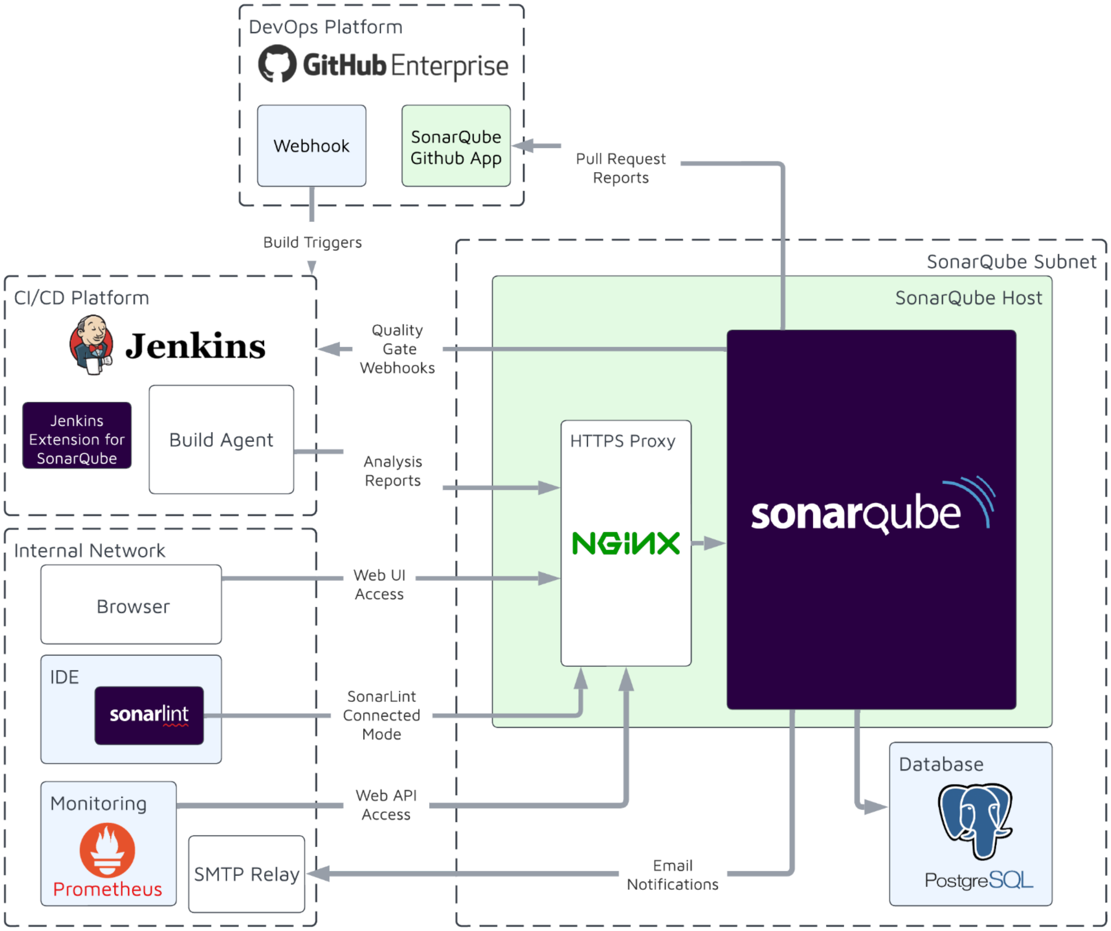
sonar

Reference Architecture:

VM-based SonarQube Enterprise Edition

This architecture describes the setup of a SonarQube Enterprise Editon instance that will support up to 50 million lines of code under normal usage patterns in a non-high availability setup. This architecture covers the following components:

- A virtual machine host with **SonarQube Enterprise Edition** installed and an **nginx** HTTPS proxy
- **PostgreSQL** database on a dedicated host
- Analysis integrated with **Jenkins CI**
- Pull request reporting and authentication through **GitHub Enterprise**
- Monitoring with **Prometheus**
- Outbound email notifications using an SMTP relay



Component Detail

This architecture favors use of open-source components when available. These may be substituted with other similarly-capable components and it is recommended that organizations use components that they are comfortable supporting.

SonarQube Host

The SonarQube Host will have the SonarQube software installed as well as nginx acting as an HTTPS proxy.

