

Installing and configuring SonarQube with an Nginx reverse proxy on an AWS EC2 Ubuntu instance

SonarQube is an open-source platform used for continuous inspection of code quality to perform static code analysis and identify bugs, vulnerabilities, and code smells. Nginx can be configured as a reverse proxy to provide a public-facing URL for SonarQube, which typically runs on a non-standard port.

Nginx is an open-source, high-performance web server that functions as a reverse proxy, load balancer, and content cache. It is designed to handle large volumes of traffic and many simultaneous connections, serving static content, improving application performance by caching and managing requests, and enhancing security by acting as a barrier between clients and backend servers

Steps to follow:

1. Prepare the AWS EC2 Ubuntu Instance:

- Launch an EC2 instance with an Ubuntu Server AMI (e.g., Ubuntu Server 22.04 LTS).
- Ensure the instance type meets SonarQube's memory requirements (at least 2GB RAM for small instances).
- Configure the security group to allow inbound traffic on port 22 (SSH), 9000 (SonarQube direct access, if needed for initial setup), 80 (HTTP), and 443 (HTTPS for Nginx).
- Connect to the instance via SSH.

2. Install Prerequisites on Ubuntu:

- Update package lists.
- Install Java (OpenJDK 17 or higher is recommended for recent SonarQube versions):
- Install PostgreSQL (SonarQube requires a database).
- Create a PostgreSQL user and database for SonarQube:
- Modify Kernel System Limits for SonarQube.
 - Increase file descriptor limits.

3. Install SonarQube:

- Create a dedicated user for SonarQube.
- Download the latest SonarQube LTS version from the official website.
- Configure SonarQube to connect to PostgreSQL.
- Create a systemd service for SonarQube.
- Enable and start SonarQube.

4. Configure Nginx Reverse Proxy:

- Install Nginx.
- Add the DNS records
- Create a new Nginx configuration file for SonarQube
- Enable the Nginx site and restart Nginx
- Make sure your configuration file has no syntax errors
- Enable the Nginx site and restart Nginx

5. Secure with SSL using Certbot:

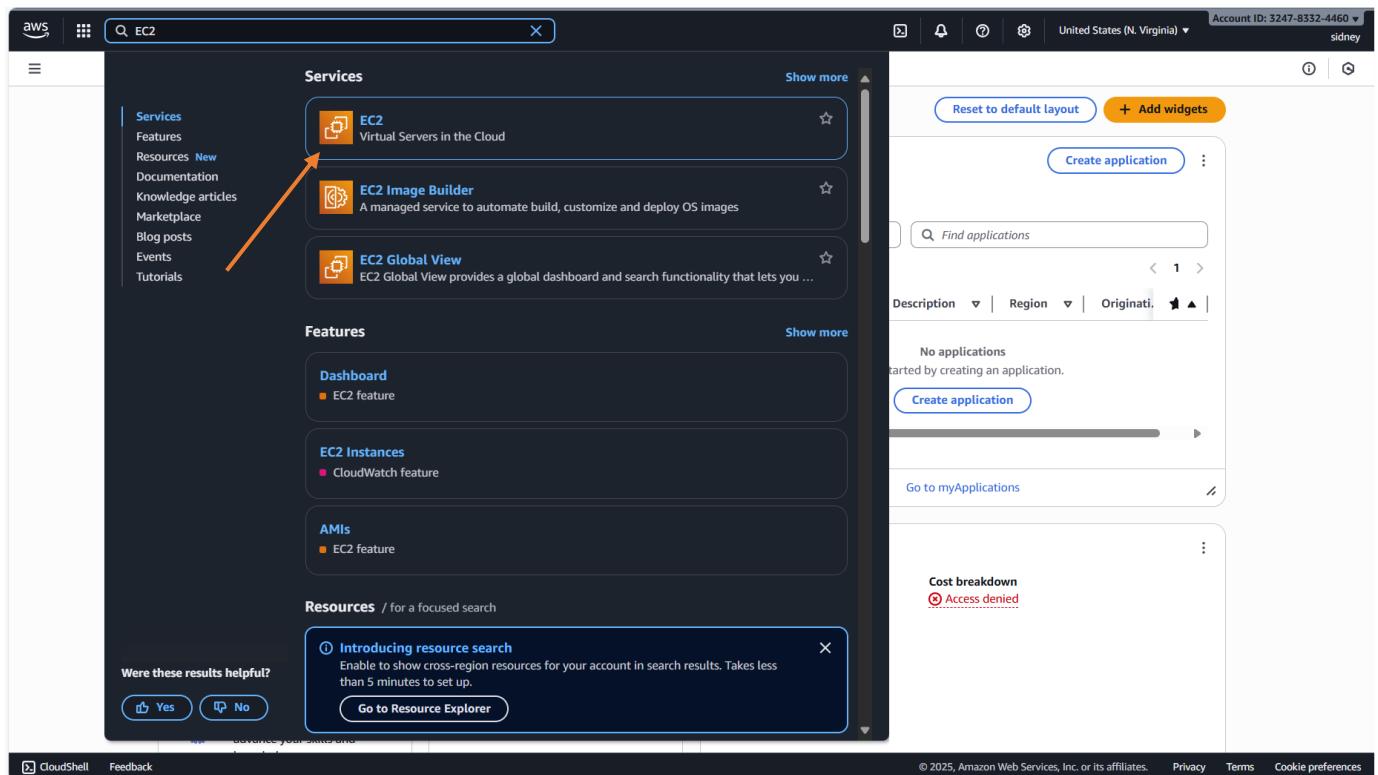
- Install Certbot
- Obtain and install an SSL certificate on the subdomain
- Verify if Certificate deployed successfully

STEP 1: Prepare the AWS EC2 Ubuntu Instance

We will launch an ubuntu EC2 instance, enable ports 22 for SSH, port 9000 for SonarQube, port 80 for HTTP and port 43 for HTTPS and later on SSH connect to it to install our dependencies.

PART 1: Launch an Ubuntu EC2 instance

Go to AWS Management console and search for “EC2” Instance.



Click on “EC2”

Click on “Launch Instance”

We will call the instance “sonar-server”

The screenshot shows the AWS EC2 "Launch an instance" wizard. In the "Name and tags" step, a "Name" field contains "jenkins-server". A "Add additional tags" link is visible.

Scroll down to “AMI”, select “ubuntu”

The screenshot shows the "Application and OS Images (Amazon Machine Image)" section. Under the "Quick Start" tab, the "Ubuntu" icon is selected. A search bar at the top says "Search our full catalog including 1000s of application and OS images". To the right, there's a "Browse more AMIs" button with a magnifying glass icon. Below the tabs, a list of AMI categories includes Amazon Linux, macOS, Ubuntu, Windows, Red Hat, SUSE Linux, and Debian. The Ubuntu entry shows details: "Ubuntu Server 24.04 LTS (HVM), SSD Volume Type", "ami-0360c520857e3138f (64-bit (x86)) / ami-026fcdd88446aa0bf (64-bit (Arm))", "Virtualization: hvm", "ENA enabled: true", "Root device type: ebs", and "Free tier eligible".

Scroll down to “Instance Type” and select “t2.medium”

The screenshot shows the "Instance type" selection step. The "t2.micro" instance type is selected. Details for t2.micro include: Family: t2, 1 vCPU, 1 GiB Memory, Current generation: true, On-Demand Windows base pricing: 0.0162 USD per Hour, On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour, On-Demand SUSE base pricing: 0.0116 USD per Hour, On-Demand RHEL base pricing: 0.026 USD per Hour, and On-Demand Linux base pricing: 0.0116 USD per Hour. A note states "Additional costs apply for AMIs with pre-installed software". There are also "Free tier eligible", "All generations", and "Compare instance types" buttons.

Scroll down to “Key Pair”

▼ 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**

Select

[Create new key pair](#)

Click on “Create new key pair”

Learn more'."/>

Create key pair

Key pair name
Key pairs allow you to connect to your instance securely.

Enter key pair name

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

RSA RSA encrypted private and public key pair

ED25519 ED25519 encrypted private and public key pair

Private key file format

.pem For use with OpenSSH

.ppk For use with PuTTY

⚠ When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. [Learn more](#)

Cancel [Create key pair](#)

Give the key pair a name, I will call it “sonar-key”

Create key pair

Key pair name
Key pairs allow you to connect to your instance securely.

sonar-key

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

RSA RSA encrypted private and public key pair

ED25519 ED25519 encrypted private and public key pair

Private key file format

.pem For use with OpenSSH

.ppk For use with PuTTY

⚠ When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. [Learn more](#)

Cancel [Create key pair](#)

Click on “Create key pair”

▼ 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**

sonar-key

[Create new key pair](#)

Scroll down to “**Network Settings**”. Select “Allow HTTP traffic from the internet”, “Allow HTTP traffic from the internet” and “Allow SSH traffic from”.

▼ Network settings [Info](#)

[Edit](#)

Network [Info](#)

vpc-0128e9209eaef1c37

Subnet [Info](#)

No preference (Default subnet in any availability zone)

Auto-assign public IP [Info](#)

Enable

Additional charges apply when outside of free tier allowance

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

We'll create a new security group called 'launch-wizard-5' with the following rules:

Allow SSH traffic from

Helps you connect to your instance

Anywhere

▼

0.0.0.0/0

Allow HTTPS traffic from the internet

To set up an endpoint, for example when creating a web server

Allow HTTP traffic from the internet

To set up an endpoint, for example when creating a web server

⚠️ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

X

Scroll down to the end

▼ Configure storage [Info](#)

[Advanced](#)

1x 8 GiB gp3 Root volume, 3000 IOPS, Not encrypted

ⓘ Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage

X

[Add new volume](#)

The selected AMI contains instance store volumes, however the instance does not allow any instance store volumes. None of the instance store volumes from the AMI will be accessible from the instance

ⓘ Click refresh to view backup information

The tags that you assign determine whether the instance will be backed up by any Data Lifecycle Manager policies.

C

0 x File systems

[Edit](#)

► Advanced details [Info](#)

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

ⓘ Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included as part of the free tier allowance.

Cancel

Launch instance

[Preview code](#)

Click on “**Launch Instance**”

A screenshot of the AWS EC2 Instances launch success page. At the top, there's a navigation bar with 'EC2 > Instances > Launch an instance'. A red arrow points from the text 'Click on "Instances"' in the previous slide to the 'Instances' link in the navigation. Below the navigation is a green success message box containing the text 'Successfully initiated launch of instance (i-093e6e5bae1cd3540)'. Underneath the message is a 'Launch log' button. The main content area is titled 'Next Steps' and contains eight cards:

- Create billing and free tier usage alerts**: To manage costs and avoid surprise bills, set up email notifications for billing and free tier usage thresholds. Includes a 'Create billing alerts' button.
- Connect to your instance**: Once your instance is running, log into it from your local computer. Includes a 'Connect to instance' button and a 'Learn more' link.
- Connect an RDS database**: Configure the connection between an EC2 instance and a database to allow traffic flow between them. Includes a 'Connect an RDS database' button and a 'Create a new RDS database' link.
- Create EBS snapshot policy**: Create a policy that automates the creation, retention, and deletion of EBS snapshots. Includes a 'Create EBS snapshot policy' button.
- Manage detailed monitoring**: Enable or disable detailed monitoring for the instance. If you enable detailed monitoring, the Amazon EC2 console displays monitoring graphs with a 1-minute period. Includes a 'Manage detailed monitoring' button.
- Create Load Balancer**: Create a application, network gateway or classic Elastic Load Balancer. Includes a 'Create Load Balancer' button.
- Create AWS budget**: AWS Budgets allows you to create budgets, forecast spend, and take action on your costs and usage from a single location. Includes a 'Create AWS budget' button.
- Manage CloudWatch alarms**: Create or update Amazon CloudWatch alarms for the instance. Includes a 'Manage CloudWatch alarms' button.

At the bottom of the page, there are links for 'CloudShell', 'Feedback', and copyright information: '© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences'.

Click on “Instances”

A screenshot of the AWS EC2 Instances list page. On the left is a navigation sidebar with 'EC2' selected, followed by 'Instances', 'Images', 'Elastic Block Store', and 'Network & Security'. A red arrow points from the text 'Click on "Instances"' in the previous slide to the 'Instances' link in the sidebar. The main content area shows a table titled 'Instances (1) Info' with one row:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
sonar-server	i-03386f8b488c0086d	Running	t2.micro	Initializing	View alarms +	us-east-1d

The 'Status check' column for the instance shows a circular icon with a question mark and the text 'Initializing'. At the bottom of the page, there are links for 'CloudShell', 'Feedback', and copyright information: '© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences'.

You can see that our just created instance is initializing. Wait for it to pass the “2/2 check”

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with sections like EC2, Dashboard, EC2 Global View, Events, Instances (selected), 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, and Network Interfaces. The main content area is titled "Instances (1) Info". It shows a table with one row for the instance "sonar-server". The columns include Name (sonar-server), Instance ID (i-03386f8b488c0086d), Instance state (Running), Instance type (t2.micro), Status check (2/2 checks passed), Alarm status (View alarms +), and Availability Zone (us-east-1d). A red arrow points to the "2/2 checks passed" status check link.

The instance has passed the “2/2 check”

PART 2: Add Port 9000 for SonarQube

We will add port 9000 that will be used to access SonarQube through the browser. Now, we have to add port 9000 to enable access SonarQube from our browser. Go to our EC2 instance

This screenshot is identical to the one above, showing the AWS EC2 Instances page with the same instance details. The "2/2 checks passed" status check link is again highlighted with a red arrow.

Select the EC2 instance

This screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with sections like EC2, Instances, Images, and Network & Security. The main area displays a table of instances. A red arrow points to the 'Security' tab in the instance details header. The instance listed is 'i-03386f8b488c0086d (sonar-server)'. The security section shows details such as Public IPv4 address (18.208.247.55), Instance state (Running), and IAM Role (None). It also lists security groups (sg-0fc... launch-wizard-11) and inbound rules for ports 443, 22, and 80.

Click on the “Security” tab

This screenshot shows the same AWS EC2 Instances page as the previous one, but with a red arrow pointing to the 'Security groups' link under the 'Security details' section. This link leads to a detailed view of the security group 'sg-0fc... launch-wizard-11'.

Click on the “Security Group” url

sg-0fc6cfee12d64c86 - launch-wizard-11

Inbound rules (3)

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source
-	sgr-02cebe2cbcba6c711	IPv4	HTTPS	TCP	443	0.0.0.0/0
-	sgr-004e5cad248863dca	IPv4	SSH	TCP	22	0.0.0.0/0
-	sgr-0435312b1bf37ae0	IPv4	HTTP	TCP	80	0.0.0.0/0

Click on “Edit Inbound rules”

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-02cebe2cbcba6c711	HTTPS	TCP	443	Custom	0.0.0.0/0
sgr-004e5cad248863dca	SSH	TCP	22	Custom	0.0.0.0/0
sgr-0435312b1bf37ae0	HTTP	TCP	80	Custom	0.0.0.0/0

Add rule

Cancel **Preview changes** **Save rules**

Click on “Add Rule”

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-02cebe2cbcba6c711	HTTPS	TCP	443	Custom	0.0.0.0/0
sgr-004e5cad248863dca	SSH	TCP	22	Custom	0.0.0.0/0
sgr-0435312b1bf37ae0	HTTP	TCP	80	Custom	0.0.0.0/0
-	Custom TCP	TCP	0	Custom	0.0.0.0/0

Add rule

Cancel **Preview changes** **Save rules**

Click on the drop down on “**Port Range**” and enter the value “**9000**”

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-02cebe2cbcba6c711	HTTPS	TCP	443	Custom	0.0.0.0/0
sgr-004e5cad248863dca	SSH	TCP	22	Custom	0.0.0.0/0
sgr-0435312b1bf37ae0	HTTP	TCP	80	Custom	0.0.0.0/0
-	Custom TCP	TCP	9000	Custom	0.0.0.0/0

Add rule

Cancel **Preview changes** **Save rules**

Click on the drop down on “**source**” and select “**Anywhere-IPv4**”

aws Search [Alt+S] Account ID: 3247-8332-4460 sidney

☰ EC2 > Security Groups > sg-0fcfa6cfee12d64c86 - launch-wizard-11 > Edit inbound rules

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Security group rule ID	Type	Protocol	Port range	Source	Description - optional	Info	Delete
sgr-02cebe2cbcba6c711	HTTPS	TCP	443	Custom	Q	0.0.0.0/0 X	<button>Delete</button>
sgr-004e5cad248863dca	SSH	TCP	22	Custom	Q	0.0.0.0/0 X	<button>Delete</button>
sgr-0435312b1bf37ae0	HTTP	TCP	80	Custom	Q	0.0.0.0/0 X	<button>Delete</button>
-	Custom TCP	TCP	9000	Anywhe...	Q	0.0.0.0/0 X	<button>Delete</button>

[Add rule](#)

[Cancel](#) [Preview changes](#) [Save rules](#)

Click on “Save rules”

aws Search [Alt+S] Account ID: 3247-8332-4460 sidney

☰ EC2 > Security Groups > sg-0fcfa6cfee12d64c86 - launch-wizard-11

ⓘ Inbound security group rules successfully modified on security group (sg-0fcfa6cfee12d64c86 | launch-wizard-11) Actions

sg-0fcfa6cfee12d64c86 - launch-wizard-11

Details

Security group name	Security group ID	Description	VPC ID
launch-wizard-11	sg-0fcfa6cfee12d64c86	launch-wizard-11 created 2025-09-06T23:03:28.836Z	vpc-0128e9209eaef1c37
Owner	Inbound rules count	Outbound rules count	
524783324460	4 Permission entries	1 Permission entry	

[Inbound rules](#) [Outbound rules](#) [Sharing - new](#) [VPC associations - new](#) [Tags](#)

Inbound rules (4)

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source
-	sgr-02cebe2cbcba6c711	IPv4	HTTPS	TCP	443	0.0.0.0/0
-	sgr-004e5cad248863dca	IPv4	SSH	TCP	22	0.0.0.0/0
-	sgr-0435312b1bf37ae0	IPv4	HTTP	TCP	80	0.0.0.0/0
-	sgr-02aff83fa1a9b9a8a	IPv4	Custom TCP	TCP	9000	0.0.0.0/0

[Manage tags](#) [Edit inbound rules](#)

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

Port 9000 has been added.

PART 4: SSH Connect to the EC2 instance

Now, let us connect to our EC2 instance through SSH

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with various navigation options like Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, etc. The main area shows a table titled 'Instances (1) Info' with one row. The instance is named 'sonar-server' with an ID of 'i-03386f8b488c0086d'. It is listed as 'Running' with an 'Actions' button. Below the table, a section titled 'Select an instance' is visible.

