find applications

Name	Description	Region	Origin
No applications Get started by creating an application.			

Create application

CloudShell Feedback View all services Go to myApplications © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

NAVIGATE TO IAM:

The screenshot shows the AWS IAM console with the 'Recently visited' section open. The 'Recently visited' section contains links to 'IAM', 'Console Home', and other AWS services like 'AWS Account', 'Quick Links', and 'Tools'. The 'AWS Account' section displays account details such as 'Account ID' (209644362016), 'Account Alias' (Create), and 'Sign-in URL for IAM users in this account' (https://209644362016.signin.aws.amazon.com/console). The 'Quick Links' section provides access to 'My security credentials' and 'Tools'.

add users , grant him certain permission/Policies

The screenshot shows the 'Users' page in the AWS IAM console. It lists four users: 'group_user', 'sid', 'user1', and 'user3'. Each user entry includes a checkbox, the user name, path, group (1), last activity (indicated by a timestamp), MFA status, password age, and console last sign-in information.

	User name	Path	Group	Last activity	MFA	Password age	Console last sign-in
<input type="checkbox"/>	group_user	/	1	4 months ago	-	-	-
<input type="checkbox"/>	sid	/	1	4 months ago	-	-	-
<input type="checkbox"/>	user1	/	1	5 ago	-	-	-
<input type="checkbox"/>	user3	/	0	5 ago	-	-	-

create group with few policies

The screenshot shows the 'User groups' section of the AWS IAM console. It displays one user group named 'Group1'. The group has 3 users assigned and permissions defined. It was created 1 month ago.

Group name	Users	Permissions	Creation time
Group1	3	Defined	1 month ago

permission override for user in group

Using IAM to assign a user to access S3 bucket

→ Go to IAM [root user]

→ create users

The screenshot shows the 'Specify user details' step of the IAM user creation wizard. The user name is set to 'test_user1'. A note indicates that the user name can have up to 64 characters and includes a link to learn more about best practices for console access.

User details

User name: test_user1

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = . @ _ - (hyphen)

Provide user access to the AWS Management Console - optional
If you're providing console access to a person, it's a [best practice](#) to manage their access in IAM Identity Center.

Info: If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keypairs, you can generate them after you create this IAM user. [Learn more](#)

Cancel **Next**

★ARN

test_user1 Info

Summary

ARN
arn:aws:iam::331409392843:user/test_user1

Console access
Disabled

Access key 1
[Create access key](#)

Created
May 14, 2025, 22:45 (UTC+05:30)

Last console sign-in
-

Permissions **Groups** **Tags** **Security credentials** **Last Accessed**

Permissions policies (0)

Permissions are defined by policies attached to the user directly or through groups.

Filter by Type

Search All types

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

→ to enable console access to that user [test-user1]

