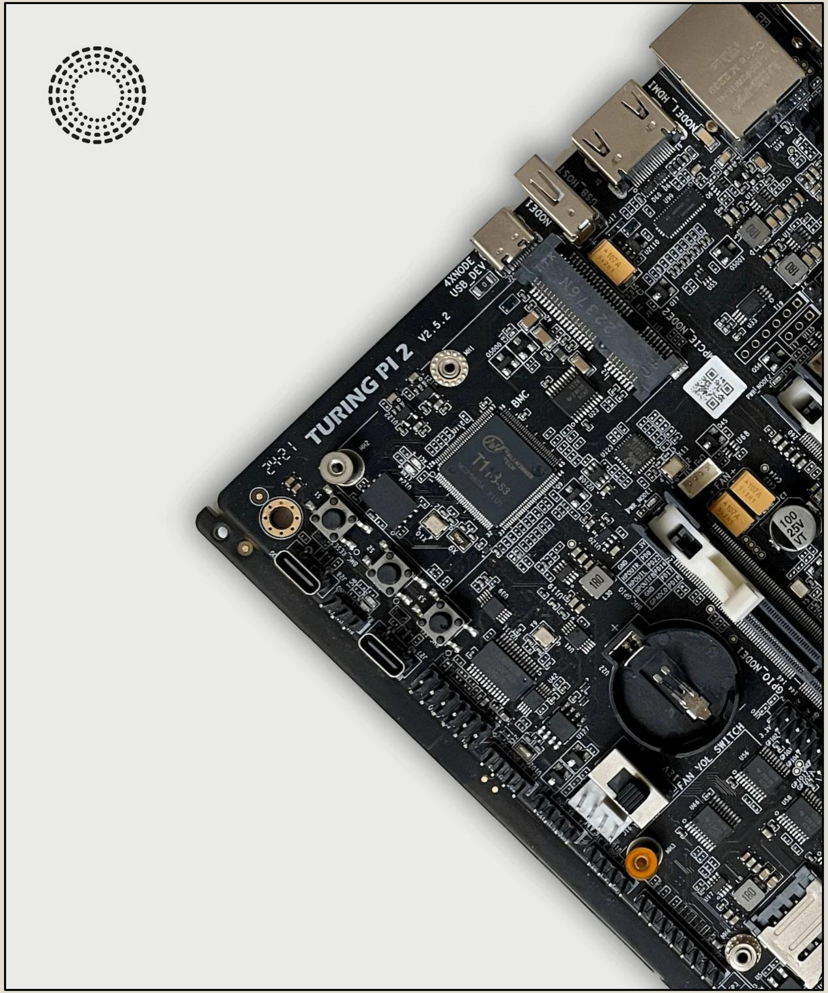
MAR 8, 2025

SCaLE

BAKING A TURINGPI IN THE HOME LAB

MATTHEW SANABRIA



INTRODUCTION

MATTHEW SANABRIA

Solutions Software Engineer
Oxide Computer Company
https://matthewsanabria.dev

WHAT EXACTLY IS A HOME LAB?

FAMOUS HOME LABS

PROJECT MINI RACK BY JEFF GEERLING

https://mini-rack.jeffgeerling.com/

MY FRIEND'S RACK (NO PUN INTENDED)

I asked for permission to share this.

TECHNO TIM'S MINI NETWORK RACK

https://x.com/TechnoTimLive/status/1891227742793765132

HOME LAB GOALS

Learn Baby Learn!

Have Fun

Use Kubernetes

Don't Use Helm

Keep It Simple

Publicly Accessible

HARDWARE

WHAT HARDWARE SHOULD I USE?

HARDWARE GOALS

Low Power Usage

Small Form Factor

Quiet Operation

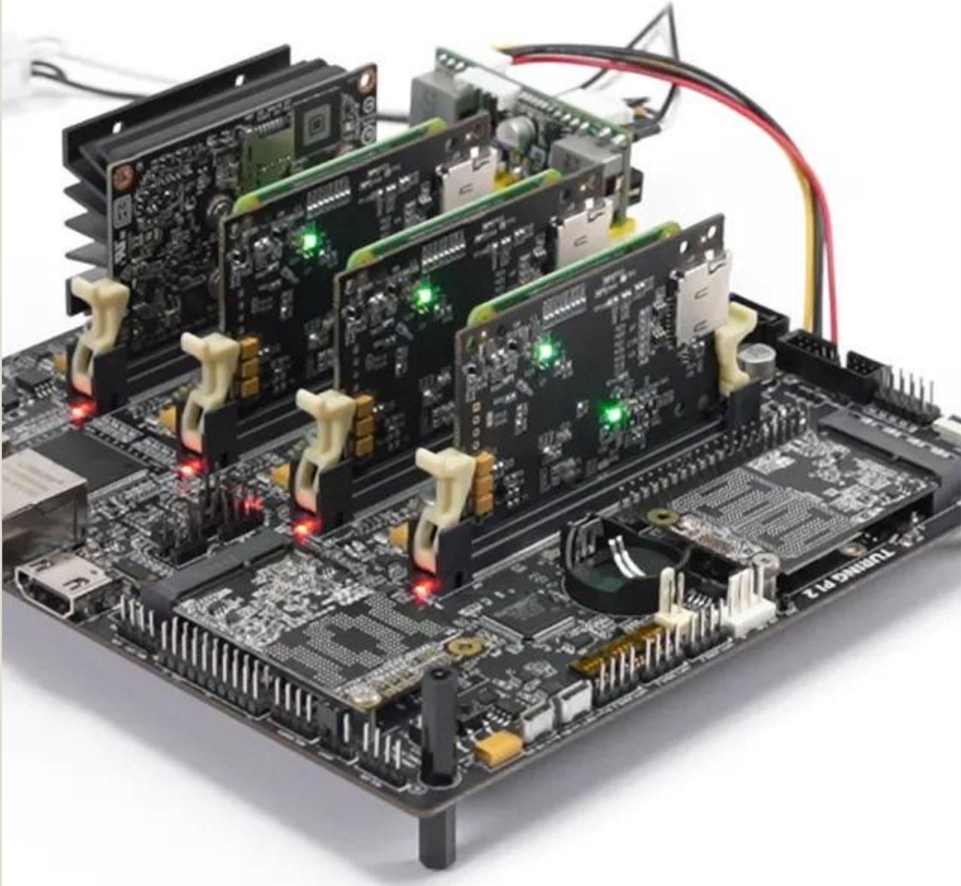
Multiple Nodes

Powerful CPU

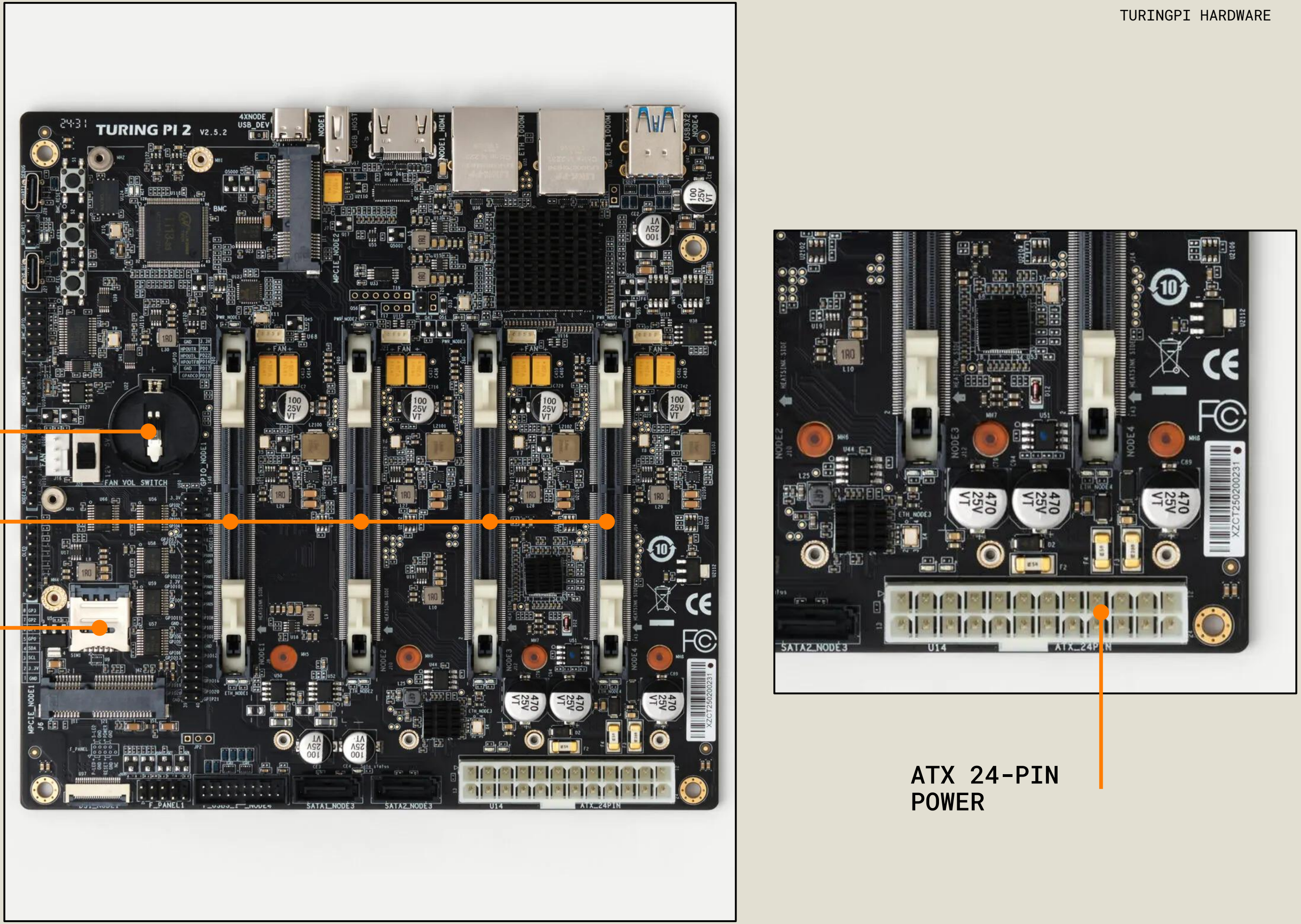
High RAM Capacity

TURINGPI 2.5 CLUSTER BOARD

- 4-node mITX cluster board
- Support for different compute modules
- Shared power & network
- "A home lab blade chassis" - Me



TURINGPI 2.5 CLUSTER BOARD - FRONT



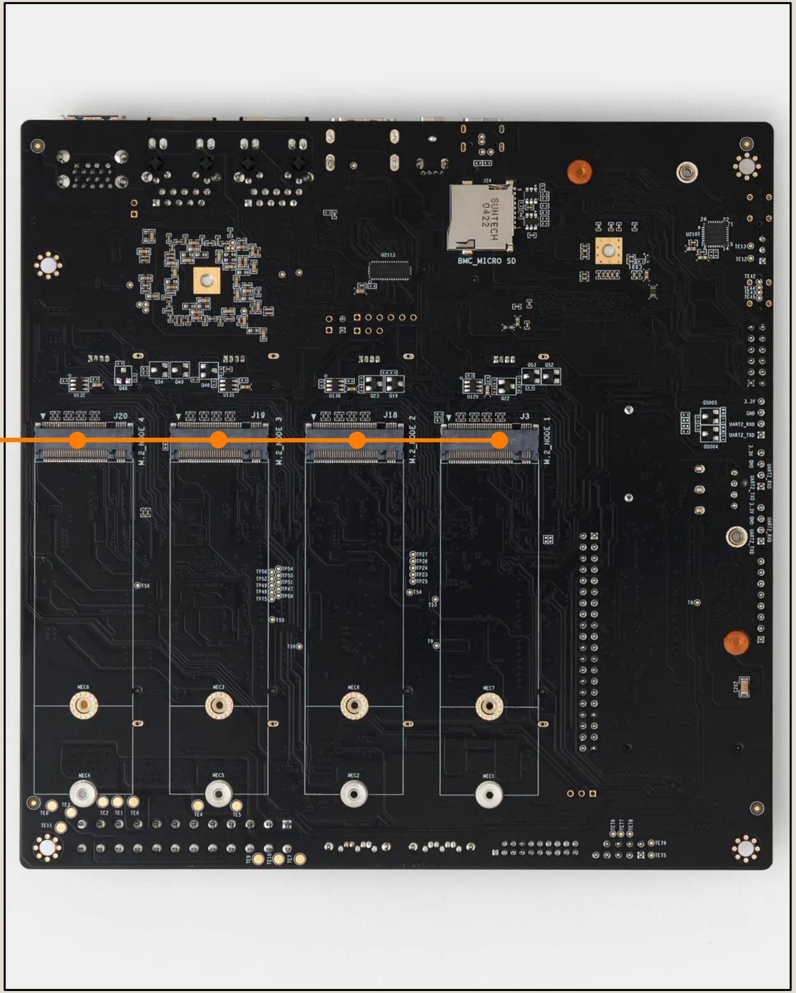
CR2032 BATTERY

4X DDR4 260-PIN

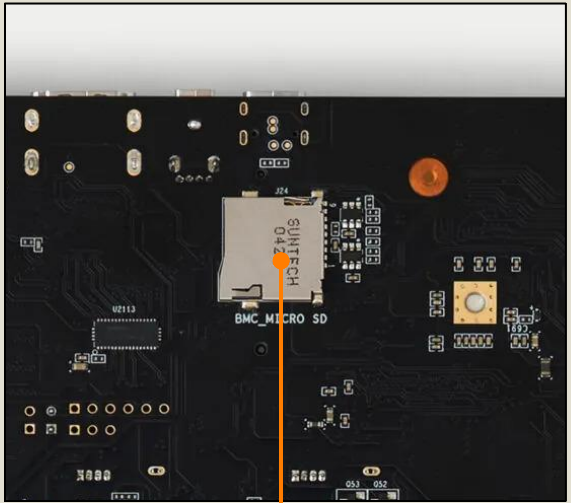
SIM CARD SLOT

ATX 24-PIN POWER

TURINGPI 2.5 CLUSTER BOARD - BACK

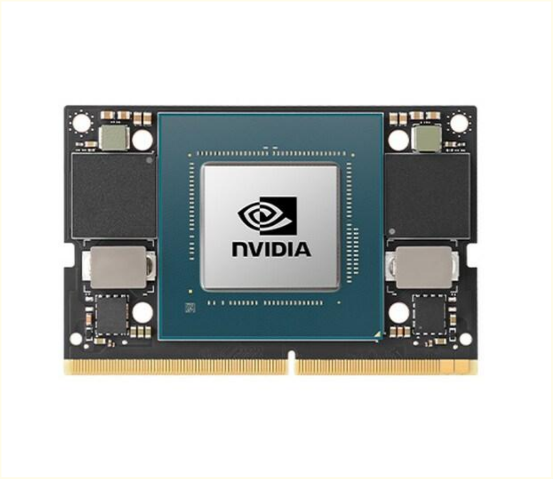


4X M.2 SLOTS 2260/2280



MICROSD CARD SLOT

SUPPORTED COMPUTE MODULES

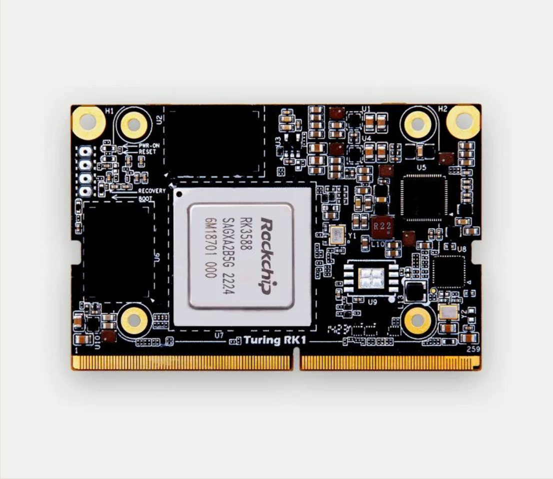


NVIDIA JETSON

Perfect for running AI/ML workloads.

RASPBERRY PI CM4

Bring your existing Raspberry Pi CM4 compute devices.

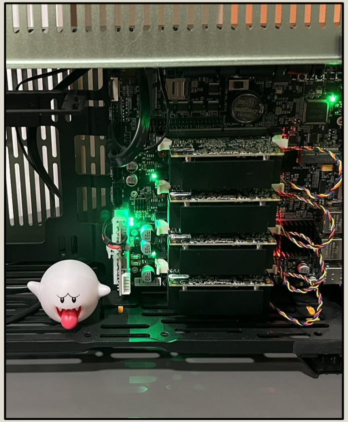


TURING RK1

Rockchip RK3588 CPU, up to 32GiB memory, and 32GiB flash.

