intelligaia

Sonarqube Installation and Setup

By

Ashutosh Gupta

DevOps Engineer

DevOps/Networking Department

Intelligaia Technologies

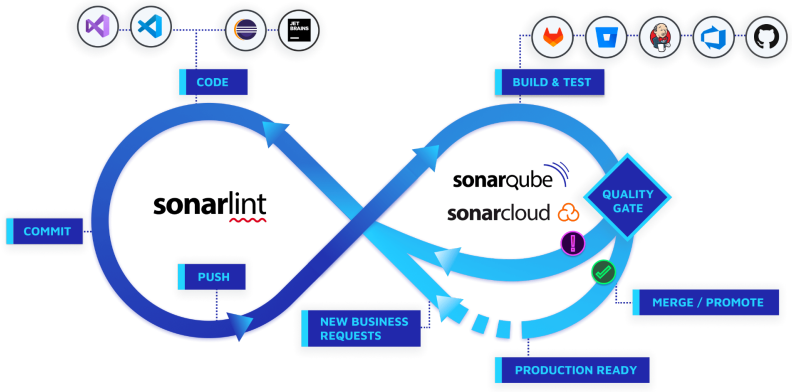
	Author	Version	Change Reference
21/08/2023	Ashutosh Gupta	1.0	Creation of document – Draft

Table Of contents

1. Introduction:

- SonarQube is a self-managed, automatic code review tool that systematically helps you deliver clean code. As a core element of our Sonar solution, SonarQube integrates into your existing workflow and detects issues in your code to help you perform continuous code inspections of your projects. The tool analyses 30+ different programming languages and integrates into your CI pipeline and DevOps platform to ensure that your code meets high-quality standards.
- Writing clean code
- Clean Code is the standard for all code that results in secure, reliable, and maintainable software therefore, writing clean code is essential to maintaining a healthy codebase. This applies to all code: source code, test code, infrastructure as code, glue code, scripts, and more.
- Sonar's Clean as You Code approach eliminates many of the pitfalls that arise from reviewing code at a late stage in the development process. The Clean as You Code approach uses your quality gate to alert/inform you when there's something to fix or review in your new code (code that has been added or changed), allowing you to maintain high standards and focus on code quality.

- Developing with Sonar



- The Sonar solution performs checks at every stage of the development process:

- SonarLint provides immediate feedback in your IDE as you write code so you can find and fix issues before a commit.
- SonarQube's PR analysis fits into your CI/CD workflows with SonarQube's PR analysis and use of quality gates.
- Quality gates keep code with issues from being released to production, a key tool in helping you incorporate the Clean as You Code methodology.
- The Clean as You Code approach helps you focus on submitting new, clean code for production, knowing that your existing code will be improved over time.

2. Prerequisites:

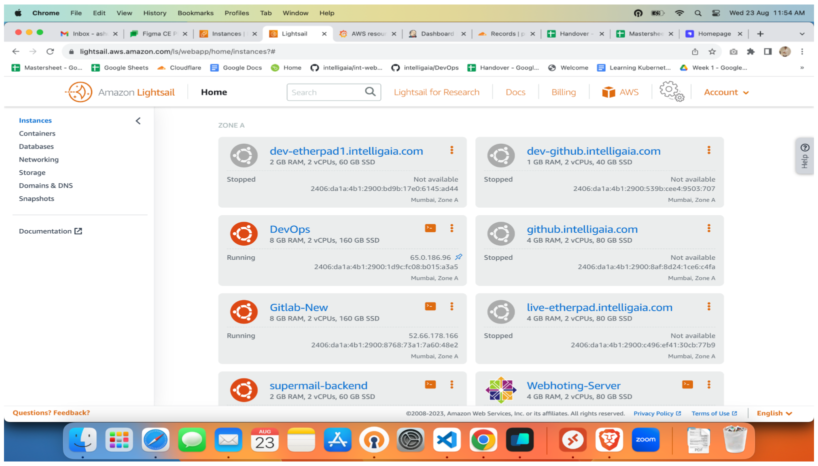
- We have setup the Sonarqube with docker-compose
- Prerequisites are for setting up sonarqube with docker-compose are listed below
- Linux Server (Ubuntu 20.04 LTS)