Permissions **Groups** **Tags** **Security credentials** **Last Accessed**

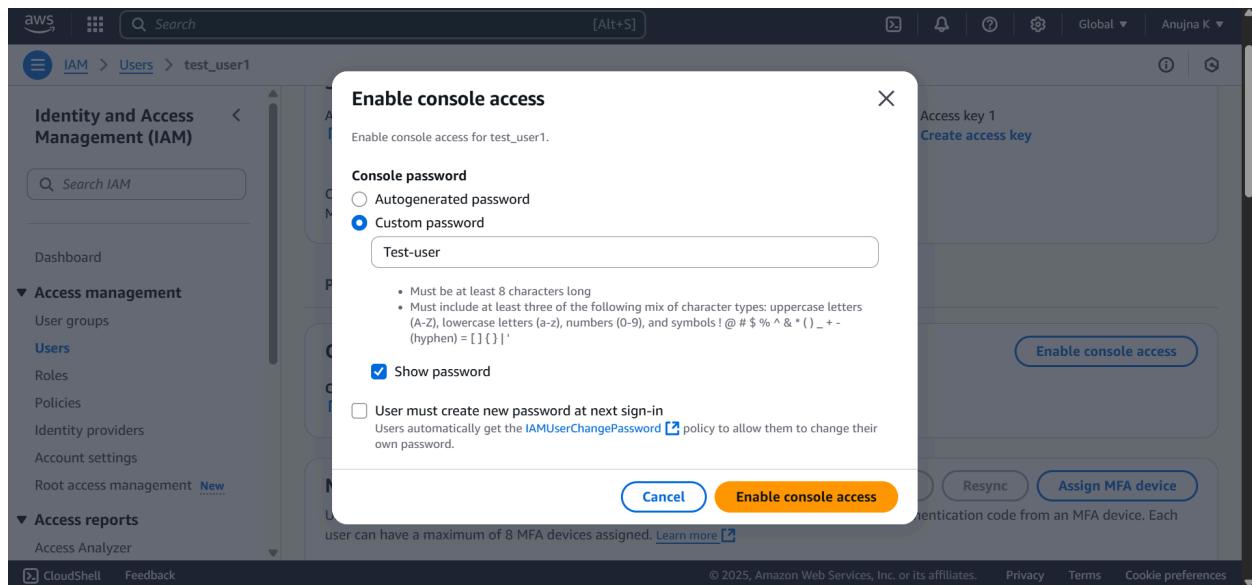
Console sign-in

Console sign-in link
<https://331409392843.signin.aws.amazon.com/console>

Enable console access

Console password
Not enabled

password → Test-user



Console password

You have successfully enabled the user's new password.

This is the only time you can view this password. After you close this window, if the password is lost, you must create a new one.

Console sign-in URL

<https://331409392843.signin.aws.amazon.com/console>

User name

test_user1

Console password

Test-user [Hide](#)