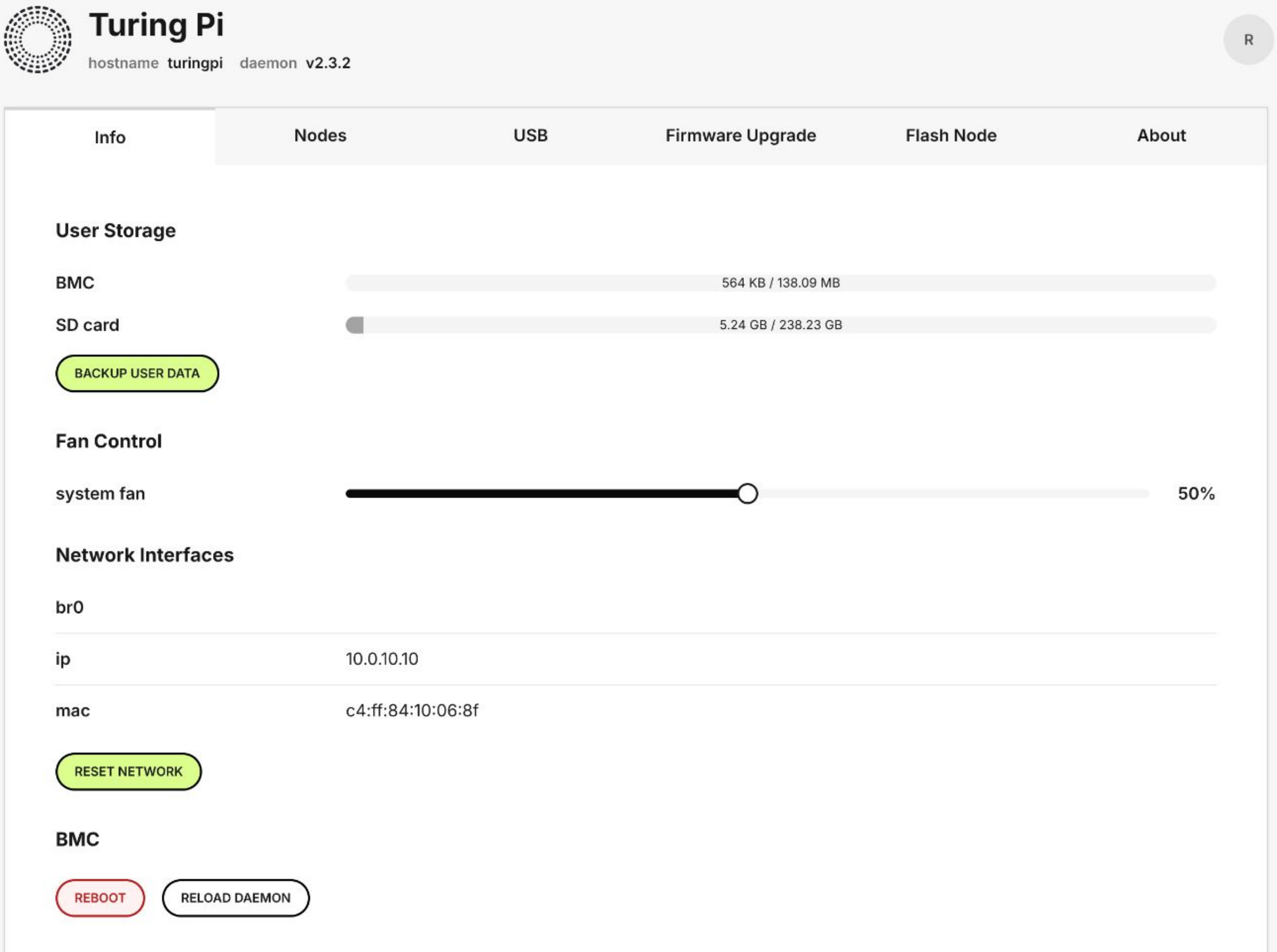
THE BUILD

- 1x TuringPi 2.5 Cluster Board
- 4x RK1 32 GiB
- 4x RK1 Heatsink
- 4× 2 TiB SN850x NVMe
- 1x Fractal Terra (Jade)
- Total Cost: ~$2,250.00
- Ordered Feb 2024
- Arrived Oct 2024



SOFTWARE

HOW DOES THIS TURINGPI WORK?



hostname turingpi daemon v2.3.2

User Storage

BMC 564 KB / 138.09 MB

SD card 5.24 GB / 238.23 GB

Fan Control

system fan 50%

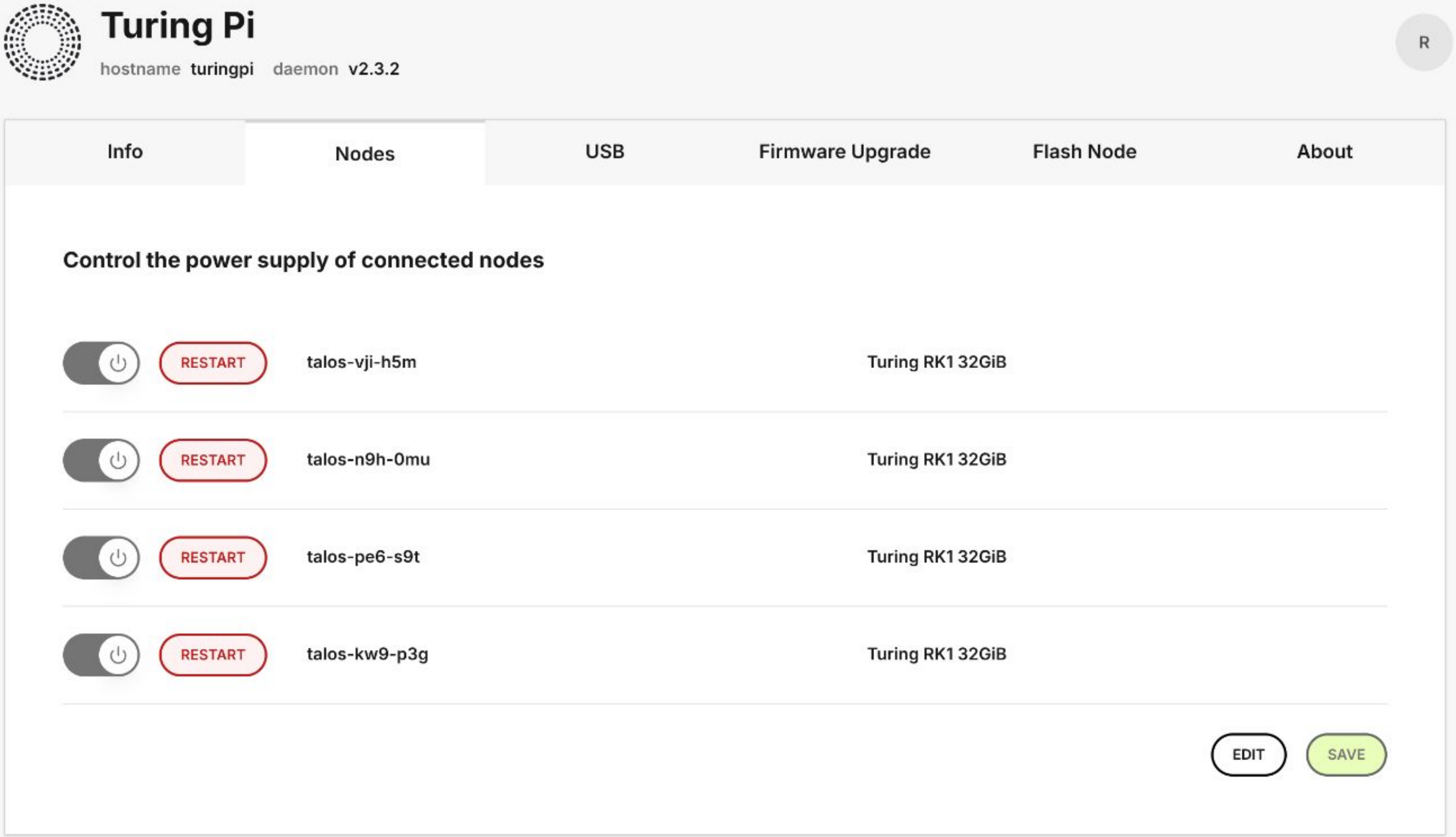
Network Interfaces

br0

ip 10.0.10.10

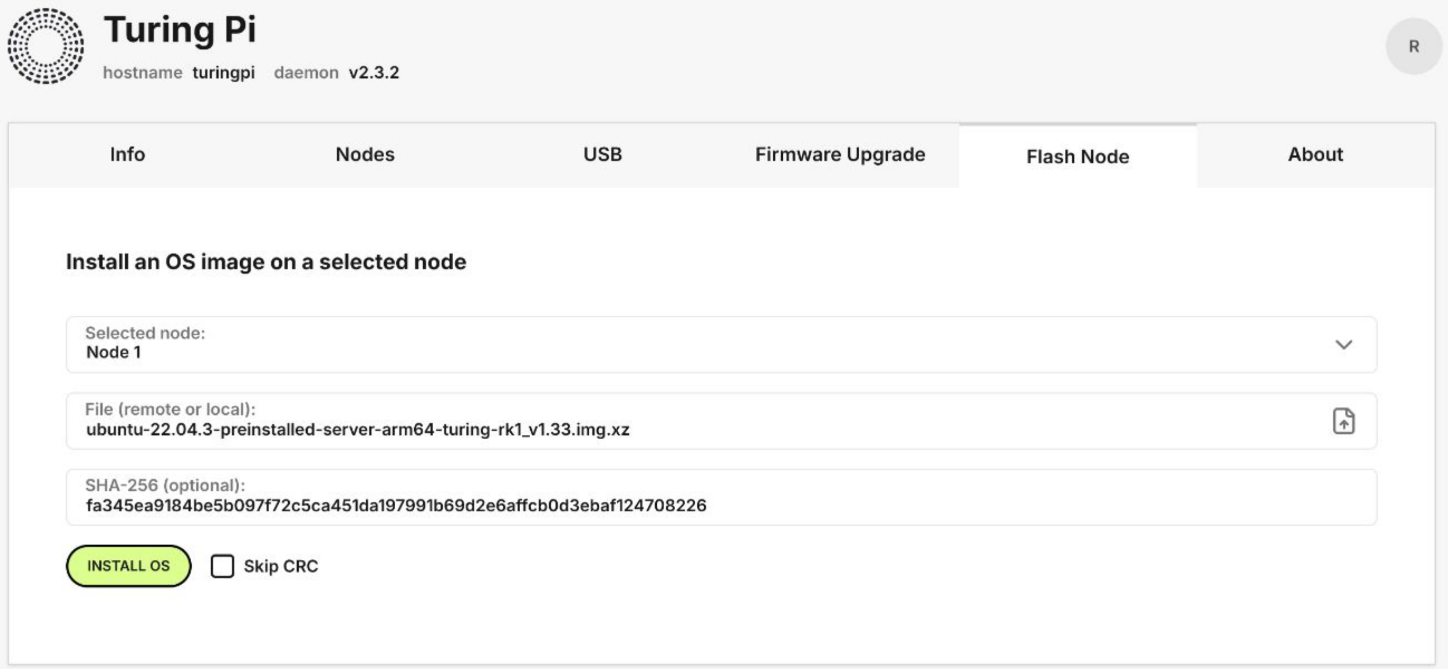
mac c4:ff:84:10:06:8f

BMC



Control the power supply of connected nodes

⏻	RESTART	talos-vji-h5m	Turing RK1 32GiB
⏻	RESTART	talos-n9h-0mu	Turing RK1 32GiB
⏻	RESTART	talos-pe6-s9t	Turing RK1 32GiB
⏻	RESTART	talos-kw9-p3g	Turing RK1 32GiB



Turing Pi

hostname turingpi daemon v2.3.2

Install an OS image on a selected node

Selected node:
Node 1

File (remote or local):
ubuntu-22.04.3-preinstalled-server-arm64-turing-rk1_v1.33.img.xz

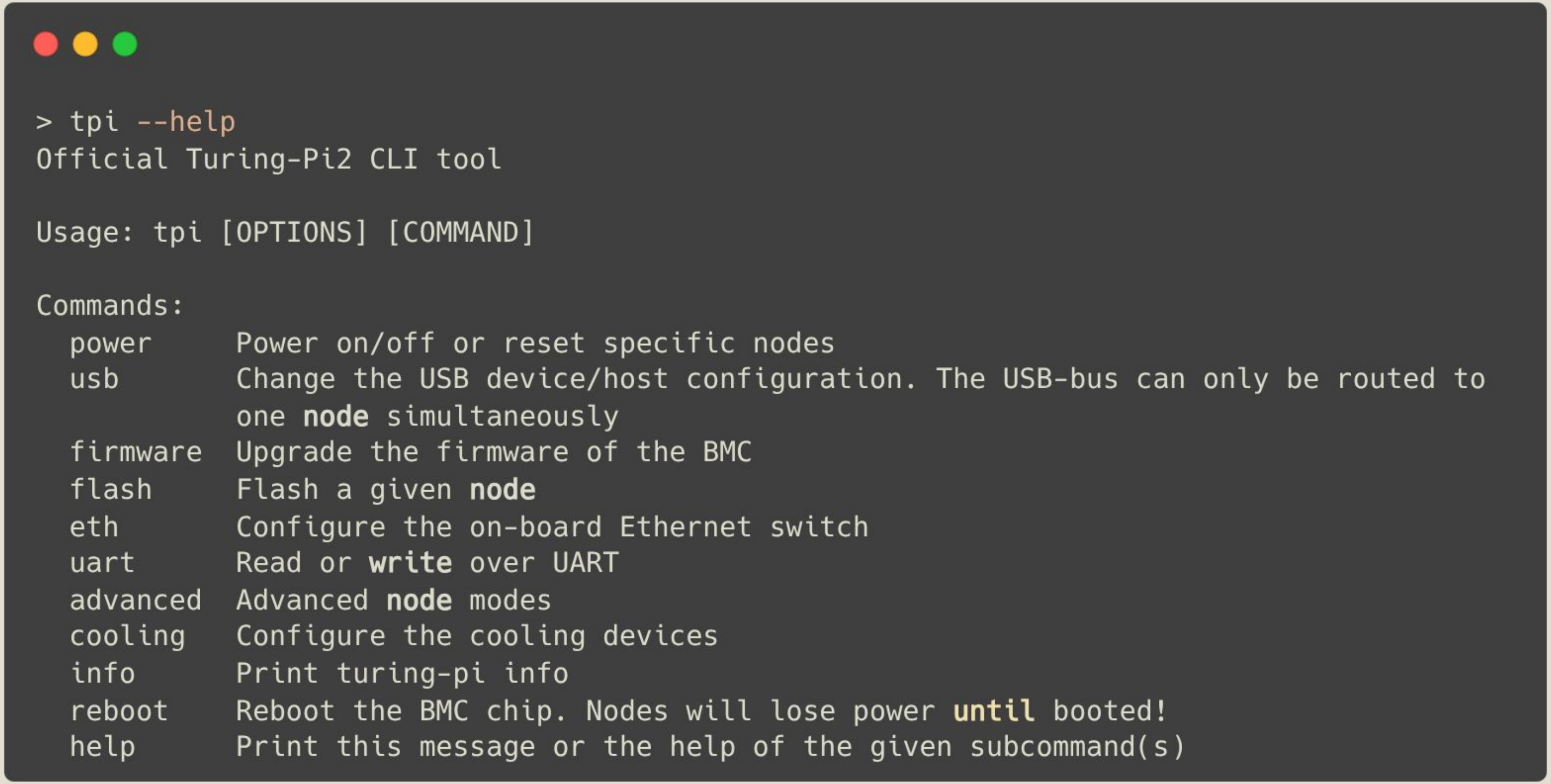
SHA-256 (optional):
fa345ea9184be5b097f72c5ca451da197991b69d2e6affcb0d3ebaf124708226

□ Skip CRC

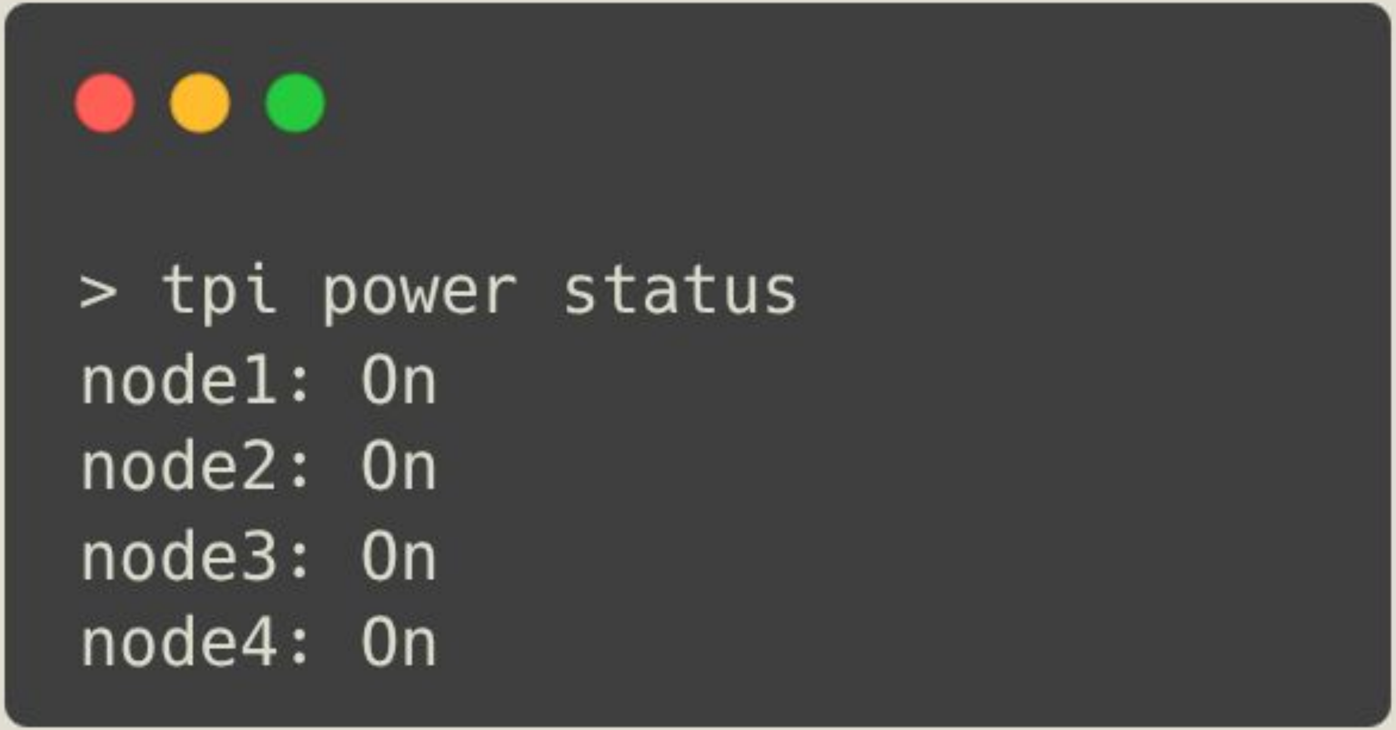
OBLIGATORY DRAKE MEME

GUI? HATOOEY!