Select the instance

This screenshot is similar to the previous one but focuses on the 'Connect' button. The 'sonar-server' instance is selected, and the 'Connect' button is highlighted with a large orange arrow. The rest of the interface is identical to the first screenshot.

Click on “Connect”

The screenshot shows the AWS EC2 Connect interface. The top navigation bar includes the AWS logo, a search bar, and account information (Account ID: 3247-8332-4460, sidney). Below the navigation is a breadcrumb trail: EC2 > Instances > i-03386f8b48c0086d > Connect to instance. The main content area has a title 'Connect info' and a sub-section 'Connect to an instance using the browser-based client.' There are four tabs: EC2 Instance Connect, Session Manager, **SSH client**, and EC2 serial console. Under the SSH client tab, there's a section for 'Instance ID' (i-03386f8b48c0086d sonar-server) with a copy link. Below it are numbered steps: 1. Open an SSH client, 2. Locate your private key file. The key used to launch this instance is sonar-key.pem, 3. Run this command, if necessary, to ensure your key is not publicly viewable. (chmod 400 'sonar-key.pem'), 4. Connect to your instance using its Public DNS: ec2-18-208-247-55.compute-1.amazonaws.com. An example command is provided: ssh -i "sonar-key.pem" ubuntu@ec2-18-208-247-55.compute-1.amazonaws.com. A note at the bottom states: '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.' A red arrow points to the copy link for the command.

Copy this command and open PowerShell

```
ssh -i "sonar-key.pem" ubuntu@ec2-18-208-247-55.compute-1.amazonaws.com
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\ebots> |
```

Navigate to the “Downloads” folder where our “.pem” file is stored by using the command:

```
cd Downloads
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\ebots> cd Downloads
PS C:\Users\ebots\Downloads> |
```

Then run the copied command:

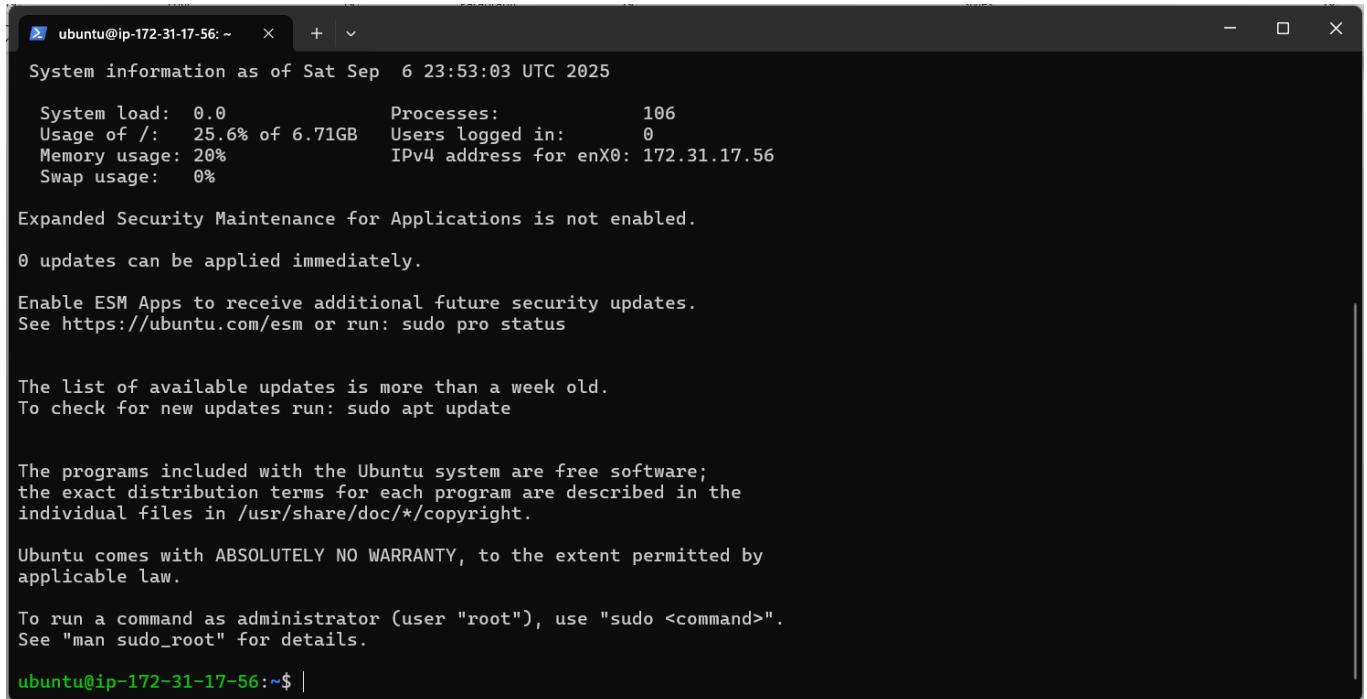
```
ssh -i "sonar-key.pem" ubuntu@ec2-18-208-247-55.compute-1.amazonaws.com
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\ebots> cd Downloads
PS C:\Users\ebots\Downloads> ssh -i "sonar-key.pem" ubuntu@ec2-18-208-247-55.compute-1.amazonaws.com
The authenticity of host 'ec2-18-208-247-55.compute-1.amazonaws.com (18.208.247.55)' can't be established.
ED25519 key fingerprint is SHA256:UMgvEHfY5MZHNBwu6WEGYktQCmoiswCwz0UktEnUjw.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? |
```

Type “yes” and press ENTER



```
ubuntu@ip-172-31-17-56: ~ + v
System information as of Sat Sep 6 23:53:03 UTC 2025
System load: 0.0 Processes: 106
Usage of /: 25.6% of 6.71GB Users logged in: 0
Memory usage: 20% IPv4 address for enX0: 172.31.17.56
Swap usage: 0%
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

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

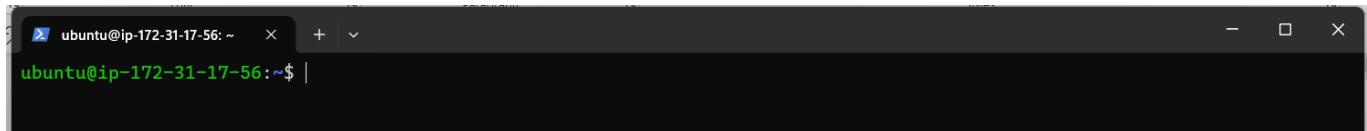
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-17-56:~$ |
```

We have successfully SSH connected to our EC2 instance

Clear the terminal using the command:

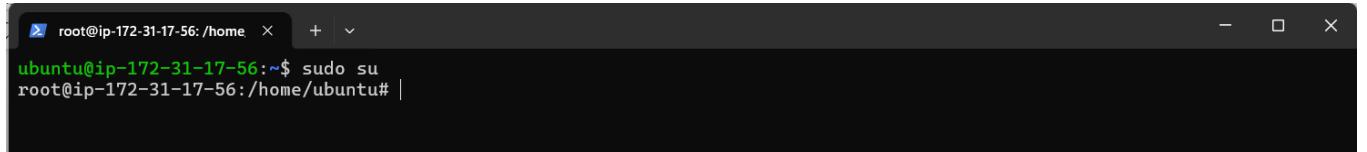
```
clear
```



```
ubuntu@ip-172-31-17-56: ~ + v
ubuntu@ip-172-31-17-56:~$ |
```

Grant Root user access using the command:

```
sudo su
```



```
root@ip-172-31-17-56: /home ~ + v
ubuntu@ip-172-31-17-56:~$ sudo su
root@ip-172-31-17-56:/home/ubuntu# |
```

STEP 2: Install Prerequisites on Ubuntu:

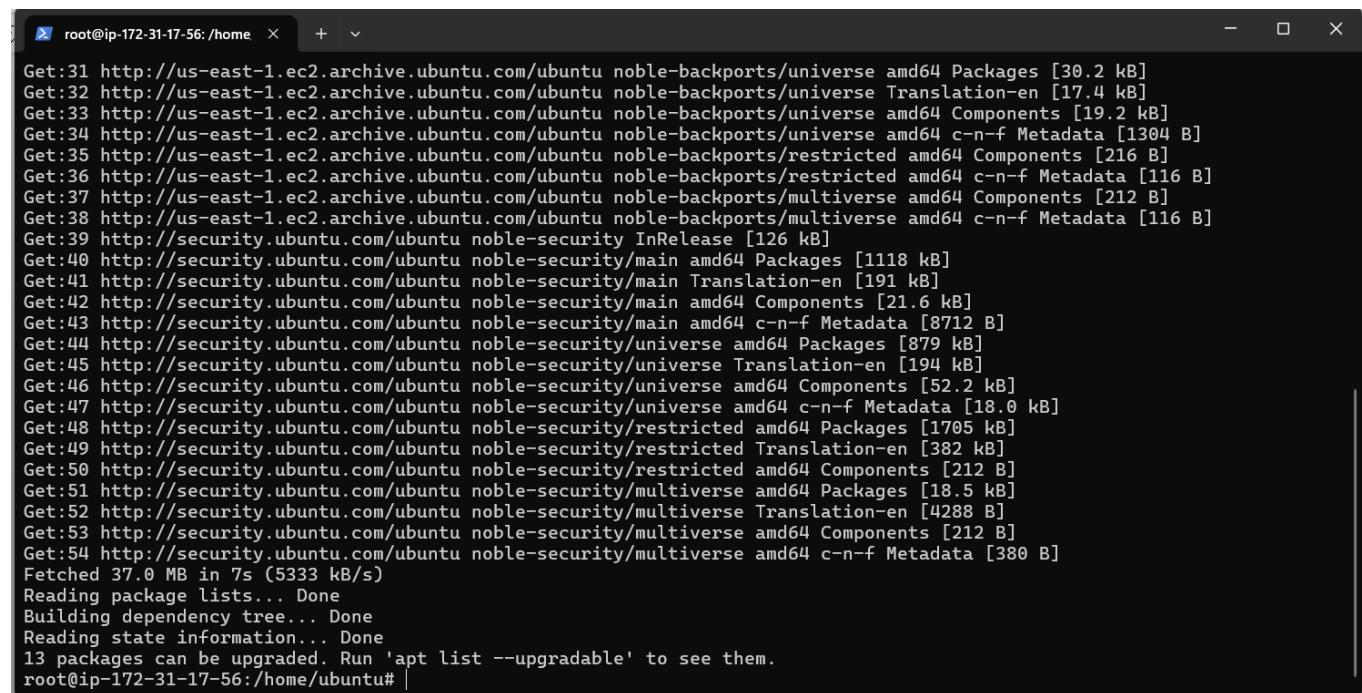
Here we will Update package lists, Install Java (OpenJDK 17 or higher is recommended for recent SonarQube versions), Install PostgreSQL (SonarQube requires a database), Create a PostgreSQL user and database for SonarQube, adjust kernel parameters for SonarQube, and Increase file descriptor limits.

Part 1: Update package lists

We will update the package and upgrade the package list here

Update packages on server

```
sudo apt update
```



The screenshot shows a terminal window with the title bar "root@ip-172-31-17-56: /home". The window contains the output of the "sudo apt update" command. The output shows various HTTP requests being made to the Ubuntu archive servers to fetch package lists and metadata for different components like universe, restricted, and multiverse. It includes details like the URL, component, file type (e.g., Packages, Translation-en, Metadata), and size. The process is shown to have fetched 37.0 MB in 7 seconds at 5333 kB/s. It then moves on to building the dependency tree, reading state information, and finding 13 upgradable packages.

```
Get:31 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [30.2 kB]
Get:32 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [17.4 kB]
Get:33 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [19.2 kB]
Get:34 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1304 B]
Get:35 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:36 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:37 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:38 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Get:39 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:40 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [1118 kB]
Get:41 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [191 kB]
Get:42 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [21.6 kB]
Get:43 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [8712 B]
Get:44 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [879 kB]
Get:45 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [194 kB]
Get:46 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [52.2 kB]
Get:47 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [18.0 kB]
Get:48 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [1705 kB]
Get:49 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [382 kB]
Get:50 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:51 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [18.5 kB]
Get:52 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [4288 B]
Get:53 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]
Get:54 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [380 B]
Fetched 37.0 MB in 7s (5333 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
13 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ip-172-31-17-56:/home/ubuntu# |
```

Update the server

```
sudo apt upgrade -y
```

```
root@ip-172-31-17-56:/home ~ + | ^

Found initrd image: /boot/microcode.cpio /boot/initrd.img-6.14.0-1011-aws
Warning: os-prober will not be executed to detect other bootable partitions.
Systems on them will not be added to the GRUB boot configuration.
Check GRUB_DISABLE_OS_PROBER documentation entry.
Adding boot menu entry for UEFI Firmware Settings ...
done
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
6.14.0-1011-aws
Diagnostics:
The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu#
```

Part 2: Install OpenJDK 17

Install OpenJDK 17 (needed for the latest version of SonarQube (version 10.0)).

```
sudo apt install -y openjdk-17-jdk
```

```
root@ip-172-31-17-56:/home ~ + | ^

Setting up openjdk-17-jre:amd64 (17.0.16+8~us1-0ubuntu1~24.04.1) ...
Setting up openjdk-17-jdk:amd64 (17.0.16+8~us1-0ubuntu1~24.04.1) ...
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jconsole to provide /usr/bin/jconsole (jconsole) in auto mode
Processing triggers for libc-bin (2.39-0ubuntu8.5) ...
Processing triggers for libgdk-pixbuf-2.0-0:amd64 (2.42.10+dfsg-3ubuntu3.2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
6.14.0-1011-aws
Diagnostics:
The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu#
```

Let's check the installed version of Java. VALIDATION IS IMPORTANT.

```
java -version
```

```
root@ip-172-31-17-56:/home ~ + - X
Processing triggers for libc-bin (2.39-0ubuntu8.5) ...
Processing triggers for libgdk-pixbuf-2.0-0:amd64 (2.42.10+dfsg-3ubuntu3.2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
 6.14.0-1011-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
  systemctl restart networkd-dispatcher.service
  systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# java -version
openjdk version "17.0.16" 2025-07-15
OpenJDK Runtime Environment (build 17.0.16+8-Ubuntu-0ubuntu124.04.1)
OpenJDK 64-Bit Server VM (build 17.0.16+8-Ubuntu-0ubuntu124.04.1, mixed mode, sharing)
root@ip-172-31-17-56:/home/ubuntu# |
```

Part 3: Install and Configure PostgreSQL

Add the PostgreSQL repository

```
sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb_release -cs`-pgdg main" /etc/apt/sources.list.d/pgdg.list'
```

```
root@ip-172-31-17-56:/home/ubuntu ~ + - X
root@ip-172-31-17-56:/home/ubuntu# sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb_release -cs`-pgdg main" /etc/apt/sources.list.d/pgdg.list'
deb http://apt.postgresql.org/pub/repos/apt/ noble-pgdg main /etc/apt/sources.list.d/pgdg.list
root@ip-172-31-17-56:/home/ubuntu# |
```

Add the PostgreSQL signing key

```
wget -q https://www.postgresql.org/media/keys/ACCC4CF8.asc -O - | sudo apt-key add -
```

```
root@ip-172-31-17-56:/home/ubuntu ~ + - X
root@ip-172-31-17-56:/home/ubuntu# sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb_release -cs`-pgdg main" /etc/apt/sources.list.d/pgdg.list'
deb http://apt.postgresql.org/pub/repos/apt/ noble-pgdg main /etc/apt/sources.list.d/pgdg.list
root@ip-172-31-17-56:/home/ubuntu# wget -q https://www.postgresql.org/media/keys/ACCC4CF8.asc -O - | sudo apt-key add -
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
root@ip-172-31-17-56:/home/ubuntu# |
```

Install PostgreSQL

```
sudo apt install postgresql postgresql-contrib -y
```

```
root@ip-172-31-17-56:/home ~ + | 
performing post-bootstrap initialization ... ok
syncing data to disk ... ok
Setting up postgresql-contrib (16+257build1.1) ...
Setting up postgresql (16+257build1.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.5) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
6.14.0-1011-aws
Diagnostics:
The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# |
```

Enable the database server to start automatically on reboot

```
sudo systemctl enable postgresql
```

```
root@ip-172-31-17-56:/home ~ + | 
Setting up postgresql (16+257build1.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.5) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
6.14.0-1011-aws
Diagnostics:
The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# |
```

Start the database server

```
sudo systemctl start postgresql
```

```
root@ip-172-31-17-56:/home × + ▾
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.5) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
6.14.0-1011-aws
Diagnostics:
The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# |
```

Check the status of the database server

```
sudo systemctl status postgresql
```

```
root@ip-172-31-17-56:/home × + ▾
Diagnostics:
The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
   Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
     Main PID: 9879 (code=exited, status=0/SUCCESS)
        CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# |
```

Let's check the installed version of the install Postgres DB. VALIDATION IS IMPORTANT

```
psql --version
```

```
root@ip-172-31-17-56:/home ~ + - x
Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.
Restarting services...
Service restarts being deferred:
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service
No containers need to be restarted.
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
   Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
     Main PID: 9879 (code=exited, status=0/SUCCESS)
       CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# |
```

Switch to the Postgres user

```
sudo -i -u postgres
```

```
root@ip-172-31-17-56:/home ~ + - x
Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.
Restarting services...
Service restarts being deferred:
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service
No containers need to be restarted.
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
   Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
     Main PID: 9879 (code=exited, status=0/SUCCESS)
       CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:~$ |
```

Part 4: Create a database user named ddsonar

Note: You can provide the name of your own but make a note of it, as we will be needing this name in further steps.

```
createuser ddsonar
```

```
root@ip-172-31-17-56:/home ~ + - X
Restarting services...
Service restarts being deferred:
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
    Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
    Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
      Main PID: 9879 (code=exited, status=0/SUCCESS)
        CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:~$ createuser ddsonar
postgres@ip-172-31-17-56:~$ |
```

Log in to PostgreSQL

psql

```
root@ip-172-31-17-56:/home ~ + - X
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
    Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
    Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
      Main PID: 9879 (code=exited, status=0/SUCCESS)
        CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:~$ createuser ddsonar
postgres@ip-172-31-17-56:~$ psql
psql (16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))
Type "help" for help.

postgres=# |
```

Set a password for the ddsonar user. Use a strong password in place of my_strong_password.

```
ALTER USER [Created_user_name] WITH ENCRYPTED password 'my_strong_password';
```

For example (In my case it will be):

```
ALTER USER ddsonar WITH ENCRYPTED password 'myfirstsonar77!';
```

```

root@ip-172-31-17-56:/home ~ + - X
No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
    Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
      Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
        Main PID: 9879 (code=exited, status=0/SUCCESS)
          CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:~$ createuser ddsonar
postgres@ip-172-31-17-56:~$ psql
psql (16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))
Type "help" for help.

postgres=# ALTER USER ddsonar WITH ENCRYPTED password 'myfirstsonar77!';
ALTER ROLE
postgres=# |