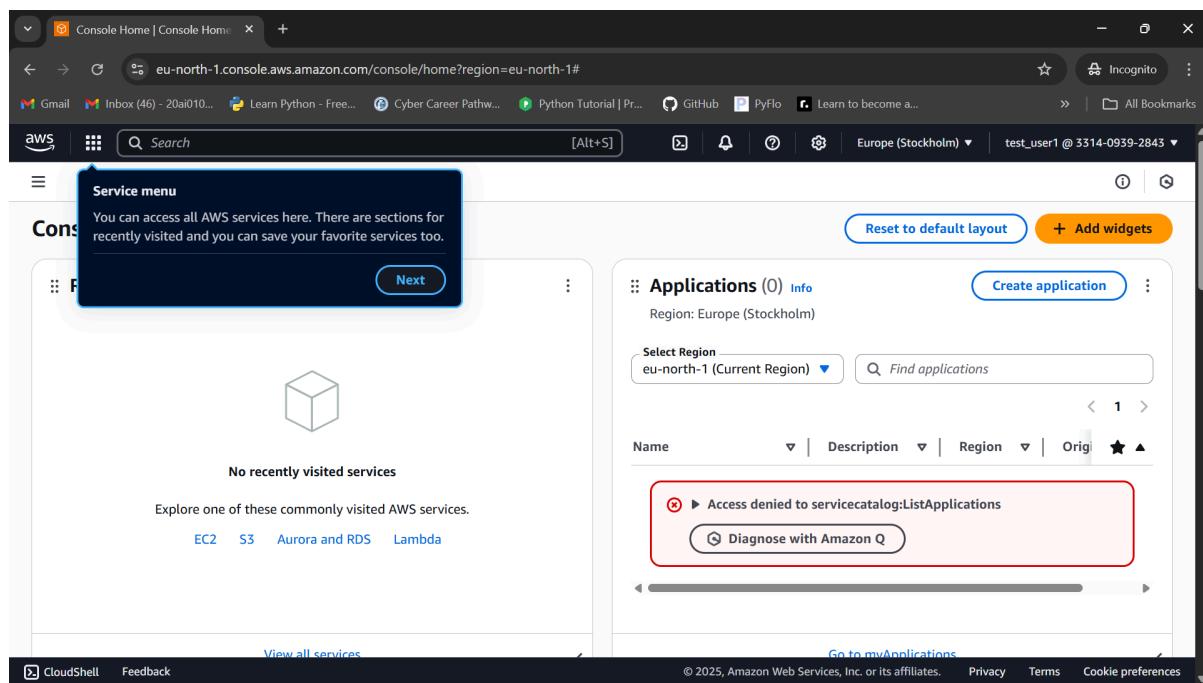
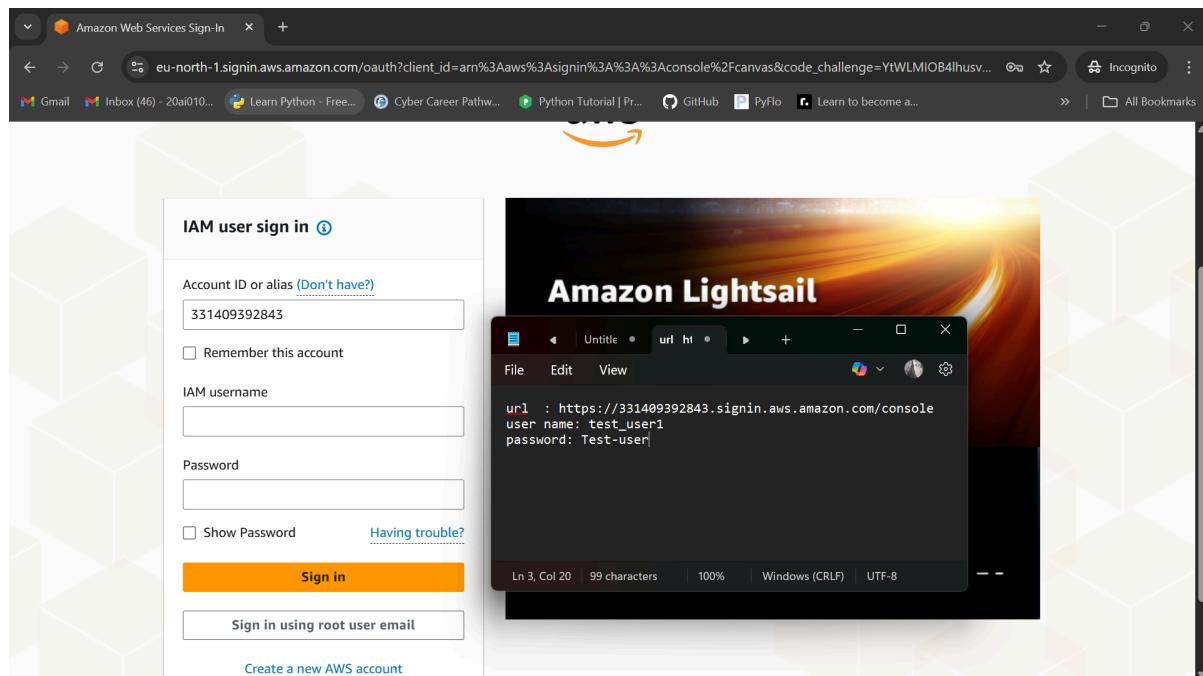
[Download .csv file](#)