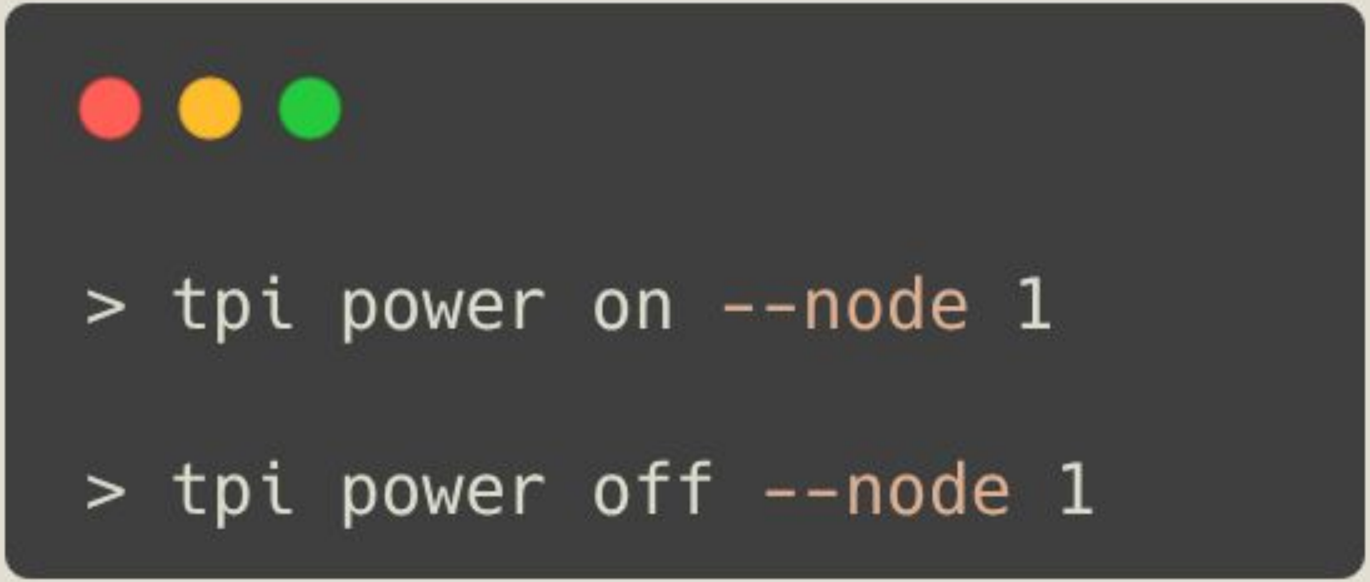
We use the CLI around here.



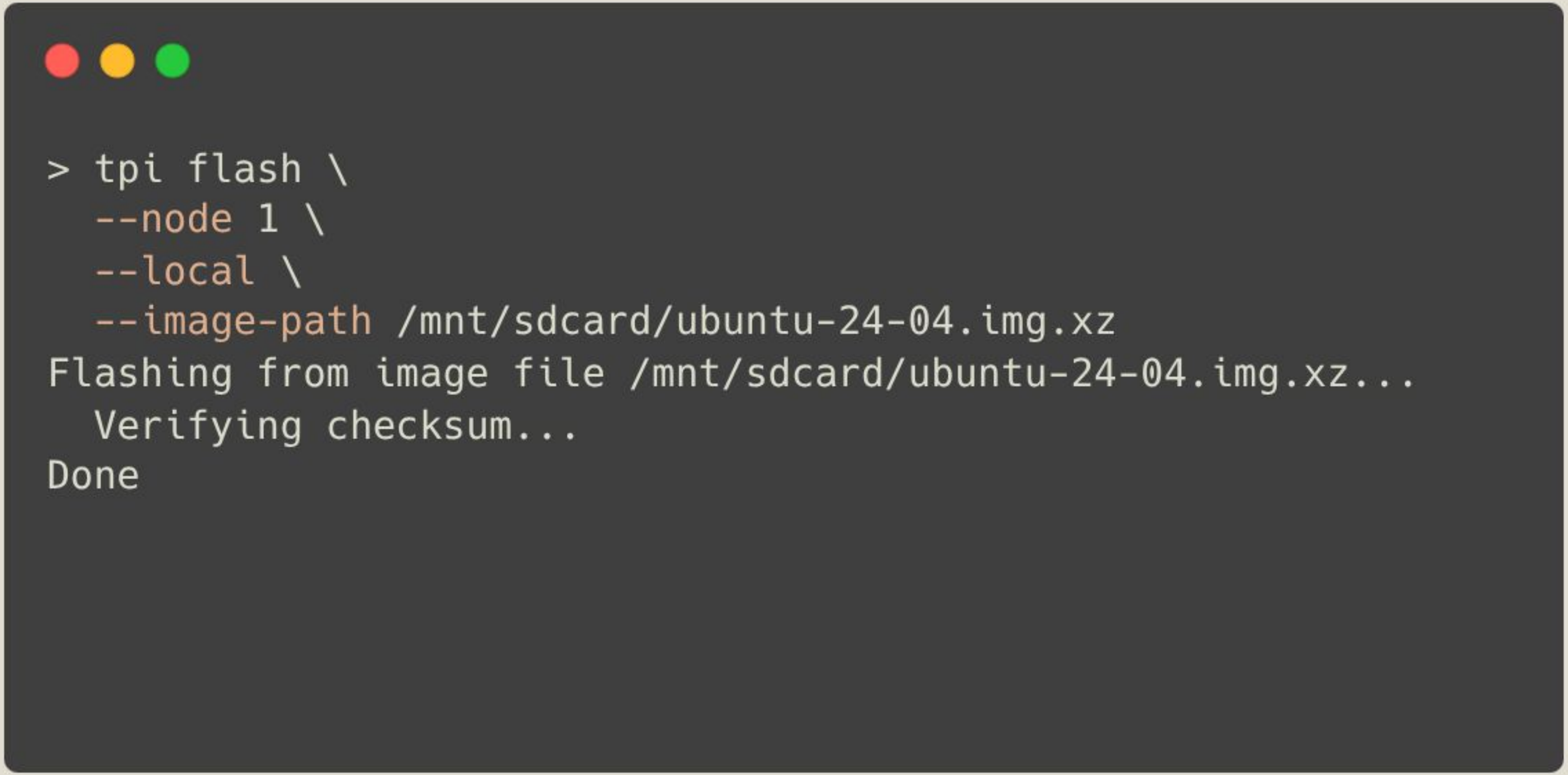
```
> tpi --help
Official Turing-Pi2 CLI tool

Usage: tpi [OPTIONS] [COMMAND]

Commands:
  power     Power on/off or reset specific nodes
  usb       Change the USB device/host configuration. The USB-bus can only be routed to
            one node simultaneously
  firmware  Upgrade the firmware of the BMC
  flash     Flash a given node
  eth       Configure the on-board Ethernet switch
  uart      Read or write over UART
  advanced  Advanced node modes
  cooling   Configure the cooling devices
  info      Print turing-pi info
  reboot    Reboot the BMC chip. Nodes will lose power until booted!
  help      Print this message or the help of the given subcommand(s)
```



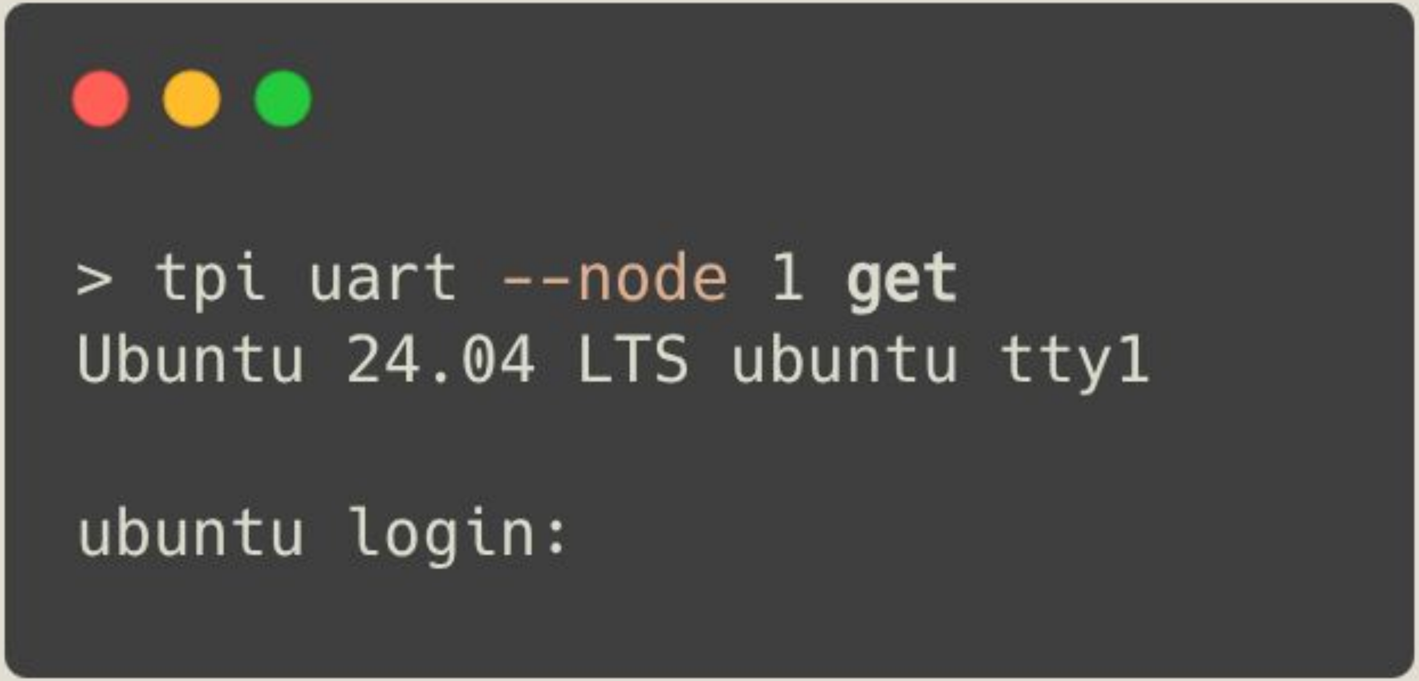
```
> tpi power status
node1: On
node2: On
node3: On
node4: On
```



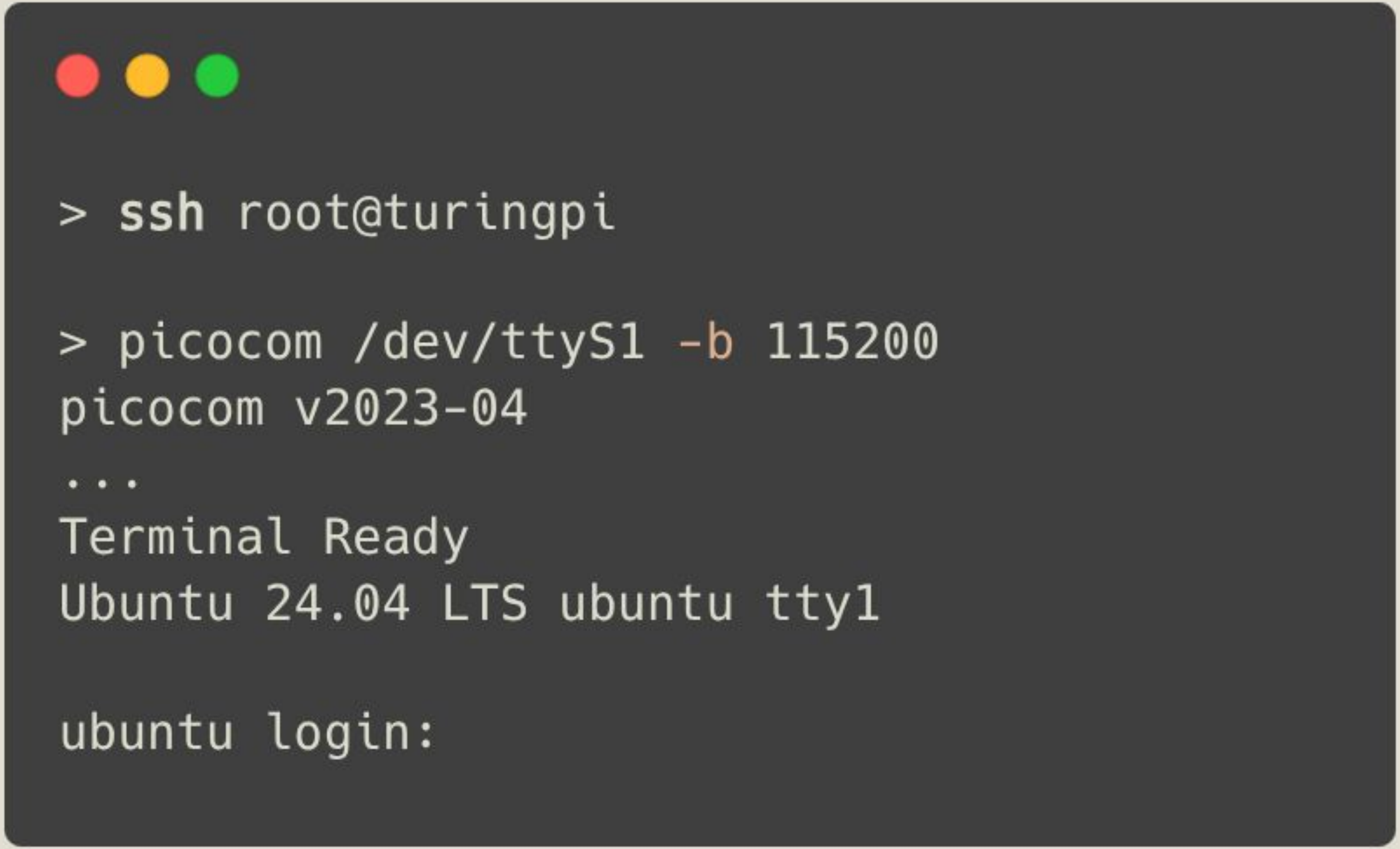
```
> tpi power on --node 1

> tpi power off --node 1
```



```
> tpi flash \
  --node 1 \
  --local \
  --image-path /mnt/sdcard/ubuntu-24-04.img.xz
Flashing from image file /mnt/sdcard/ubuntu-24-04.img.xz...
  Verifying checksum...
Done
```



```
> tpi uart --node 1 get
Ubuntu 24.04 LTS ubuntu tty1

ubuntu login:
```



```
> ssh root@turingpi

> picocom /dev/ttyS1 -b 115200
picocom v2023-04
...
Terminal Ready
Ubuntu 24.04 LTS ubuntu tty1

ubuntu login:
```

OPERATING SYSTEM

WHAT OPERATING SYSTEM DO I RUN?

OPERATING SYSTEM GOALS

TuringPi Support

Kubernetes Support

Open Source

Minimal Bloat

Secure

Upgradable

TALOS LINUX

- Designed for Kubernetes
- Managed via API, not SSH
- Open source
- Wide platform support

WHERE DO I FIND A TURINGPI TALOS IMAGE?

