ARM compiling through kubernetes backed CI/CD

Summer student two months work sumup

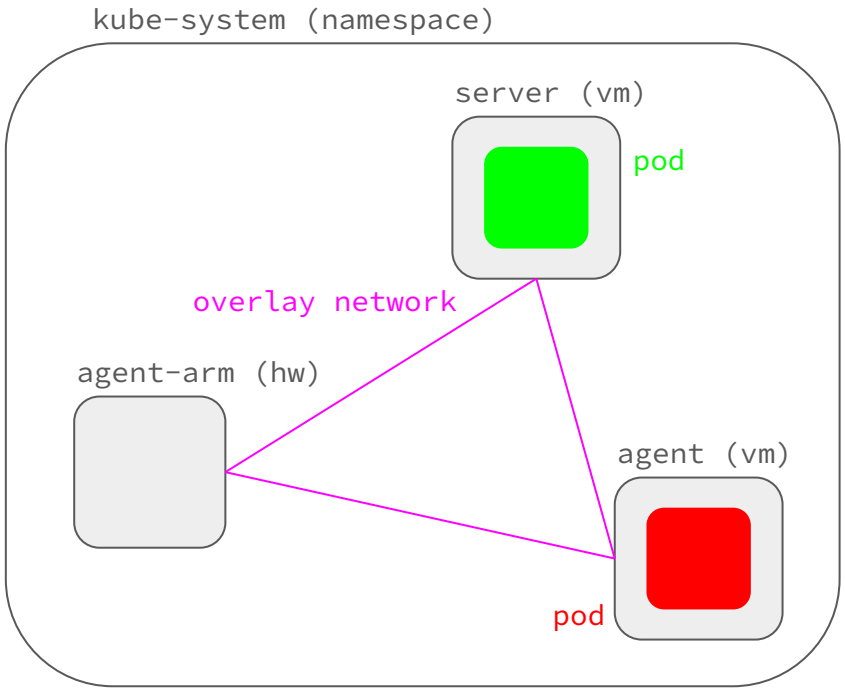
The Kubernetes (k8s) cluster

Components:

- nodes:
  - server (vm)
  - agent (vm)
  - agent-arm (hw)
- pods
- deployments and services
- kubelets
- overlay network
- namespaces

RKE2 installation provides:

- `kubectl` utility
- `kubeconfig` file
- server token to register other nodes

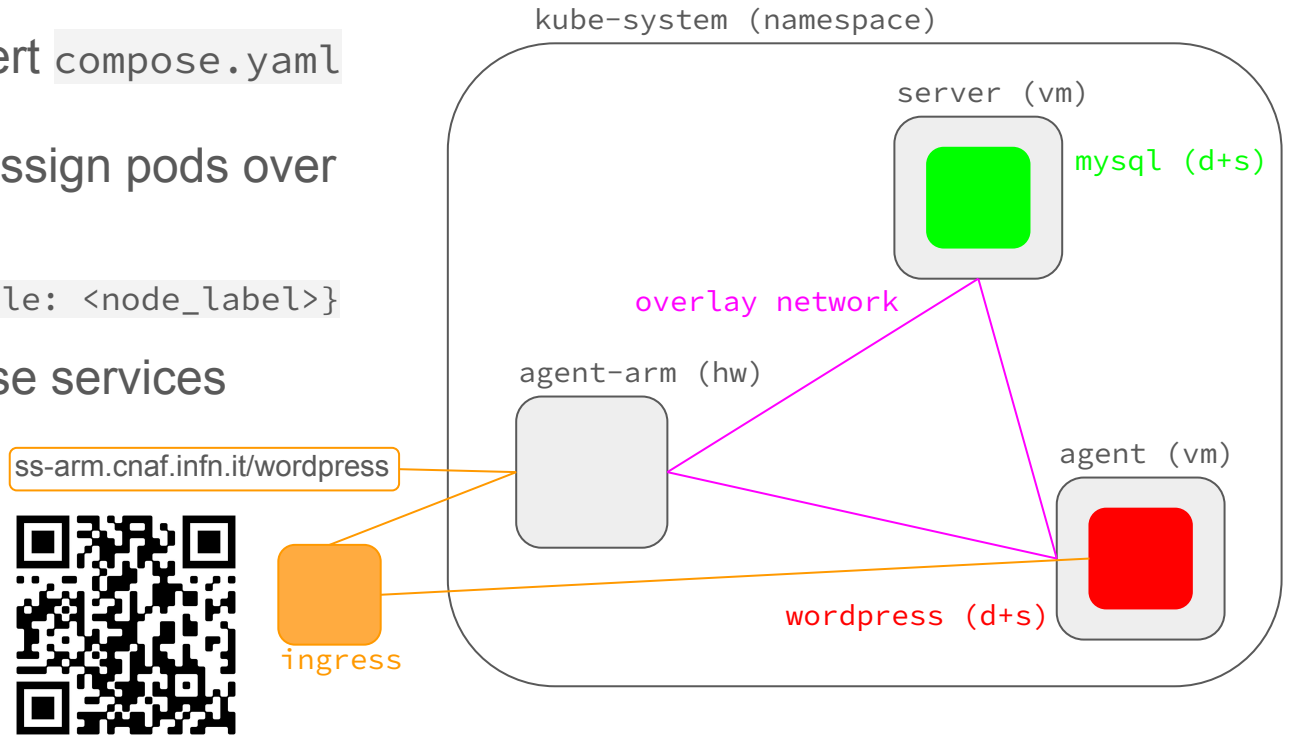


A k8s hosted Wordpress website at CNAF

- kompose: convert `compose.yaml` into k8s confs
- label nodes to assign pods over the cluster

```
nodeSelector: {role: <node_label>}
```