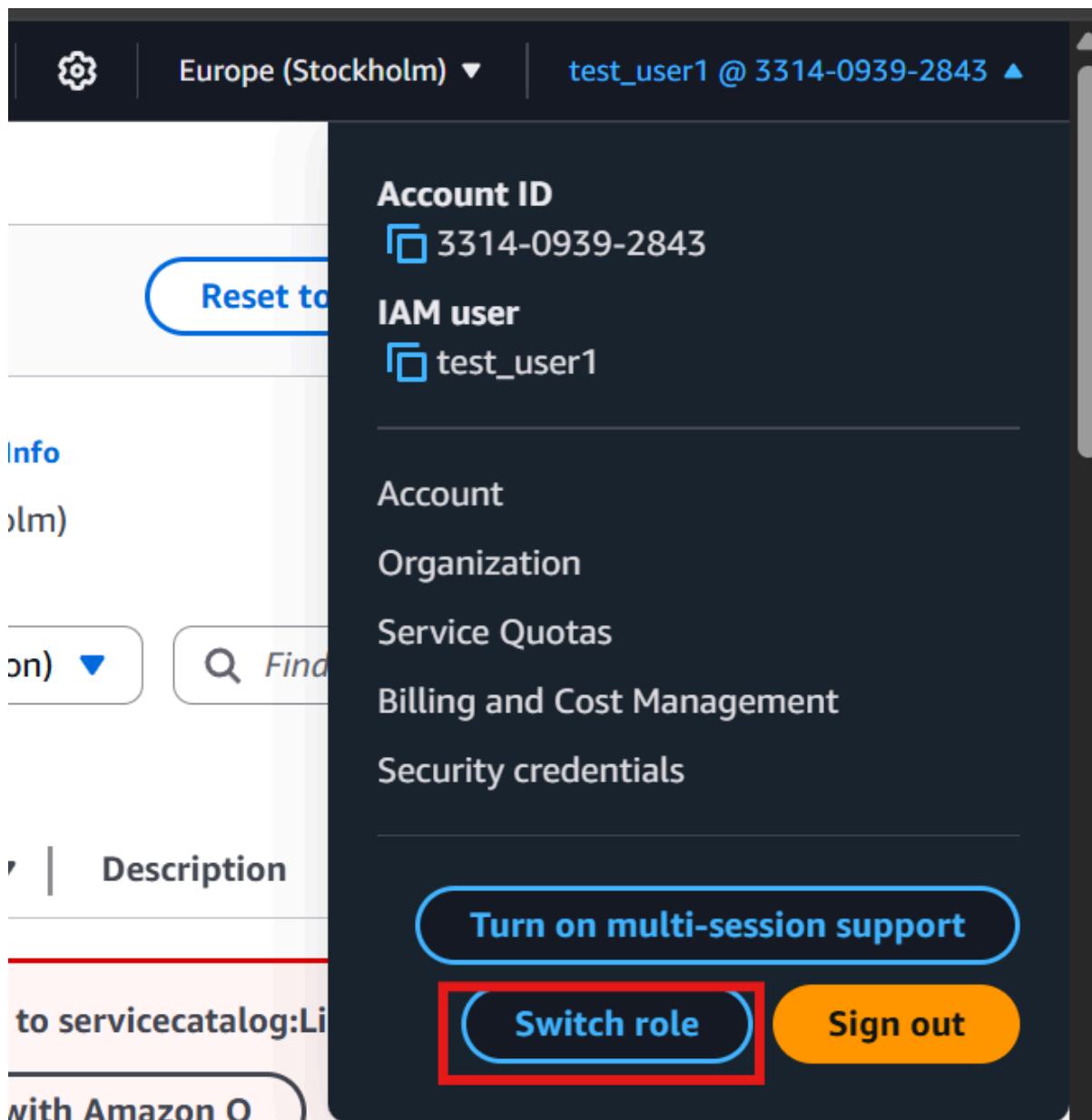
[Close](#)

→ to perform login for this test user :

→ open a new window and copy the console sign-in url [can use incognito mode]

→ copy and paste username and password





→ switch role means its not a root user, is an IAM user

→ click on S3 on dashboard:

The screenshot shows the Amazon S3 landing page. On the left, there's a sidebar with navigation links like 'General purpose buckets', 'Directory buckets', etc. The main area features a large 'Amazon S3' logo with the tagline 'Store and retrieve any amount of data from anywhere'. A prominent 'Create a bucket' button is visible. Below it, a text box explains that every object in S3 is stored in a bucket and provides instructions to upload files and folders. At the bottom, there's a 'How it works' section and a 'Pricing' section.

The screenshot shows the 'Buckets' page under the 'Amazon S3' service. The sidebar is identical to the landing page. The main content area displays an 'Account snapshot' and a table for 'General purpose buckets'. The table has one row, which is highlighted with a red border and labeled 'Error' with the message 'Access Denied'. There are buttons for 'Copy ARN', 'Empty', 'Delete', and 'Create bucket'.

This user doesn't have the permission to access S3 buckets. The root user have the permission to access S3 buckets.

→ To create a policy for accessing S3 bucket for that user:

→ go to root user → then IAM → Policies → Create policy

we can then use the built-in policies or create a default policy(as in our case)

Screenshot of the AWS IAM Policies page. The page shows a list of 1348 policies. A red box highlights the "Create policy" button.