TALOS FOR TURING RK1: UNOFFICIAL COMMUNITY SUPPORT

- Found via Turing Pi Discord
- Best-effort support
- Difficult to customize

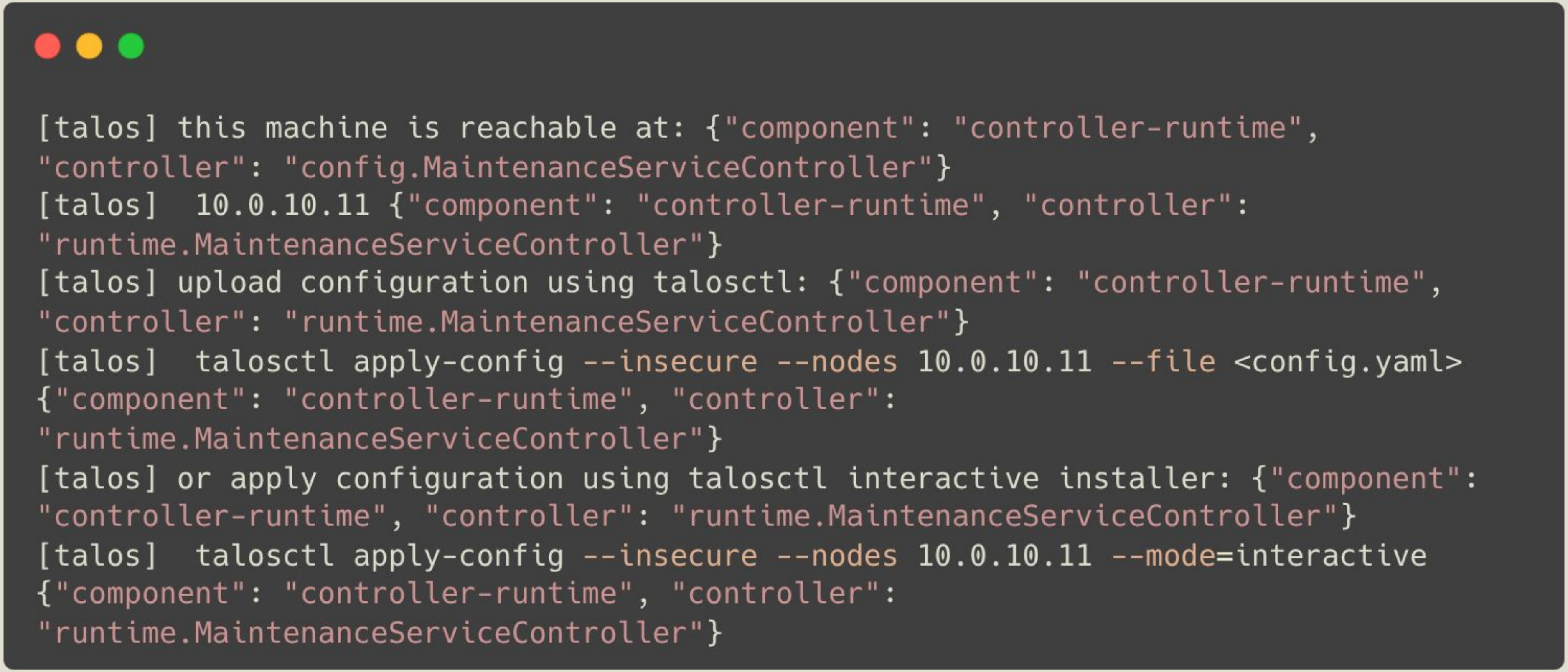
Only option for a while

TALOS FOR TURING RK1: OFFICIAL IMAGE FACTORY SUPPORT

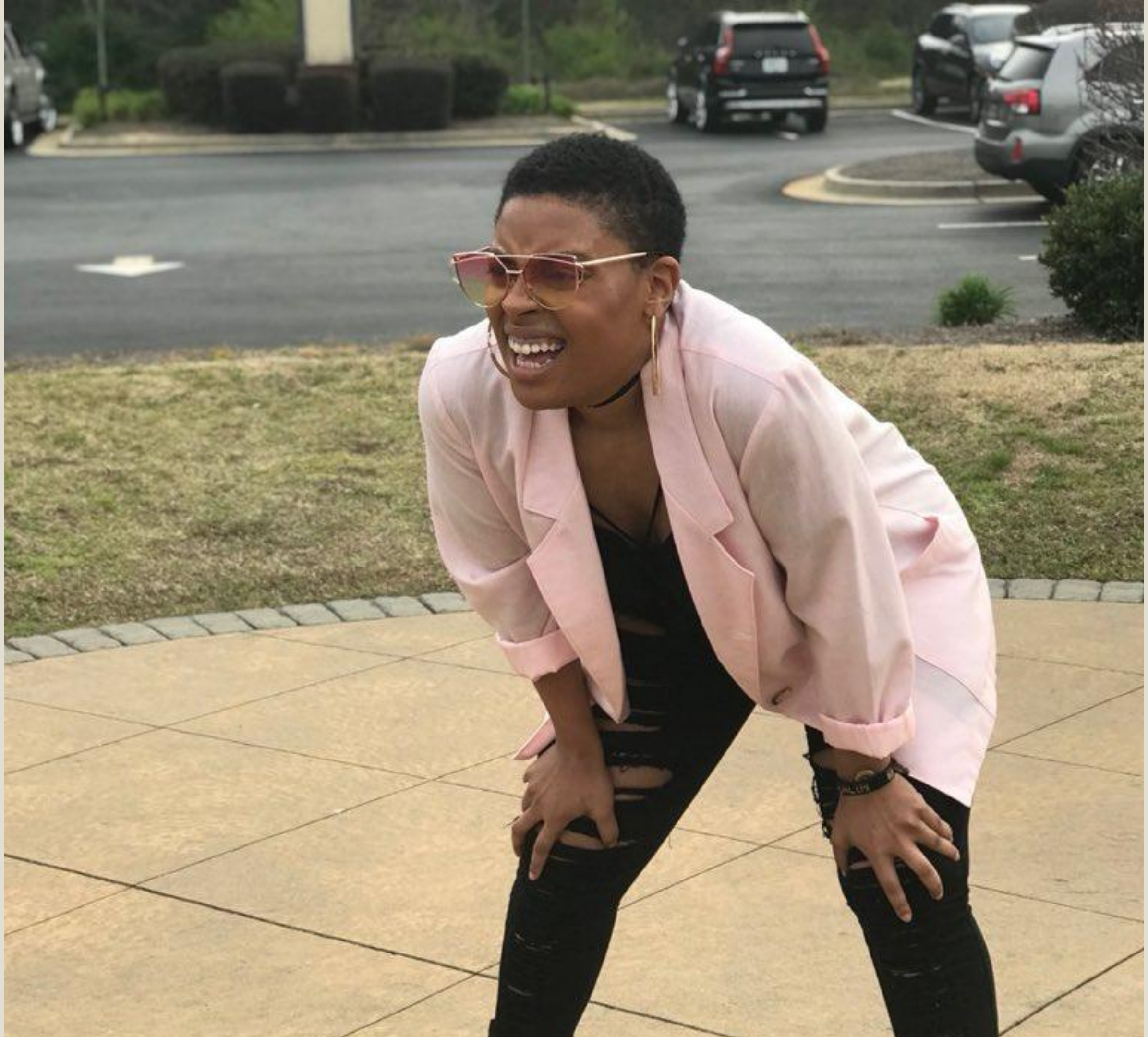
- siderolabs/sbc-rockchip#35
- siderolabs/image-factory#171
- siderolabs/talos#9864

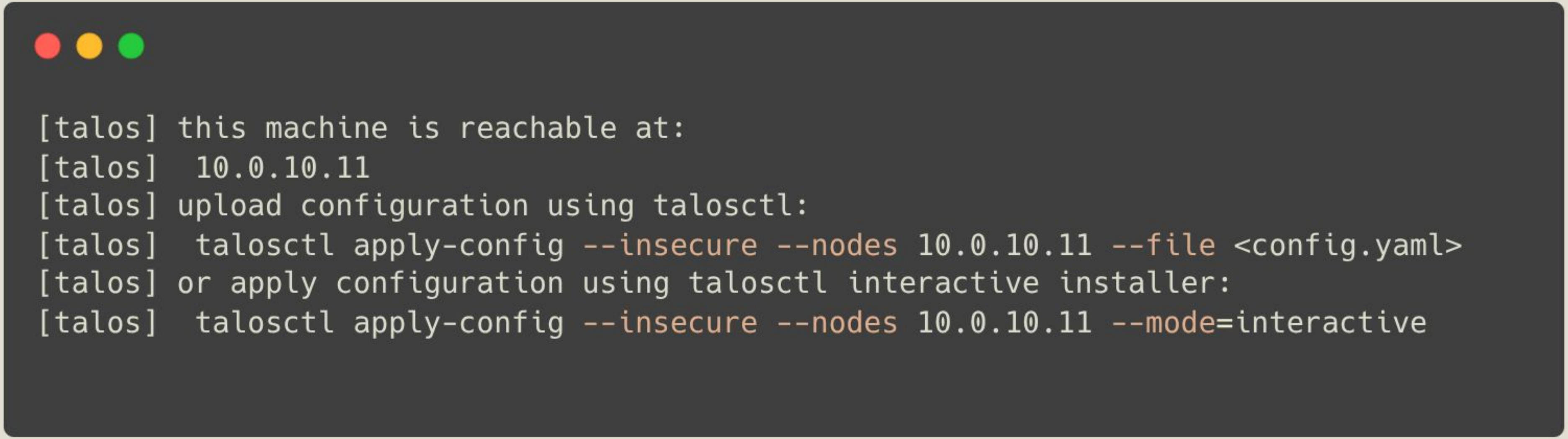
Available on factory.talos.dev

LET'S BOOT TALOS!

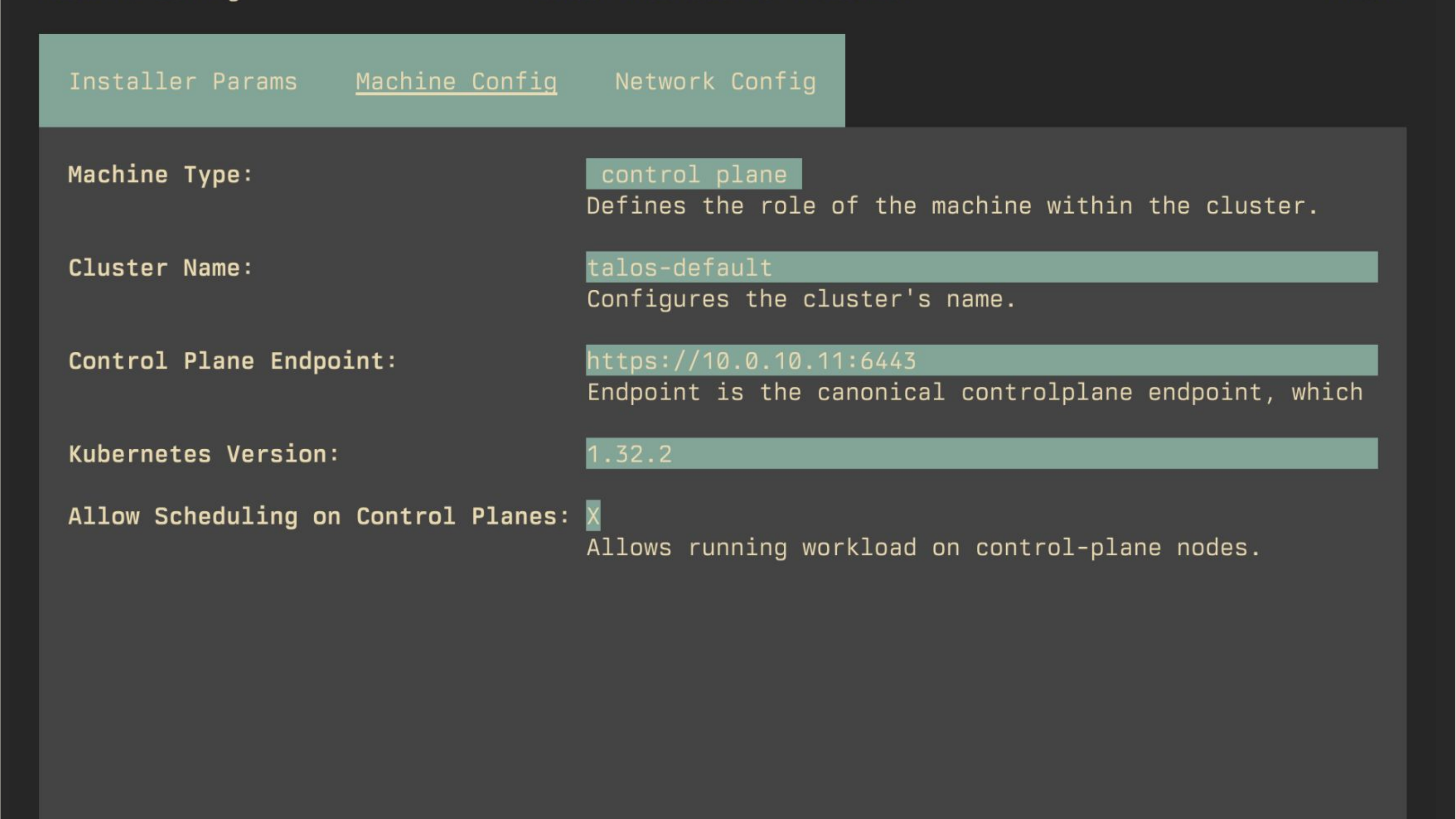


```
[talos] this machine is reachable at: {"component": "controller-runtime",
"controller": "config.MaintenanceServiceController"}
[talos]  10.0.10.11 {"component": "controller-runtime", "controller":
"runtime.MaintenanceServiceController"}
[talos] upload configuration using talosctl: {"component": "controller-runtime",
"controller": "runtime.MaintenanceServiceController"}
[talos]  talosctl apply-config --insecure --nodes 10.0.10.11 --file <config.yaml>
{"component": "controller-runtime", "controller":
"runtime.MaintenanceServiceController"}
[talos] or apply configuration using talosctl interactive installer: {"component":
"controller-runtime", "controller": "runtime.MaintenanceServiceController"}
[talos]  talosctl apply-config --insecure --nodes 10.0.10.11 --mode=interactive
{"component": "controller-runtime", "controller":
"runtime.MaintenanceServiceController"}
```





```
[talos] this machine is reachable at:
[talos]  10.0.10.11
[talos] upload configuration using talosctl:
[talos]  talosctl apply-config --insecure --nodes 10.0.10.11 --file <config.yaml>
[talos] or apply configuration using talosctl interactive installer:
[talos]  talosctl apply-config --insecure --nodes 10.0.10.11 --mode=interactive
```



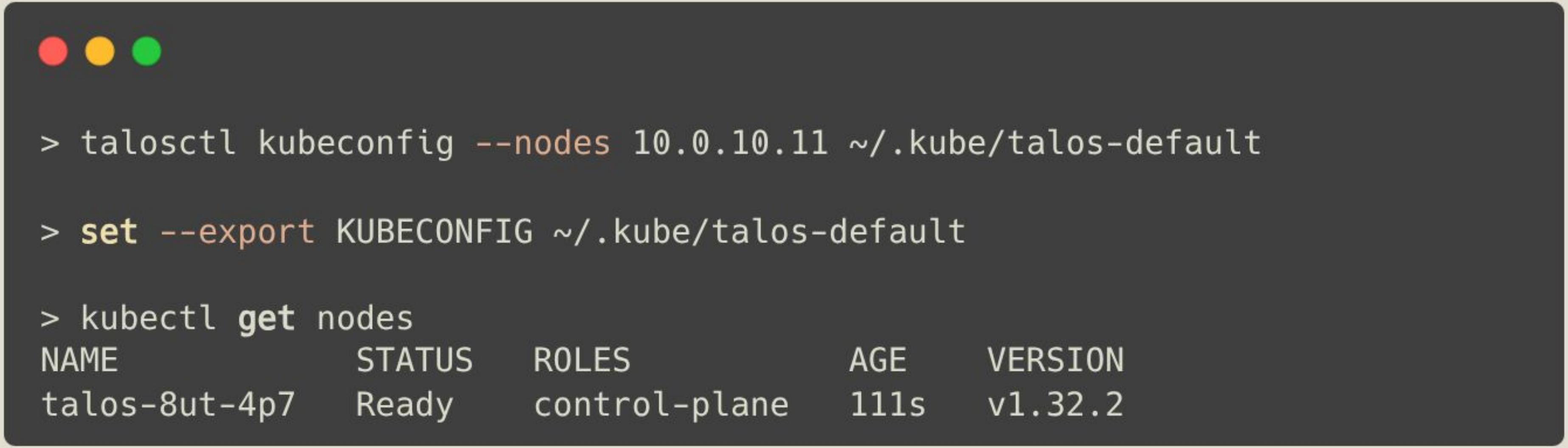
Machine Type: control plane
Defines the role of the machine within the cluster.

Cluster Name: talos-default
Configures the cluster's name.

Control Plane Endpoint: https://10.0.10.11:6443
Endpoint is the canonical controlplane endpoint, which

Kubernetes Version: 1.32.2

Allow Scheduling on Control Planes: X
Allows running workload on control-plane nodes.



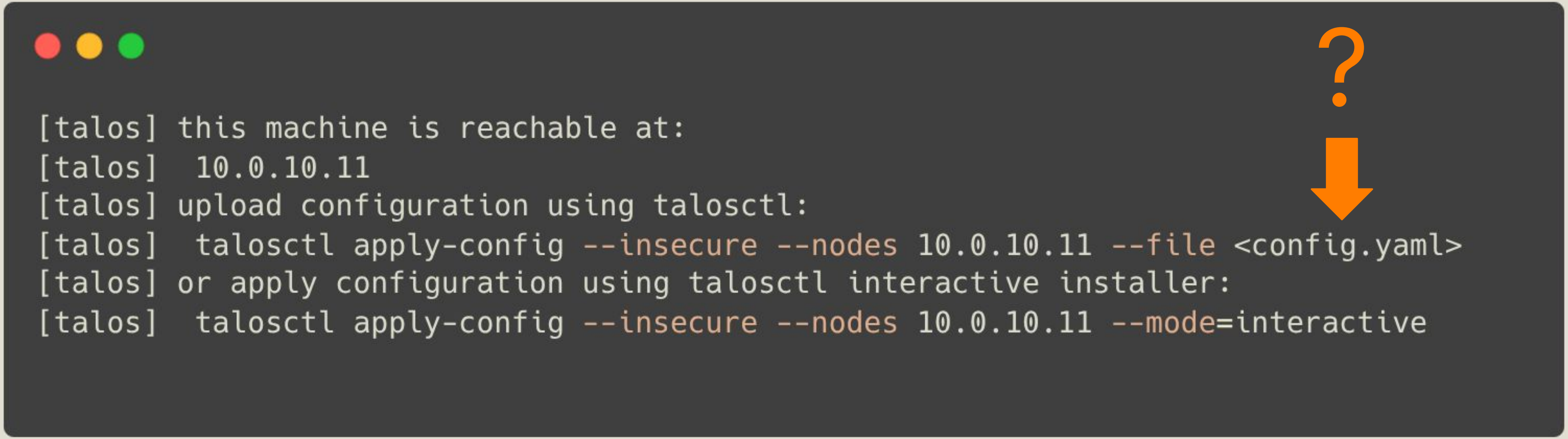
```
> talosctl kubeconfig --nodes 10.0.10.11 ~/.kube/talos-default

> set --export KUBECONFIG ~/.kube/talos-default

> kubectl get nodes
NAME              STATUS   ROLES           AGE    VERSION
talos-8ut-4p7     Ready    control-plane   111s   v1.32.2
```

HECK YEAH, KUBERNETES!

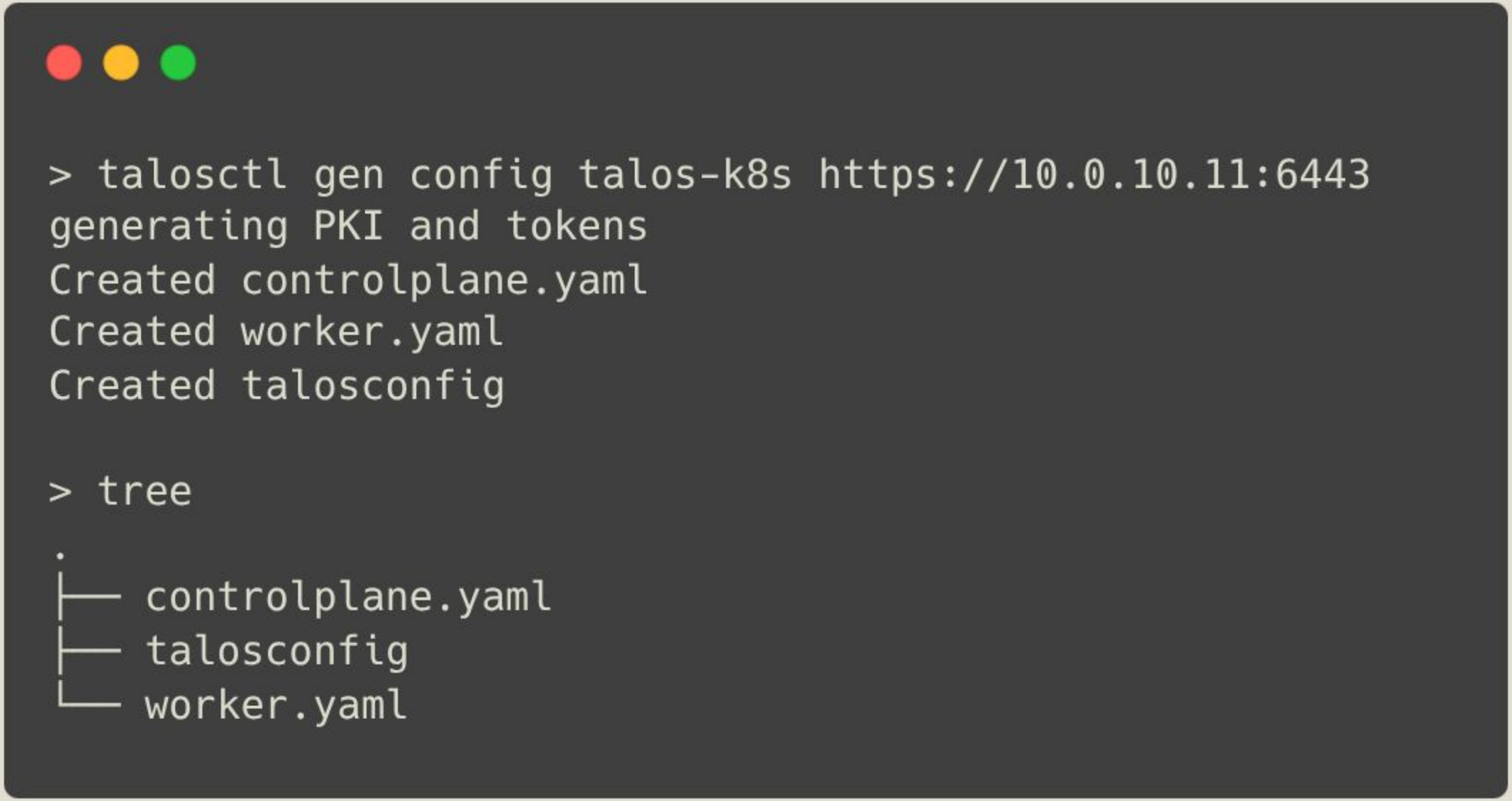
TALOS CAN UPDATE ITSELF...