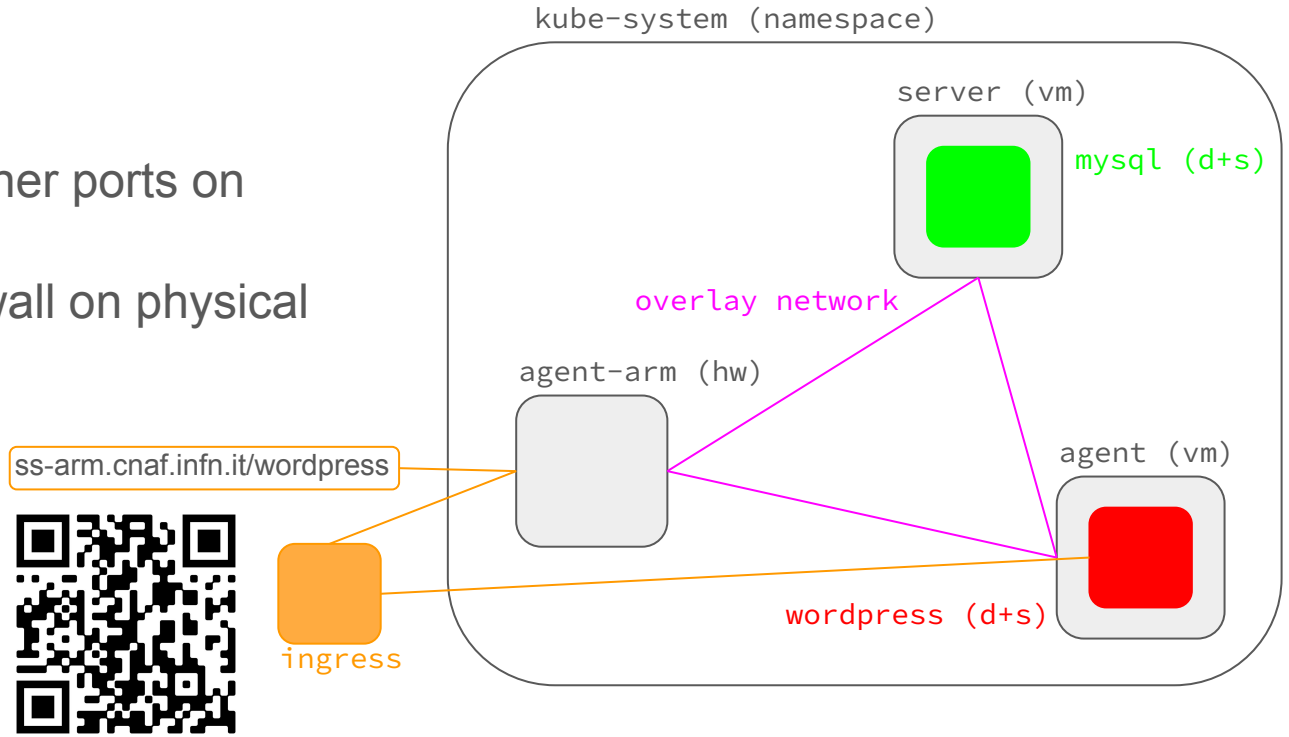
- ingress to expose services



A k8s hosted Wordpress website at CNAF

NB:

- open container ports on nodes!
- disable firewall on physical machines…



Create and run a Baltig CI/CD pipeline

Prerequisites:

being owner of a baltig project

Steps:

A. Having runners available

- **A.1** Install gitlab runner on the cluster
- **A.2** Register the runner for the project

B. Define the pipeline through `gitlab-ci.yml` file at root of the project

A. Having runners available

A.1 Install gitlab runner on the cluster

```
$ helm install --namespace gitlab-runner gitlab-runner -f values.yaml gitlab/gitlab-runner
```

- new pod for each job ⟶ which node?

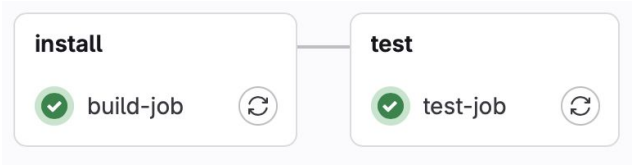
values.yaml

```
gitlabUrl: baltig.infn.it
rbac: {create: true}
runnerToken: <token>
nodeSelector: {role: agent-arm}
```

A.2 Register the runner for the project

`baltig.infn.it/<username>/<project>/-/runners/new` to get the runner token

B. Define the pipeline



- install (19 min)
- test (4 min)

gitlab-ci.yml (1)

```
stages:
  - install
  - test

default:
  image: alpine
  before_script:
    - <install_dependencies>
    - <create_python_venv>

variables:
  …
  PIP_CACHE_DIR: "$CI_PROJECT_DIR/.cache/pip"

…
```

gitlab-ci.yml (2)

```
…

build-job:
  stage: install
  tags:
    - arm
  script:
    - pip install -v .
  artifacts:
    paths:
      - .cache/pip

test-job:
  stage: test
  tags:
    - arm
  script:
    - cd tests
    - pip install -v .
    - ./run_test.sh
```

Specify ARM gitlab-runner-helper image!

values.yaml

```
gitlabUrl: baltig.infn.it
rbac: {create: true}
runnerToken: <token>
nodeSelector: {role: agent-arm}

runners:
  config: |
    [[runners]]
      [runners.kubernetes]
        helper_image = "registry.gitlab.com/gitlab-org/gitlab-runner/gitlab-runner-helper:arm64-latest"
```

- `gitlab-runner`: main application. Runs `gitlab-ci.yml`
- `gitlab-runner-helper`: supporting application. Downloaded by `gitlab-runner` as needed.

Sum up

What I knew:

- operate linux environment
- use containers (docker, podman)
- use docker-compose

What I learnt:

- K8s fundamentals from scratch
- Gitlab CI/CD
- Networking
- Ask for help
- Responsibility in a work environment

Job succeeded