Policy name	Type	Used as	Description
AccessAnalyzerSer...	AWS managed	None	-
AdministratorAccess	AWS managed - job fu...	None	Provides full access to AWS services an
AdministratorAcce...	AWS managed	None	Grants account administrative permis
AdministratorAcce...	AWS managed	None	Grants account administrative permis
AIOpsAssistantPolicy	AWS managed	None	Provides ReadOnly permissions requir.
AIOpsConsoleAdmi...	AWS managed	None	Grants full access to Amazon AI Opera
AIOpsOperatorAcc...	AWS managed	None	Grants access to the Amazon AI Opera

Screenshot of the "Create policy" wizard, Step 1: Specify permissions. The "Visual" tab is selected. A red box highlights the "Statement1" entry in the JSON editor.

```

1▼ {
2    "Version": "2012-10-17",
3    "Statement": [
4        {
5            "Sid": "Statement1",
6            "Effect": "Allow",
7            "Action": [],
8            "Resource": []
9        }
10    ]
11 }

```

The right panel shows the "Edit statement" section with "Statement1" selected, and the "Available" services list including AI Operations, API Gateway, and API Gateway V2.

```

1 ▼ {
2     "Version": "2012-10-17",
3     "Statement": [
4         {
5             "Sid": "AllowS3AccessToTestUser",
6             "Effect": "Allow",
7             "Action": "s3:*",
8             "Resource": "*"
9         }
10    ]
11 }

```

The screenshot shows the AWS IAM 'Create policy' wizard at Step 2: Review and create. The policy document is displayed in a code editor-like interface. The policy content is:

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3AccessToTestUser",
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": "*"
        }
    ]
}

```

The 'Policy name' field contains 'test-user-allow-s3'. The 'Description - optional' field is empty. The 'Permissions defined in this policy' section shows the policy document. The bottom right corner has an 'Edit' button.

→ Assigning the policy to the user:

IAM → users → test_user → Add Permission

The screenshot shows the AWS IAM User Details page for a user named 'test_user1'. The 'Summary' section displays the ARN (arn:aws:iam::331409392843:user/test_user1), which is highlighted with a red box. Other details shown include 'Console access Enabled without MFA', 'Created May 14, 2025, 22:45 (UTC+05:30)', and 'Last console sign-in Never'. The 'Permissions' tab is selected, showing 'Permissions policies (0)'. A button labeled 'Add permissions' is also highlighted with a red box. The left sidebar shows navigation options like Dashboard, Access management (User groups, Roles, Policies, Identity providers, Account settings, Root access management), and Access reports.

This screenshot shows the 'Add permissions' step in the IAM User creation wizard. It features three options: 'Add user to group', 'Copy permissions', and 'Attach policies directly', with 'Attach policies directly' selected. Below this is a search results table for 'Permissions policies (1/1349)' containing two entries: 'AWSIAMIdentityCenterAllowAll' (AWS managed, 0 matches) and 'test-user-allow-s3' (Customer managed, 0 matches). The table includes filters for 'Policy name' and 'Type'.

The screenshot shows the AWS IAM 'Add permissions' review step. The navigation path is IAM > Users > test_user1 > Add permissions. A progress bar indicates Step 1: Add permissions is complete, and Step 2: Review is in progress. The 'Review' section displays the following details:

- User details:** User name: test_user1
- Permissions summary (1):**

Name	Type	Used as
test-user-allow-s3	Customer managed	Permissions policy

At the bottom, there are 'Cancel', 'Previous', and 'Add permissions' buttons.

After adding the policy, now the test user has the access to S3 buckets -

The screenshot shows the AWS S3 Buckets page. The URL is eu-north-1.console.aws.amazon.com/s3/buckets?region=eu-north-1&bucketType=general. The sidebar includes links for General purpose buckets, Directory buckets, Table buckets, Access Grants, Access Points, Object Lambda Access Points, Multi-Region Access Points, Batch Operations, and IAM Access Analyzer for S3. The main content area shows the 'General purpose buckets' tab selected, with a message: "Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets." Below this, it says "General purpose buckets (0) All AWS Regions" and "Buckets are containers for data stored in S3." A search bar and a table header for "Name", "AWS Region", "IAM Access Analyzer", and "Creation date" are shown. The table body displays the message: "No buckets. You don't have any buckets." A "Create bucket" button is located at the bottom right of the table area.

→ Creating a group with multiple users

The screenshot shows the AWS IAM User groups page. The left sidebar has sections for Identity and Access Management (IAM) like Dashboard, Access management (User groups, Users, Roles, Policies, Identity providers, Account settings, Root access management), and Access reports (Access Analyzer). The main area title is "User groups (0)" with an info link. It says "A user group is a collection of IAM users. Use groups to specify permissions for a collection of users." A search bar and a table header with columns Group name, Users, Permissions, and Creation time are shown. Below the table, it says "No resources to display". The bottom navigation bar includes CloudShell, Feedback, and links to Privacy, Terms, and Cookie preferences.

attach policies to the group, those policies are applicable to all users in the group

The screenshot shows the AWS IAM User groups page for "Group-1". The left sidebar is identical to the previous screenshot. The main area shows a "Summary" card with User group name: Group-1, Creation time: May 15, 2025, 06:26 (UTC+05:30), and ARN: arn:aws:iam::331409392843:group/Group-1. Below the summary are tabs for Users (1) and Permissions. The Permissions tab is selected, showing "Permissions policies (1) Info" with a note "You can attach up to 10 managed policies." A table lists one policy: test-user-allow-s3, which is Customer managed. A "Filter by Type" dropdown is set to "All types". The bottom navigation bar is the same as the first screenshot.

Doker

Recommended: GUI via Web Browser (VNC/noVNC)

This method lets you run a full Linux GUI inside Docker, and access it via your browser on localhost.



Option A: Use

dorowu/ubuntu-desktop-Ixde-vnc

as a base

Step 1: Pull GUI-Enabled Docker Base