TALOS CAN UPDATE ITSELF...AND KUBERNETES!

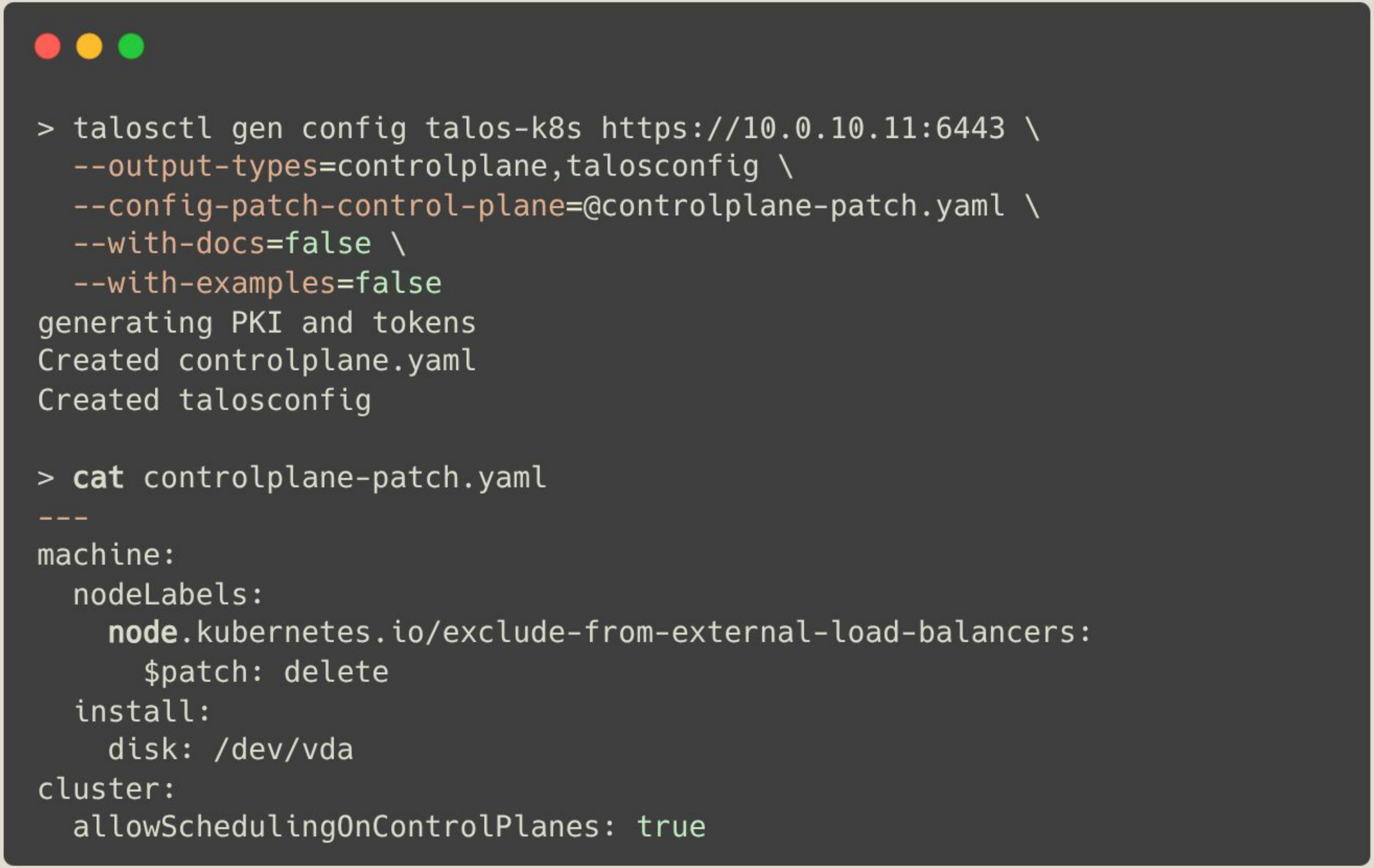
HOW DO I CUSTOMIZE MY INSTALLATION?



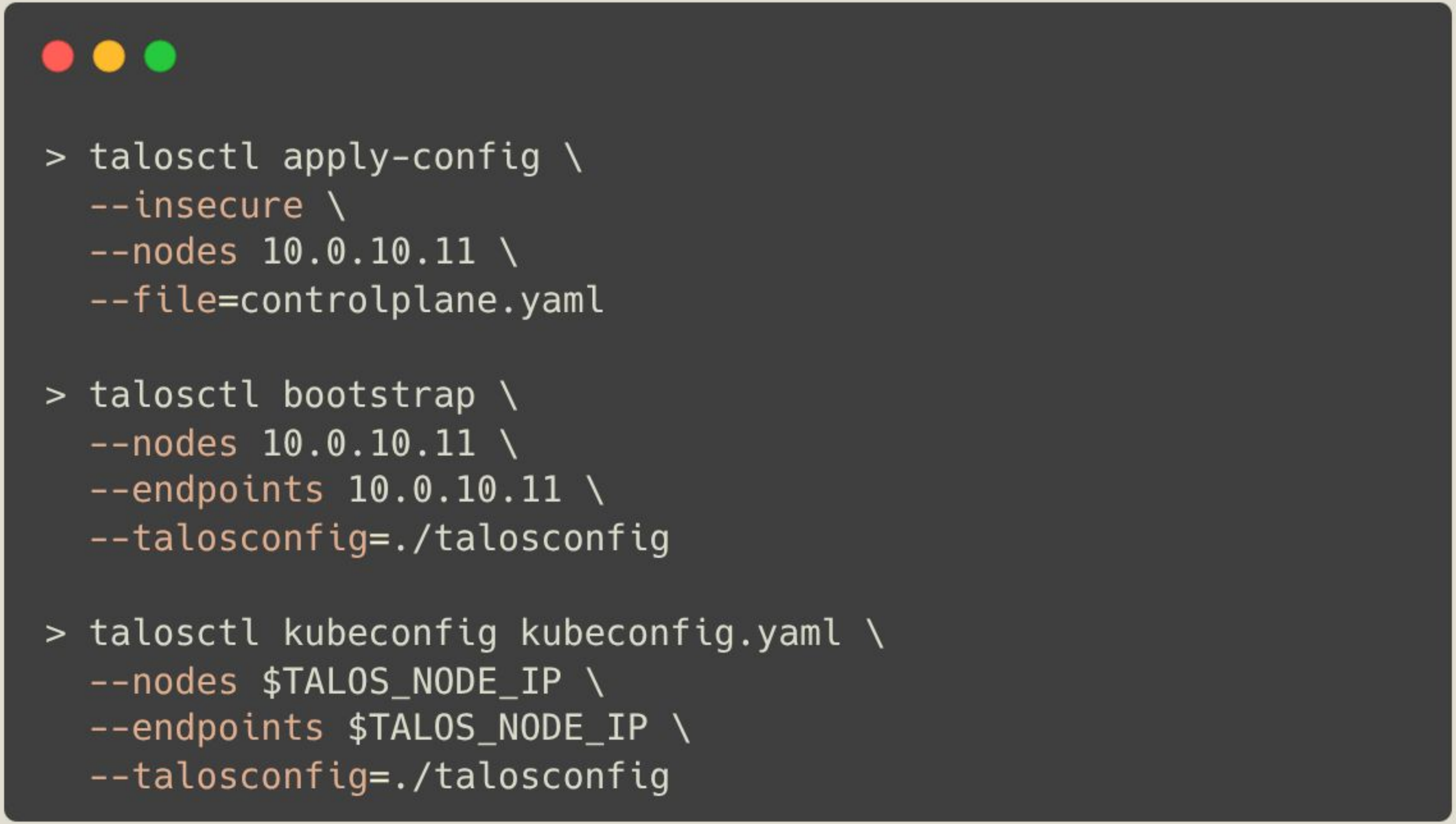
```
[talos] this machine is reachable at:
[talos]  10.0.10.11
[talos] upload configuration using talosctl:
[talos]  talosctl apply-config --insecure --nodes 10.0.10.11 --file <config.yaml>
[talos] or apply configuration using talosctl interactive installer:
[talos]  talosctl apply-config --insecure --nodes 10.0.10.11 --mode=interactive
```



```
> talosctl gen config talos-k8s https://10.0.10.11:6443
generating PKI and tokens
Created controlplane.yaml
Created worker.yaml
Created talosconfig

> tree
.
├── controlplane.yaml
├── talosconfig
└── worker.yaml
```



```
> talosctl gen config talos-k8s https://10.0.10.11:6443 \
  --output-types=controlplane,talosconfig \
  --config-patch-control-plane=@controlplane-patch.yaml \
  --with-docs=false \
  --with-examples=false
generating PKI and tokens
Created controlplane.yaml
Created talosconfig

> cat controlplane-patch.yaml
---
machine:
  nodeLabels:
    node.kubernetes.io/exclude-from-external-load-balancers:
      $patch: delete
  install:
    disk: /dev/vda
cluster:
  allowSchedulingOnControlPlanes: true
```

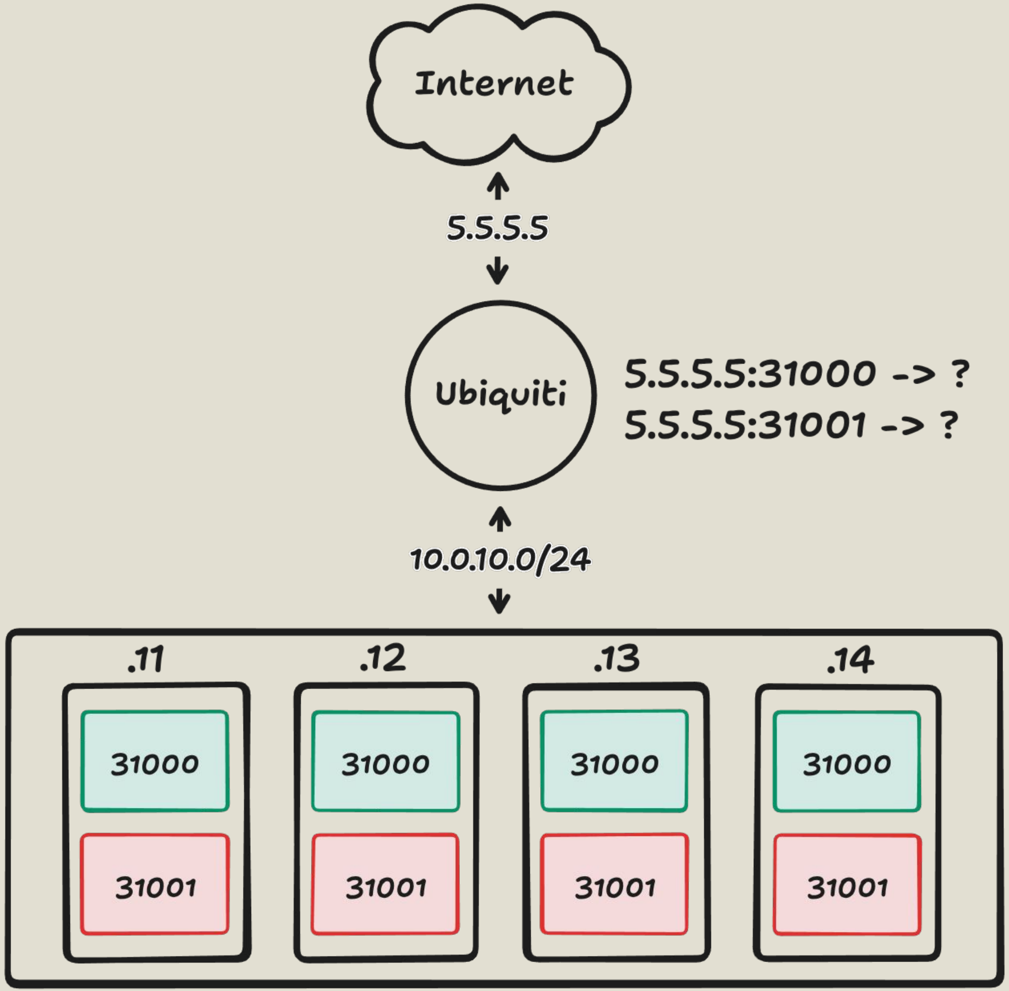


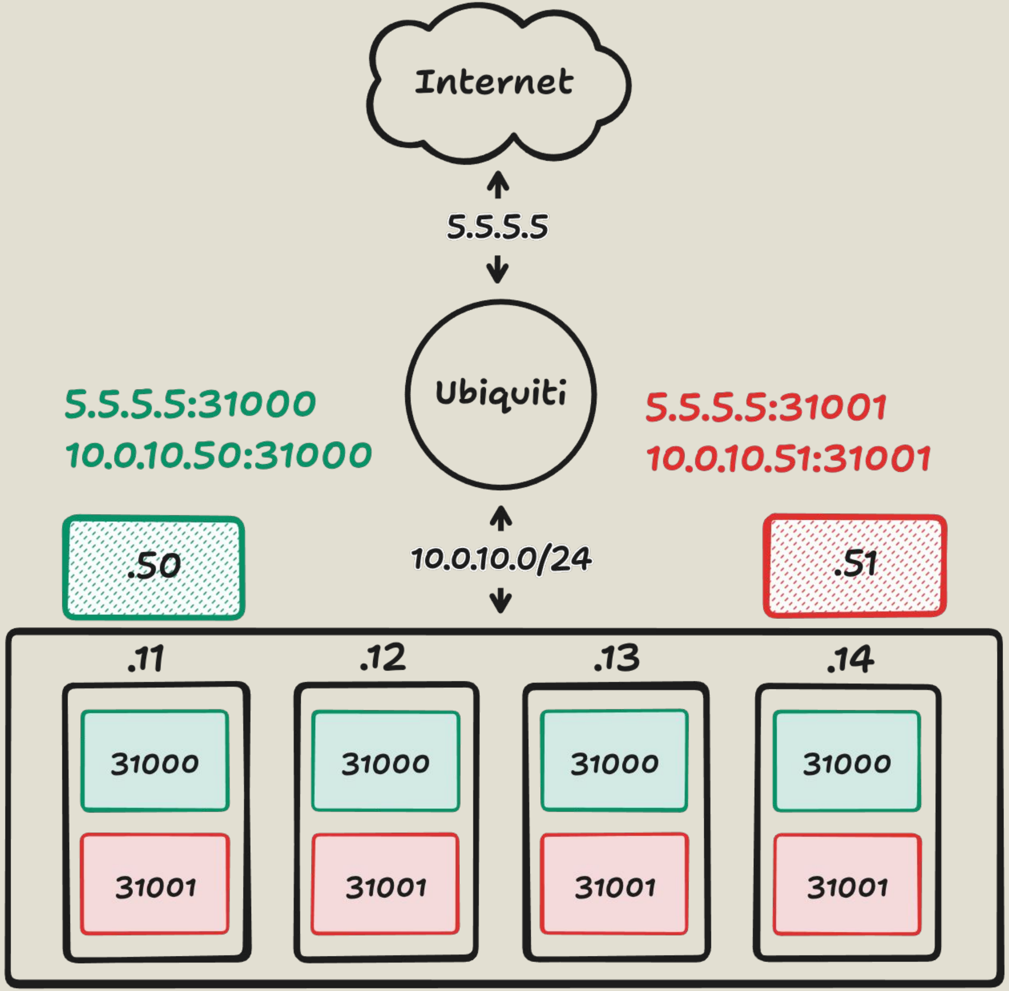
```
> talosctl apply-config \
  --insecure \
  --nodes 10.0.10.11 \
  --file=controlplane.yaml

> talosctl bootstrap \
  --nodes 10.0.10.11 \
  --endpoints 10.0.10.11 \
  --talosconfig=./talosconfig

> talosctl kubeconfig kubeconfig.yaml \
  --nodes $TALOS_NODE_IP \
  --endpoints $TALOS_NODE_IP \
  --talosconfig=./talosconfig
```

HECK YEAH, TALOS KUBERNETES!

