Università di Genova
DIBRIS DIPARTIMENTO
DI INFORMATICA, BIOINGEGNERIA,
ROBOTICA E INGEGNERIA DEI SISTEMI

Exam 2024

Virtualization and Cloud Computing

Giacomo Longo, Enrico Russo

Revisions

This is the first revision of this exam project

Structure

- Context
- Project rules
- Tasks
- Examination procedure

Università di Genova
DIBRIS DIPARTIMENTO
DI INFORMATICA, BIOINGEGNERIA,
ROBOTICA E INGEGNERIA DEI SISTEMI

Context

Why

- Representative example of a real-world infrastructure
- Leveraging state of the art technologies
- Allows you to get your feet wet with plenty challenging problems
- Allows you to learn to read technical documentation and implements third party requirements
- Very close to what some professionals do:
  - DevOps engineers
  - Cloud engineers
  - System administrators
  - System integration companies
  - ...

What

- You will deploy a Forgejo instance with Single-Sign-On
- Such deployment will be
  - Reproducible
  - Automated
  - Scalable thanks to Docker Swarm
  - Centrally monitored

Università di Genova
DIBRIS DIPARTIMENTO
DI INFORMATICA, BIOINGEGNERIA,
ROBOTICA E INGEGNERIA DEI SISTEMI

Project rules

Version control

- Project should be version controlled via *git*
- Repository **must** be from GitHub classroom
- Commits should be plenty and meaningful
  - We don't care about memes in the commit history
    - Unless they are unfunny
  - It shows us the process and reasoning
- **DO NOT LEAVE ANY SECRETS INSIDE OF VERSION CONTROL**
  - SSH keys, tokens, credentials
  - If they are necessary, put them in files that are .gitignore
  - Ansible-vault is your friend

Template

What to do

- Leverage the template we give you
- Template is automatically setup for you via GitHub classroom
- Use vagrant and the make targets we gave you
- Read the original README.md to see usage instructions

Ansible

Project should be able to automatically be installed with no human intervention from two empty Ubuntu VMs (from Vagrant).

I.e. for the base template:

- Running *make setup-all*

Should allow the users to access every service belonging to the project.

This will be **automatically checked** by a **program** so please ensure your project can be installed in this automatic and unattended way.

Ansible

Principles

- Never hardcode IPs, hostnames, network interfaces, and configuration values.
  Use variables
  - Leverage magic variables such as groups['all'] and inventory_hostname in your conditionals
- Ensure that you are not running operations where it's unnecessary:
  - If you copy a file to an NFS share you don't need to copy it from two nodes
  - If you are running Docker swarm commands, you only need to do it from the master
- Leverage proper collections / tasks instead of relying on commands
- Do not use docker exec (and equivalents) anywhere

Ansible

During development, do whatever you want, but in the final version

do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control do not leave secrets in version control

Docker

- Use the images versions we indicate on these slides.
- Whenever an image is custom (i.e. built by you) it **must** be hosted on your registry, accessible by **both** nodes (see later on).
- Unless a container requires access to the Docker Swarm master API (available only on the first node) do not put placement constraints on services.

Docker

During development, do whatever you want, but in the final version

- Do not use externally forwarded ports (except for the reverse proxy)
- Do not use externally forwarded ports (except for the reverse proxy)
- Do not use externally forwarded ports (except for the reverse proxy)
- Do not use externally forwarded ports (except for the reverse proxy)
- Do not use externally forwarded ports (except for the reverse proxy)
- Do not use externally forwarded ports (except for the reverse proxy)
- Do not use externally forwarded ports (except for the reverse proxy)
- Do not use externally forwarded ports (except for the reverse proxy)

Università di Genova
DIBRIS DIPARTIMENTO
DI INFORMATICA, BIOINGEGNERIA,
ROBOTICA E INGEGNERIA DEI SISTEMI

Tasks

Tasks

Notice on ordering

- You will be presenting the **final** project
  - And graded only on that
- You can do these tasks in any order you like
  - Respecting their dependencies (!)
- You can add more entries in your ansible files than the ones already present in the templates

Docker

- Add Docker APT repository key
- Add Docker APT repository
- Install Docker and the docker compose plugin

Docker Swarm

- Initialize Docker Swarm on the manager node
- Join the second node as a Docker Swarm worker node

NFS

- Install NFS Client and Server
- Configure NFS to export /data to the other node
- Ensure that
  a. The server is running at boot
  b. The client is mounting the share at boot
  c. The playbook does not require a reboot to utilize the share

Registry

- Create an htpasswd file for registry authentication
- Configure the registry to require authentication using the htpasswd file
- Configure the registry to expose metrics
- Ensure that both nodes are logged in to the registry

Swarm services

Database

- Deploy a "postgres:17" database
- Use the default entrypoint to create databases and users for
  a. dex
  b. forgejo
  c. grafana
- Ensure that database data is persistent across restarts and changes in node

Swarm services

Traefik certificate generation

- Create a TLS certificate for traefik
- Ensure that its Subject Alternate Name contains the DNS name "*.vcc.internal"

Swarm services

Traefik certificate generation

To verify run `openssl x509 -text -noout -in <cert>`

And look for the following

```
X509v3 extensions:
    X509v3 Subject Alternative Name:
        DNS:*.vcc.internal
```

Swarm services

Traefik

- Deploy public.ecr.aws/docker/library/traefik:v3.2.2
- Expose it on ports 80 and 443
- Configure traefik for service discovery via Docker Swarm labels
- Configure the TLS endpoint on port 443 to use the self signed certificate we generated previously
- Redirect HTTP requests to HTTPS
- Enable access logging and prometheus metrics

Tip: to make internal services reach the xxx.vcc.internal domains you can use the "networks.xxx.aliases" parameter on the traefik service