  We have create lightsail instance with the name DevOps in Mumbai regions with configuration

  RAM 8G

  2vCPUs

  160GB SSD storage

- 

ZONE A

dev-etherpad1.intelligaia.com
2 GB RAM, 2 vCPUs, 60 GB SSD
Stopped — Not available
2406:da1a:4b1:2900:bd9b:17e0:6145:ad44
Mumbai, Zone A

dev-github.intelligaia.com
1 GB RAM, 2 vCPUs, 40 GB SSD
Stopped — Not available
2406:da1a:4b1:2900:539b:cee4:9503:707
Mumbai, Zone A

DevOps
8 GB RAM, 2 vCPUs, 160 GB SSD
Running — 65.0.186.96
2406:da1a:4b1:2900:1d9c:fc08:b015:a3a5
Mumbai, Zone A

github.intelligaia.com
4 GB RAM, 2 vCPUs, 80 GB SSD
Stopped — Not available
2406:da1a:4b1:2900:8af:8d24:1ce6:c4fa
Mumbai, Zone A

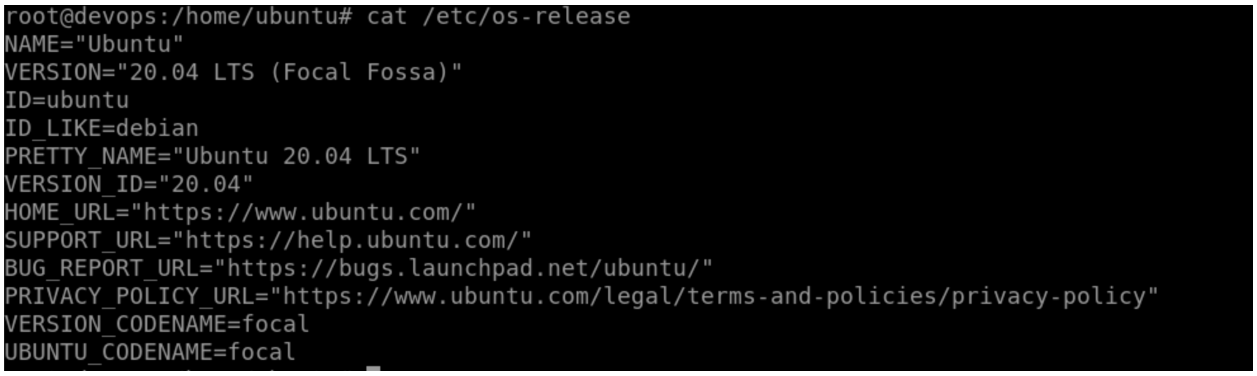
Gitlab-New
8 GB RAM, 2 vCPUs, 160 GB SSD
Running — 52.66.178.166
2406:da1a:4b1:2900:8768:73a1:7a60:48e2
Mumbai, Zone A

live-etherpad.intelligaia.com
4 GB RAM, 2 vCPUs, 80 GB SSD
Stopped — Not available
2406:da1a:4b1:2900:c496:ef41:30cb:77b9
Mumbai, Zone A

supermail-backend
2 GB RAM, 2 vCPUs, 60 GB SSD

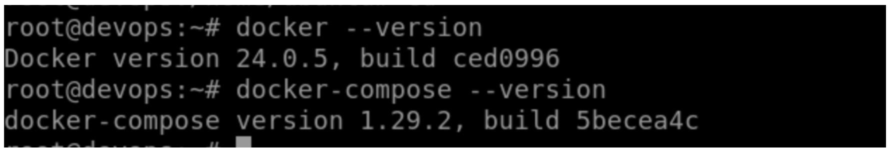
Webhoting-Server
4 GB RAM, 2 vCPUs, 80 GB SSD

Here we have selected Ubuntu 20.04 LTS



```
root@devops:/home/ubuntu# cat /etc/os-release
NAME="Ubuntu"
VERSION="20.04 LTS (Focal Fossa)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 20.04 LTS"
VERSION_ID="20.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=focal
UBUNTU_CODENAME=focal
```