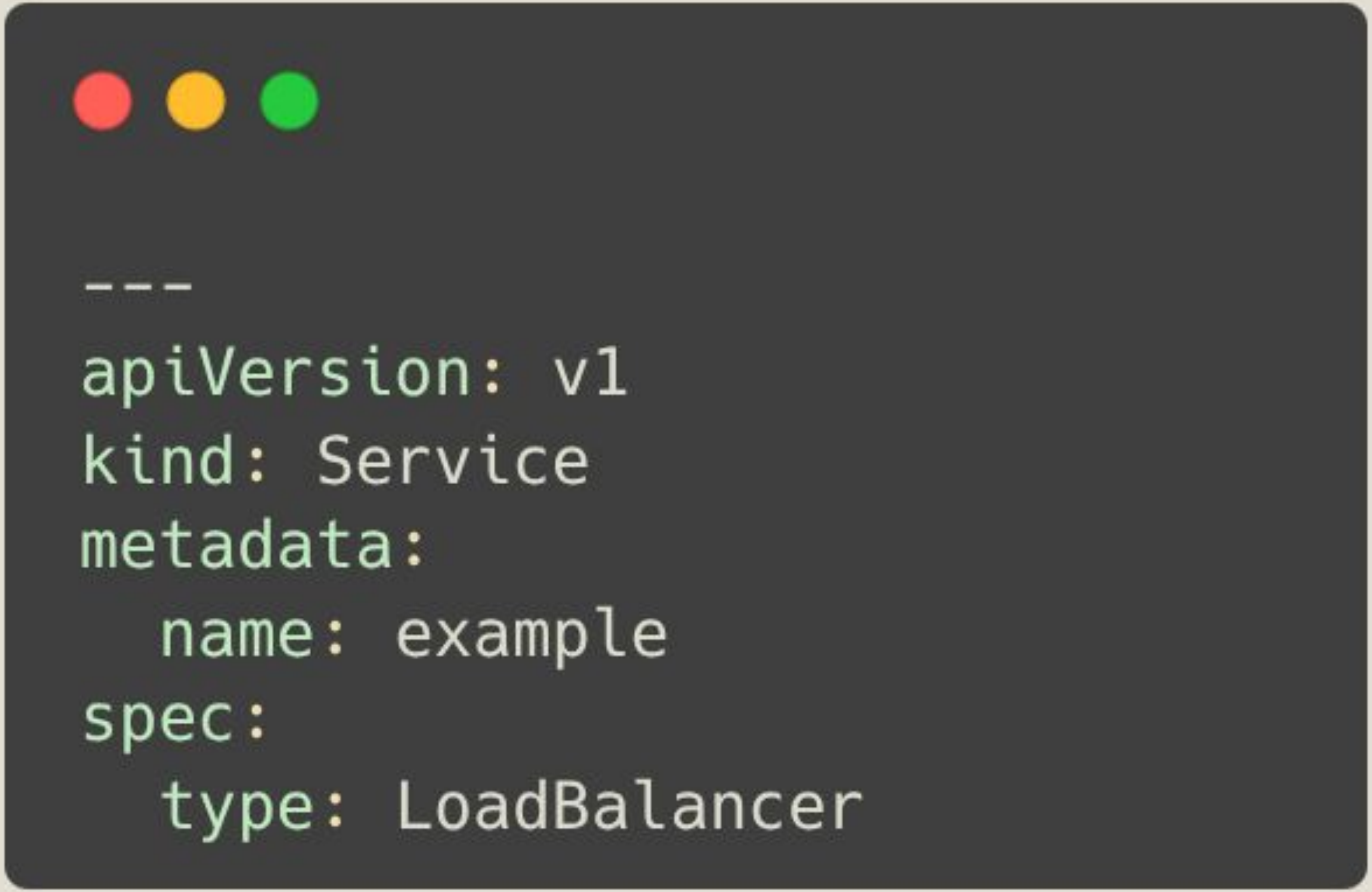
LOAD BALANCER

HOW DOES LAYER 4 CONNECTIVITY WORK?


Internet
5.5.5.5
Ubiquiti
5.5.5.5:31000 -> ?
5.5.5.5:31001 -> ?
10.0.10.0/24
.11
31000
31001
.12
31000
31001
.13
31000
31001
.14
31000
31001


Internet
5.5.5.5
Ubiquiti
5.5.5.5:31000
10.0.10.50:31000
5.5.5.5:31001
10.0.10.51:31001
.50
10.0.10.0/24
.51
.11
.12
.13
.14
31000
31001
31000
31001
31000
31001
31000
31001

A QUICK LOAD BALANCER ~~RANT~~ TANGENT



```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: example
spec:
  type: LoadBalancer
```

METALLB

- Supports bare-metal Kubernetes
- Provides L4 load balancer
- Uses standard protocols
  - ARP
  - BGP

L2ADVERTISEMENT

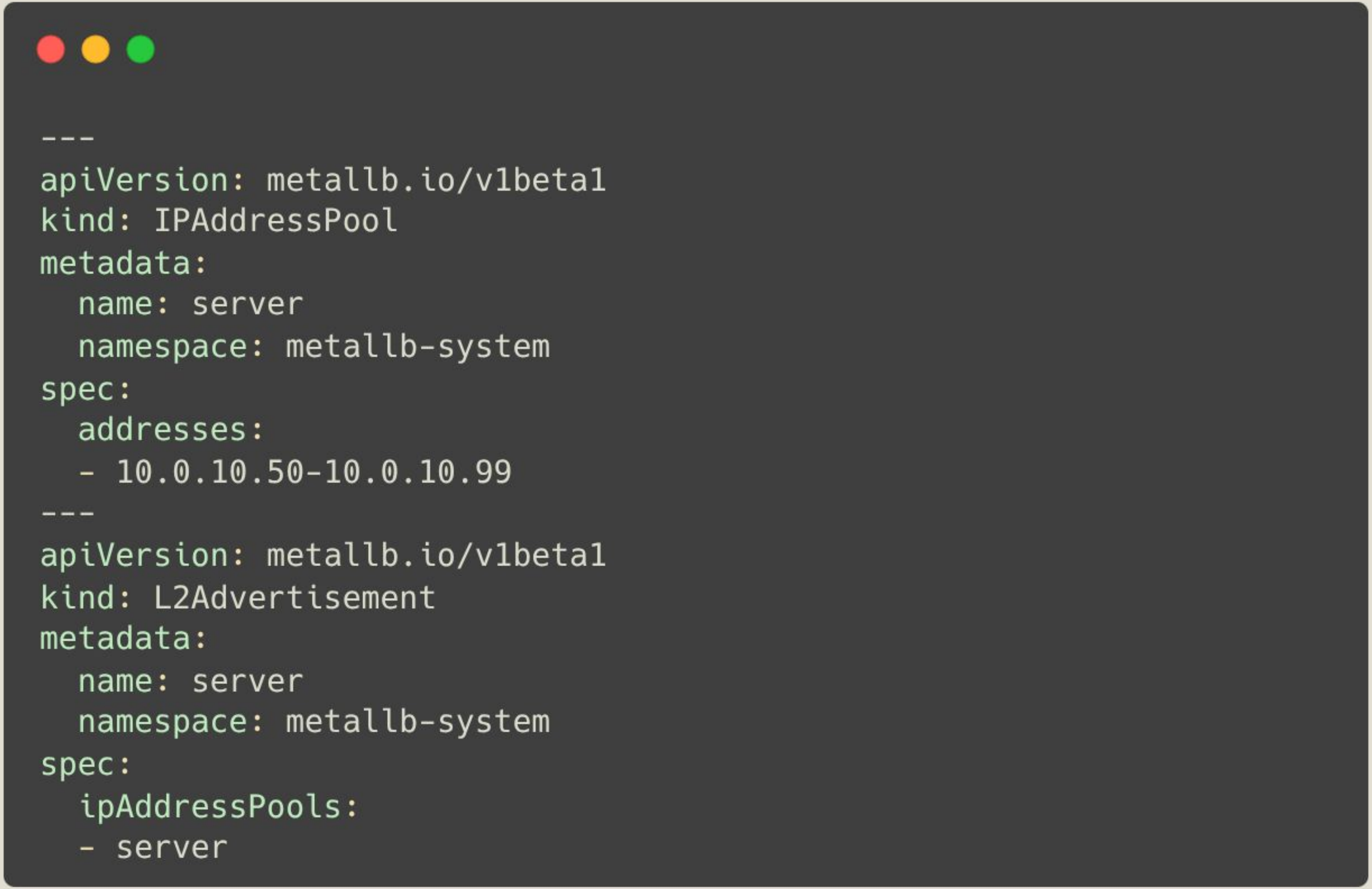
- Listens on an IP from a pool
- Advertises the IP using ARP
- Waits for traffic on the IP

More network noise

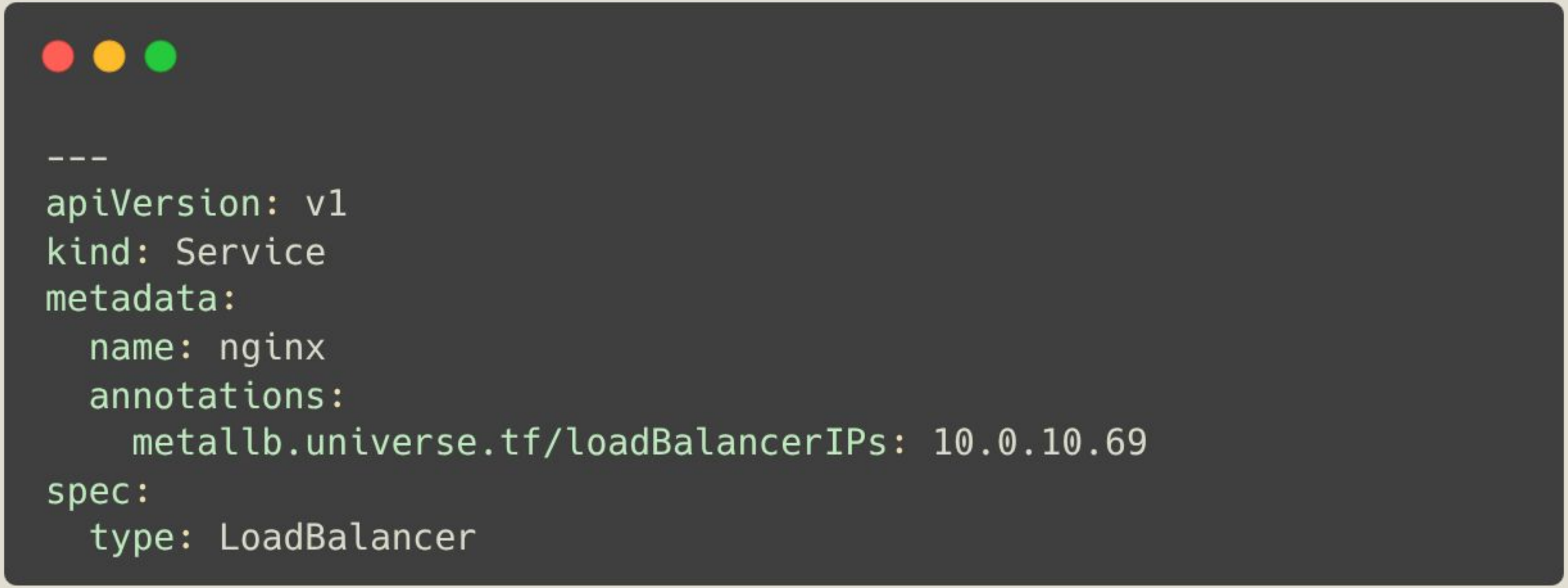
BGPADVERTISEMENT

- Listens on an IP from a pool
- Advertises the IP using BGP
- Waits for traffic on the IP

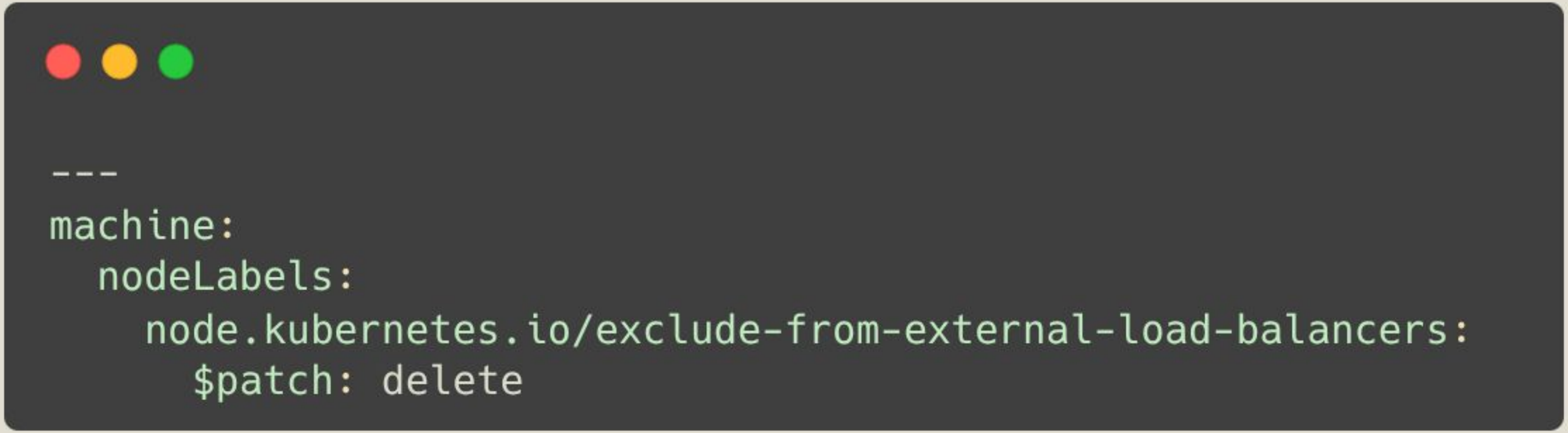
More involved setup



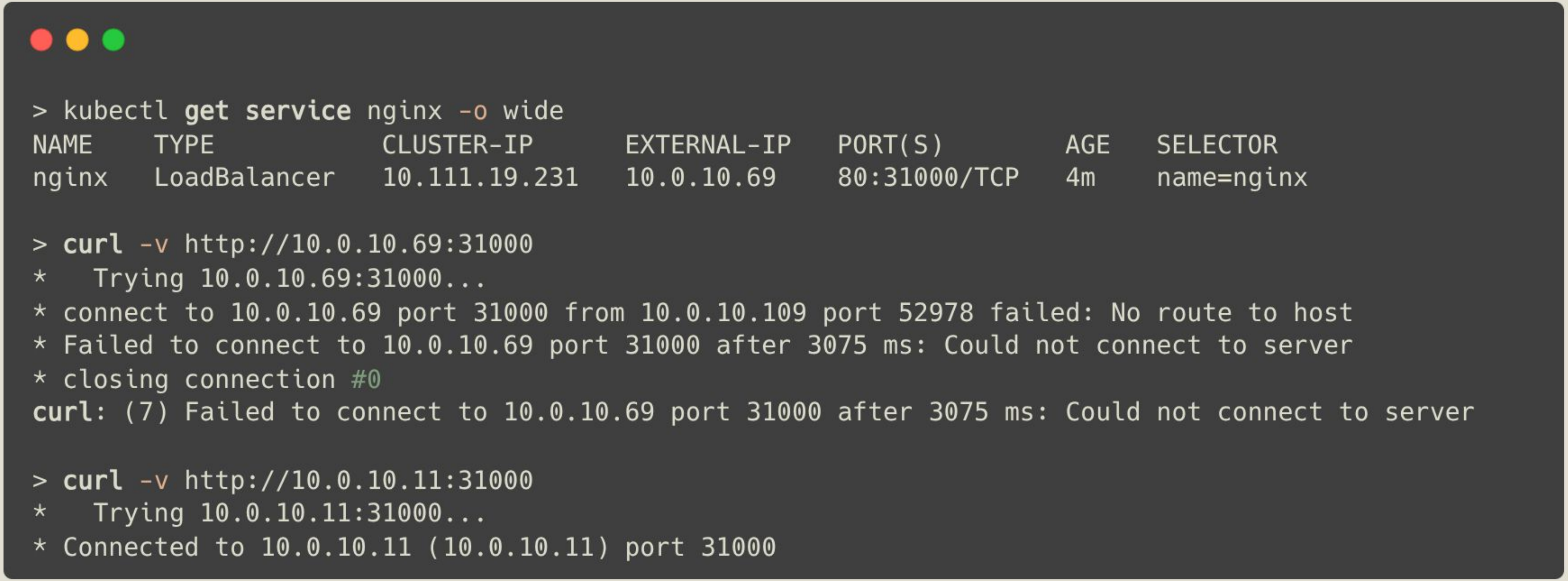
```yaml
---
apiVersion: metallb.io/v1beta1
kind: IPAddressPool
metadata:
  name: server
  namespace: metallb-system
spec:
  addresses:
  - 10.0.10.50-10.0.10.99
---
apiVersion: metallb.io/v1beta1
kind: L2Advertisement
metadata:
  name: server
  namespace: metallb-system
spec:
  ipAddressPools:
  - server
```



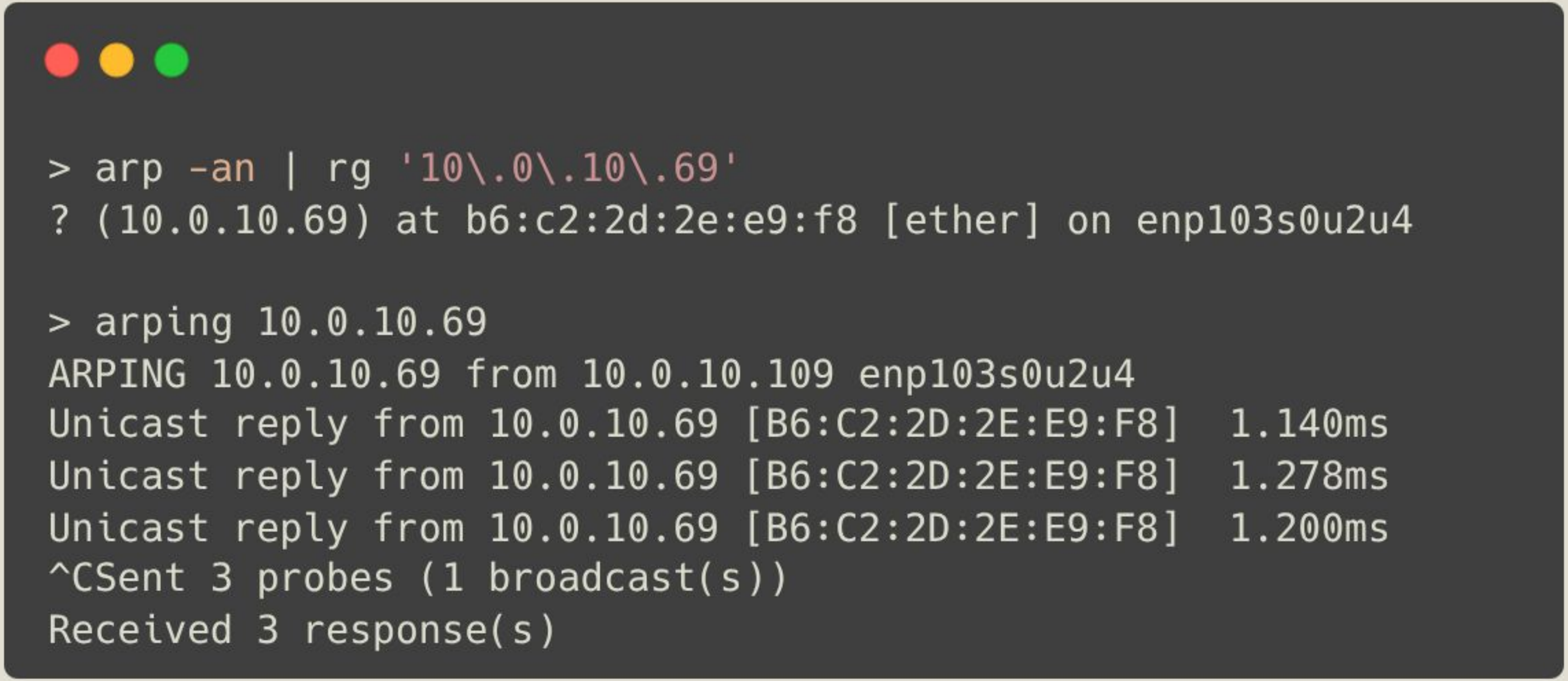
```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: nginx
  annotations:
    metallb.universe.tf/loadBalancerIPs: 10.0.10.69
spec:
  type: LoadBalancer
```