```

Create a SonarQube database and set the owner to ddsonar.

```
CREATE DATABASE [database_name] OWNER [Created_user_name];
```

For example (In our case it will be) — feel free to give an awesome database name as per your requirement:

```
CREATE DATABASE ddsonarqube OWNER ddsonar;
```

```

root@ip-172-31-17-56:/home ~ + - X
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
    Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
      Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
        Main PID: 9879 (code=exited, status=0/SUCCESS)
          CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:~$ createuser ddsonar
postgres@ip-172-31-17-56:~$ psql
psql (16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))
Type "help" for help.

postgres=# ALTER USER ddsonar WITH ENCRYPTED password 'myfirstsonar77!';
ALTER ROLE
postgres=# CREATE DATABASE ddsonarqube OWNER ddsonar;
CREATE DATABASE
postgres=# |

```

Grant all the privileges on the ddsonarqube database to the ddsonar user.

```
GRANT ALL PRIVILEGES ON DATABASE ddsonarqube to ddsonar;
```

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/system
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
    Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
      Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
        Main PID: 9879 (code=exited, status=0/SUCCESS)
           CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS...
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:~$ createuser ddsonar
postgres@ip-172-31-17-56:~$ psql
psql (16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))
Type "help" for help.

postgres=# ALTER USER ddsonar WITH ENCRYPTED password 'myfirstsonar77!';
ALTER ROLE
postgres=# CREATE DATABASE ddsonarqube OWNER ddsonar;
CREATE DATABASE
postgres=# GRANT ALL PRIVILEGES ON DATABASE ddsonarqube to ddsonar;
GRANT
postgres=# |
```

Let's check the created user and the database.

a) To check the created database using the command:

\ 1

Press “q”

```

root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable postgresql
Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start postgresql
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
   Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
     Main PID: 9879 (code=exited, status=0/SUCCESS)
        CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:~$ createuser ddsonar
postgres@ip-172-31-17-56:~$ psql
psql (16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))
Type "help" for help.

postgres=# ALTER USER ddsonar WITH ENCRYPTED password 'myfirstsonar77!';
ALTER ROLE
postgres=# CREATE DATABASE ddsonarqube OWNER ddsonar;
CREATE DATABASE
postgres=# GRANT ALL PRIVILEGES ON DATABASE ddsonarqube to ddsonar;
GRANT
postgres=# \l
postgres-# \l
postgres-#

```

b) To check the created database user

\du

```

root@ip-172-31-17-56:/home# + v
Active: active (exited) since Sun 2025-09-07 00:02:17 UTC; 1min 46s ago
Main PID: 9879 (code=exited, status=0/SUCCESS)
CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:~$ createuser ddsonar
postgres@ip-172-31-17-56:~$ psql
psql (16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))
Type "help" for help.

postgres=# ALTER USER ddsonar WITH ENCRYPTED password 'myfirstsonar77!';
ALTER ROLE
postgres=# CREATE DATABASE ddsonarqube OWNER ddsonar;
CREATE DATABASE
postgres=# GRANT ALL PRIVILEGES ON DATABASE ddsonarqube to ddsonar;
GRANT
postgres=# \l
postgres-# \l
postgres-# \du
      List of roles
Role name |          Attributes
-----+-----
ddsonar  |
postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS
postgres-|

```

Exit PostgreSQL.

\q

```

Main PID: 9879 (code=exited, status=0/SUCCESS)
CPU: 1ms

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:~$ createuser ddsonar
postgres@ip-172-31-17-56:~$ psql
psql (16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))
Type "help" for help.

postgres=# ALTER USER ddsonar WITH ENCRYPTED password 'myfirstsonar77!';
ALTER ROLE
postgres=# CREATE DATABASE ddsonarqube OWNER ddsonar;
CREATE DATABASE
postgres=# GRANT ALL PRIVILEGES ON DATABASE ddsonarqube to ddsonar;
GRANT
postgres=# \l
postgres=# \l
postgres=# \du
          List of roles
Role name | Attributes
-----+-----
ddsonar  |
postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS

postgres=# \q
postgres@ip-172-31-17-56:~$ |

```

Return to your non-root sudo user account.

exit

```

Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Sep 07 00:02:17 ip-172-31-17-56 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
root@ip-172-31-17-56:/home/ubuntu# psql --version
psql (PostgreSQL) 16.9 (Ubuntu 16.9-0ubuntu0.24.04.1)
root@ip-172-31-17-56:/home/ubuntu# sudo -i -u postgres
postgres@ip-172-31-17-56:~$ createuser ddsonar
postgres@ip-172-31-17-56:~$ psql
psql (16.9 (Ubuntu 16.9-0ubuntu0.24.04.1))
Type "help" for help.

postgres=# ALTER USER ddsonar WITH ENCRYPTED password 'myfirstsonar77!';
ALTER ROLE
postgres=# CREATE DATABASE ddsonarqube OWNER ddsonar;
CREATE DATABASE
postgres=# GRANT ALL PRIVILEGES ON DATABASE ddsonarqube to ddsonar;
GRANT
postgres=# \l
postgres=# \l
postgres=# \du
          List of roles
Role name | Attributes
-----+-----
ddsonar  |
postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS

postgres=# \q
postgres@ip-172-31-17-56:~$ exit
logout
root@ip-172-31-17-56:/home/ubuntu# |

```

Part 5: Modify Kernel System Limits

SonarQube uses Elasticsearch to store its indices in an MMap FS directory. It requires some changes to the system defaults.

Edit the sysctl configuration file.

`sudo vim /etc/sysctl.conf`

```
[root@ip-172-31-17-56: /home] + - x
#
# /etc/sysctl.conf - Configuration file for setting system variables
# See /etc/sysctl.d/ for additional system variables.
# See sysctl.conf (5) for information.
#
#kernel.domainname = example.com

# Uncomment the following to stop low-level messages on console
#kernel.printk = 3 4 1 3

#####
# Functions previously found in netbase
#
# Uncomment the next two lines to enable Spoof protection (reverse-path filter)
# Turn on Source Address Verification in all interfaces to
# prevent some spoofing attacks
#net.ipv4.conf.default.rp_filter=1
#net.ipv4.conf.all.rp_filter=1

# Uncomment the next line to enable TCP/IP SYN cookies
# See http://lwn.net/Articles/277146/
# Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1

# Uncomment the next line to enable packet forwarding for IPv4
#net.ipv4.ip_forward=1

"/etc/sysctl.conf" 64L, 2209B
1,1
Top
```

Increase file descriptor limits

```
vm.max_map_count=262144
fs.file-max=65536
ulimit -n 65536
ulimit -u 4096
```

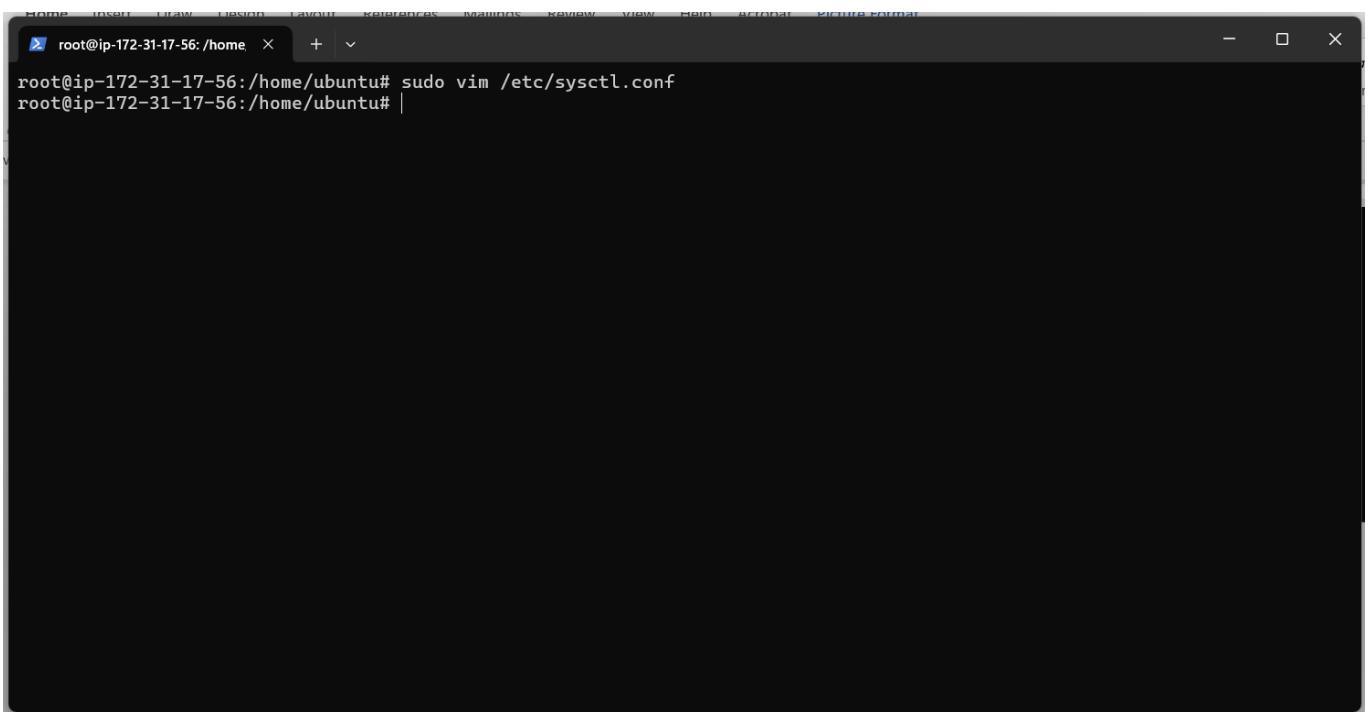
```
[root@ip-172-31-17-56: /home] + - x
#
# settings are disabled so review and enable them as needed.
#
# Do not accept ICMP redirects (prevent MITM attacks)
#net.ipv4.conf.all.accept_redirects = 0
#net.ipv4.conf.default.accept_redirects = 0
# _or_
# Accept ICMP redirects only for gateways listed in our default
# gateway list (enabled by default)
# net.ipv4.conf.all.secure_redirects = 1
#
# Do not send ICMP redirects (we are not a router)
#net.ipv4.conf.all.send_redirects = 0
#
# Log Martian Packets
#net.ipv4.conf.all.log_martians = 1
#
#####
# Magic system request Key
# 0=disable, 1=enable all, >1 bitmask of sysrq functions
# See https://www.kernel.org/doc/html/latest/admin-guide/sysrq.html
# for what other values do
#kernel.sysrq=438

|vm.max_map_count=262144
fs.file-max=65536
ulimit -n 65536
ulimit -u 4096

-- INSERT --
65,1
Bot
```

Save and exit the file

Press ESC, followed by :wq and press ENTER



A screenshot of a terminal window titled "root@ip-172-31-17-56:/home". The window has a dark background and white text. At the top, there is a menu bar with options like "HOME", "DESKTOP", "LAYOUT", "KINDESS", "MANUALS", "REVIEW", "FIELD", "ACTIONAL", and "PICTURE FORMAT". Below the menu, the terminal prompt shows "root@ip-172-31-17-56:/home/ubuntu#". The user then enters the command "sudo vim /etc/sysctl.conf". The command is completed, and the cursor is shown at the end of the command line.

```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/sysctl.conf
root@ip-172-31-17-56:/home/ubuntu# |
```

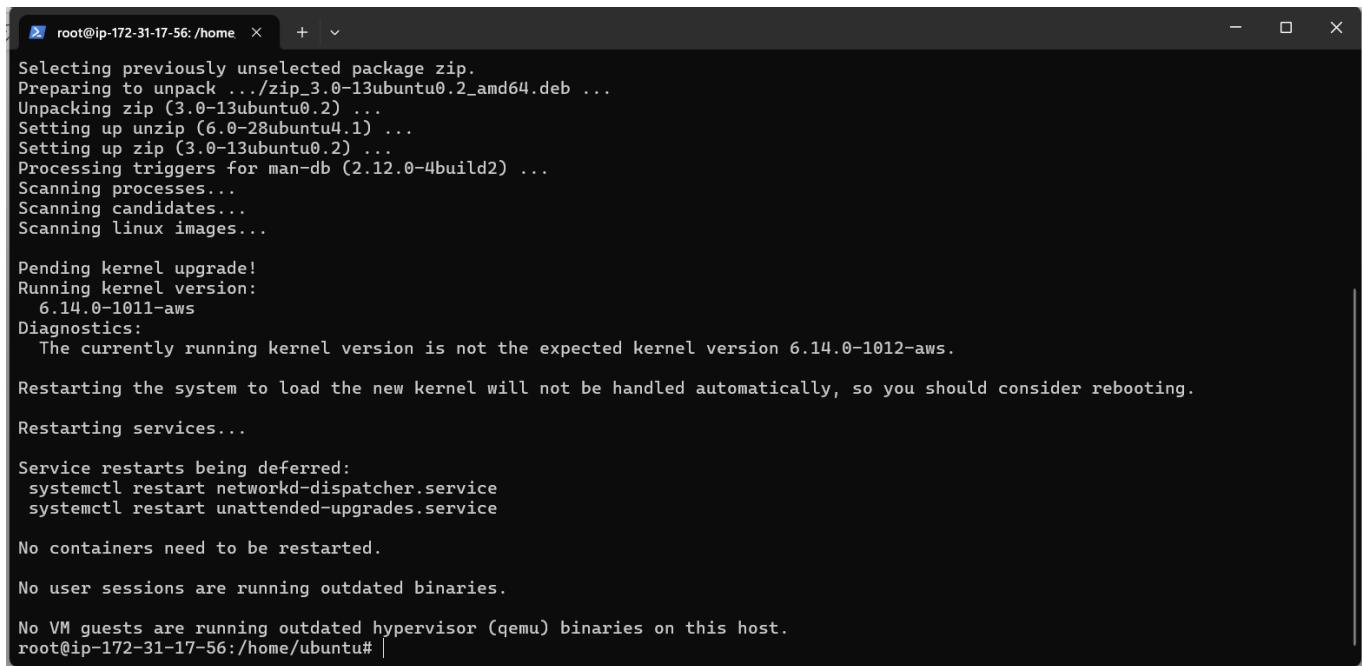
STEP 3: Install SonarQube

We will Download the latest SonarQube LTS version from the official website, Create a dedicated user for SonarQube, Configure SonarQube to connect to PostgreSQL, Create a systemd service for SonarQube, Enable and start SonarQube,

Part 1: Download and Install SonarQube

Install the zip utility, which is needed to unzip the SonarQube files.

```
sudo apt install zip -y
```



```
root@ip-172-31-17-56: /home root ~ + - x
Selecting previously unselected package zip.
Preparing to unpack .../zip_3.0-13ubuntu0.2_amd64.deb ...
Unpacking zip (3.0-13ubuntu0.2) ...
Setting up unzip (6.0-28ubuntu4.1) ...
Setting up zip (3.0-13ubuntu0.2) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
 6.14.0-1011-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
  systemctl restart networkd-dispatcher.service
  systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# |
```

Locate the latest download URL from the SonarQube official download page.

Download the SonarQube distribution files. (You can download the latest SonarQube distribution using the following link)

<https://www.sonarsource.com/products/sonarqube/downloads/>

Here we are installing the latest version of SonarQube 10.0 community edition (free one)

```
sudo wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-10.0.0.68432.zip
```

```

root@ip-172-31-17-56:/home ~ + - X
Diagnostics:
The currently running kernel version is not the expected kernel version 6.14.0-1012-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
systemctl restart networkd-dispatcher.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu# sudo wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-10.0.0.68432.zip
--2025-09-07 00:15:59-- https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-10.0.0.68432.zip
Resolving binaries.sonarsource.com (binaries.sonarsource.com)... 99.84.188.21, 99.84.188.106, 99.84.188.45, ...
Connecting to binaries.sonarsource.com (binaries.sonarsource.com)|99.84.188.21|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 352909963 (337M) [binary/octet-stream]
Saving to: 'sonarqube-10.0.0.68432.zip'

sonarqube-10.0.0.68432.zip      100%[=====] 336.56M   113MB/s    in 3.0s
2025-09-07 00:16:02 (113 MB/s) - 'sonarqube-10.0.0.68432.zip' saved [352909963/352909963]
root@ip-172-31-17-56:/home/ubuntu#

```

Unzip the downloaded file.