Swarm services

Dex

- Deploy ghcr.io/dexidp/dex:v2.41.1-alpine
- Make it use the database you created
- Set its administrative credentials
- Enable metrics
- Create clients for
  - forgejo
  - grafana

Swarm services

Forgejo

- Customize the image codeberg.org/forgejo/forgejo:9.0.3
- Use the custom entrypoint found in configs/images/forgejo
  a. Complete the contents of the forgejo_cli function
  b. Complete the database health check
  c. Run the database migration command
  d. Create the administrator user
  e. Wait for forgejo to be alive
  f. Add the certificate from before to the system certificates
  g. Wait for dex to be alive
  h. Create the openid client to use https://auth.vcc.internal as authentication source with the "forgejo" OAuth client

Swarm services

Forgejo

- Customize the forgejo.ini configuration
  a. Use the database we created
  b. Enable metrics
- Ensure that the data **and sessions** are persisted across reboots and movements between nodes
- Expose forgejo via traefik on https://git.vcc.internal

Swarm services

Grafana

- Customize the image docker.io/grafana/grafana:11.4.0
- Use a custom entrypoint
  a. Perform a database health check
  b. Add the certificate from to the system certificates
  c. Wait for dex to be alive
  d. Set environment variables (using "export NAME=value") to use https://auth.vcc.internal as authentication source with the "grafana" OAuth client

Tip: https://grafana.com/docs/grafana/latest/setup-grafana/configure-security/configure-authentication/generic-oauth/

Swarm services

Grafana

- Configure Grafana URL as https://mon.vcc.internal
- Configure Grafana to use the database you created
- Configure Grafana admin credentials
- Enable Grafana metrics
- Enable Grafana provisioning
- Ensure that Grafana data is persisted
- Expose grafana via traefik at https://mon.vcc.internal

Swarm services

Prometheus

- Deploy quay.io/prometheus/prometheus:v3.0.1
- Ensure that Prometheus data is persisted
- Set 14 days as the metrics retention period
- Expose prometheus via traefik on https://prom.vcc.internal

Tip: try to understand what prometheus.yml does for the latter points

Swarm services

Node exporter

- Deploy quay.io/prometheus/node-exporter:v1.8.2

Swarm services

Loki

- Deploy docker.io/grafana/loki:3.3.1
- Ensure that loki data is persistent
- Enable metrics

Swarm services

Promtail

- Deploy docker.io/grafana/promtail:3.2.2
- Make it so it gathers log from your services

Swarm services

Monitoring stack

- Ensure that prometheus scrapes traefik
- Ensure that prometheus scrapes node-exporter
- Ensure that prometheus scrapes grafana
- Ensure that prometheus scrapes forgejo
- Ensure that prometheus scrapes dex
- Ensure that prometheus scrapes the registry on the nodes
    - Tip dockerswarm_sd with role *nodes*
- Using provisioning, create a Grafana dashboard with metrics and logs from traefik
- Using provisioning, create a Grafana dashboard with metrics about the swarm nodes

Use docker swarm service discovery unless impossible

Swarm services

Monitoring stack

- Ensure that prometheus scrapes loki
- Ensure that prometheus scrapes promtail
- Ensure that metrics from nodes can be attributed to a specific node
- Ensure that logs can be attributed to a specific node and/or service
- Using provisioning, create a Grafana dashboard for the monitoring stack
- Using provisioning, create a Grafana dashboard for the logging stack

Mandatory tasks

Do not attempt to give the exam if

- Some service is not accessible via the reverse proxy
- SSO does not work at all
- Monitoring is unavailable for unknown reasons
- We need to perform some manual task to make your project work
- You have no idea what your project does and why

Examination procedure

Groups

- Any size $\in [1, 3]$
- Project is **the same** regardless of size
- Evaluation will take group size into account
- During evaluation **each** member will receive **at least one** question like
  - In *<file>* you wrote *<something>* why is that?
  - In *<file>* this setting might have a problem, do you know why?
  - Why did you *<something>* instead of *<something_else>*?
- I did it because *<website told me/the project I copied from did/it did not work otherwise>* are not valid answers to these questions
- Final mark is shared by the whole group

Exam workflow

- **Before October 2025**
- Ensure your repo is in our GitHub classroom
- Send **both of us** an email (Subject: [VCC24] Exam request)
  - Enrico (enrico.russo@dibris.unige.it)
  - Giacomo (giacomo.longo@dibris.unige.it)
  - Dates and times will be published periodically to Aulaweb
    - (if we find a way to handle bookings via Aulaweb, book there)
- (resend if we forget 😀)
- We will contact you to confirm the exam slot
- Do the UniGE evaluation form for the course
- Do the exam (see next slide)
- Send **both of us** an email with date, names, student ids, and mark

Performance problems

- If your computer is too slow to run the project at the exam
- Ask us **beforehand** to provision your project on our computers
- Exam will be held there

Exam timeline

- Start the project
- Exam starts
- Show us your student id cards (have them ready)
- Give us a presentation
  - We already know the requirements, so do not show us them again
  - Instead, present which challenges you faced, solutions you adopted, reasoning behind your choices (convince us they are correct 😏)
- Questions on the presentation
- Showcase of the project working
- Questions on the project

Evaluation factors

- Repository organization
- Presentation and question answers
- Technical quality
- Time taken
- Group size
- Anti-plagiarism results

With honors designation

- Respond constructively and correctly to our questions
  - We don't like groups with freeloaders
  - We appreciate knowledge more than technical prowess
- Earn a good enough score to be eligible
- We will vote on the designation

Grading

Grade := (n_done / n_tasks) * 30 * oral_correction_factor

oral_correction_factor $\in$ [0.8, 1.2]

UniGe

DIBRIS