- Docker and Docker-compose installed on ubuntu 20.04



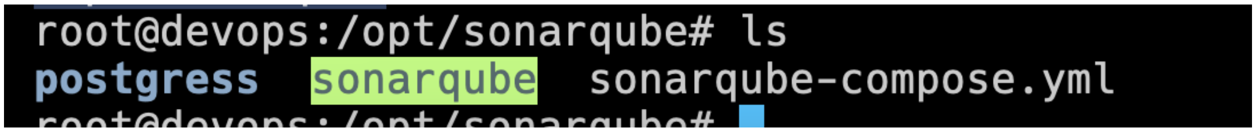
```
root@devops:~# docker --version
Docker version 24.0.5, build ced0996
root@devops:~# docker-compose --version
docker-compose version 1.29.2, build 5becea4c
```

We can use below link to install docker and docker-compose on ubuntu

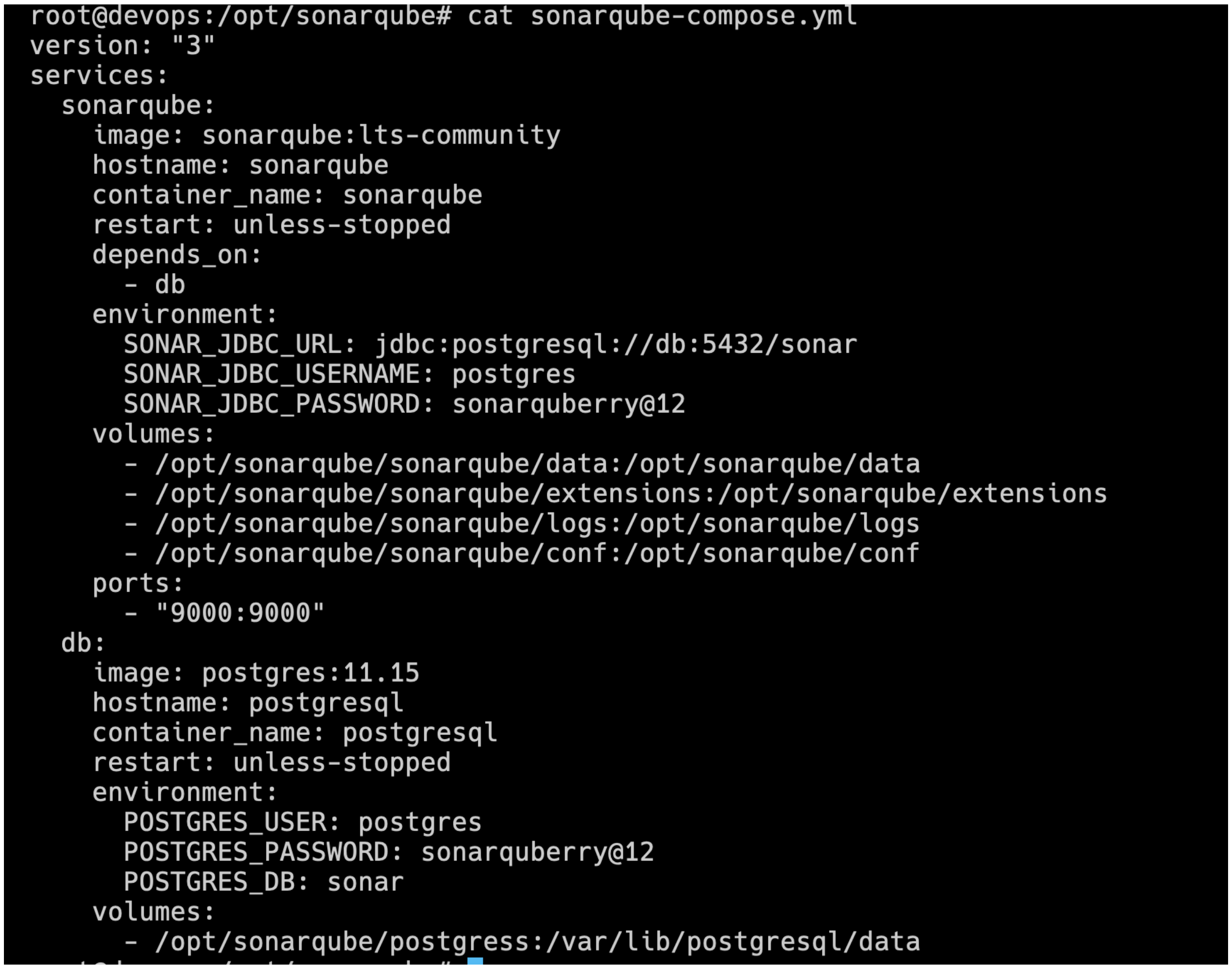
```
Unset
https://docs.docker.com/desktop/install/windows-install/

https://docs.docker.com/compose/install/
```