```
sudo unzip sonarqube-10.0.0.68432.zip
```

```

root@ip-172-31-17-56:/home ~ + - X
inflating: sonarqube-10.0.0.68432/web/images/SonarLint-connection-ok.png
inflating: sonarqube-10.0.0.68432/web/images/SonarLint-connection-request.png
inflating: sonarqube-10.0.0.68432/web/images/source-code.svg
inflating: sonarqube-10.0.0.68432/web/images/sq-sl.svg
creating: sonarqube-10.0.0.68432/web/images/tutorials/
inflating: sonarqube-10.0.0.68432/web/images/tutorials/azure-pipelines.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/commit.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/github-actions.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/jenkins.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/manual.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/refresh.svg
inflating: sonarqube-10.0.0.68432/web/index.html
creating: sonarqube-10.0.0.68432/web/js/
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js.map
inflating: sonarqube-10.0.0.68432/web/js/outPKBK1YFT.css
inflating: sonarqube-10.0.0.68432/web/js/outPKBK1YFT.css.map
inflating: sonarqube-10.0.0.68432/web/mstile-512x512.png
inflating: sonarqube-10.0.0.68432/web/robots.txt
inflating: sonarqube-10.0.0.68432/web/WEB-INF/web.xml
creating: sonarqube-10.0.0.68432/lib/jdbc/
creating: sonarqube-10.0.0.68432/lib/jdbc/mssql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/mssql/mssql-jdbc-11.2.3.jre17.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/postgresql-42.5.1.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/h2/
inflating: sonarqube-10.0.0.68432/lib/jdbc/h2/h2-2.1.214.jar
inflating: sonarqube-10.0.0.68432/lib/sonar-shutdowner-10.0.0.68432.jar
creating: sonarqube-10.0.0.68432/elasticsearch/plugins/
root@ip-172-31-17-56:/home/ubuntu# 

```

Move the unzipped files to /opt/sonarqube directory

```
sudo mv sonarqube-10.0.0.68432 sonarqube
```

```
sudo mv sonarqube /opt/
```

```
root@ip-172-31-17-56:/home root ~ + - < >
inflating: sonarqube-10.0.0.68432/web/images/source-code.svg
inflating: sonarqube-10.0.0.68432/web/images/sq-sl.svg
creating: sonarqube-10.0.0.68432/web/images/tutorials/
inflating: sonarqube-10.0.0.68432/web/images/tutorials/azure-pipelines.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/commit.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/github-actions.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/jenkins.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/manual.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/refresh.svg
inflating: sonarqube-10.0.0.68432/web/index.html
creating: sonarqube-10.0.0.68432/web/js/
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js.map
inflating: sonarqube-10.0.0.68432/web/js/outPKBKIYFT.css
inflating: sonarqube-10.0.0.68432/web/js/outPKBKIYFT.css.map
inflating: sonarqube-10.0.0.68432/web/mstile-512x512.png
inflating: sonarqube-10.0.0.68432/web/robots.txt
inflating: sonarqube-10.0.0.68432/web/WEB-INF/web.xml
creating: sonarqube-10.0.0.68432/lib/jdbc/
creating: sonarqube-10.0.0.68432/lib/jdbc/mssql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/mssql/mssql-jdbc-11.2.3.jre17.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/postgresql-42.5.1.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/h2/
inflating: sonarqube-10.0.0.68432/lib/jdbc/h2/h2-2.1.214.jar
inflating: sonarqube-10.0.0.68432/lib/sonar-shutdowner-10.0.0.68432.jar
creating: sonarqube-10.0.0.68432/elasticsearch/plugins/
root@ip-172-31-17-56:/home/ubuntu# sudo mv sonarqube-10.0.0.68432 sonarqube
sudo mv sonarqube /opt/
root@ip-172-31-17-56:/home/ubuntu# |
```

Part 2: Add SonarQube Group and User

Create a dedicated user and group for SonarQube, which can not run as the root user.

Note: You can give any name for the sonar user and group. I have here given the user and group name to be the same i.e **ddsonar**.

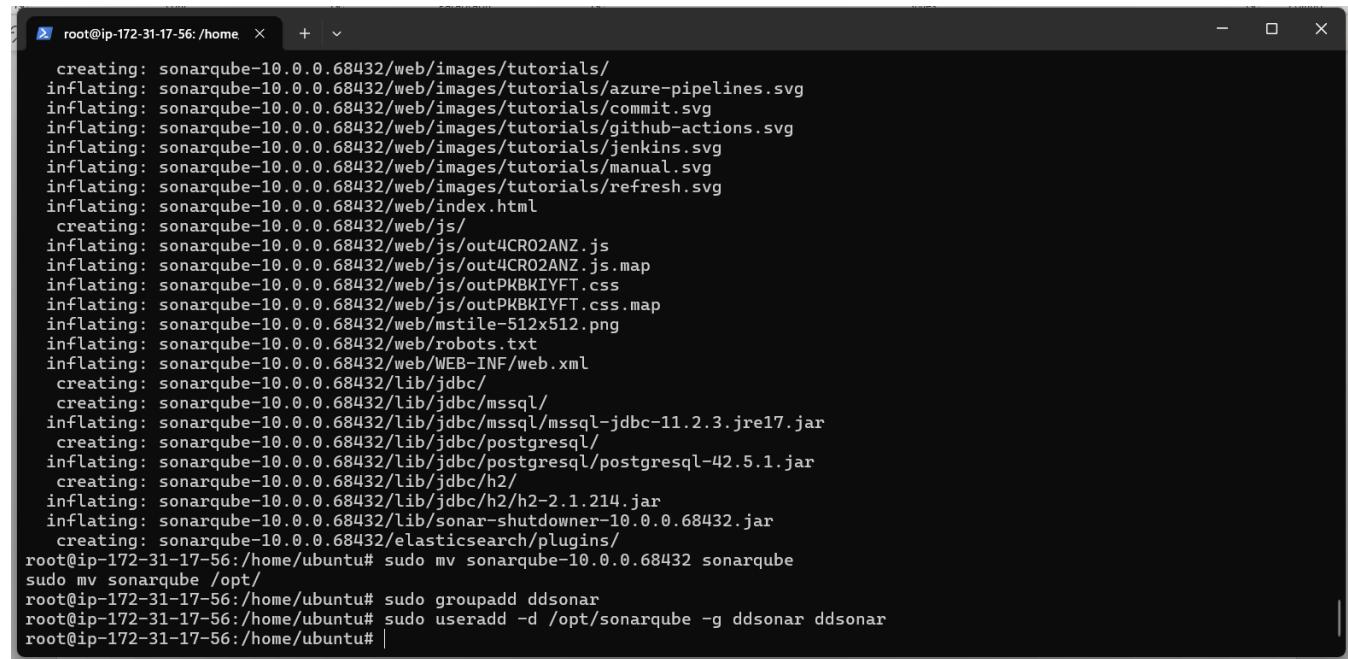
Create a sonar group.

```
sudo groupadd ddsonar
```

```
root@ip-172-31-17-56:/home root ~ + - < >
inflating: sonarqube-10.0.0.68432/web/images/sq-sl.svg
creating: sonarqube-10.0.0.68432/web/images/tutorials/
inflating: sonarqube-10.0.0.68432/web/images/tutorials/azure-pipelines.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/commit.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/github-actions.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/jenkins.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/manual.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/refresh.svg
inflating: sonarqube-10.0.0.68432/web/index.html
creating: sonarqube-10.0.0.68432/web/js/
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js.map
inflating: sonarqube-10.0.0.68432/web/js/outPKBKIYFT.css
inflating: sonarqube-10.0.0.68432/web/js/outPKBKIYFT.css.map
inflating: sonarqube-10.0.0.68432/web/mstile-512x512.png
inflating: sonarqube-10.0.0.68432/web/robots.txt
inflating: sonarqube-10.0.0.68432/web/WEB-INF/web.xml
creating: sonarqube-10.0.0.68432/lib/jdbc/
creating: sonarqube-10.0.0.68432/lib/jdbc/mssql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/mssql/mssql-jdbc-11.2.3.jre17.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/postgresql-42.5.1.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/h2/
inflating: sonarqube-10.0.0.68432/lib/jdbc/h2/h2-2.1.214.jar
inflating: sonarqube-10.0.0.68432/lib/sonar-shutdowner-10.0.0.68432.jar
creating: sonarqube-10.0.0.68432/elasticsearch/plugins/
root@ip-172-31-17-56:/home/ubuntu# sudo mv sonarqube-10.0.0.68432 sonarqube
sudo mv sonarqube /opt/
root@ip-172-31-17-56:/home/ubuntu# sudo groupadd ddsonar
root@ip-172-31-17-56:/home/ubuntu# |
```

Create a sonar user and set /opt/sonarqube as the home directory.

```
sudo useradd -d /opt/sonarqube -g ddsonar ddsonar
```

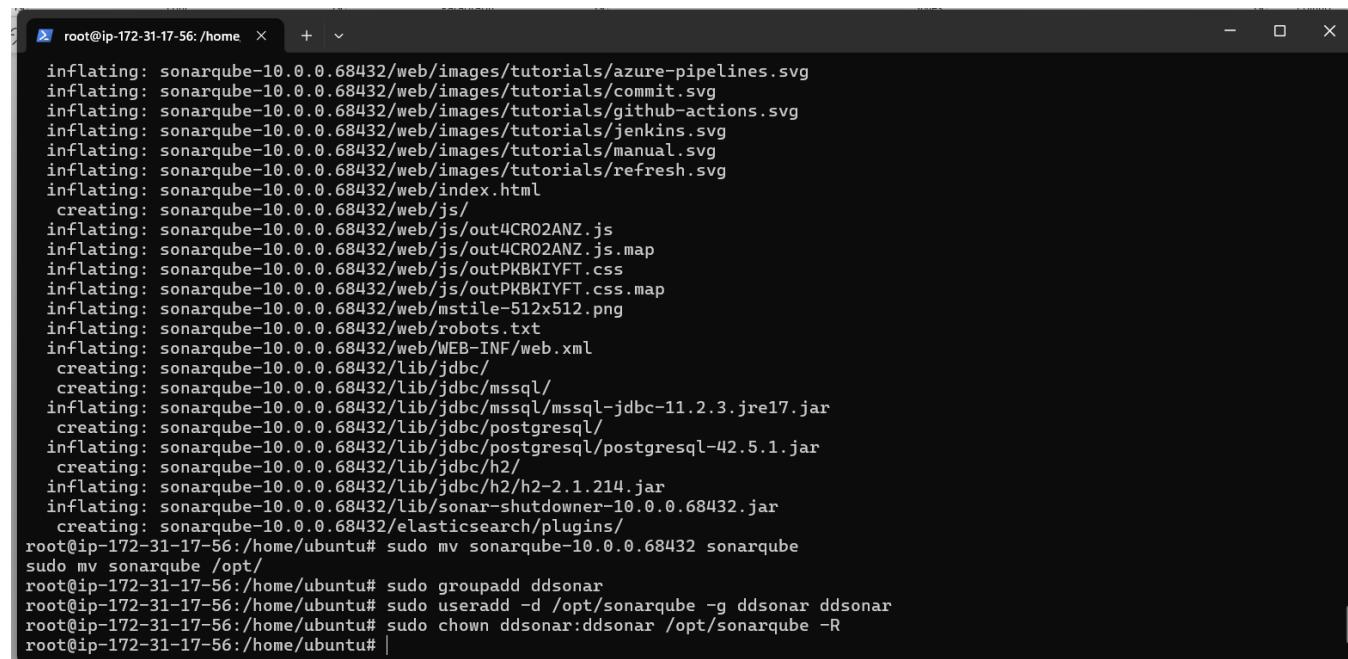


A terminal window titled "root@ip-172-31-17-56: /home" showing the extraction of a SonarQube archive. The output shows various files being created and inflated, including images, JavaScript files, and JDBC drivers. After extraction, the user "ddsonar" is created with a home directory of "/opt/sonarqube".

```
creating: sonarqube-10.0.0.68432/web/images/tutorials/
inflating: sonarqube-10.0.0.68432/web/images/tutorials/azure-pipelines.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/commit.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/github-actions.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/jenkins.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/manual.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/refresh.svg
inflating: sonarqube-10.0.0.68432/web/index.html
creating: sonarqube-10.0.0.68432/web/js/
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js.map
inflating: sonarqube-10.0.0.68432/web/js/outPKBKIYFT.css
inflating: sonarqube-10.0.0.68432/web/js/outPKBKIYFT.css.map
inflating: sonarqube-10.0.0.68432/web/mstile-512x512.png
inflating: sonarqube-10.0.0.68432/web/robots.txt
inflating: sonarqube-10.0.0.68432/web/WEB-INF/web.xml
creating: sonarqube-10.0.0.68432/lib/jdbc/
creating: sonarqube-10.0.0.68432/lib/jdbc/mssql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/mssql-jdbc-11.2.3.jre17.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/postgresql-42.5.1.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/h2/
inflating: sonarqube-10.0.0.68432/lib/jdbc/h2/h2-2.1.214.jar
inflating: sonarqube-10.0.0.68432/lib/sonar-shutdowner-10.0.0.68432.jar
creating: sonarqube-10.0.0.68432/elasticsearch/plugins/
root@ip-172-31-17-56:/home/ubuntu# sudo mv sonarqube-10.0.0.68432 sonarqube
sudo mv sonarqube /opt/
root@ip-172-31-17-56:/home/ubuntu# sudo groupadd ddsonar
root@ip-172-31-17-56:/home/ubuntu# sudo useradd -d /opt/sonarqube -g ddsonar ddsonar
root@ip-172-31-17-56:/home/ubuntu# |
```

Grant the sonar user access to the /opt/sonarqube directory.

```
sudo chown ddsonar:ddsonar /opt/sonarqube -R
```



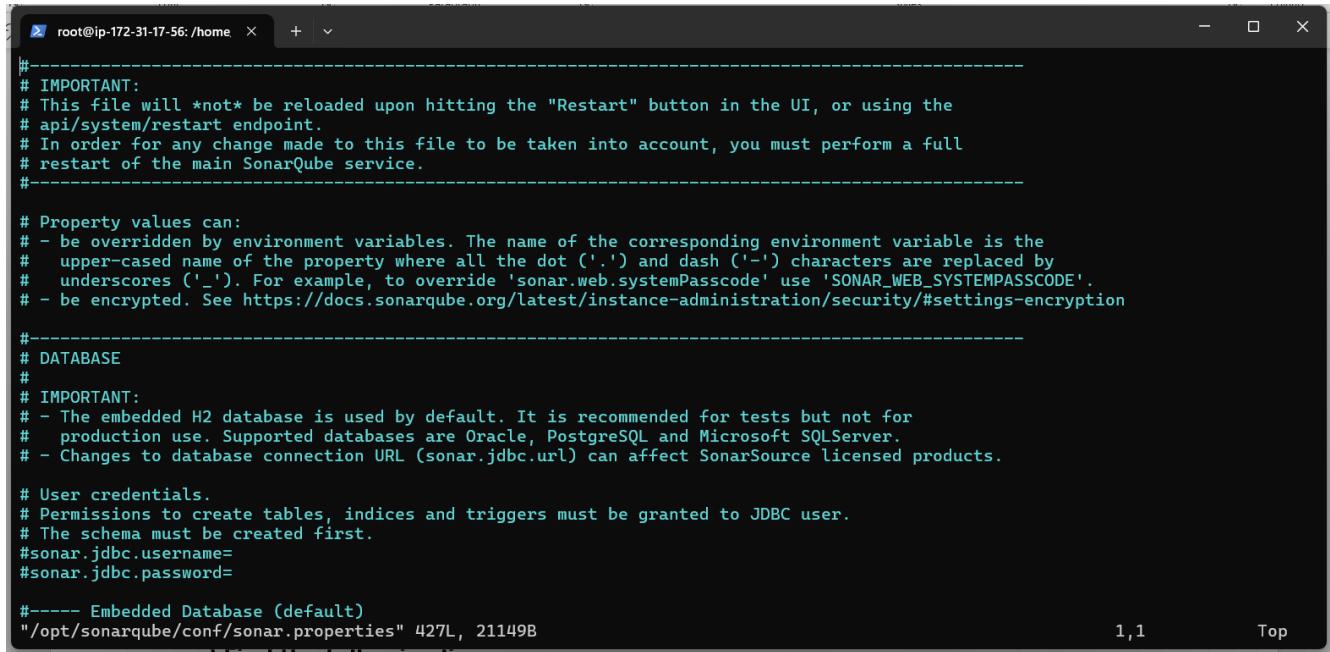
A terminal window titled "root@ip-172-31-17-56: /home" showing the ownership of the SonarQube directory being changed from "root" to "ddsonar:ddsonar" with recursive permissions (-R). The output is identical to the previous terminal window, showing the extraction of the SonarQube archive and the creation of the "ddsonar" user.

```
inflating: sonarqube-10.0.0.68432/web/images/tutorials/azure-pipelines.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/commit.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/github-actions.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/jenkins.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/manual.svg
inflating: sonarqube-10.0.0.68432/web/images/tutorials/refresh.svg
inflating: sonarqube-10.0.0.68432/web/index.html
creating: sonarqube-10.0.0.68432/web/js/
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js
inflating: sonarqube-10.0.0.68432/web/js/out4CRO2ANZ.js.map
inflating: sonarqube-10.0.0.68432/web/js/outPKBKIYFT.css
inflating: sonarqube-10.0.0.68432/web/js/outPKBKIYFT.css.map
inflating: sonarqube-10.0.0.68432/web/mstile-512x512.png
inflating: sonarqube-10.0.0.68432/web/robots.txt
inflating: sonarqube-10.0.0.68432/web/WEB-INF/web.xml
creating: sonarqube-10.0.0.68432/lib/jdbc/
creating: sonarqube-10.0.0.68432/lib/jdbc/mssql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/mssql-jdbc-11.2.3.jre17.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/
inflating: sonarqube-10.0.0.68432/lib/jdbc/postgresql/postgresql-42.5.1.jar
creating: sonarqube-10.0.0.68432/lib/jdbc/h2/
inflating: sonarqube-10.0.0.68432/lib/jdbc/h2/h2-2.1.214.jar
inflating: sonarqube-10.0.0.68432/lib/sonar-shutdowner-10.0.0.68432.jar
creating: sonarqube-10.0.0.68432/elasticsearch/plugins/
root@ip-172-31-17-56:/home/ubuntu# sudo mv sonarqube-10.0.0.68432 sonarqube
sudo mv sonarqube /opt/
root@ip-172-31-17-56:/home/ubuntu# sudo groupadd ddsonar
root@ip-172-31-17-56:/home/ubuntu# sudo useradd -d /opt/sonarqube -g ddsonar ddsonar
root@ip-172-31-17-56:/home/ubuntu# sudo chown ddsonar:ddsonar /opt/sonarqube -R
root@ip-172-31-17-56:/home/ubuntu# |
```

Part 3: Configure SonarQube to connect with PostgreSQL

Edit the SonarQube configuration file.

```
sudo vim /opt/sonarqube/conf/sonar.properties
```



```
# -----
# IMPORTANT:
# This file will *not* be reloaded upon hitting the "Restart" button in the UI, or using the
# api/system/restart endpoint.
# In order for any change made to this file to be taken into account, you must perform a full
# restart of the main SonarQube service.
# -----

# Property values can:
# - be overridden by environment variables. The name of the corresponding environment variable is the
#   upper-cased name of the property where all the dot ('.') and dash ('-') characters are replaced by
#   underscores ('_'). For example, to override 'sonar.web.systemPasscode' use 'SONAR_WEB_SYSTEMPASSCODE'.
# - be encrypted. See https://docs.sonarqube.org/latest/instance-administration/security/#settings-encryption

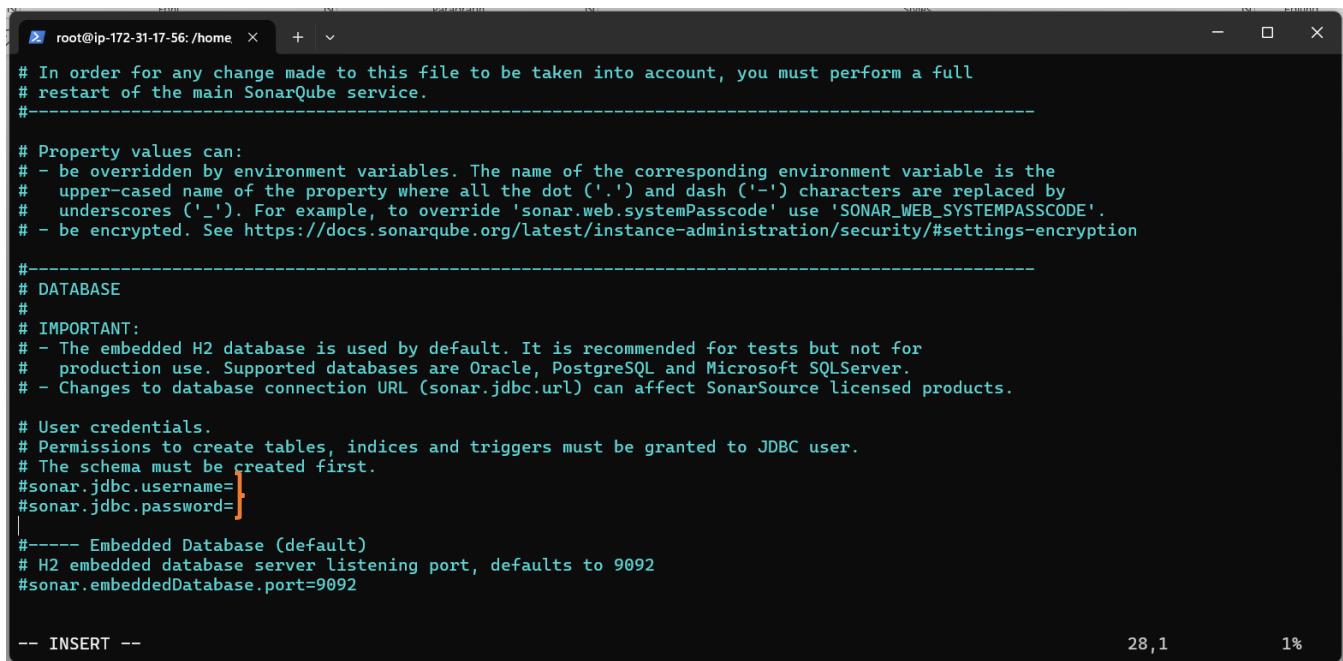
# -----
# DATABASE
#
# IMPORTANT:
# - The embedded H2 database is used by default. It is recommended for tests but not for
#   production use. Supported databases are Oracle, PostgreSQL and Microsoft SQLServer.
# - Changes to database connection URL (sonar.jdbc.url) can affect SonarSource licensed products.

