Inception of Things

42 School Project -- Full Architecture Overview

P1 K3s Cluster

P2 Ingress Apps

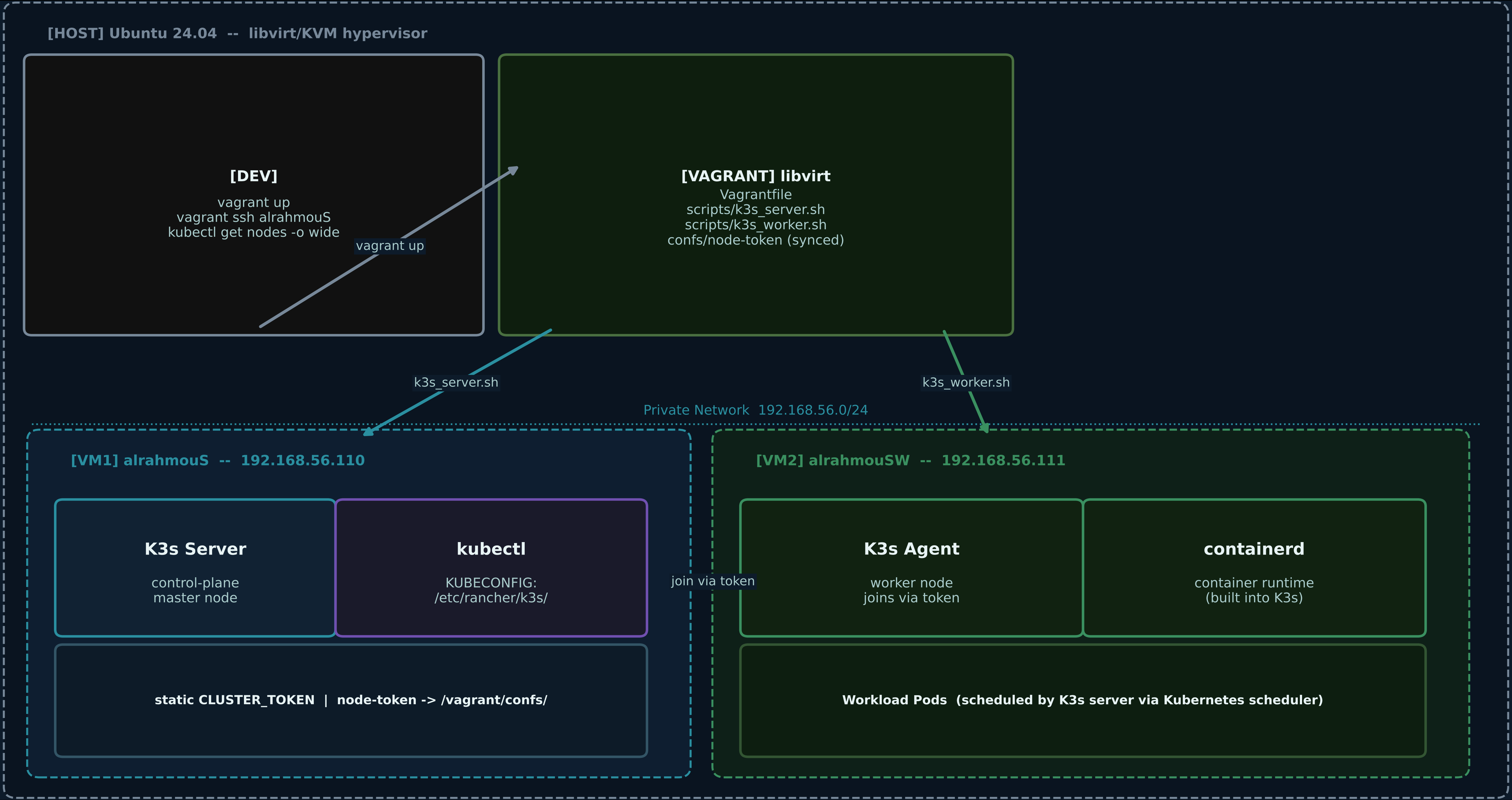
P3 GitOps/ArgoCD

Bonus GitLab+ArgoCD

alrahmou -- Luxembourg, 2026

Part 1 -- K3s Cluster with Vagrant

2 VMs | libvirt/KVM | K3s server + worker



[HOST] Ubuntu 24.04 -- libvirt/KVM hypervisor

[DEV]