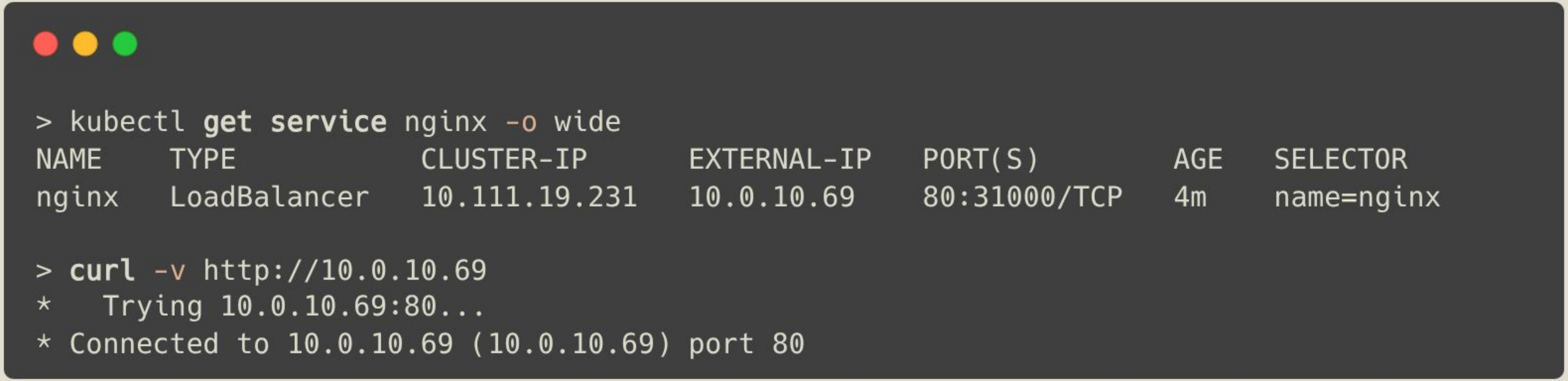
```yaml
---
machine:
  nodeLabels:
    node.kubernetes.io/exclude-from-external-load-balancers:
      $patch: delete
```



```
> kubectl get service nginx -o wide
NAME    TYPE           CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE   SELECTOR
nginx   LoadBalancer   10.111.19.231  10.0.10.69    80:31000/TCP   4m    name=nginx

> curl -v http://10.0.10.69:31000
*   Trying 10.0.10.69:31000...
* connect to 10.0.10.69 port 31000 from 10.0.10.109 port 52978 failed: No route to host
* Failed to connect to 10.0.10.69 port 31000 after 3075 ms: Could not connect to server
* closing connection #0
curl: (7) Failed to connect to 10.0.10.69 port 31000 after 3075 ms: Could not connect to server

> curl -v http://10.0.10.11:31000
*   Trying 10.0.10.11:31000...
* Connected to 10.0.10.11 (10.0.10.11) port 31000
```



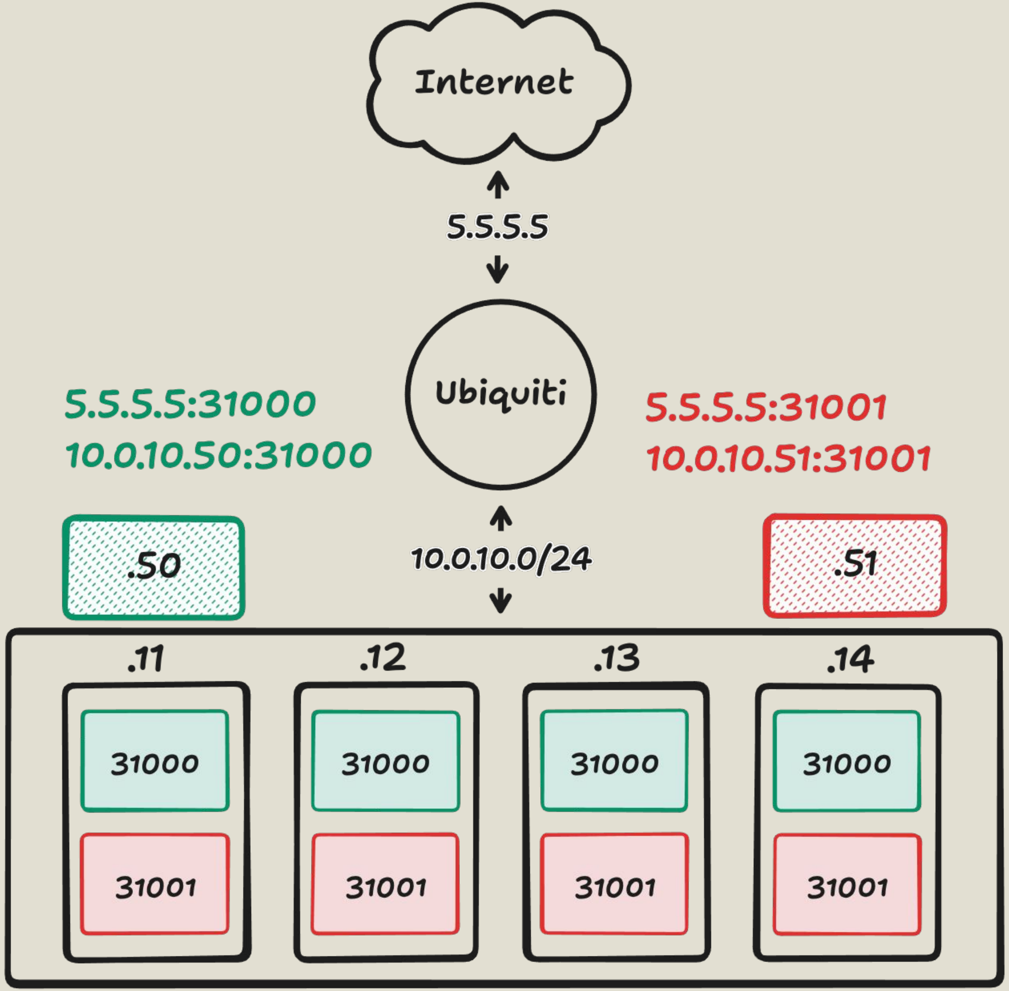
```
> arp -an | rg '10\.0\.10\.69'
? (10.0.10.69) at b6:c2:2d:2e:e9:f8 [ether] on enp103s0u2u4

> arping 10.0.10.69
ARPING 10.0.10.69 from 10.0.10.109 enp103s0u2u4
Unicast reply from 10.0.10.69 [B6:C2:2D:2E:E9:F8]  1.140ms
Unicast reply from 10.0.10.69 [B6:C2:2D:2E:E9:F8]  1.278ms
Unicast reply from 10.0.10.69 [B6:C2:2D:2E:E9:F8]  1.200ms
^CSent 3 probes (1 broadcast(s))
Received 3 response(s)
```

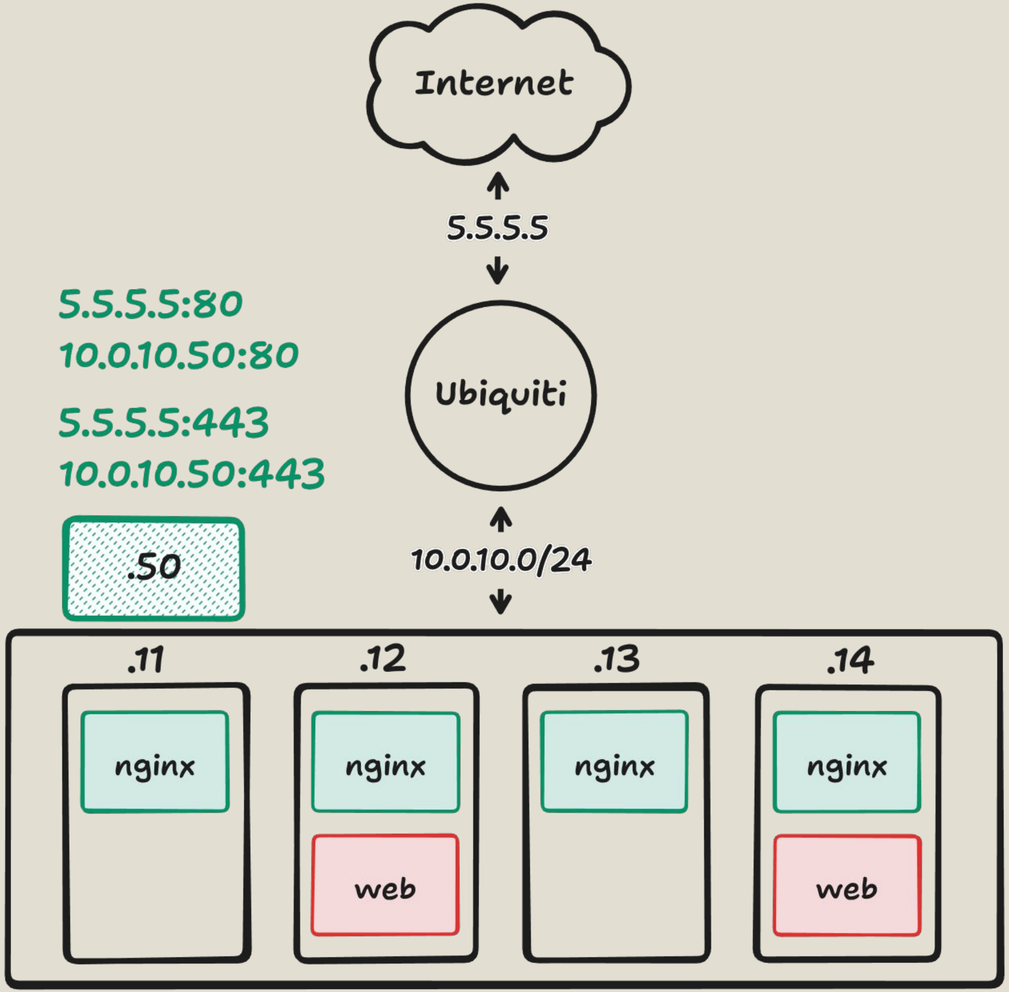


```
> kubectl get service nginx -o wide
NAME    TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE   SELECTOR
nginx   LoadBalancer   10.111.19.231   10.0.10.69    80:31000/TCP   4m    name=nginx

> curl -v http://10.0.10.69
*   Trying 10.0.10.69:80...
* Connected to 10.0.10.69 (10.0.10.69) port 80
```

INGRESS CONTROLLER

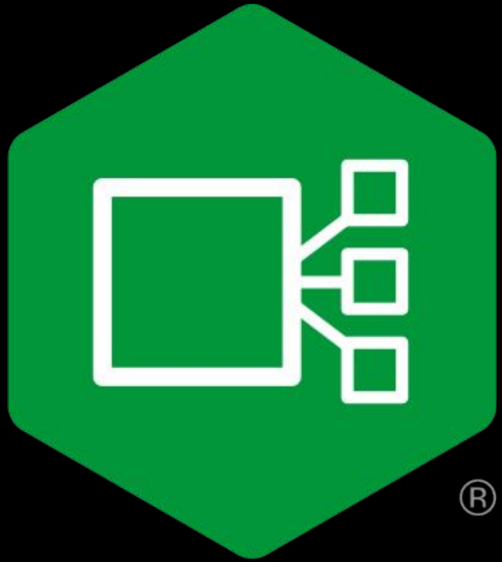
HOW DOES LAYER 7 CONNECTIVITY WORK?

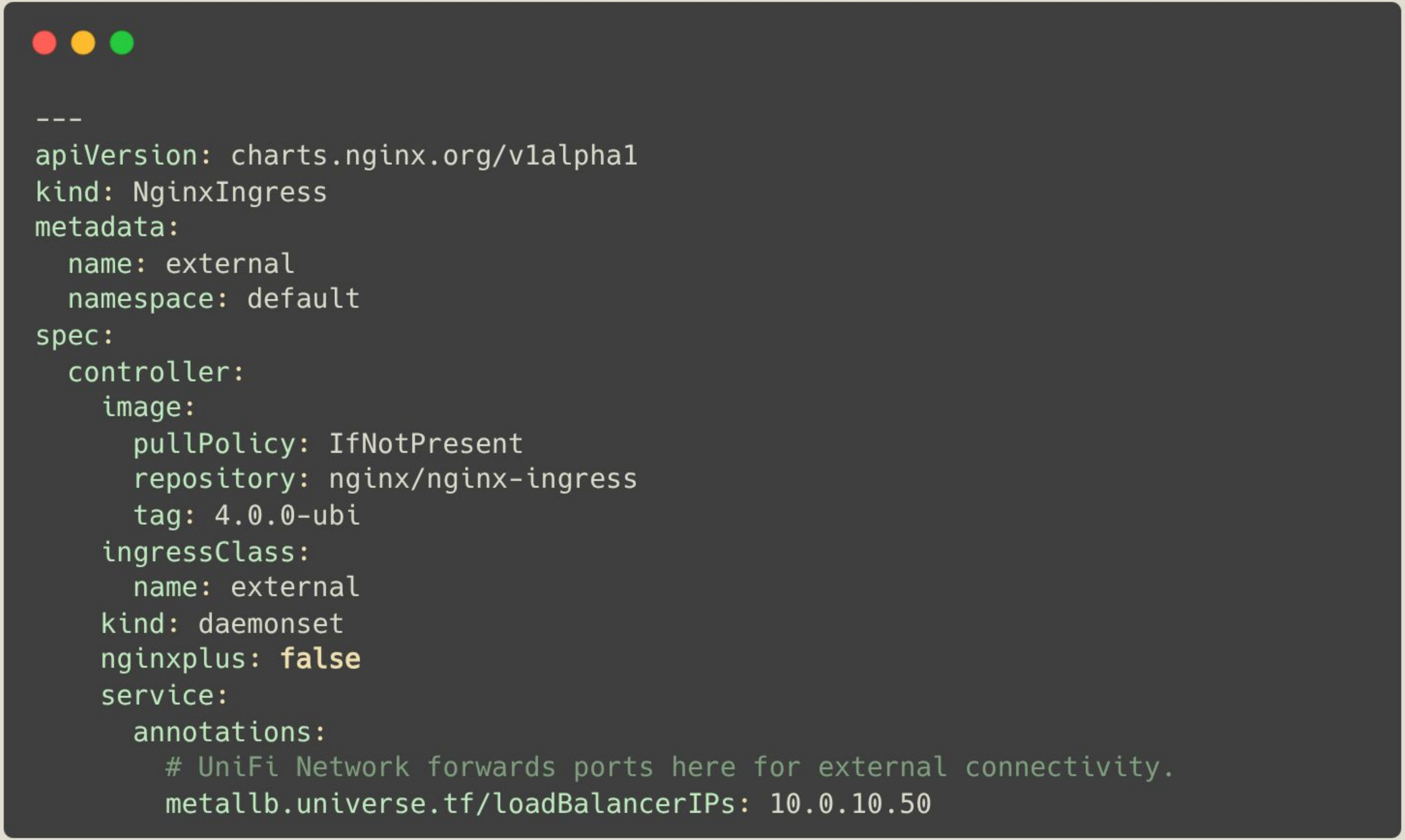

Internet
5.5.5.5
Ubiquiti
5.5.5.5:31000
10.0.10.50:31000
5.5.5.5:31001
10.0.10.51:31001
.50
10.0.10.0/24
.51
.11
.12
.13
.14
31000
31001
31000
31001
31000
31001
31000
31001


Internet
5.5.5.5
Ubiquiti
5.5.5.5:80
10.0.10.50:80
5.5.5.5:443
10.0.10.50:443
.50
10.0.10.0/24
.11
.12
.13
.14
nginx
nginx
nginx
nginx
web
web