# User credentials.
# Permissions to create tables, indices and triggers must be granted to JDBC user.
# The schema must be created first.
#sonar.jdbc.username=
#sonar.jdbc.password=

#----- Embedded Database (default)
"/opt/sonarqube/conf/sonar.properties" 427L, 21149B
```

Find the following lines:

```
#sonar.jdbc.username=
#sonar.jdbc.password=
```



```
# In order for any change made to this file to be taken into account, you must perform a full
# restart of the main SonarQube service.
# -----

# Property values can:
# - be overridden by environment variables. The name of the corresponding environment variable is the
#   upper-cased name of the property where all the dot ('.') and dash ('-') characters are replaced by
#   underscores ('_'). For example, to override 'sonar.web.systemPasscode' use 'SONAR_WEB_SYSTEMPASSCODE'.
# - be encrypted. See https://docs.sonarqube.org/latest/instance-administration/security/#settings-encryption

# -----
# DATABASE
#
# IMPORTANT:
# - The embedded H2 database is used by default. It is recommended for tests but not for
#   production use. Supported databases are Oracle, PostgreSQL and Microsoft SQLServer.
# - Changes to database connection URL (sonar.jdbc.url) can affect SonarSource licensed products.

# User credentials.
# Permissions to create tables, indices and triggers must be granted to JDBC user.
# The schema must be created first.
#sonar.jdbc.username=
#sonar.jdbc.password=}

#----- Embedded Database (default)
# H2 embedded database server listening port, defaults to 9092
#sonar.embeddedDatabase.port=9092

-- INSERT --
```

Uncomment the lines, and add the database user and Database password you created in Step 3. For me, it's:

```
sonar.jdbc.username=ddsonar
sonar.jdbc.password=myfirstsonar77!
```

```
root@ip-172-31-17-56:/home ~ + - X
# In order for any change made to this file to be taken into account, you must perform a full
# restart of the main SonarQube service.
#-
# Property values can:
# - be overridden by environment variables. The name of the corresponding environment variable is the
#   upper-cased name of the property where all the dot ('.') and dash ('-') characters are replaced by
#   underscores ('_'). For example, to override 'sonar.web.systemPasscode' use 'SONAR_WEB_SYSTEMPASSCODE'.
# - be encrypted. See https://docs.sonarqube.org/latest/instance-administration/security/#settings-encryption
#-
# DATABASE
#
# IMPORTANT:
# - The embedded H2 database is used by default. It is recommended for tests but not for
#   production use. Supported databases are Oracle, PostgreSQL and Microsoft SQLServer.
# - Changes to database connection URL (sonar.jdbc.url) can affect SonarSource licensed products.
#
# User credentials.
# Permissions to create tables, indices and triggers must be granted to JDBC user.
# The schema must be created first.
#sonar.jdbc.username=ddsonar
#sonar.jdbc.password=myfirstsonar77!
#
----- Embedded Database (default)
# H2 embedded database server listening port, defaults to 9092
#sonar.embeddedDatabase.port=9092

-- INSERT --
```

27,37

1%

Below these two lines, add the following line of code.

```
sonar.jdbc.url=jdbc:postgresql://localhost:5432/ddsonarqube
```

```
root@ip-172-31-17-56:/home ~ + - X
# In order for any change made to this file to be taken into account, you must perform a full
# restart of the main SonarQube service.
#-
# Property values can:
# - be overridden by environment variables. The name of the corresponding environment variable is the
#   upper-cased name of the property where all the dot ('.') and dash ('-') characters are replaced by
#   underscores ('_'). For example, to override 'sonar.web.systemPasscode' use 'SONAR_WEB_SYSTEMPASSCODE'.
# - be encrypted. See https://docs.sonarqube.org/latest/instance-administration/security/#settings-encryption
#-
# DATABASE
#
# IMPORTANT:
# - The embedded H2 database is used by default. It is recommended for tests but not for
#   production use. Supported databases are Oracle, PostgreSQL and Microsoft SQLServer.
# - Changes to database connection URL (sonar.jdbc.url) can affect SonarSource licensed products.
#
# User credentials.
# Permissions to create tables, indices and triggers must be granted to JDBC user.
# The schema must be created first.
#sonar.jdbc.username=ddsonar
#sonar.jdbc.password=myfirstsonar77!
sonar.jdbc.url=jdbc:postgresql://localhost:5432/ddsonarqube

----- Embedded Database (default)
# H2 embedded database server listening port, defaults to 9092
#sonar.embeddedDatabase.port=9092

-- INSERT --
```

26,1

1%

Here, ddsonarqube is the database name created.

Save and exit the file.

Press ESC, followed by :wq and press ENTER

```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/conf/sonar.properties
root@ip-172-31-17-56:/home/ubuntu# |
```

Edit the sonar script file.

```
sudo vim /opt/sonarqube/bin/linux-x86-64/sonar.sh
```

```
#!/bin/sh
APP_NAME="SonarQube"

# By default, java from the PATH is used, except if SONAR_JAVA_PATH env variable is set
findjava() {
    if [ -z "${SONAR_JAVA_PATH}" ]; then
        if ! command -v java 2>&1; then
            echo "Java not found. Please make sure that the environmental variable SONAR_JAVA_PATH points to a Java executable"
            exit 1
        fi
        JAVA_CMD=java
    else
        if ! [ -x "${SONAR_JAVA_PATH}" ] || ! [ -f "${SONAR_JAVA_PATH}" ]; then
            echo "File '${SONAR_JAVA_PATH}' is not executable. Please make sure that the environmental variable SONAR_JAVA_PATH points to a Java executable"
            exit 1
        fi
        JAVA_CMD="${SONAR_JAVA_PATH}"
    fi
}
findjava

# Get the fully qualified path to the script
case $0 in
    /*)
        SCRIPT="$0"
        ;;
    /*
"/opt/sonarqube/bin/linux-x86-64/sonar.sh" 317L, 7136B
```

1,1

Top

Add the following line

```
RUN_AS_USER=ddsonar
```

```
root@ip-172-31-17-56:/home| + | - | X
#!/bin/sh

RUN_AS_USER=ddsonar
APP_NAME="SonarQube"

# By default, java from the PATH is used, except if SONAR_JAVA_PATH env variable is set
findjava() {
    if [ -z "${SONAR_JAVA_PATH}" ]; then
        if ! command -v java 2>&1; then
            echo "Java not found. Please make sure that the environmental variable SONAR_JAVA_PATH points to a Java executable"
            exit 1
        fi
        JAVA_CMD=java
    else
        if ! [ -x "${SONAR_JAVA_PATH}" ] || ! [ -f "${SONAR_JAVA_PATH}" ]; then
            echo "File '${SONAR_JAVA_PATH}' is not executable. Please make sure that the environmental variable SONAR_JAVA_PATH points to a Java executable"
            exit 1
        fi
        JAVA_CMD="${SONAR_JAVA_PATH}"
    fi
}

findjava

# Get the fully qualified path to the script
case $0 in
/*)
    SCRIPT="$0"
-- INSERT --
```

3,20 Top

Here, ddsonar is the name of the user that we have created in step number 6 (ii).

Save and exit the file.

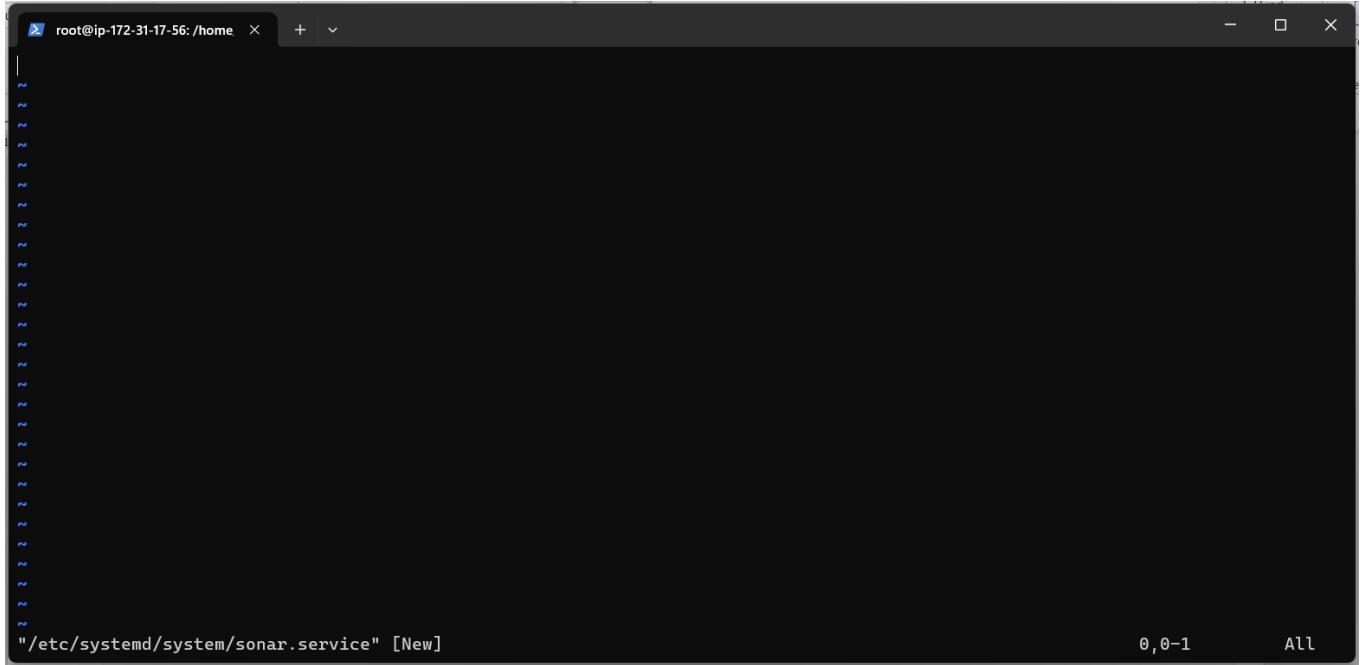
Press ESC, followed by :wq and press ENTER

```
root@ip-172-31-17-56:/home| + | - | X
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/conf/sonar.properties
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/bin/linux-x86-64/sonar.sh
root@ip-172-31-17-56:/home/ubuntu# |
```

Part 4: Create Systemd service for SonarQube

Create a systemd service file to start SonarQube at system boot.

```
sudo vim /etc/systemd/system/sonar.service
```



```
root@ip-172-31-17-56: /home ~
```

"`/etc/systemd/system/sonar.service`" [New]

0,0-1

All

Paste the following lines to the file.

```
[Unit]
Description=SonarQube service
After=syslog.target network.target
[Service]
Type=forking
ExecStart=/opt/sonarqube/bin/linux-x86-64/sonar.sh start
ExecStop=/opt/sonarqube/bin/linux-x86-64/sonar.sh stop
User=ddsonar
Group=ddsonar
Restart=always
LimitNOFILE=65536
LimitNPROC=4096
[Install]
WantedBy=multi-user.target
```

```
root@ip-172-31-17-56:/home ~ + | - X
[Unit]
Description=SonarQube service
After=syslog.target network.target
[Service]
Type=forking
ExecStart=/opt/sonarqube/bin/linux-x86-64/sonar.sh start
ExecStop=/opt/sonarqube/bin/linux-x86-64/sonar.sh stop
User=ddsonar
Group=ddsonar
Restart=always
LimitNOFILE=65536
LimitNPROC=4096
[Install]
WantedBy=multi-user.target
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
-- INSERT --
```

15,1 All

Note: Here in the above script, make sure to change the User and Group section with the value that you have created. For me its:

```
User=ddsonar  
Group=ddsonar
```

Save and exit the file.

Press ESC, followed by :wq then press ENTER

```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/conf/sonar.properties
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/bin/linux-x86-64/sonar.sh
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/systemd/system/sonar.service
root@ip-172-31-17-56:/home/ubuntu# |
```

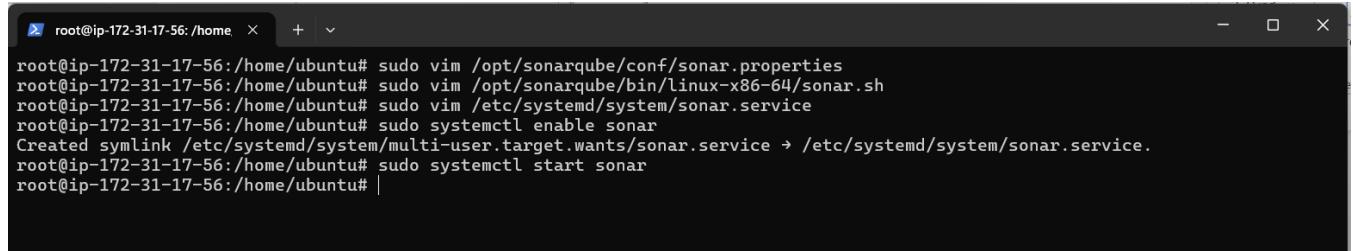
Part 5: Enable the SonarQube service to run at system startup

```
sudo systemctl enable sonar
```

```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/conf/sonar.properties
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/bin/linux-x86-64/sonar.sh
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/systemd/system/sonar.service
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable sonar
Created symlink /etc/systemd/system/multi-user.target.wants/sonar.service → /etc/systemd/system/sonar.service.
root@ip-172-31-17-56:/home/ubuntu# |
```

Start the SonarQube service

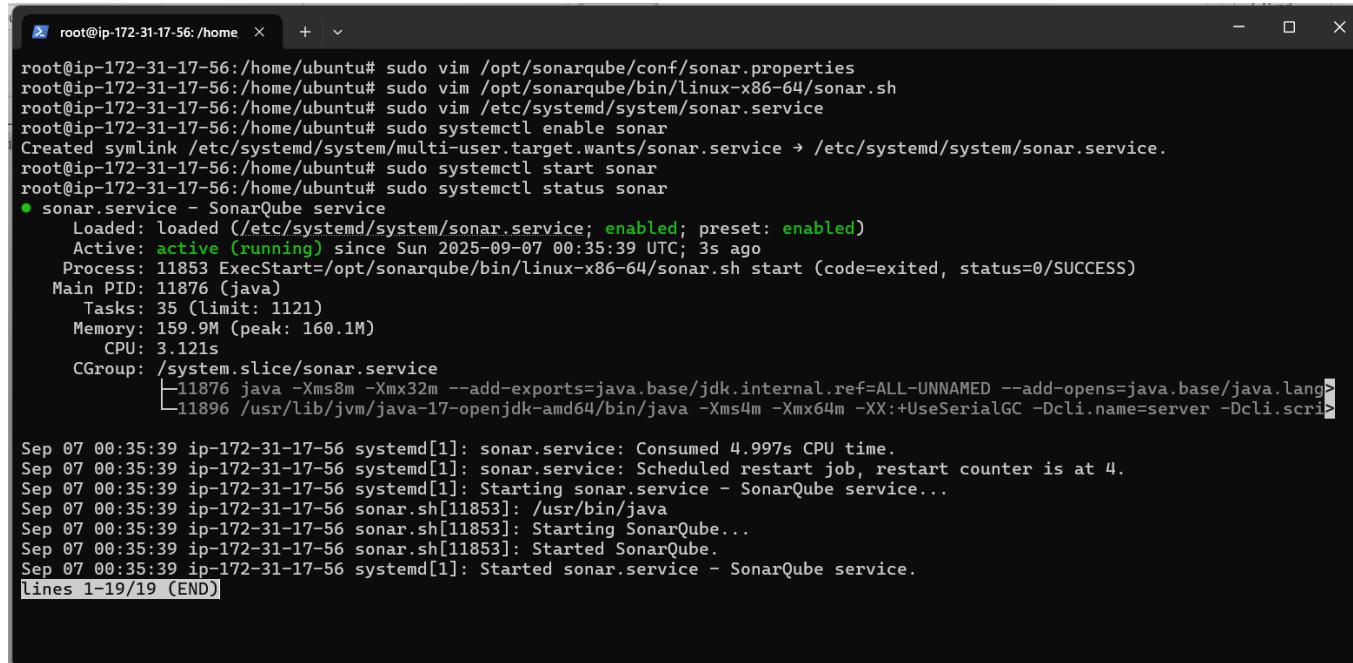
```
sudo systemctl start sonar
```



```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/conf/sonar.properties
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/bin/linux-x86-64/sonar.sh
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/systemd/system/sonar.service
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable sonar
Created symlink /etc/systemd/system/multi-user.target.wants/sonar.service → /etc/systemd/system/sonar.service.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start sonar
root@ip-172-31-17-56:/home/ubuntu# |
```

Check the service status

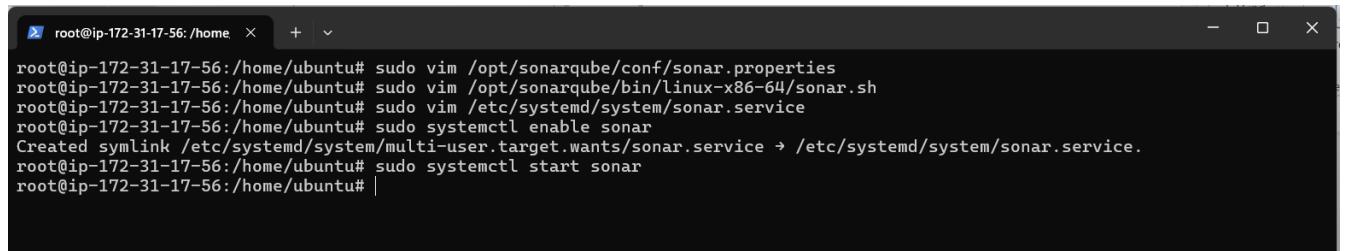
```
sudo systemctl status sonar
```