vagrant up
vagrant ssh alrahmouS
kubectl get nodes -o wide

vagrant up

[VAGRANT] libvirt

Vagrantfile
scripts/k3s_server.sh
scripts/k3s_worker.sh
confs/node-token (synced)

k3s_server.sh

k3s_worker.sh

Private Network 192.168.56.0/24

[VM1] alrahmouS -- 192.168.56.110

K3s Server

control-plane
master node

kubectl

KUBECONFIG:
/etc/rancher/k3s/

static CLUSTER_TOKEN | node-token -> /vagrant/confs/

join via token

[VM2] alrahmouSW -- 192.168.56.111

K3s Agent

worker node
joins via token

containerd

container runtime
(built into K3s)

Workload Pods (scheduled by K3s server via Kubernetes scheduler)

Part 1 -- Key Facts & Defense Commands

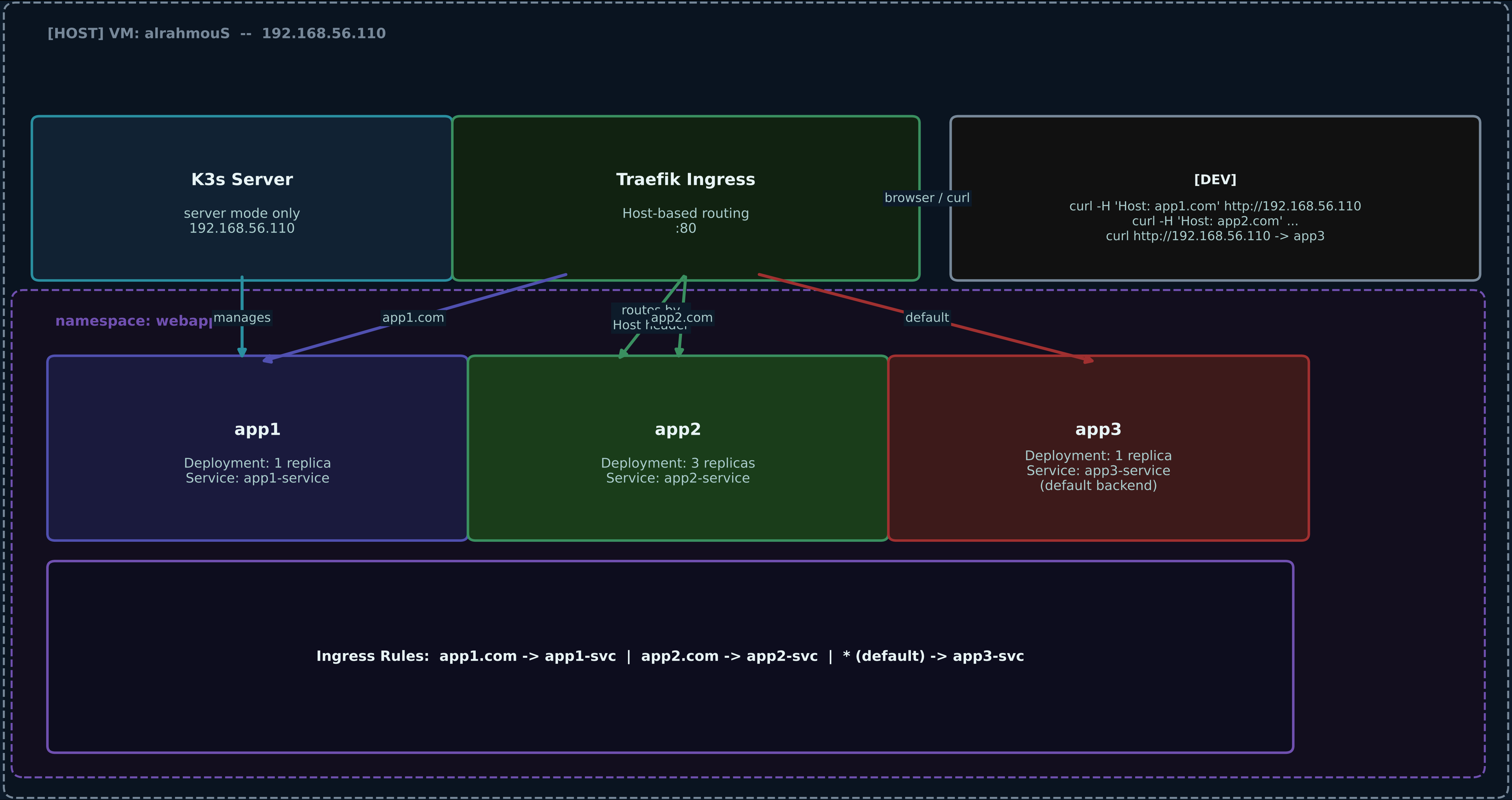
K3s | Vagrant | 2 VMs

VMs	2 VMs: alrahmouS (server) + alrahmouSW (worker)
IPs	192.168.56.110 (server) -- 192.168.56.111 (worker)
K3s mode	Server: --cluster-init Agent: K3S_URL + K3S_TOKEN
Token	Static CLUSTER_TOKEN in Vagrantfile -- written to /vagrant/confs/node-token
Scripts	k3s_server.sh provisions VM1 | k3s_worker.sh provisions VM2
Provision	Fully automated via Vagrant shell provisioners -- no manual steps
Verify	kubectl get nodes -o wide -> 2 nodes, both Ready

```
vagrant up   ->   vagrant ssh alrahmouS   ->   kubectl get nodes -o wide
```

Part 2 -- K3s + Traefik Ingress + 3 Apps

1 VM | webapps namespace | Host-based routing



[HOST] VM: alrahmouS -- 192.168.56.110

K3s Server

server mode only
192.168.56.110

Traefik Ingress

Host-based routing
:80

browser / curl

[DEV]

curl -H 'Host: app1.com' http://192.168.56.110
curl -H 'Host: app2.com' ...
curl http://192.168.56.110 -> app3

namespace: webapps

manages

app1.com

routes by
Host header

app2.com

default

app1

Deployment: 1 replica
Service: app1-service

app2

Deployment: 3 replicas
Service: app2-service

app3

Deployment: 1 replica
Service: app3-service
(default backend)

Ingress Rules: app1.com -> app1-svc | app2.com -> app2-svc | * (default) -> app3-svc

