HW 2-1 Kubernetes

hytsao (2026, CC)

Goals

Build a complete Kubernetes production environment from scratch:

- **Self-managed cluster** bootstrapped with kubeadm (via Kubespray)
- **Multi-tier web application** with three independently deployed services
- **Cluster networking** using Calico CNI + Traefik Ingress Controller
- **Advanced configuration**: config management, health checks, network policies

Requirements

Requirement	Specification
Kubernetes version	**1.34.2**
Bootstrap tool	**kubeadm** (Kubespray)
Nodes	**3 control-plane + 2 workers**
Node resources	**≥ 2 vCPU, ≥ 4 GiB RAM** per node
CRI	**CRI-O**
CNI	**Calico**
Ingress Controller	**Traefik**

Node IP Assignment

- In the **private zone** of HW1

Machine	Internal IP	Role
router	`172.16.1.254`	Judge entry point (NOT a Kubernetes node)
na-cp1	`172.16.1.2`	Control plane
na-cp2	`172.16.1.3`	Control plane
na-cp3	`172.16.1.4`	Control plane
na-w1	`172.16.1.5`	Worker
na-w2	`172.16.1.6`	Worker

Cluster Bootstrap

Control Plane High Availability

- **Keepalived**
  - Configure across all three control-plane nodes
  - Manages VIP: `172.16.1.100`
- **HAProxy**
  - Load balance port `8443` in TCP mode

HAProxy frontend	Backend servers
`172.16.1.100:8443`	`172.16.1.2:6443`, `172.16.1.3:6443`, `172.16.1.4:6443`

Calico CNI

- Install Calico in the `calico` namespace
- Pod CIDR MUST be `10.244.0.0/16`
- Resource settings MUST satisfy

Kind	Component	CPU request	CPU limit	Memory request	Memory limit
DaemonSet	`calico-node`	`100m`	`250m`	`100Mi`	`200Mi`
Deployment	`calico-kube-controllers`	`50m`	`100m`	`50Mi`	`100Mi`

Traefik Ingress Controller

- Deployed as a **Deployment** with `replicas 2`
- Exposed via a **NodePort** Service
  - HTTP: `30080` (service port `80`)
  - HTTPS: `30443` (service port `443`)
- Uses `traefik` IngressClass set as default
- Applies global HTTP → HTTPS redirect
- Resource settings MUST satisfy

Component	CPU request	CPU limit	Memory request	Memory limit
Traefik	`100m`	`300m`	`50Mi`	`150Mi`

Static Pod

- Create a static Pod on **one** control-plane node:

Field	Value
Pod name	`hello-static`
Namespace	`kube-system`
Image	`testcontainers/helloworld:1.3.0`
Container port	`80`

Namespace & Resource Management

Namespace & Resource Management

- Create **Namespace** named `nycu-na`
- Create **ResourceQuota** named `nycu-na-quota`

Resource	Hard limit
Total Pods	`20`
Total CPU requests	`500m`
Total CPU limits	`1`
Total memory requests	`400Mi`
Total memory limits	`640Mi`

Workloads

APP-A

- **Deployment** - name: `app-a`, namespace: `nycu-na` , replica: `1`

Container	Image	Port	CPU req	CPU limit	Mem req	Mem limit
`dozzle`	`amir20/dozzle:v10`	8080	`50m`	`100m`	`40Mi`	`80Mi`

- **Service** - name: `app-a-svc`, ClusterIP with port: `6666`
- Set up a ServiceAccount, ClusterRole, and ClusterRoleBinding so that Dozzle can query the Kubernetes API (pods and their logs)

APP-B

- **Deployment**: name: `app-b`, namespace: `nycu-na`, replicas: 2
- **Service** - name: `app-b-svc`, ClusterIP with port: 1234
- **Pod anti-affinity**: Pods MUST be scheduled on different nodes

Container	Image	Port	CPU req	CPU limit	Mem req	Mem limit
`main`	`traefik/whoami:v1.9.0`	80	`50m`	`100m`	`40Mi`	`80Mi`
`datetime-writer`	`busybox:1.36`	—	`10m`	`50m`	`20Mi`	`40Mi`
`datetime-follower`	`busybox:1.36`	—	`10m`	`50m`	`20Mi`	`40Mi`