3. Installation

- Here for installation of sonarqube we are using database postgres
- We have setup sonarqube with docker-compose persistent volume
- The persistent volume location is /opt
- We have created directory /opt/sonarqube



```
root@devops:/opt/sonarqube# ls
postgress  sonarqube  sonarqube-compose.yml
```



```
root@devops:/opt/sonarqube# cat sonarqube-compose.yml
version: "3"
services:
  sonarqube:
    image: sonarqube:lts-community
    hostname: sonarqube
    container_name: sonarqube
    restart: unless-stopped
    depends_on:
      - db
    environment:
      SONAR_JDBC_URL: jdbc:postgresql://db:5432/sonar
      SONAR_JDBC_USERNAME: postgres
      SONAR_JDBC_PASSWORD: sonarquberry@12
    volumes:
      - /opt/sonarqube/sonarqube/data:/opt/sonarqube/data
      - /opt/sonarqube/sonarqube/extensions:/opt/sonarqube/extensions
      - /opt/sonarqube/sonarqube/logs:/opt/sonarqube/logs
      - /opt/sonarqube/sonarqube/conf:/opt/sonarqube/conf
    ports:
      - "9000:9000"
  db:
    image: postgres:11.15
    hostname: postgresql
    container_name: postgresql
    restart: unless-stopped
    environment:
      POSTGRES_USER: postgres
      POSTGRES_PASSWORD: sonarquberry@12
      POSTGRES_DB: sonar
    volumes:
      - /opt/sonarqube/postgress:/var/lib/postgresql/data
```

- Cat sonarqube-compose.yml

Unset
```
version: "3"
services:
```

```
sonarqube:
 image: sonarqube:lts-community
 hostname: sonarqube
 container_name: sonarqube
 restart: unless-stopped
 depends_on:
  - db
 environment:
  SONAR_JDBC_URL: jdbc:postgresql://db:5432/sonar
  SONAR_JDBC_USERNAME: postgres
  SONAR_JDBC_PASSWORD: sonarquberry@12
 volumes:
  - /opt/sonarqube/sonarqube/data:/opt/sonarqube/data
  - /opt/sonarqube/sonarqube/extensions:/opt/sonarqube/extensions
  - /opt/sonarqube/sonarqube/logs:/opt/sonarqube/logs
  - /opt/sonarqube/sonarqube/conf:/opt/sonarqube/conf
 ports:
  - "9000:9000"
db:
 image: postgres:11.15
 hostname: postgresql
 container_name: postgresql
 restart: unless-stopped
 environment:
  POSTGRES_USER: postgres
  POSTGRES_PASSWORD: sonarquberry@12
  POSTGRES_DB: sonar
 volumes:
  - /opt/sonarqube/postgress:/var/lib/postgresql/data
```

- We need to create all the necessary directories and files for persistent volumes