```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/conf/sonar.properties
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/bin/linux-x86-64/sonar.sh
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/systemd/system/sonar.service
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable sonar
Created symlink /etc/systemd/system/multi-user.target.wants/sonar.service → /etc/systemd/system/sonar.service.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start sonar
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status sonar
● sonar.service - SonarQube service
   Loaded: loaded (/etc/systemd/system/sonar.service; enabled; preset: enabled)
   Active: active (running) since Sun 2025-09-07 00:35:39 UTC; 3s ago
     Process: 11853 ExecStart=/opt/sonarqube/bin/linux-x86-64/sonar.sh start (code=exited, status=0/SUCCESS)
   Main PID: 11876 (java)
      Tasks: 35 (limit: 1121)
        Memory: 159.9M (peak: 160.1M)
         CPU: 3.121s
        CGroup: /system.slice/sonar.service
                  └─11876 java -Xms8m -Xmx32m --add-exports=java.base/jdk.internal.ref=ALL-UNNAMED --add-opens=java.base/java.lang>
                     ├─11896 /usr/lib/jvm/java-17-openjdk-amd64/bin/java -Xms4m -Xmx64m -XX:+UseSerialGC -Dcli.name=server -Dcli.scri>

Sep 07 00:35:39 ip-172-31-17-56 systemd[1]: sonar.service: Consumed 4.997s CPU time.
Sep 07 00:35:39 ip-172-31-17-56 systemd[1]: sonar.service: Scheduled restart job, restart counter is at 4.
Sep 07 00:35:39 ip-172-31-17-56 systemd[1]: Starting sonar.service - SonarQube service...
Sep 07 00:35:39 ip-172-31-17-56 sonar.sh[11853]: /usr/bin/java
Sep 07 00:35:39 ip-172-31-17-56 sonar.sh[11853]: Starting SonarQube...
Sep 07 00:35:39 ip-172-31-17-56 sonar.sh[11853]: Started SonarQube.
Sep 07 00:35:39 ip-172-31-17-56 systemd[1]: Started sonar.service - SonarQube service.
Lines 1-19/19 (END)
```

It is up and running. Then type "q" to exit this mode



```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/conf/sonar.properties
root@ip-172-31-17-56:/home/ubuntu# sudo vim /opt/sonarqube/bin/linux-x86-64/sonar.sh
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/systemd/system/sonar.service
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable sonar
Created symlink /etc/systemd/system/multi-user.target.wants/sonar.service → /etc/systemd/system/sonar.service.
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl start sonar
root@ip-172-31-17-56:/home/ubuntu# |
```

Reboot the system to apply the changes.

```
sudo reboot
```

```
Windows PowerShell x + v
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/sysctl.conf
root@ip-172-31-17-56:/home/ubuntu# sudo reboot

Broadcast message from root@ip-172-31-17-56 on pts/2 (Sun 2025-09-07 01:14:43 UTC):
The system will reboot now!

Broadcast message from root@ip-172-31-17-56 on pts/2 (Sun 2025-09-07 01:14:43 UTC):
The system will reboot now!

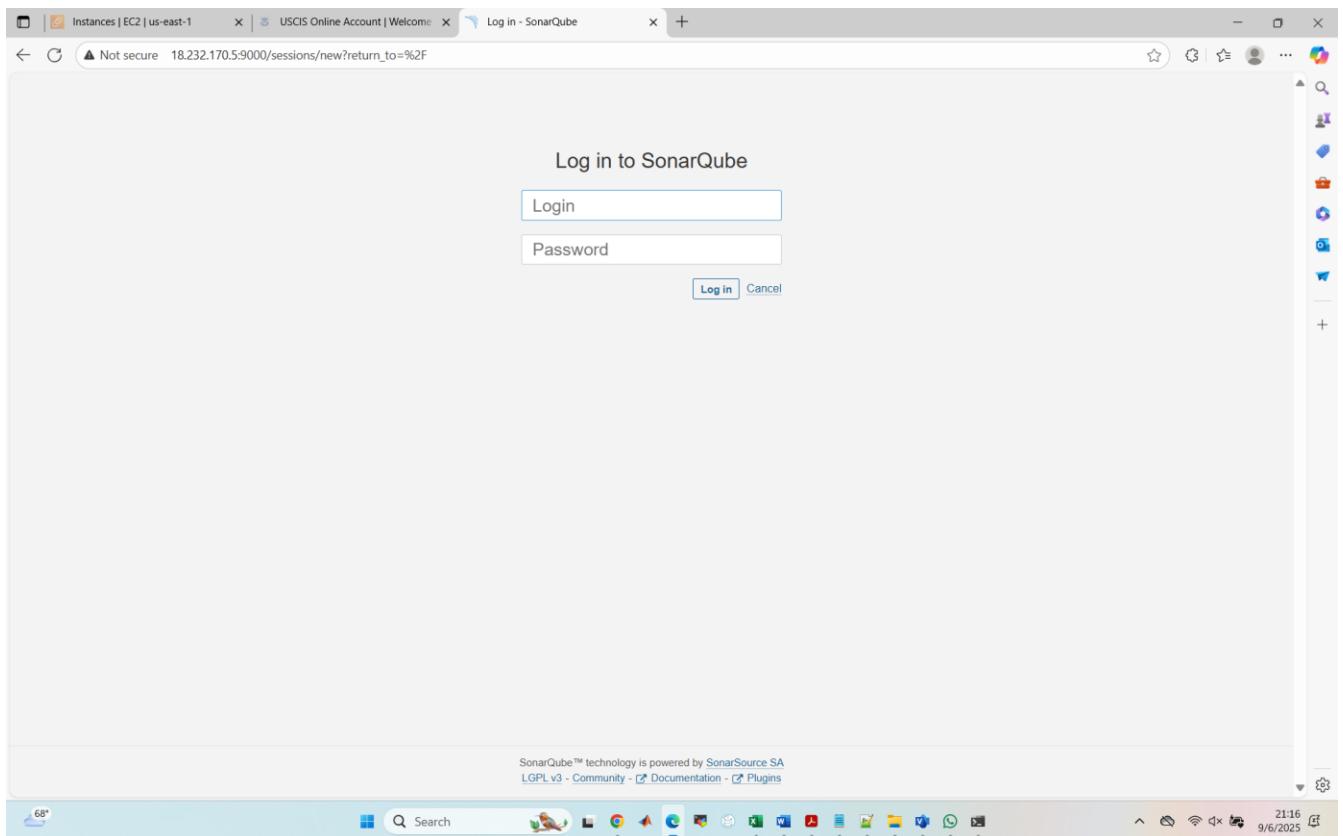
root@ip-172-31-17-56:/home/ubuntu# Connection to ec2-18-232-170-5.compute-1.amazonaws.com closed by remote host.
Connection to ec2-18-232-170-5.compute-1.amazonaws.com closed.
PS C:\Users\ebots\Downloads> |
```

STEP 5: Access SonarQube Web Interface

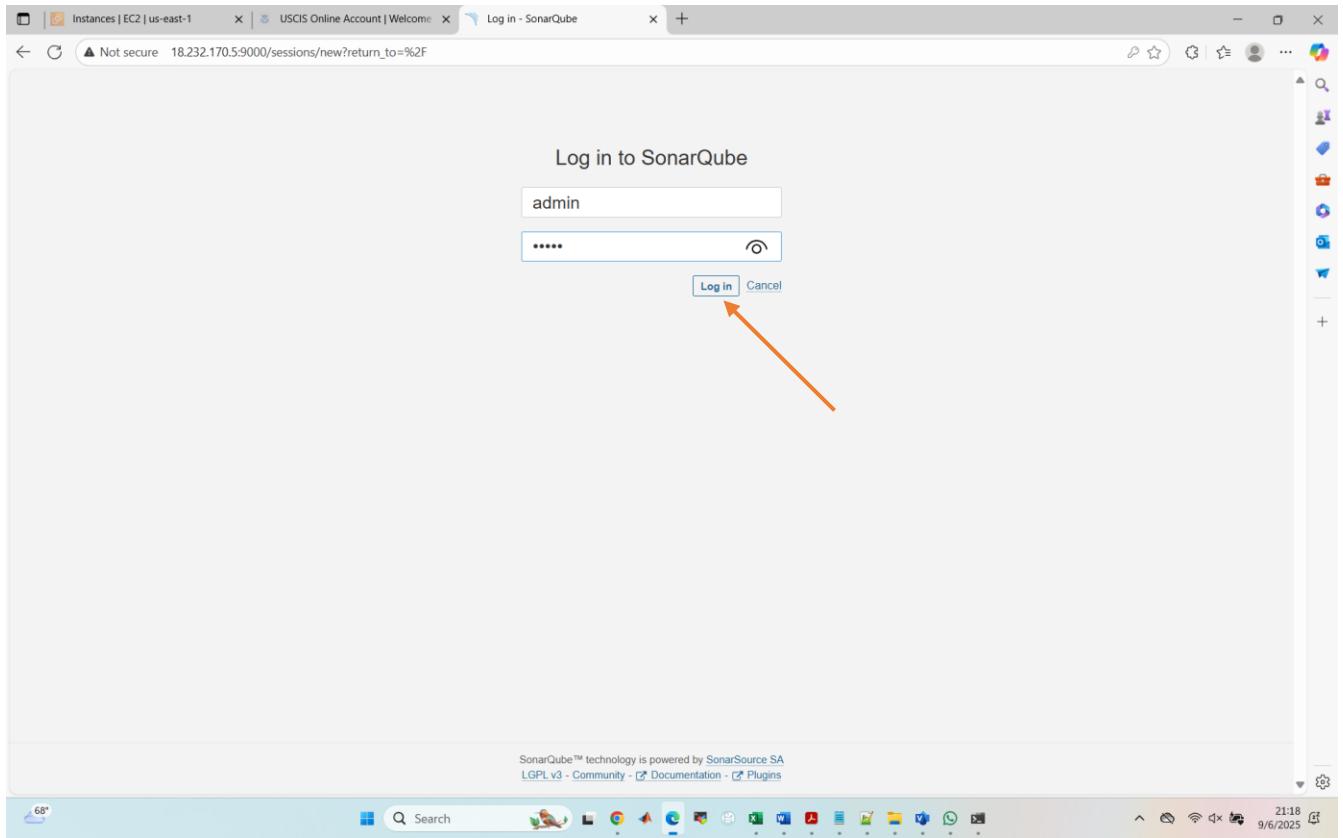
Access SonarQube in a web browser at your server's IP address on port 9000.

For example, <http://Public IPv4 Address:9000>

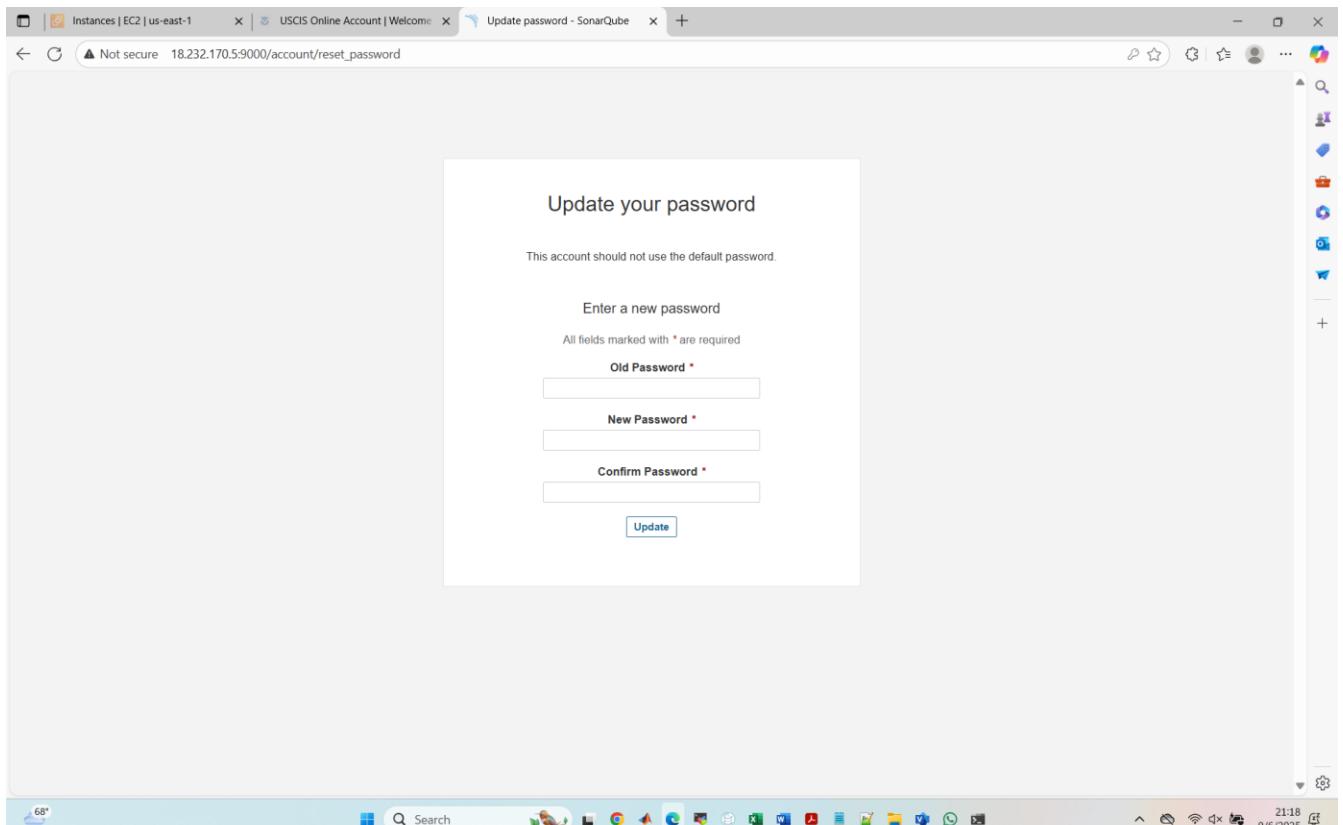
That is <http://18.232.170.5:9000>



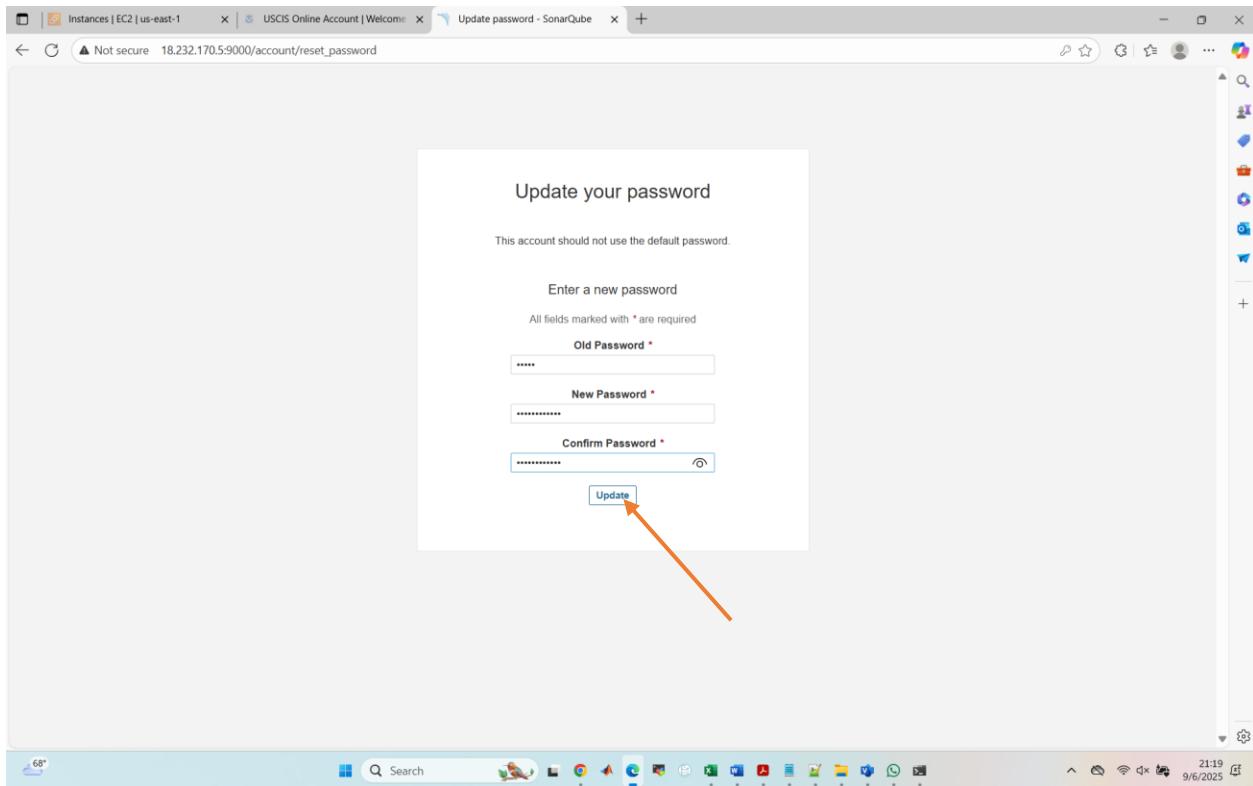
Note: the default username and password are **admin** and **admin** respectively.



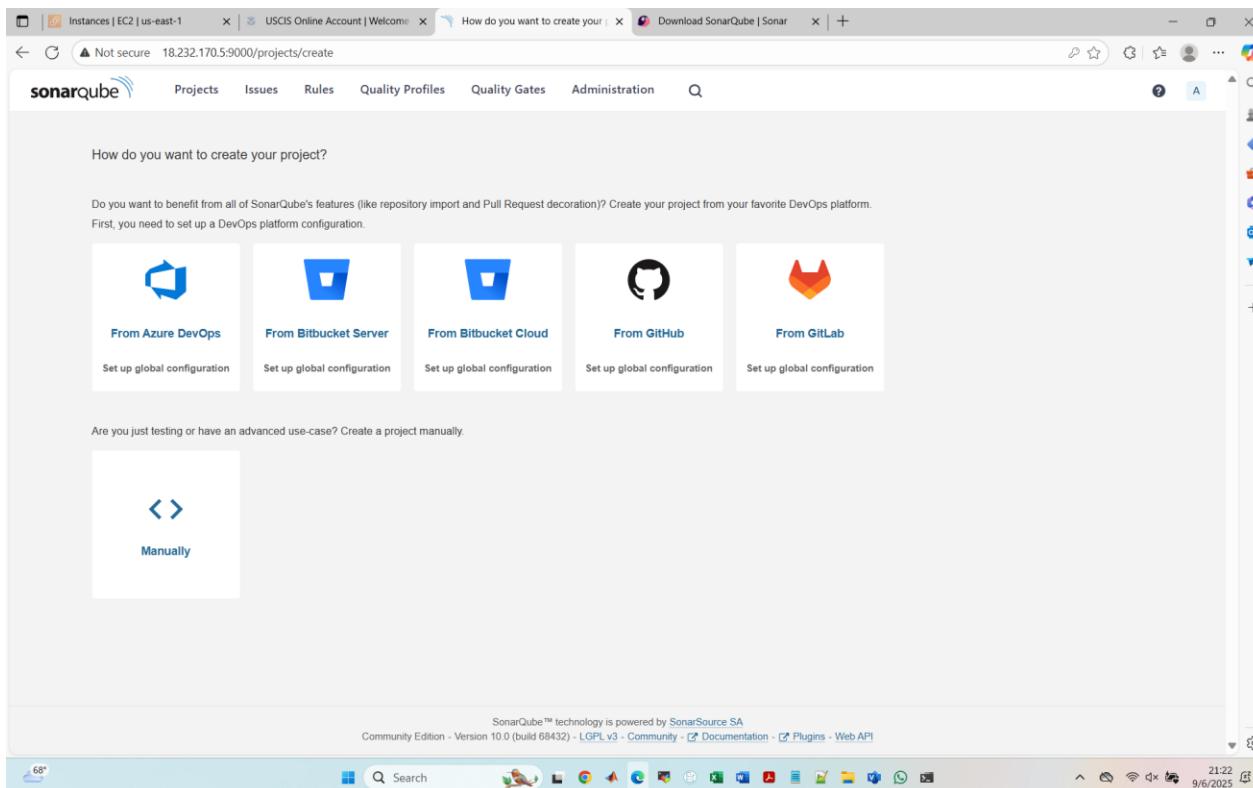
Click on “Login”. In the next step, SonarQube will prompt you to change your password. CHANGE THE PASSWORD.



Change the Old password with a New one.



Click on “Update”



We have successfully installed the SonarQube Community version 10.0 on AWS EC2 Ubuntu.

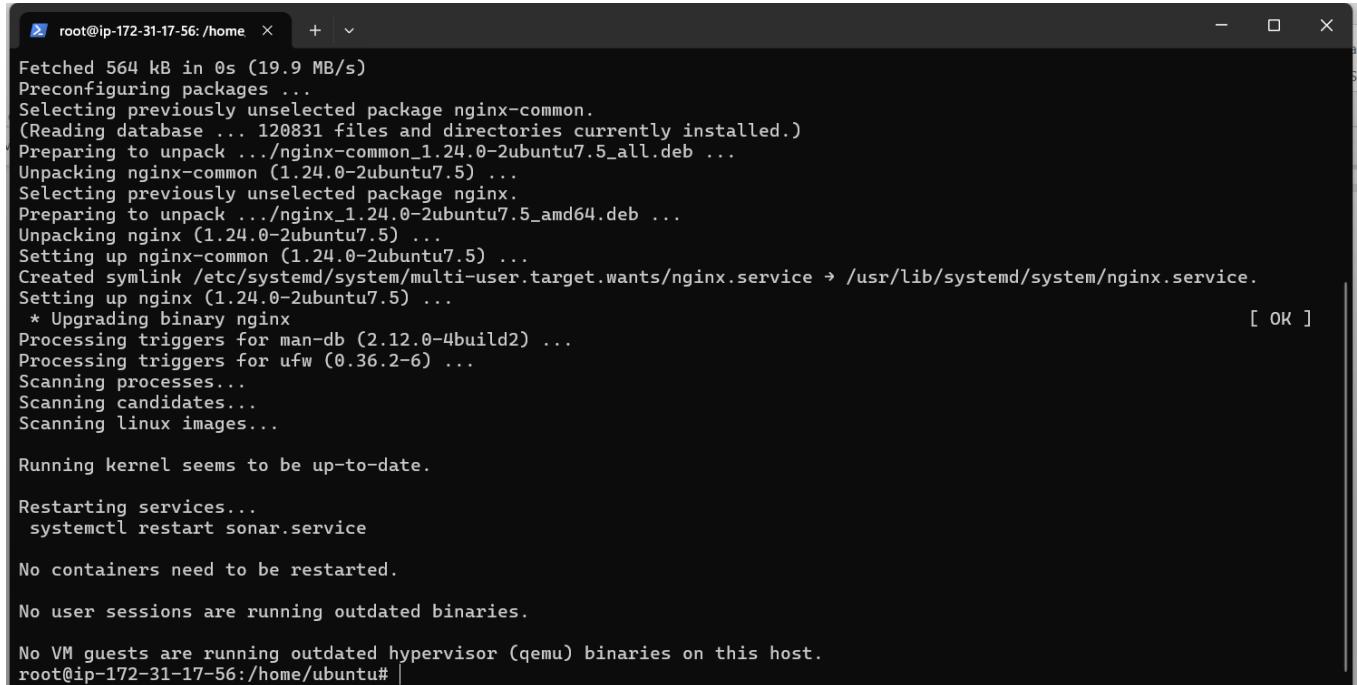
STEP 4: Configuring the Reverse Proxy using Nginx

Now we want a domain name to take over the IP of the server so that instead of hitting the IP we can use the domain name that is TLS enabled. Let's see how to do so.

I will be using the sub-domain name **sonarqube.sosoebot.org**. This is mine. Please use yours.

Part 1: Install Nginx