Specification

VM Configuration	AWS EC2	Azure VM	GCE
8 vCPU 16 GB RAM 50GB SSD Local Storage	c5d.2xlarge	F8s_v2	c3-highcpu-8

Networking

Source/Destination	Direction	Port (Protocol)	Notes
SonarQube host	Outbound	5432	Database
	Outbound	25 (SMTP)	Email notifications
Internal network (user desktops)	Inbound	443 (HTTPS)	Inbound web and API traffic
CI platform (Jenkins)	Inbound	443 (HTTPS)	Analysis reports
	Outbound	443 (HTTPS)	Quality gate webhooks
DevOps platform (GitHub Enterprise)	Outbound	443 (HTTPS)	Pull Request reports

Software

- OS - Ubuntu Server (or other Linux distribution)
- OpenJDK 17
- SonarQube Enterprise Edition
  - Four Compute Engine workers (see documentation)
- nginx
  - Configured as a reverse proxy between incoming traffic and SonarQube port 9000.
  - Secured with SSL. Use of self-signed SSL certificates will require installation of the certificate on all CI build agents, and developer desktops using SonarLint
  - May be substituted with other reverse proxy (ex. haproxy) or a solution from a cloud provider, such as an AWS Application Load Balancer (ALB)

Reference

- Prerequisites
- Installing the server
- Securing the server behind a proxy

Database

This architecture utilizes a dedicated PostgreSQL database installed on a separate host.

VM Configuration	AWS RDS	Azure SQL	Google Cloud SQL
4 vCPU 16 GB RAM	db.t3.xlarge	B4ms	4 vCPU 16 GB memory
All: 150 GB table space			

Database requirements can widely vary based on the usage patterns of each SonarQube installation. It is important that database resources are closely monitored and adjusted as needed.

PostgreSQL may be substituted with other supported database platforms.

Reference

- Supported database platforms

DevOps Platform

SonarQube submits analysis reports back to pull requests to integrate with code review processes. This functionality is enabled in GitHub Enterprise using a GitHub App.

GitHub Enterprise may be substituted with other supported DevOps platforms without changes to other components in this architecture.

Reference

- Github integration

Authentication

It is recommended that authentication and authorization be handled through an external identity provider. The architecture utilizes the GitHub App to authorize users and synchronize access to SonarQube projects.

Other external identity providers such as SAML may be substituted. *Features such as group and permission synchronization are not available for all authentication methods.*

Reference

- Github authentication

Continuous Integration

Automated analysis of source code is enabled through the installation of the various SonarScanners into continuous integration pipelines. When using Jenkins, the **Jenkins Extension for SonarQube** manages the installation of the scanners and provides functionality to ease the integration of Sonar analysis into build pipelines.

If using other CI platforms, they may be used without changes to other components in this architecture.

Reference

- Jenkins extension for SonarQube

Monitoring

SonarQube exposes endpoints that are easy to monitor using Prometheus or other monitoring solutions. In addition to the overall system health of both the SonarQube host and database, it is recommended to monitor SonarQube's Compute Engine performance statistics to ensure incoming analyses are being promptly processed.

Reference

- Monitoring SonarQube

Email

Users can be notified of new issues and events via email. SonarQube will deliver these notifications through an SMTP mail relay. The volume of emails is low, dependent on the number of users subscribed, and a dedicated SMTP server is typically not required.

Reference

- SonarQube notifications

Resiliency

As a single-host installation, this architecture relies on robust monitoring, automated backups of the database, and a rapid recovery process to maximize resiliency. If high availability is critical, SonarQube Data Center Edition is recommended.

Scalability

This architecture is designed to support typical production usage for up to 50 million lines of code. Beyond this, it is recommended that organizations use SonarQube Data Center Edition to support high-volume workloads.

The following use cases are considered outside of "normal usage" and may require additional capacity:

High-frequency analysis

Normal usage assumes a daily scan of main branches and analysis of several pull requests. Scanning code more frequently may require an increase in the number of Compute Engine workers as well as additional memory and CPU resources allocated to SonarQube's Compute Engine process. Monitoring of the Compute Engine process will ensure that your installation can keep up with demand.

Large repositories

This architecture assumes analyzed repositories average 50,000 lines of code. If your organization is scanning a majority of very large repositories (where the repositories average 500,000 lines of code or more), additional memory and CPU resources may be required for SonarQube's Compute Engine process.

Heavy API integration

SonarQube exposes a REST-based API for reporting and automation of administration tasks. This architecture assumes occasional use of this API. Heavy use of this API may require the allocation of additional memory and CPU resources to SonarQube's web process.

Third-party plugins

This architecture assumes that no third-party plugins are in use. As these extensions are developed by non-sponsored developers, their impact on the performance of a SonarQube instance varies based on the function being performed and the quality of the implementation. It is recommended that the use of third-party plugins is carefully considered and monitored for performance throughout the life of your SonarQube implementation.