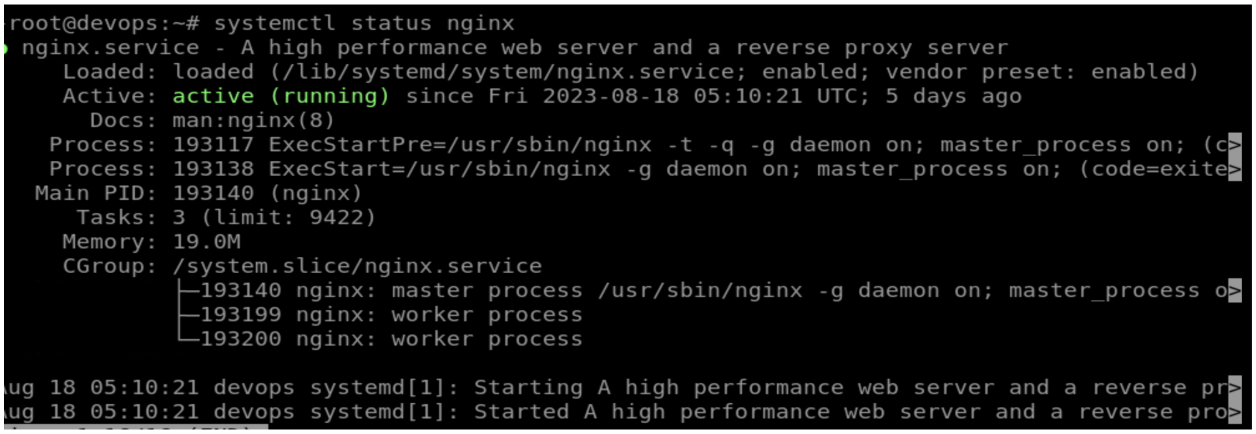
4. Configuration

At this point sonarqube is accessible to ip and port

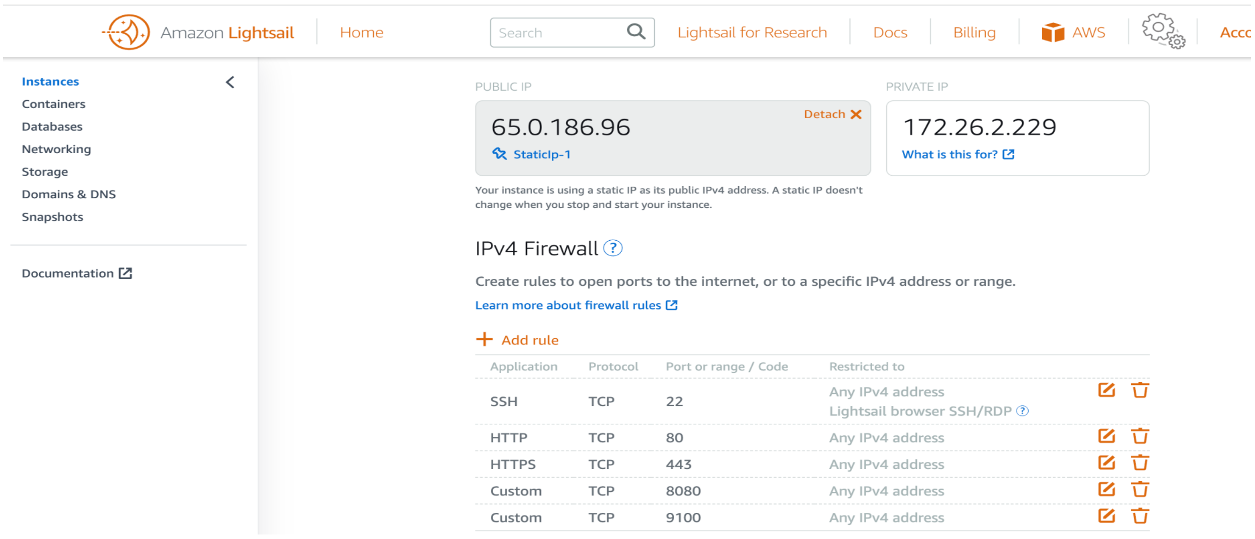
- We need to make sure that is accessible to sonarqube.intelligaia.com
- So we need to setup reverse proxy
- Nginx can be installed using below command