```
docker pull dorowu/ubuntu-desktop-Ixde-vnc
```

This gives you:

- Full GUI desktop
- Accessible on localhost:8080 via browser (noVNC)
- Also exposes traditional VNC on 5901

Step 2: Create Your Dockerfile with Vulnerable App

Create a Dockerfile like this:

```
FROM dorowu/ubuntu-desktop-Ixde-vnc
USER root
# Install vulnerable web app dependencies
RUN apt-get update && apt-get install -y \
apache2 php php-mysqli git unzip
# Set up DVWA or any other vulnerable web app
RUN git clone https://github.com/digininja/DVWA /var/www/html && \
chmod -R 755 /var/www/html &&
```

```
chown -R www-data:www-data /var/www/html  
# Start Apache automatically  
RUN echo "service apache2 start" >> /root/.bashrc  
EXPOSE 8080
```

Step 3: Build the Docker Image

```
docker build -t vuln-gui-app .
```

Step 4: Run the Container

```
docker run -d -p 8080:80 -p 6080:80 vuln-gui-app
```

- Access GUI via browser: <http://localhost:6080>
- Access DVWA web app via: <http://localhost:8080>
- Intercept with Burp by routing browser traffic through Burp's proxy.



What You Can Do

- Launch a Linux desktop GUI in your browser.
- Launch Firefox inside container, browse your vulnerable app.
- Route traffic through Burp Suite (configure Firefox proxy to 127.0.0.1:8080).
- Perform attacks (SQLi, XSS, etc.) visually.

! Notes

- The ISO itself isn't directly used in this method — instead, you build a vulnerable GUI system in Docker, similar to what the ISO contains.
- You can mimic an ISO system by copying its files or setting up the same services inside the Docker image.
- If you absolutely need to convert an ISO directly, you'll need to extract its filesystem and run it with VNC manually configured — more fragile and complex.

Do You Want:

- A prebuilt GUI + vulnerable system image to try right away?
- Instructions on using your ISO directly inside a VNC-enabled Docker container?
- Help setting up DVWA/bWAPP/Metasploitable-like environments visually?

Let me know your ISO name or send it over, and I'll tailor it exactly for you.