APP-B sidecar details

- **Shared Volume**
  - Both sidecars MUST share an emptyDir volume mounted at `/opt`
- **datetime-writer**
  - Continuously appends current datetime to `/opt/datetime.txt`
- **datetime-follower**
  - Continuously tail `/opt/datetime.txt`

APP-C

- **Deployment** - name: `app-c`, namespace: `nycu-na`, replicas: `1`
- **Service** - name: `app-c-svc`, ClusterIP with port: `5678`

Container	Image	Port	CPU req	CPU limit	Mem req	Mem limit
`main`	`nginx:1.25`	80	`100m`	`200m`	`60Mi`	`120Mi`

- **Mount Requirements**
  - `nginx.conf` from ConfigMap `app-config` → `/etc/nginx/conf.d/default.conf` (using `subPath`)
  - Secret `app-secret` as a read-only volume → `/etc/app-secret`
  - Env vars: `APP_USERNAME` (from `username`), `APP_PASSWORD` (from `password`)

Configuration & Secrets

ConfigMap - app-config

- Namespace: `nycu-na`

Key	Value
`APP_ENV`	`production`
`nginx.conf`	Complete nginx `server` block (see below)

- The `nginx.conf` MUST:
  - Listen on port `80`
  - Serve static files from the default document root
  - Return a custom `404` response with body `NA: page not found`

Secret - app-sercret

- Type: `Opaque`, namespace: `nycu-na`

Key	Plaintext value
`username`	`nasaadmin`
`password`	`s3cr3t!nasA`

- Usage in APP-C
  - Volume mount (read-only): `/etc/app-secret`
  - Environment variables:
    - `APP_USERNAME` ← `username`
    - `APP_PASSWORD` ← `password`

Health checks

Health Check Configuration

- Applied to main containers (app-a/dozzle , app-b/main , app-c/main)
- Readiness Probe

Parameter	Value
Mechanism	`httpGet` on path `/` (`8080` for dozzle, else `80`)
`initialDelaySeconds`	`5`
`periodSeconds`	`10`
`failureThreshold`	`3`
`successThreshold`	`1`

Health Check Configuration (cont.)

- Liveness Probe

Parameter	Value
Mechanism	`httpGet` on path `/` (`8080` for dozzle, else `80`)
`initialDelaySeconds`	`15`
`periodSeconds`	`20`
`timeoutSeconds`	`5`
`failureThreshold`	`3`