```
sudo apt install nginx -y
```



A terminal window titled "root@ip-172-31-17-56:/home" showing the output of the apt command. The output details the package installation process for nginx-common and nginx, including file unpacking, configuration, and service setup. It also shows system checks like kernel status, service restarting, and hypervisor binary scanning. The command "root@ip-172-31-17-56:/home/ubuntu#" is visible at the bottom.

```
Fetched 564 kB in 0s (19.9 MB/s)
Preconfiguring packages ...
Selecting previously unselected package nginx-common.
(Reading database ... 120831 files and directories currently installed.)
Preparing to unpack .../nginx-common_1.24.0-2ubuntu7.5_all.deb ...
Unpacking nginx-common (1.24.0-2ubuntu7.5) ...
Selecting previously unselected package nginx.
Preparing to unpack .../nginx_1.24.0-2ubuntu7.5_amd64.deb ...
Unpacking nginx (1.24.0-2ubuntu7.5) ...
Setting up nginx-common (1.24.0-2ubuntu7.5) ...
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /usr/lib/systemd/system/nginx.service.
Setting up nginx (1.24.0-2ubuntu7.5) ...
 * Upgrading binary nginx
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for ufw (0.36.2-6) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Running kernel seems to be up-to-date.

Restarting services...
systemctl restart sonar.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-17-56:/home/ubuntu#
```

Part 2: Add the DNS records

My domain is hosted on the AWS Route53, thus I will be adding the records in there. Go to AWS Console and search for "**Route53**"

The screenshot shows the AWS Route 53 Dashboard. The left sidebar contains a navigation menu with sections like Dashboard, Hosted zones (highlighted with an orange arrow), Health checks, Profiles, IP-based routing, Traffic flow, Domains, and Resolver. The main content area displays the Route 53 Dashboard with sections for DNS management (2 hosted zones), Traffic management, Availability monitoring, and Domain registration (1 domain). It also includes a Register domain form, Notifications (empty), and More resources links.

Adding the IP of the server on the DNS zone. Click on “Hosted Zones”

The screenshot shows the Hosted zones page with 2 entries. The table lists:

Hosted zone name	Type	Created by	Record count
sosoebot.com	Public	Route 53	3
sosoebot.org	Public	Route 53	4

An orange arrow points to the "sosoebot.org" entry.

And now our domain sonarqube.sosoebot.org is ready to be used for the nginx configuration setup. Click on “**sosoebot.org**”

Click on “Create Record”

► View existing records

The following table lists the existing records in sosoebot.org.

Enter “sonarqube”

Create record [Info](#)

Quick create record

▼ Record 1

Record name Info	.sosoebot.org	Record type Info
sonarqube	A – Routes traffic to an IPv4 address and some AWS resources	

Keep blank to create a record for the root domain.

Alias

Value [Info](#)

192.0.2.235
Enter multiple values on separate lines.

TTL (seconds) [Info](#)

300

Routing policy [Info](#)

Simple routing

[Add another record](#)

[Cancel](#) **Create records**

► View existing records

The following table lists the existing records in sosoebot.org.

Add the IP address

Create record [Info](#)

Quick create record

▼ Record 1

Record name Info	.sosoebot.org	Record type Info
sonarqube	A – Routes traffic to an IPv4 address and some AWS resources	

Keep blank to create a record for the root domain.

Alias

Value [Info](#)

18.232.170.5
Enter multiple values on separate lines.

TTL (seconds) [Info](#)

300

Routing policy [Info](#)

Simple routing

[Add another record](#)

[Cancel](#) **Create records**

► View existing records

The following table lists the existing records in sosoebot.org.

Click on “Create records”

The screenshot shows the AWS Route 53 console. On the left, a sidebar navigation includes: Dashboard, Hosted zones (selected), Health checks, Profiles, IP-based routing, Traffic flow, Domains, Resolver, VPCs, Inbound endpoints, Outbound endpoints, Rules, Query logging, and Outposts. The main area displays a success message: "Record for sosoebot.org was successfully created. Route 53 propagates your changes to all of the Route 53 authoritative DNS servers within 60 seconds. Use "View status" button to check propagation status." Below this, the "Hosted zone details" section shows a table of records. The table has columns: Record, Type, Routine, Differ..., Alias, Value/Route traffic to, TTL, Health, Evaluation, and Record. There are five entries:

Record	Type	Routine	Differ...	Alias	Value/Route traffic to	TTL	Health	Evaluation	Record
sosoebot...	A	Simple	-	Yes	d2e8xtfhr55xqp.cloudfront.n...	-	-	No	-
sosoebot...	NS	Simple	-	No	ns-578.awsdns-08.net. ns-23.awsdns-02.com. ns-2007.awsdns-58.co.uk. ns-1451.awsdns-53.org.	172800	-	-	-
sosoebot...	SOA	Simple	-	No	ns-578.awsdns-08.net.awsd...	900	-	-	-
_0e55f88c...	CNAME	Simple	-	No	_bccf75c8964d660b38bfbb...	300	-	-	-
sonarqube...	A	Simple	-	No	18.232.170.5	300	-	-	-

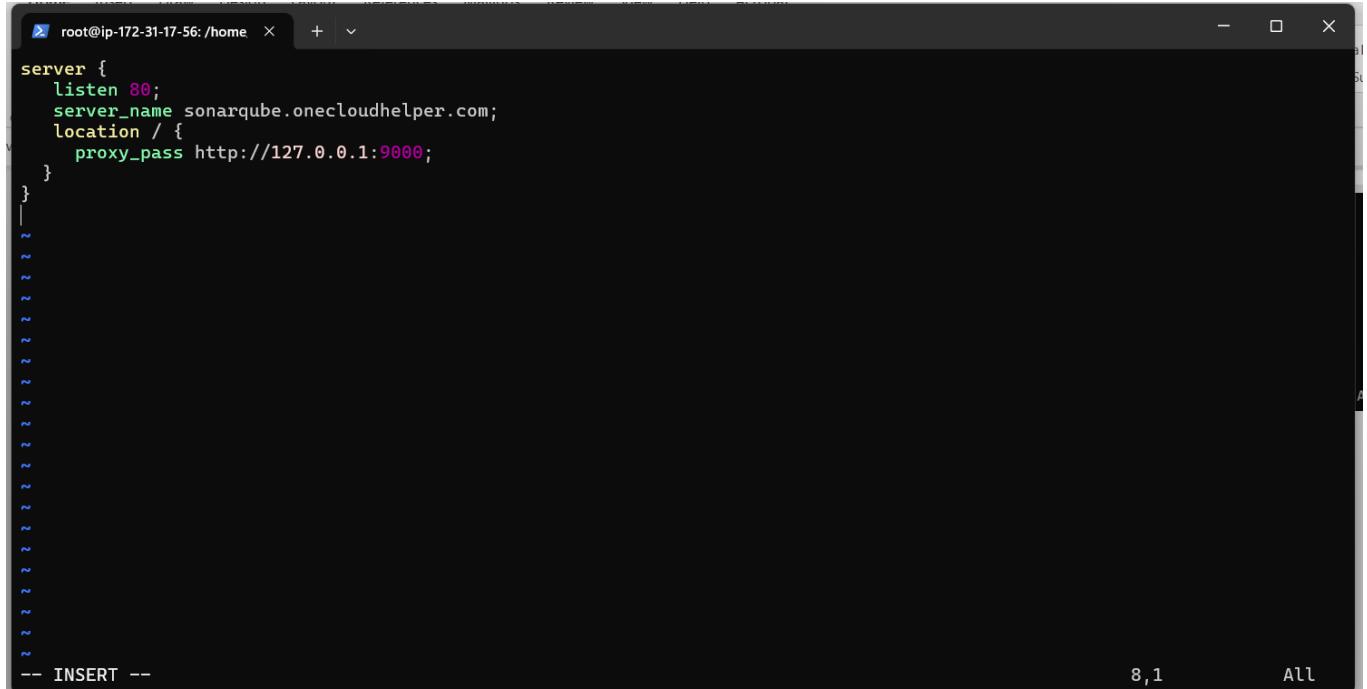
Part 3: Create a new Nginx configuration file for the site

```
sudo vim /etc/nginx/sites-enabled/ddsonarqube
```

The terminal window title bar says "root@ip-172-31-17-56: /home". The command "/etc/nginx/sites-enabled/ddsonarqube" is being typed into the terminal. The bottom status bar shows the file path "/etc/nginx/sites-enabled/ddsonarqube" and the status "[New]".

Add this configuration so that Nginx will route incoming traffic to SonarQube.

```
server {
    listen 80;
    server_name sonarqube.onecloudhelper.com;
    location / {
        proxy_pass http://127.0.0.1:9000;
    }
}
```



A screenshot of a terminal window titled "root@ip-172-31-17-56:/home". The window contains the same configuration code as the previous block. The status bar at the bottom right shows "8,1" and "All".

```
server {
    listen 80;
    server_name sonarqube.onecloudhelper.com;
    location / {
        proxy_pass http://127.0.0.1:9000;
    }
}
-- INSERT --
```

Here, replace the part **server_name** with **sonarqube.sosoebot.org**; that is your domain or sub-domain name.

```
server {
    listen 80;
    server_name sonarqube.sosoebot.org;
    location / {
        proxy_pass http://127.0.0.1:9000;
    }
}
```

A screenshot of a terminal window titled "root@ip-172-31-17-56:/home". The window shows an Nginx configuration file in Vim. The configuration includes a server block for port 80, proxying requests to localhost:9000. The file ends with a double colon and a minus sign, indicating it's in insert mode. The status bar at the bottom right shows "8,1 All".

```
server {
    listen 80;
    server_name sonarqube.sosoebot.org;
    location / {
        proxy_pass http://127.0.0.1:9000;
    }
}
-- INSERT --
```

Save it by Pressing ESC, followed by :wq and press ENTER

A screenshot of a terminal window titled "root@ip-172-31-17-56:/home/ubuntu". The user runs the command "sudo vim /etc/nginx/sites-enabled/ddsonarqube". The status bar at the bottom right shows "8,1 All".

```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/nginx/sites-enabled/ddsonarqube
root@ip-172-31-17-56:/home/ubuntu# |
```

Part 4: Make sure your configuration file has no syntax errors.

```
sudo nginx -t
```

A screenshot of a terminal window titled "root@ip-172-31-17-56:/home/ubuntu". The user runs "sudo nginx -t", which tests the configuration and outputs that it is syntax ok and the test was successful. The status bar at the bottom right shows "8,1 All".

```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/nginx/sites-enabled/ddsonarqube
root@ip-172-31-17-56:/home/ubuntu# sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
root@ip-172-31-17-56:/home/ubuntu# |
```

Part 5: Enable the Nginx site and restart Nginx

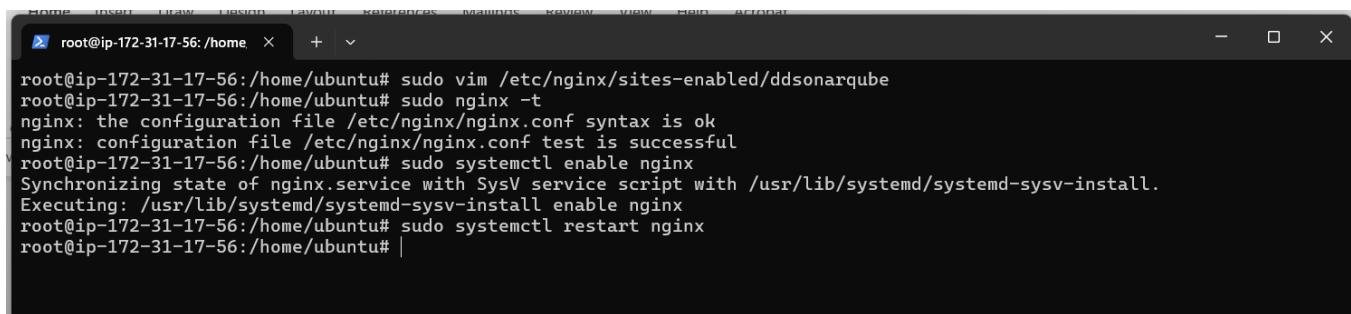
```
sudo systemctl enable nginx
```

A screenshot of a terminal window titled "root@ip-172-31-17-56:/home/ubuntu". The user runs "sudo systemctl enable nginx", which synchronizes the state of the service with the SysV service script and enables it. The status bar at the bottom right shows "8,1 All".