Part 2 -- Key Facts & Defense Commands

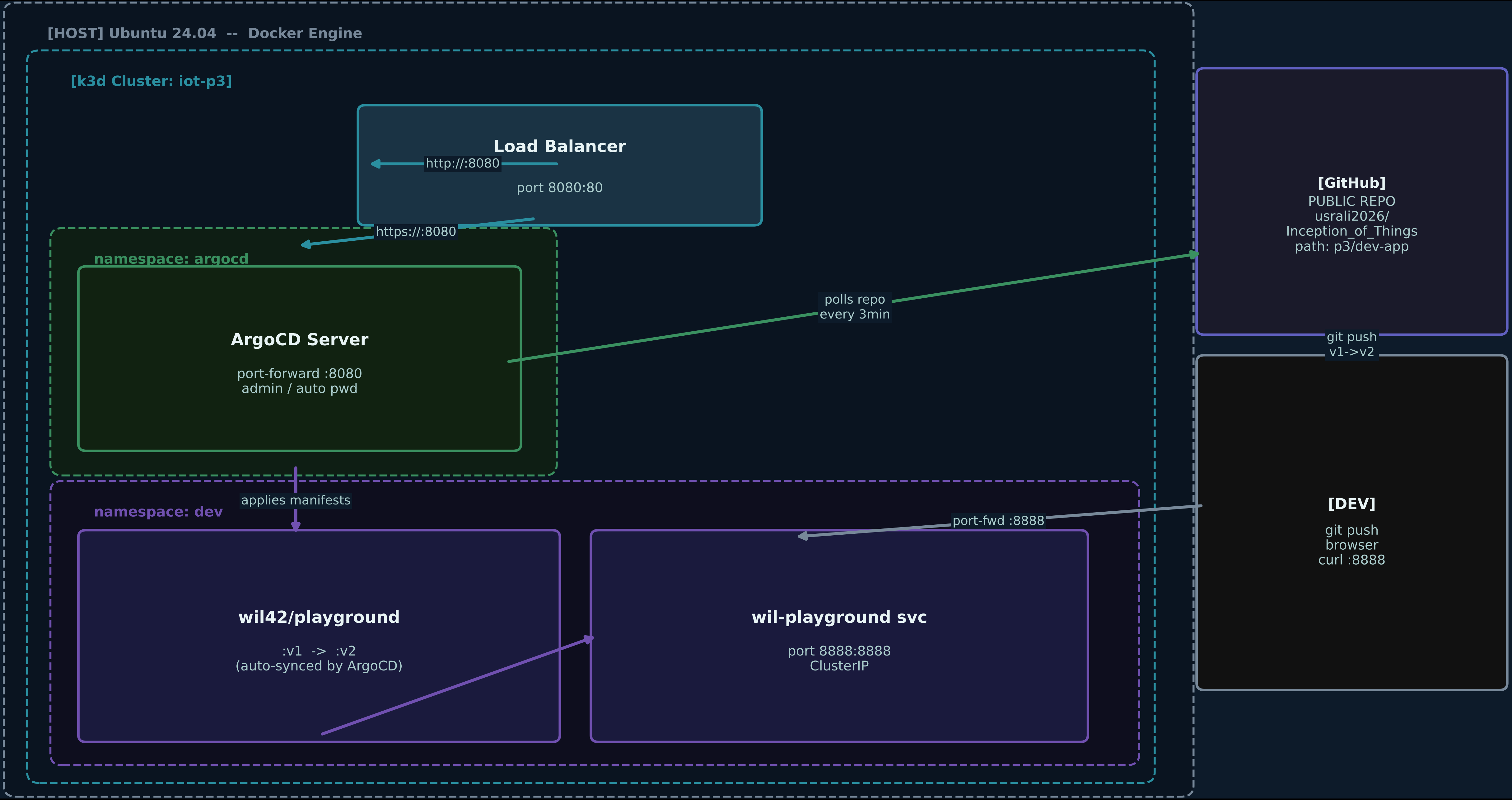
K3s | Traefik | Ingress | 3 Apps

VM	Single VM alrahmouS at 192.168.56.110 (libvirt)
K3s	Server mode only -- no worker node needed
Ingress	Traefik (built-in K3s) -- Host-header based routing
app1	1 replica -- Host: app1.com -> app1-service:80
app2	3 replicas -- Host: app2.com -> app2-service:80
app3	1 replica -- default backend (any other Host header)
/etc/hosts	192.168.56.110 app1.com | 192.168.56.110 app2.com (on host)

```
curl -H 'Host: app1.com' http://192.168.56.110   ->  app1
curl -H 'Host: app2.com' http://192.168.56.110   ->  app2
curl http://192.168.56.110                   ->  app3
```