Unset
```
apt update
apt install nginx -y
```

- After successfully installing create nginx configuration file we need to open 80 and 443 ports so that we could access nginx service



```
root@devops:~# systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2023-08-18 05:10:21 UTC; 5 days ago
       Docs: man:nginx(8)
    Process: 193117 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (c>
    Process: 193138 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exite>
   Main PID: 193140 (nginx)
      Tasks: 3 (limit: 9422)
     Memory: 19.0M
     CGroup: /system.slice/nginx.service
             ├─193140 nginx: master process /usr/sbin/nginx -g daemon on; master_process o>
             ├─193199 nginx: worker process
             └─193200 nginx: worker process

ug 18 05:10:21 devops systemd[1]: Starting A high performance web server and a reverse pr>
ug 18 05:10:21 devops systemd[1]: Started A high performance web server and a reverse pro>
```



PUBLIC IP

65.0.186.96 Detach ✕
StaticIp-1

PRIVATE IP

172.26.2.229
What is this for?

Your instance is using a static IP as its public IPv4 address. A static IP doesn't change when you stop and start your instance.

IPv4 Firewall

Create rules to open ports to the internet, or to a specific IPv4 address or range.

Learn more about firewall rules

+ Add rule

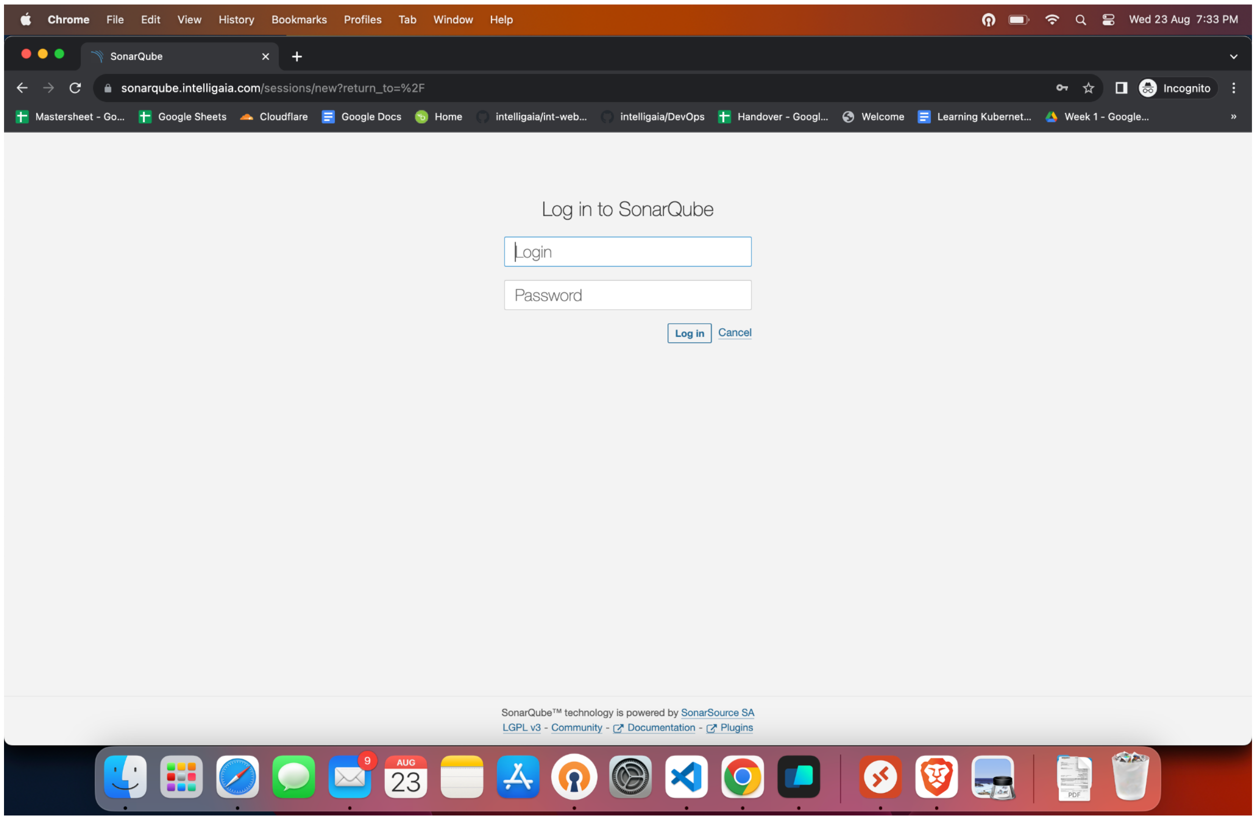
Application	Protocol	Port or range / Code	Restricted to
SSH	TCP	22	Any IPv4 address Lightsail browser SSH/RDP
HTTP	TCP	80	Any IPv4 address
HTTPS	TCP	443	Any IPv4 address
Custom	TCP	8080	Any IPv4 address
Custom	TCP	9100	Any IPv4 address