```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/nginx/sites-enabled/ddsonarqube
root@ip-172-31-17-56:/home/ubuntu# sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable nginx
Synchronizing state of nginx.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable nginx
root@ip-172-31-17-56:/home/ubuntu# |
```

Restart the nginx server

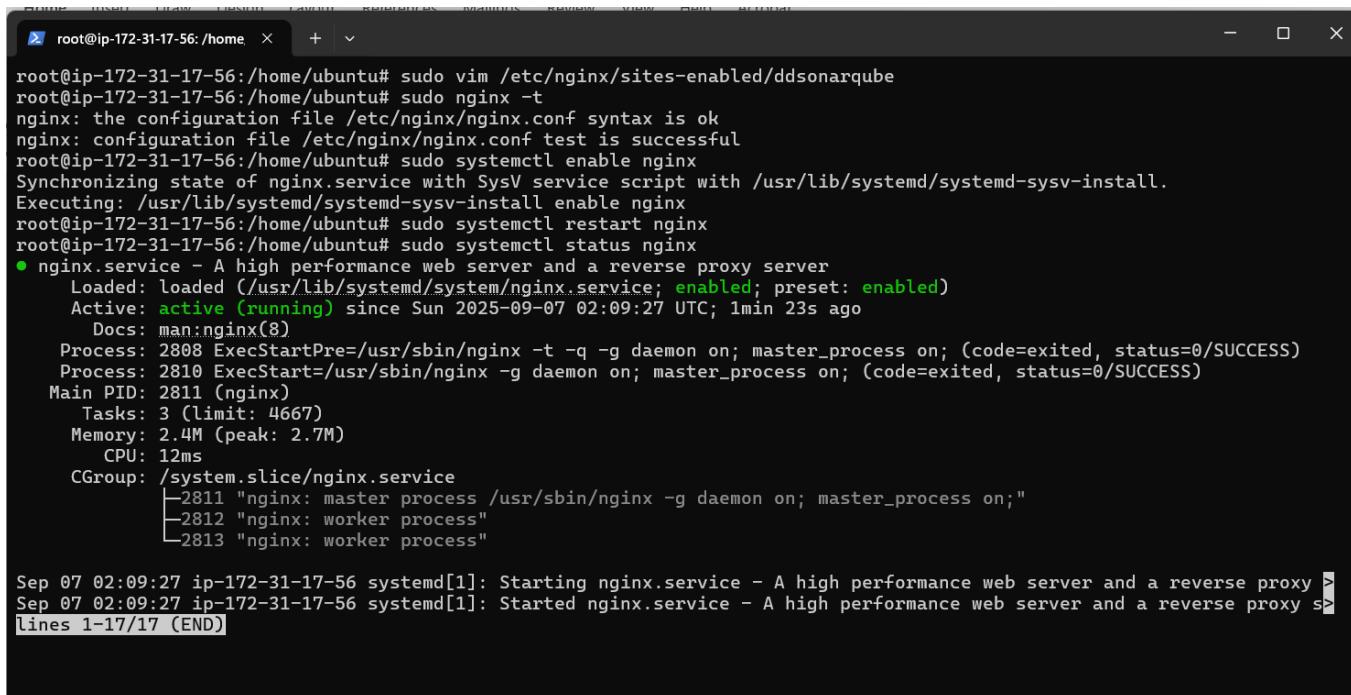
```
sudo systemctl restart nginx
```



```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/nginx/sites-enabled/ddsonarqube
root@ip-172-31-17-56:/home/ubuntu# sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable nginx
Synchronizing state of nginx.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable nginx
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl restart nginx
root@ip-172-31-17-56:/home/ubuntu# |
```

Check the status of the nginx server

```
sudo systemctl status nginx
```

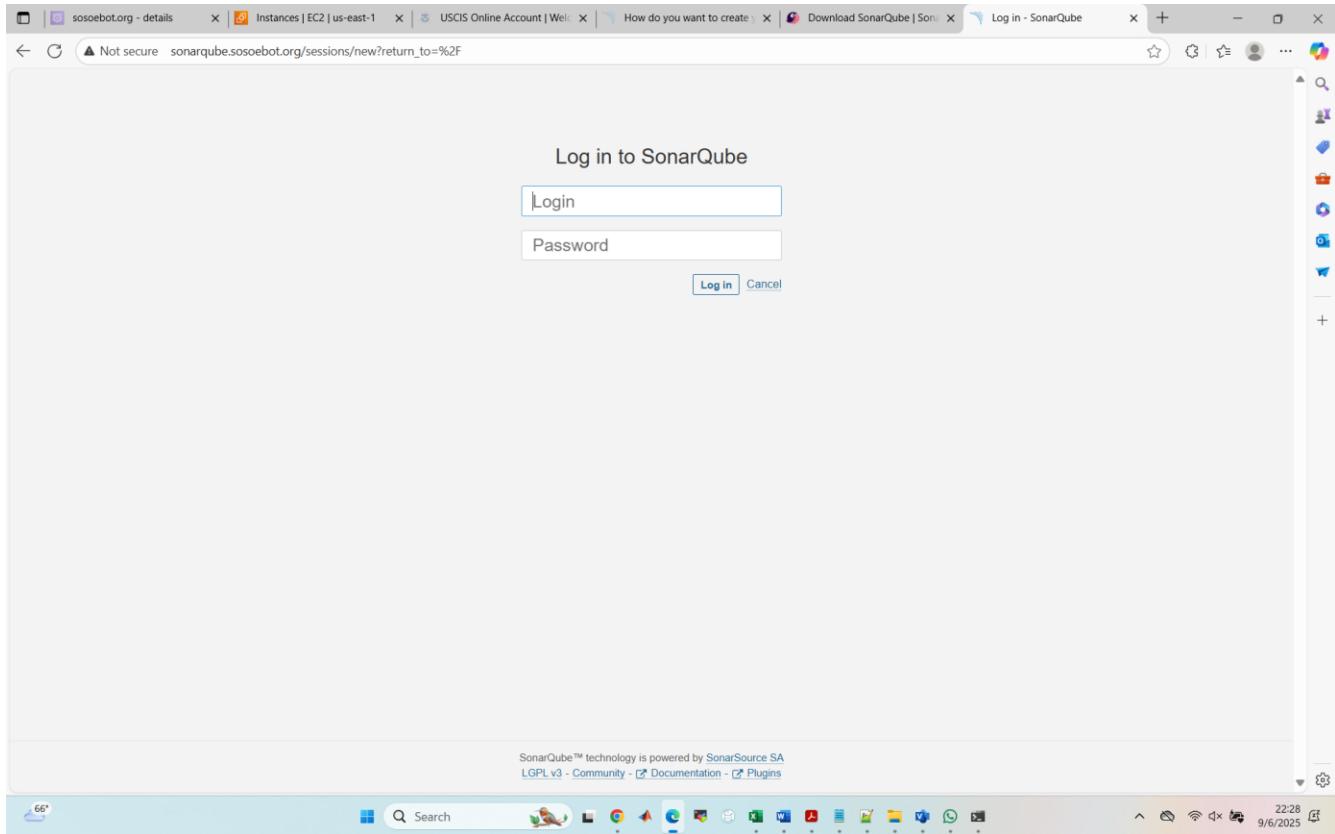


```
root@ip-172-31-17-56:/home/ubuntu# sudo vim /etc/nginx/sites-enabled/ddsonarqube
root@ip-172-31-17-56:/home/ubuntu# sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl enable nginx
Synchronizing state of nginx.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable nginx
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl restart nginx
root@ip-172-31-17-56:/home/ubuntu# sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: enabled)
   Active: active (running) since Sun 2025-09-07 02:09:27 UTC; 1min 23s ago
     Docs: man:nginx(8)
  Process: 2808 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
  Process: 2810 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
    Main PID: 2811 (nginx)
       Tasks: 3 (limit: 4667)
      Memory: 2.4M (peak: 2.7M)
        CPU: 12ms
       CGroup: /system.slice/nginx.service
               ├─2811 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
               ├─2812 "nginx: worker process"
               └─2813 "nginx: worker process"

Sep 07 02:09:27 ip-172-31-17-56 systemd[1]: Starting nginx.service - A high performance web server and a reverse proxy >
Sep 07 02:09:27 ip-172-31-17-56 systemd[1]: Started nginx.service - A high performance web server and a reverse proxy s>
lines 1-17/17 (END)
```

Use the configured domain

You can now visit <http://sonarqube.sosoebot.org> in your web browser. You'll be greeted with the SonarQube web interface.



The configured domain sonarqube.onecloudhelper.com is working perfectly.

Wait, seems there is a problem ahead, let's solve it. There seems to be a problem ahead

Problem:

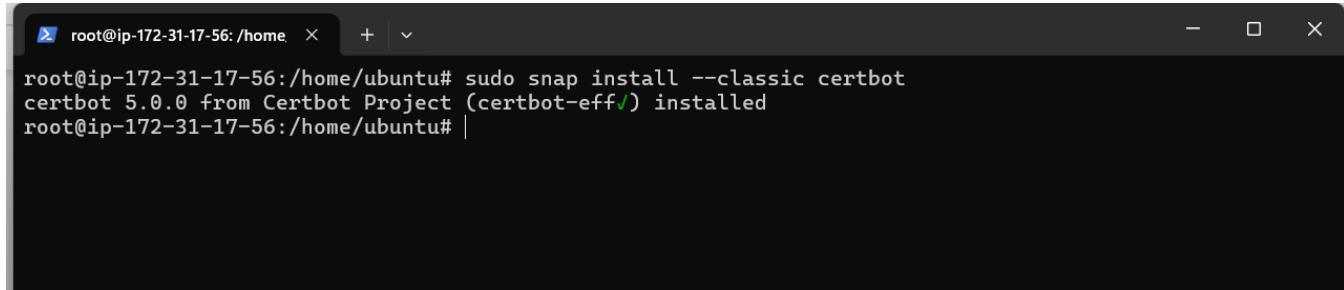
But the domain is working on HTTP protocol and we need to make it secure by making it work with the HTTPS protocol with SSL certificate.

STEP 5: Secure with SSL using Certbot

We will Install SSL certificate for maximum security. We will use the free SSL certificate provided by Certbot for our sub-domain sonarqube.sosoebot.org. Let's see how we will do this.

Part 1: Installing certbot

```
sudo snap install --classic certbot
```



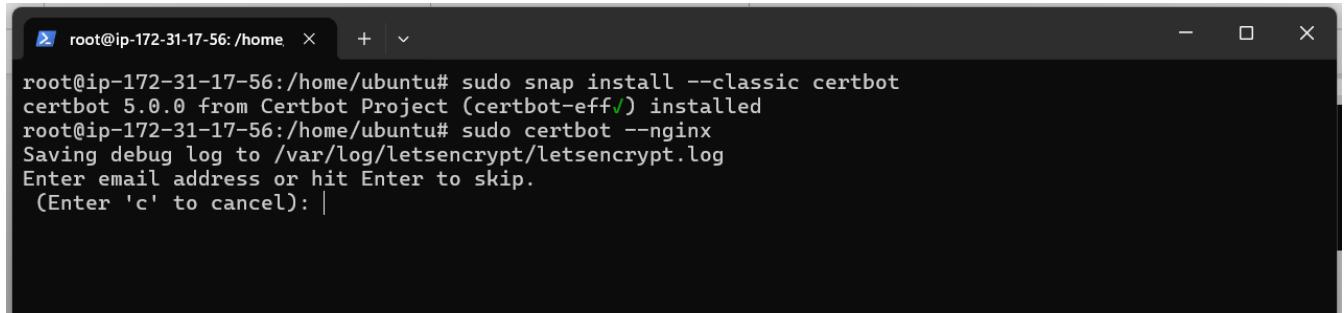
A terminal window titled "root@ip-172-31-17-56:/home" showing the command "sudo snap install --classic certbot" being run. The output indicates that certbot 5.0.0 from Certbot Project (certbot-eff) has been successfully installed.

```
root@ip-172-31-17-56:/home/ubuntu# sudo snap install --classic certbot
certbot 5.0.0 from Certbot Project (certbot-eff) installed
root@ip-172-31-17-56:/home/ubuntu# |
```

Part 2: Obtain and install an SSL certificate on the subdomain

Generating SSL certificate for the required domain. Please read the asked question carefully and answer them accordingly.

```
sudo certbot --nginx
```



A terminal window titled "root@ip-172-31-17-56:/home" showing the command "sudo certbot --nginx" being run. The output shows the process of saving a debug log to /var/log/letsencrypt/letsencrypt.log and prompting for an email address.

```
root@ip-172-31-17-56:/home/ubuntu# sudo snap install --classic certbot
certbot 5.0.0 from Certbot Project (certbot-eff) installed
root@ip-172-31-17-56:/home/ubuntu# sudo certbot --nginx
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Enter email address or hit Enter to skip.
(Enter 'c' to cancel): |
```

Enter your email address. I will enter "ebotsmith@gmail.com"

```
root@ip-172-31-17-56:/home/ubuntu# sudo snap install --classic certbot
certbot 5.0.0 from Certbot Project (certbot-eff) installed
root@ip-172-31-17-56:/home/ubuntu# sudo certbot --nginx
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Enter email address or hit Enter to skip.
(Enter 'c' to cancel): ebotsmith@gmail.com

-----
Please read the Terms of Service at:
https://letsencrypt.org/documents/LE-SA-v1.5-February-24-2025.pdf
You must agree in order to register with the ACME server. Do you agree?
-----
(Y)es/(N)o: |
```

Enter "Y" and press ENTER

```
root@ip-172-31-17-56:/home# Enter email address or hit Enter to skip.
(Enter 'c' to cancel): ebotsmith@gmail.com

-----
Please read the Terms of Service at:
https://letsencrypt.org/documents/LE-SA-v1.5-February-24-2025.pdf
You must agree in order to register with the ACME server. Do you agree?
-----
(Y)es/(N)o: Y

-----
Would you be willing, once your first certificate is successfully issued, to
share your email address with the Electronic Frontier Foundation, a founding
partner of the Let's Encrypt project and the non-profit organization that
develops Certbot? We'd like to send you email about our work encrypting the web,
EFF news, campaigns, and ways to support digital freedom.
-----
(Y)es/(N)o: |
```

Enter "N" and press ENTER

```
root@ip-172-31-17-56:/home# -----
Would you be willing, once your first certificate is successfully issued, to
share your email address with the Electronic Frontier Foundation, a founding
partner of the Let's Encrypt project and the non-profit organization that
develops Certbot? We'd like to send you email about our work encrypting the web,
EFF news, campaigns, and ways to support digital freedom.
-----
(Y)es/(N)o: N
Account registered.

Which names would you like to activate HTTPS for?
We recommend selecting either all domains, or all domains in a VirtualHost/server block.
-----
1: sonarqube.sosobot.org
-----
Select the appropriate numbers separated by commas and/or spaces, or leave input
blank to select all options shown (Enter 'c' to cancel): |
```

Enter "1" and press Enter

```

root@ip-172-31-17-56:/home ~ + -
Successfully received certificate.
Certificate is saved at: /etc/letsencrypt/live/sonarqube.sosoebot.org/fullchain.pem
Key is saved at: /etc/letsencrypt/live/sonarqube.sosoebot.org/privkey.pem
This certificate expires on 2025-12-06.
These files will be updated when the certificate renews.
Certbot has set up a scheduled task to automatically renew this certificate in the background.

Deploying certificate
Successfully deployed certificate for sonarqube.sosoebot.org to /etc/nginx/sites-enabled/ddsonarqube
Congratulations! You have successfully enabled HTTPS on https://sonarqube.sosoebot.org

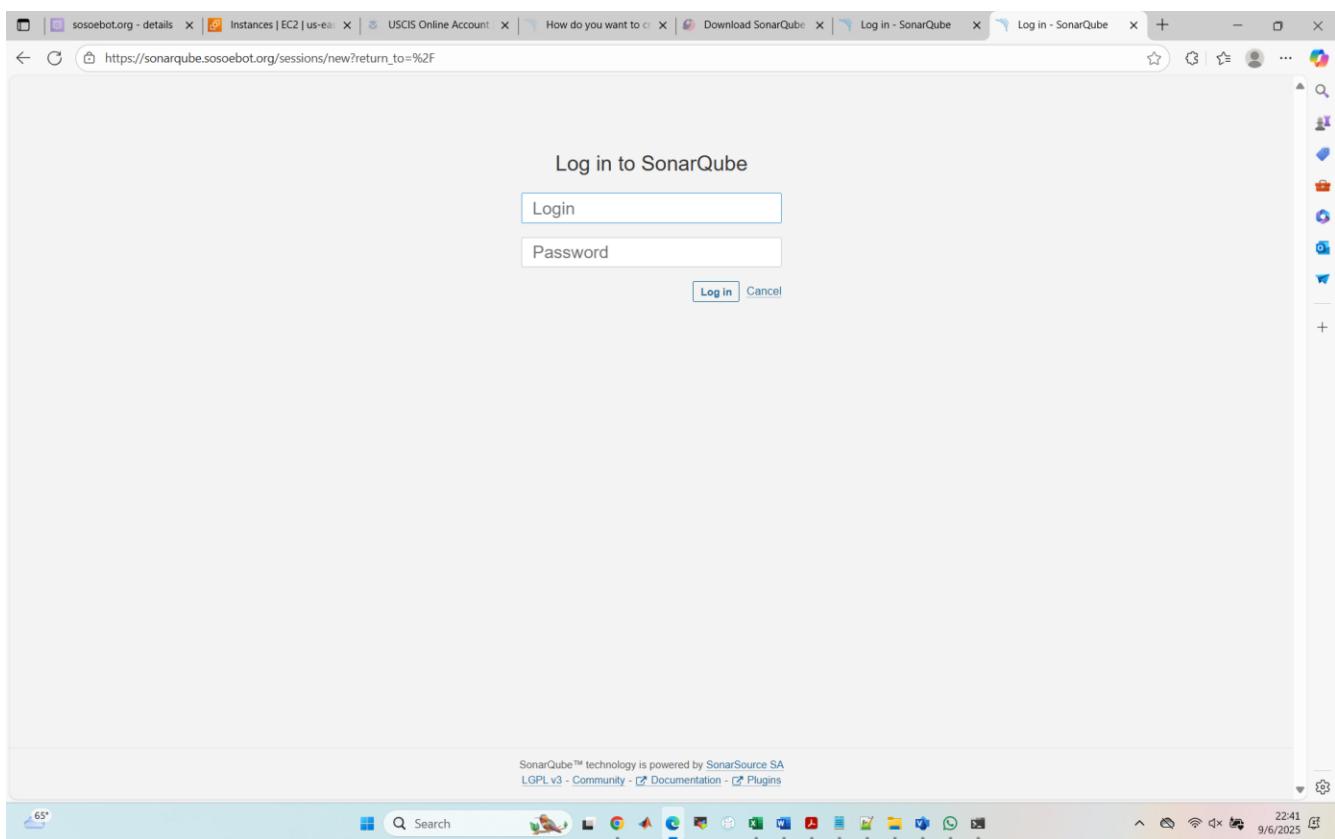
-----
If you like Certbot, please consider supporting our work by:
 * Donating to ISRG / Let's Encrypt: https://letsencrypt.org/donate
 * Donating to EFF: https://eff.org/donate-le
-----
root@ip-172-31-17-56:/home/ubuntu# |

```

Success message that the SSL certificate has been deployed on the domain sonarqube.sosoebot.org

Part 3: Verify if Certificate deployed successfully

Now if we hit the domain **sonarqube.sosoebot.org**, it automatically redirects to HTTPS.



Now you can access SonarQube through your domain name or EC2 public IP address via Nginx (on port 80 or 443 since SSL has been configured).