Ingress & TLS

TLS certificate

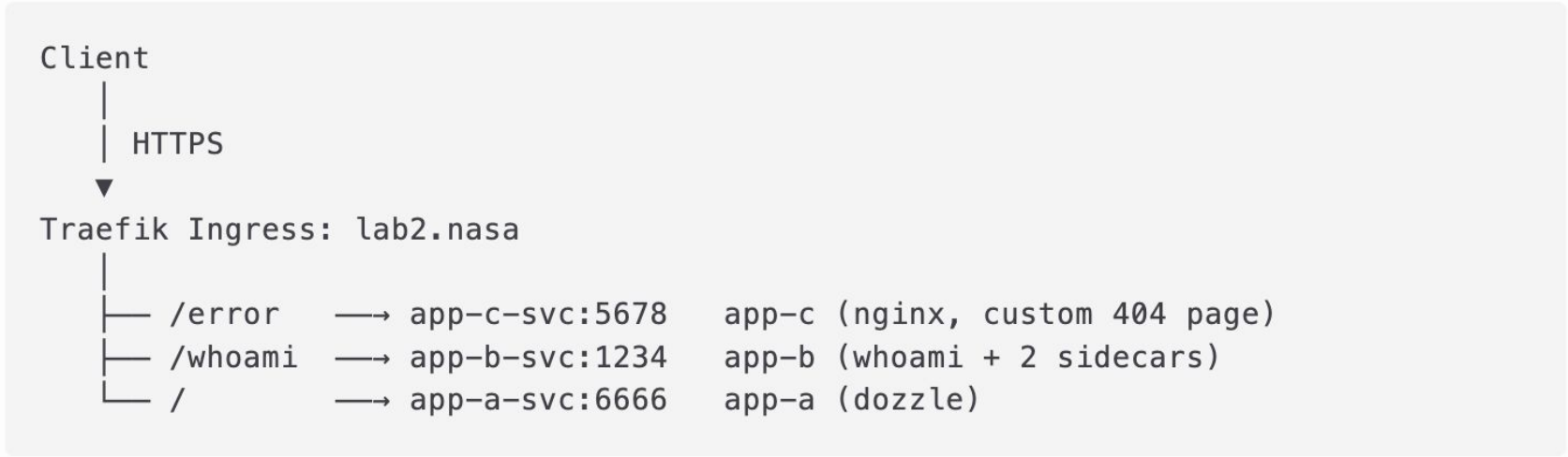
- Generate a self-signed certificate for `lab2.nasa`
  - Both **CN** and SAN must include `lab2.nasa` (modern TLS clients validate SAN)
  - Secret Store as a `kubernetes.io/tls` type
  - Secret name: `lab2-tls`, namespace: `nycu-na`
- Path routing (all Prefix match):
  - `/error` → `app-c-svc:5678`
  - `/whoami` → `app-b-svc:1234`
  - `/` → `app-a-svc:6666`

Ingress Verification

- TLS enabled for host `lab2.nasa` using Secret `lab2-tls`
- HTTP automatically redirects to HTTPS
- All paths use **Prefix** match
- IngressClass: `traefik`

Request workflow

- **Namespace**: `nycu-na`
- **Ingress hostname**: `lab2.nasa`



NetworkPolicy

NetworkPolicy

- Implement **default-deny** + **explicit allow** in namespace `nycu-na`

Source	Destination	Port	Result
Ingress Controller	app-a Pods	`8080`	ALLOWED
Ingress Controller	app-b Pods	`80`	ALLOWED
Ingress Controller	app-c Pods	`80`	ALLOWED
Any Pod	CoreDNS	`53` UDP/TCP	ALLOWED
app-a Pods	app-b Pods	`80`	BLOCKED
app-b Pods	app-c Pods	`80`	BLOCKED
app-c Pods	app-a Pods	`8080`	BLOCKED

Grading

Grading Details (1/4)

- Control Plane HA / LB Setup (15%)
  - HAProxy with all 3 CP backends on port 6443 (5%)
  - Keepalived with VIP configured (5%)
  - API server reachable via 172.16.1.100:8443 (5%)
- Cluster Bootstrap (20%)
  - All 5 nodes Ready, system pods Running, CNI and IngressClass registered (15%)
  - Static Pod hello-static running in kube-system (5%)

Grading Details (2/4)

- Workloads (25%)
  - APP-B: whoami + two sidecars + emptyDir + Service + anti-affinity (10%)
  - APP-C: nginx + ConfigMap mount + Secret mount + env vars + Service (8%)
  - APP-A: dozzle + RBAC + Service (7%)
- Configuration & Secrets (20%)
  - ConfigMap app-config with nginx.conf correctly mounted (10%)
  - Secret app-secret as volume + APP_USERNAME/APP_PASSWORD env vars (10%)

Grading Details (3/4)

- Ingress & TLS (10%)
  - TLS Secret lab2-tls of type kubernetes.io/tls (3%)
  - Correct path routing /error, /whoami, / (4%)
  - HTTP→HTTPS redirect working (3%)
- NetworkPolicy (5%)
  - All ALLOWED and BLOCKED entries enforced (5%)

Grading Details (3/4)

- Health Checks (3%)
  - Readiness probes with correct parameters (1%)
  - Liveness probes with correct parameters (2%)
- Namespace & Resource Management (2%)
  - ResourceQuota nycu-na-quota with correct hard limits (2%)

Note

- Always backup system before submission, as we may perform malicious actions during testing
- Make sure everything works correctly after reboot

Good Luck !