Part 3 -- GitOps with ArgoCD + k3d

k3d | argocd ns | dev ns | GitHub



Part 3 -- Key Facts & Defense Commands

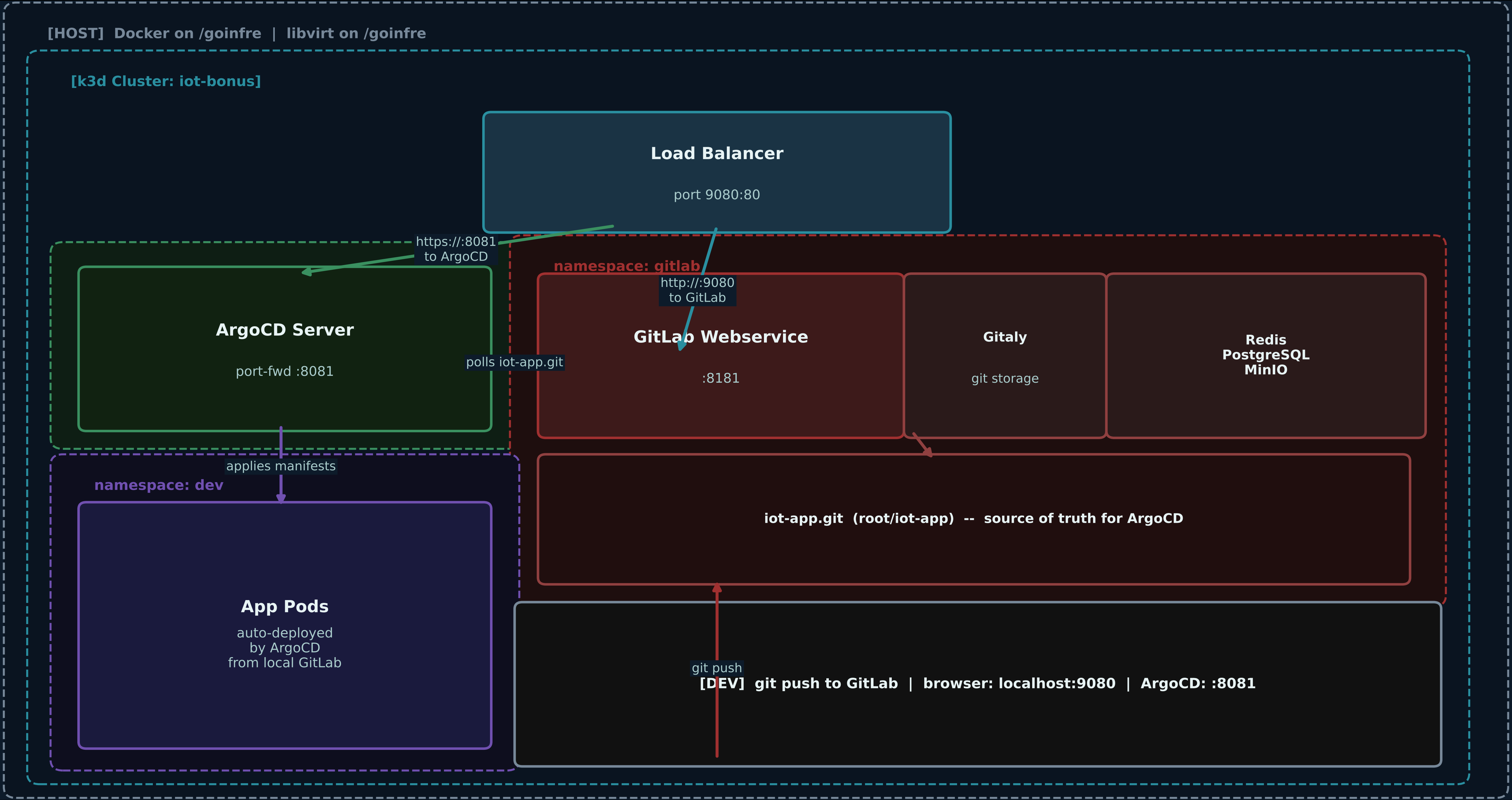
GitOps | ArgoCD | k3d | GitHub

Cluster	k3d iot-p3 -- 1 server + 2 agents (Docker containers)
Namespaces	argocd (ArgoCD) + dev (app pods)
App image	wil42/playground:v1 -> change tag to :v2 in Git to prove GitOps
Repo	Public GitHub: usrali2026/Inception_of_Things | path: p3/dev-app
Sync	Auto-sync enabled -- ArgoCD detects Git drift every ~3 min, reconciles
Access	ArgoCD: port-fwd svc/argocd-server 8080:443 App: port-fwd :8888
GitOps	git push tag v1->v2 -> ArgoCD detects drift -> pod rolls over

```
kubectl get pods -n argocd   |   kubectl get pods -n dev
kubectl port-forward svc/argocd-server -n argocd 8080:443
kubectl -n dev port-forward svc/wil-playground 8888:8888
curl http://127.0.0.1:8888  ->  v1    [edit Git tag to v2]    curl -> v2
```

Bonus -- Self-Hosted GitLab + ArgoCD

k3d | gitlab ns | argocd ns | dev ns | No GitHub needed



[HOST] Docker on /goinfre | libvirt on /goinfre

[k3d Cluster: iot-bonus]

Load Balancer
port 9080:80

https://:8081
to ArgoCD

http://:9080
to GitLab

ArgoCD Server
port-fwd :8081

polls iot-app.git

namespace: gitlab

GitLab Webservice
:8181

Gitaly
git storage

Redis
PostgreSQL
MinIO

applies manifests

namespace: dev

App Pods
auto-deployed
by ArgoCD
from local GitLab

iot-app.git (root/iot-app) -- source of truth for ArgoCD