Sonarqube nginx configuration is kept at **/etc/nginx/conf.d/sonarqube.conf**

- Cat cat /etc/nginx/conf.d/sonarqube.conf

```
Unset
server {
  server_name sonarqube.intelligaia.com;
  access_log off;

  location / {

    proxy_pass     http://127.0.0.1:9000;

    proxy_set_header  Host      $host;
    proxy_set_header  X-Real-IP    $remote_addr;
    proxy_set_header  X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header  X-Forwarded-Proto http;
    proxy_max_temp_file_size 0;
    proxy_connect_timeout   150;
    proxy_send_timeout    100;
    proxy_read_timeout    100;

    proxy_buffer_size     8k;
    proxy_buffers       4 32k;
    proxy_busy_buffers_size  64k;
    proxy_temp_file_write_size 64k;

      }



  listen 443 ssl; # managed by Certbot
  ssl_certificate /etc/letsencrypt/live/sonarqube.intelligaia.com/fullchain.pem;
# managed by Certbot
  ssl_certificate_key
/etc/letsencrypt/live/sonarqube.intelligaia.com/privkey.pem; # managed by Certbot
  include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
  ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

}
server {
  if ($host = sonarqube.intelligaia.com) {
    return 301 https://$host$request_uri;
  } # managed by Certbot
  server_name sonarqube.intelligaia.com;
  listen 80;
  return 404; # managed by Certbot


}
```

- Now the sonarqube is accessible to sonarqube.intelligaia.com
- We can use active directory credentials to login for that we need to configure ldap


sonarqube.intelligaia.com/sessions/new?return_to=%2F
Log in to SonarQube
Login
Password
Log in
Cancel
SonarQube™ technology is powered by SonarSource SA
LGPL v3 - Community - Documentation - Plugins

5. Integration

- We need to configure ldap so we need to create sonar.properties inside /opt/sonarqube/sonarqube/conf
- Cat sonar.properties

```
Unset
# LDAP configuration

# General Configuration
sonar.security.realm=LDAP
ldap.url=ldap://dc1.ad.intelligaia.com:389
ldap.bindDn=CN=sonarqube-svc,OU=ServiceAccounts,DC=ad,DC=intelligaia,DC=com
ldap.bindPassword=mEhn@7k@r#630
ldap.authentication=simple

# User Configuration
ldap.user.baseDn=DC=ad,DC=intelligaia,DC=com
ldap.user.request=(sAMAccountName={0})
ldap.user.realNameAttribute=cn
ldap.user.emailAttribute=mail

# Group Configuration
ldap.group.baseDn=CN=sonarqube-admin,OU=ServiceAccounts,DC=ad,DC=intelligaia,DC
=com
ldap.group.request=(&(objectClass=group)(member={dn}))
```

- You need to add necessary parameters such as
  sonar.security.realm=LDAP
  ldap.url=ldap://dc1.ad.intelligaia.com:389
  ldap.bindDn=CN=sonarqube-svc,OU=ServiceAccounts,DC=ad,DC=intelligaia,DC=com
  ldap.bindPassword=mEhn@7k@r#630
  ldap.authentication=simple
  ldap.user.baseDn=DC=ad,DC=intelligaia,DC=com
  ldap.user.request=(sAMAccountName={0})
  ldap.user.realNameAttribute=cn
  ldap.user.emailAttribute=mail
  ldap.group.baseDn=CN=sonarqube-admin,OU=ServiceAccounts,DC=ad,DC=intelligaia,DC=com
  ldap.group.request=(&(objectClass=group)(member={dn}))

- These are the ldap configuration could be get from Active directory server