git push

[DEV] git push to GitLab | browser: localhost:9080 | ArgoCD: :8081

Bonus -- Key Facts & Defense Commands

GitLab local | ArgoCD | k3d | Fully self-hosted

Cluster	k3d iot-bonus -- 1 server + 2 agents | port 9080:80
Namespaces	argocd + dev + gitlab (Webservice, Gitaly, PostgreSQL, Redis, MinIO)
GitLab	Fully local -- no GitHub | user: root / initial-root-password secret
ArgoCD repo	http://gitlab-webservice-default.gitlab.svc.cluster.local:8181/root/iot-app.git
GitOps	git push to local GitLab -> ArgoCD polls -> auto-deploys to dev ns
vs P3	P3 uses GitHub (external). Bonus replaces GitHub with self-hosted GitLab
Setup	Run: bash bonus/scripts/setup.sh -- installs everything end-to-end

```
bash bonus/scripts/setup.sh
kubectl get secret gitlab-gitlab-initial-root-password -n gitlab \
  -o jsonpath='{.data.password}' | base64 -d
kubectl get pods -n gitlab  |  kubectl get pods -n argocd  |  kubectl get pods -n dev
```

Full Comparison -- P1 P2 P3 Bonus

Side-by-side overview

	P1	P2	P3	Bonus
VMs	2 (server+worker)	1 (server only)	0 (Docker)	0 (Docker)
K8s distro	K3s bare	K3s bare	K3s in k3d	K3s in k3d
Provisioner	Vagrant/libvirt	Vagrant/libvirt	k3d CLI	k3d CLI
Namespaces	--	webapps	argocd+dev	argocd+dev +gitlab
Apps	None	app1/2/3	wil42/play	custom app
Ingress	None	Traefik (Host)	LB :8080	LB :9080
GitOps	No	No	Yes (GitHub)	Yes (GitLab)
CI/CD	No	No	ArgoCD	ArgoCD+GitLab