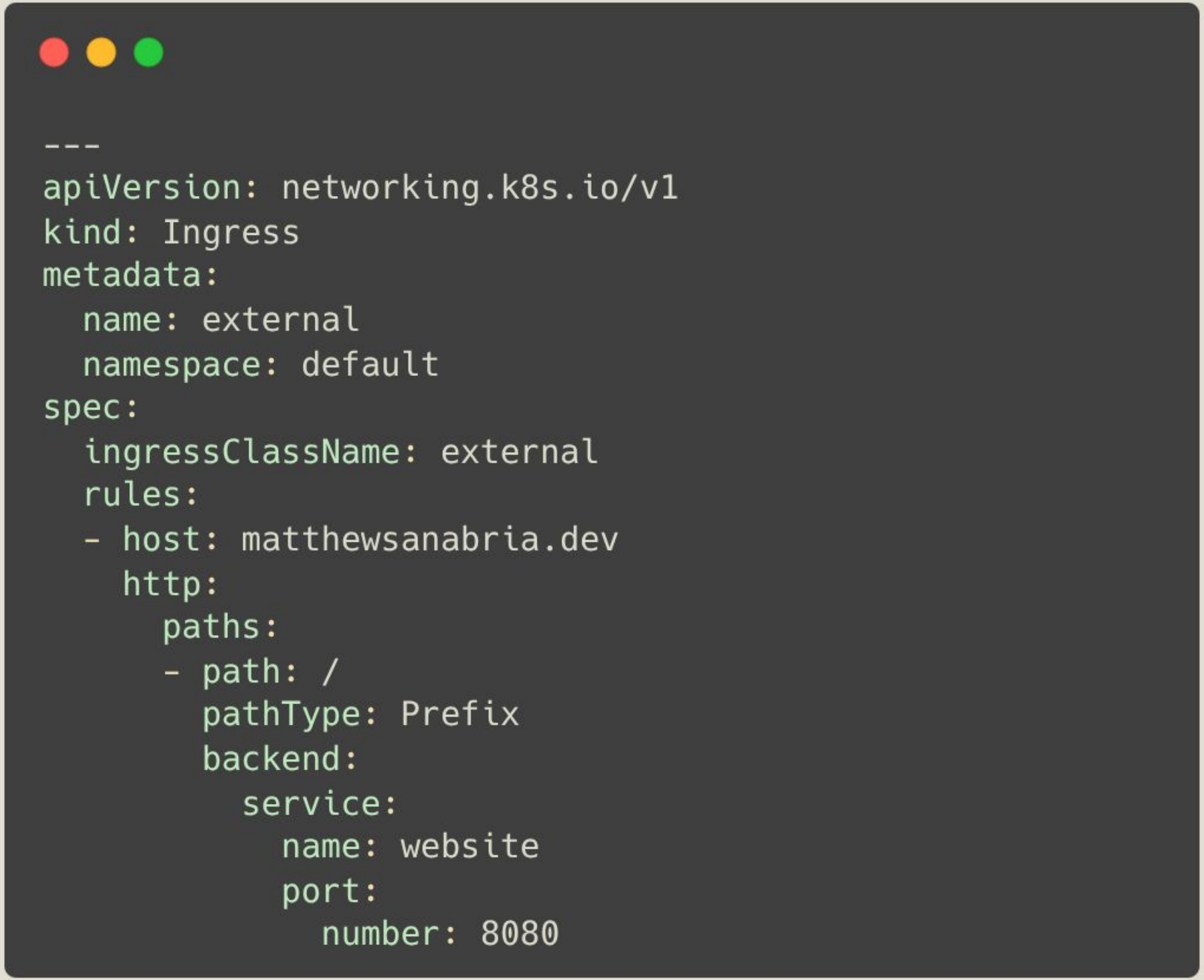
NGINX INGRESS CONTROLLER

- Ingress Controller implementation
- Extremely popular
- Supports TCP and UDP

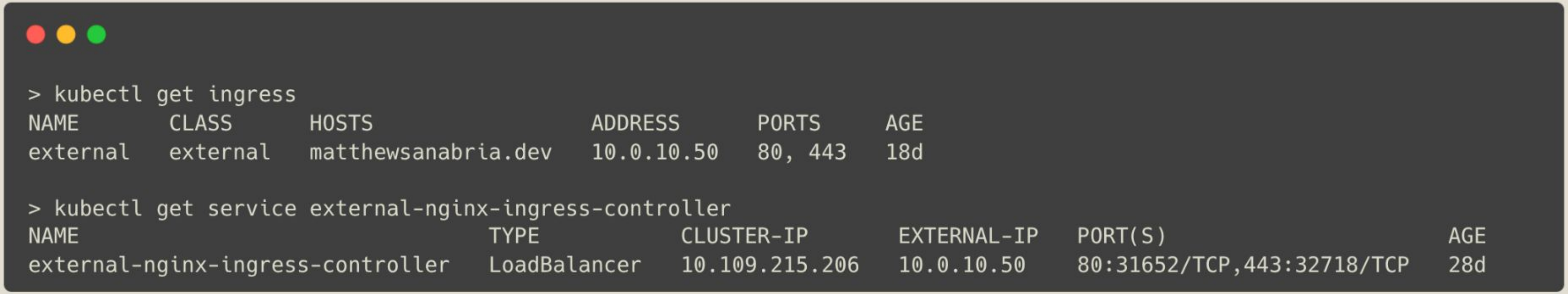




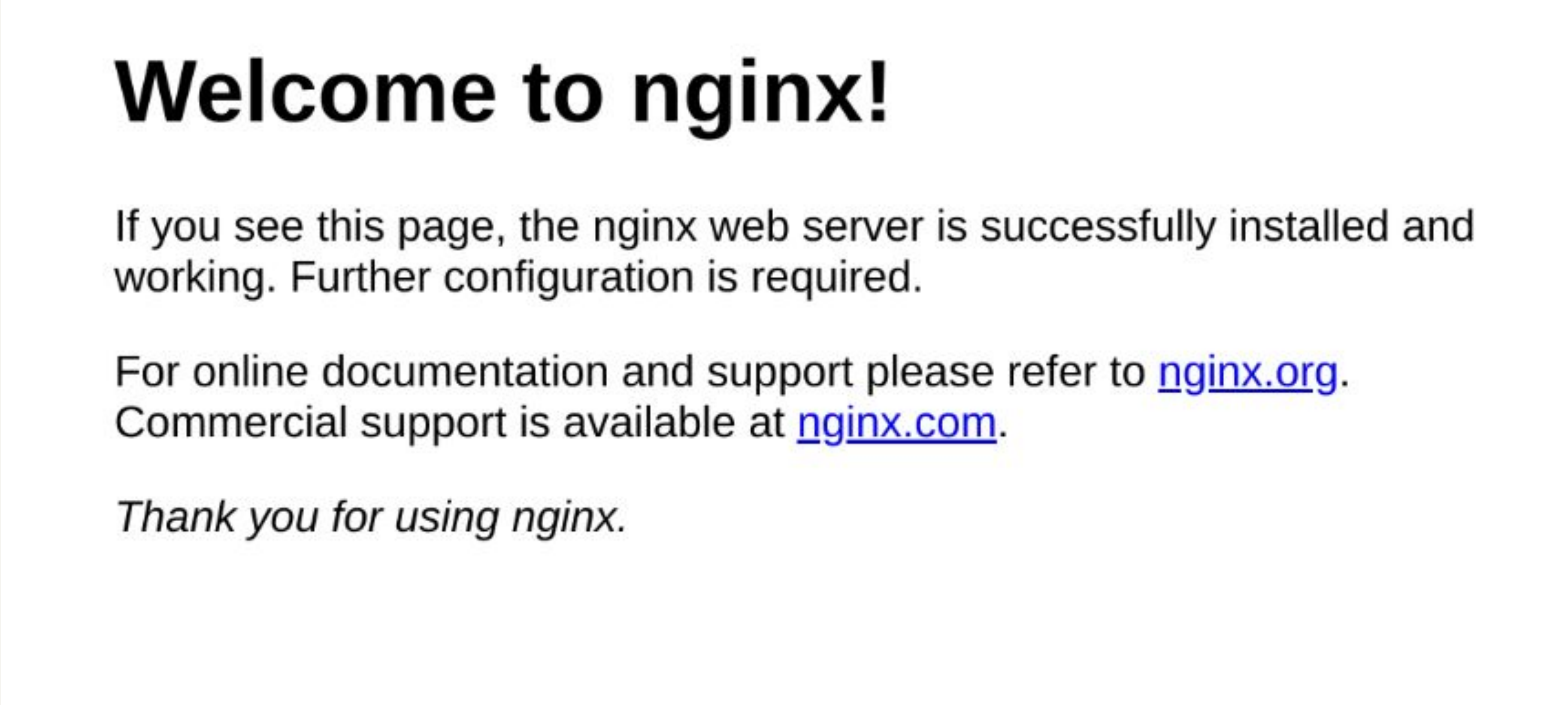
```yaml
---
apiVersion: charts.nginx.org/v1alpha1
kind: NginxIngress
metadata:
  name: external
  namespace: default
spec:
  controller:
    image:
      pullPolicy: IfNotPresent
      repository: nginx/nginx-ingress
      tag: 4.0.0-ubi
    ingressClass:
      name: external
    kind: daemonset
    nginxplus: false
    service:
      annotations:
        # UniFi Network forwards ports here for external connectivity.
        metallb.universe.tf/loadBalancerIPs: 10.0.10.50
```



```yaml
---
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: external
  namespace: default
spec:
  ingressClassName: external
  rules:
  - host: matthewsanabria.dev
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: website
            port:
              number: 8080
```



```
> kubectl get ingress
NAME       CLASS      HOSTS                 ADDRESS      PORTS     AGE
external   external   matthewsanabria.dev   10.0.10.50   80, 443   18d

> kubectl get service external-nginx-ingress-controller
NAME                                TYPE           CLUSTER-IP       EXTERNAL-IP    PORT(S)                      AGE
external-nginx-ingress-controller   LoadBalancer   10.109.215.206   10.0.10.50     80:31652/TCP,443:32718/TCP   28d
```



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

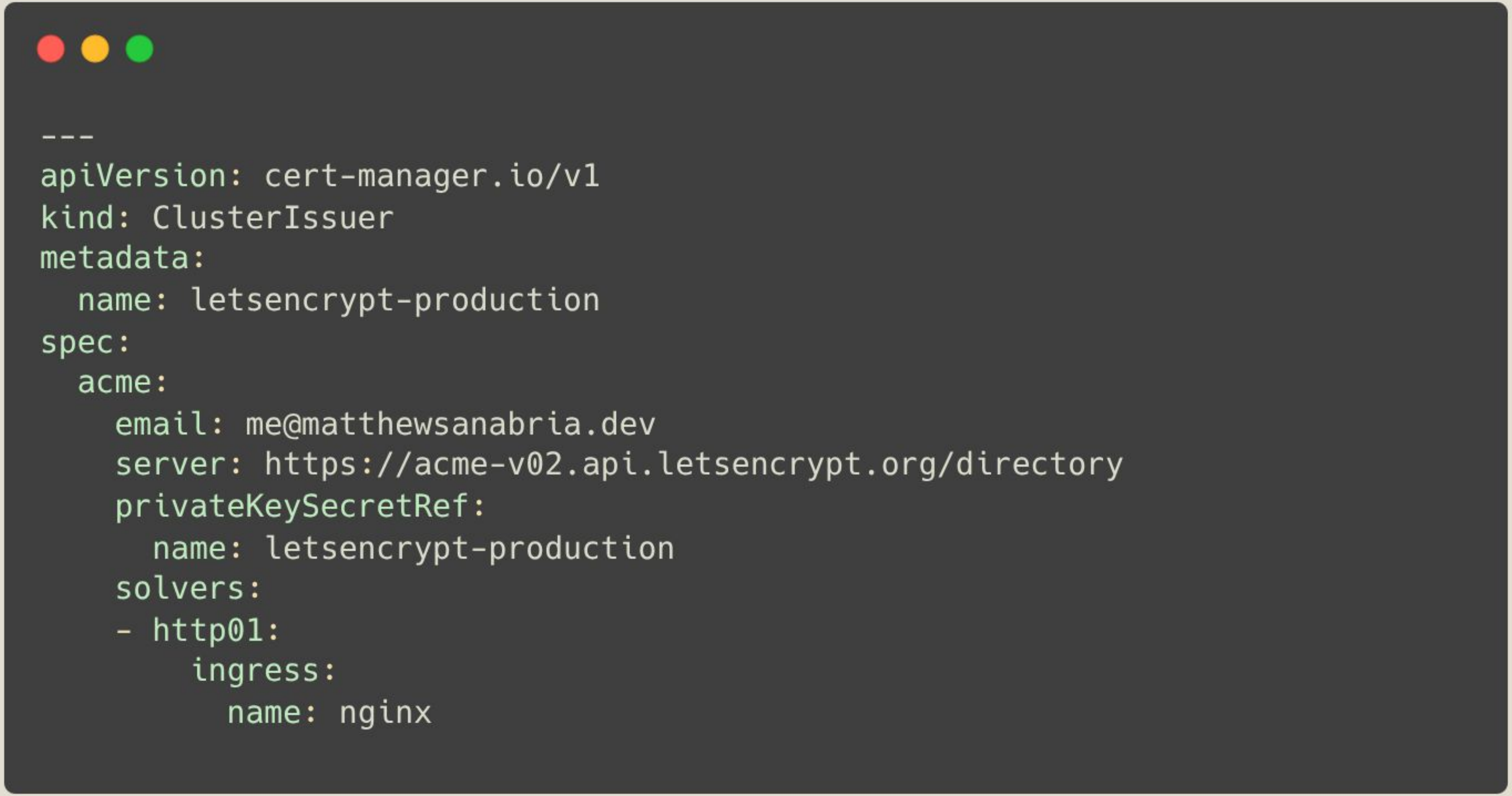
Thank you for using nginx.

TLS CERTIFICATES

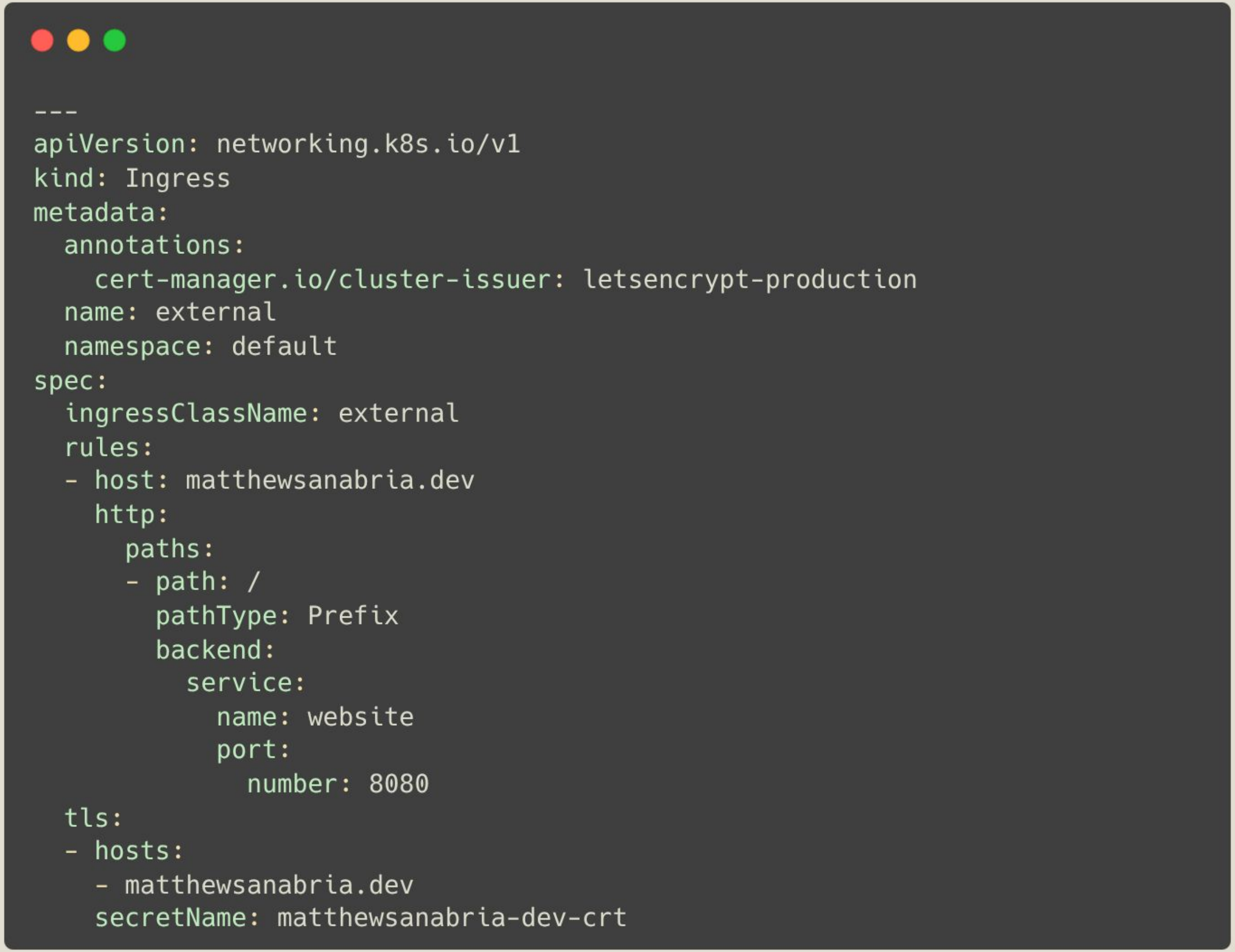
HTTP IS GREAT, BUT WHAT ABOUT HTTPS?

CERT-MANAGER

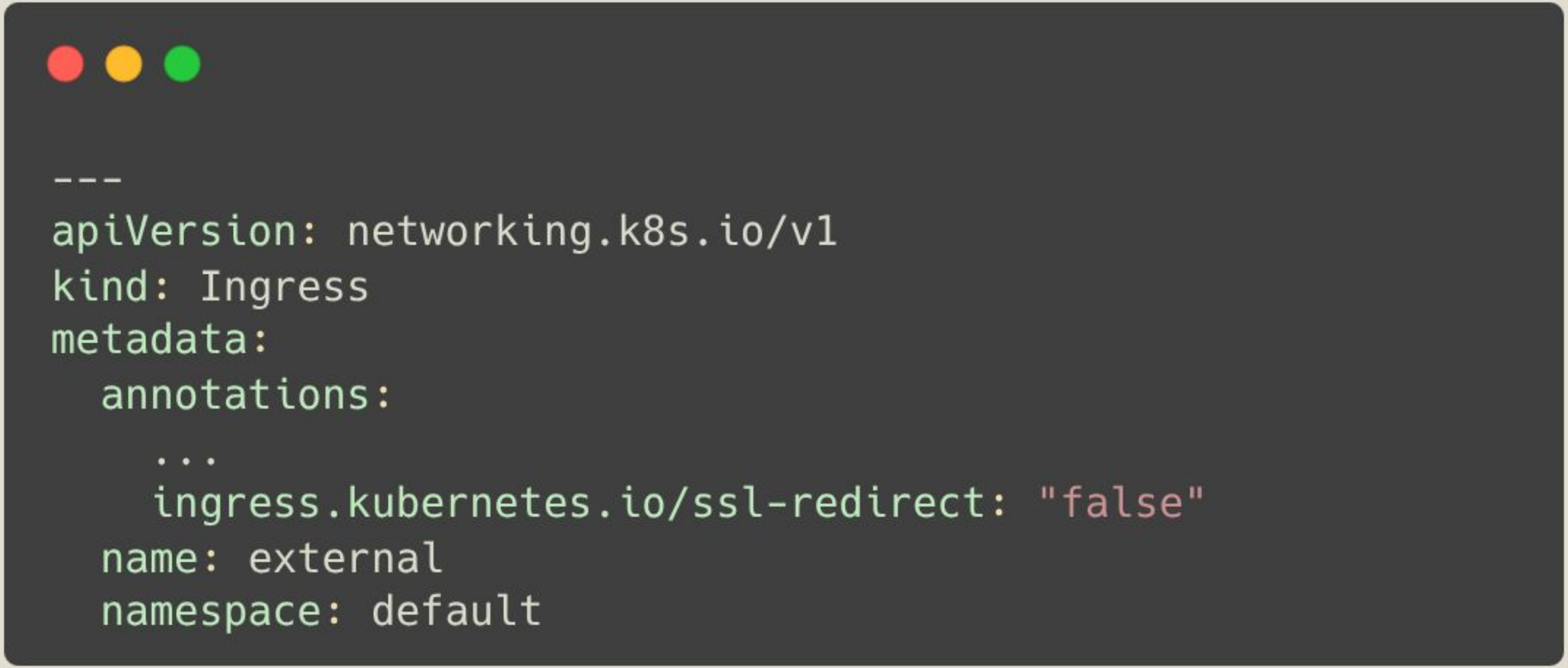
- X.509 certificate management
- Supports LetsEncrypt/ACME
- Handles certificate renewal



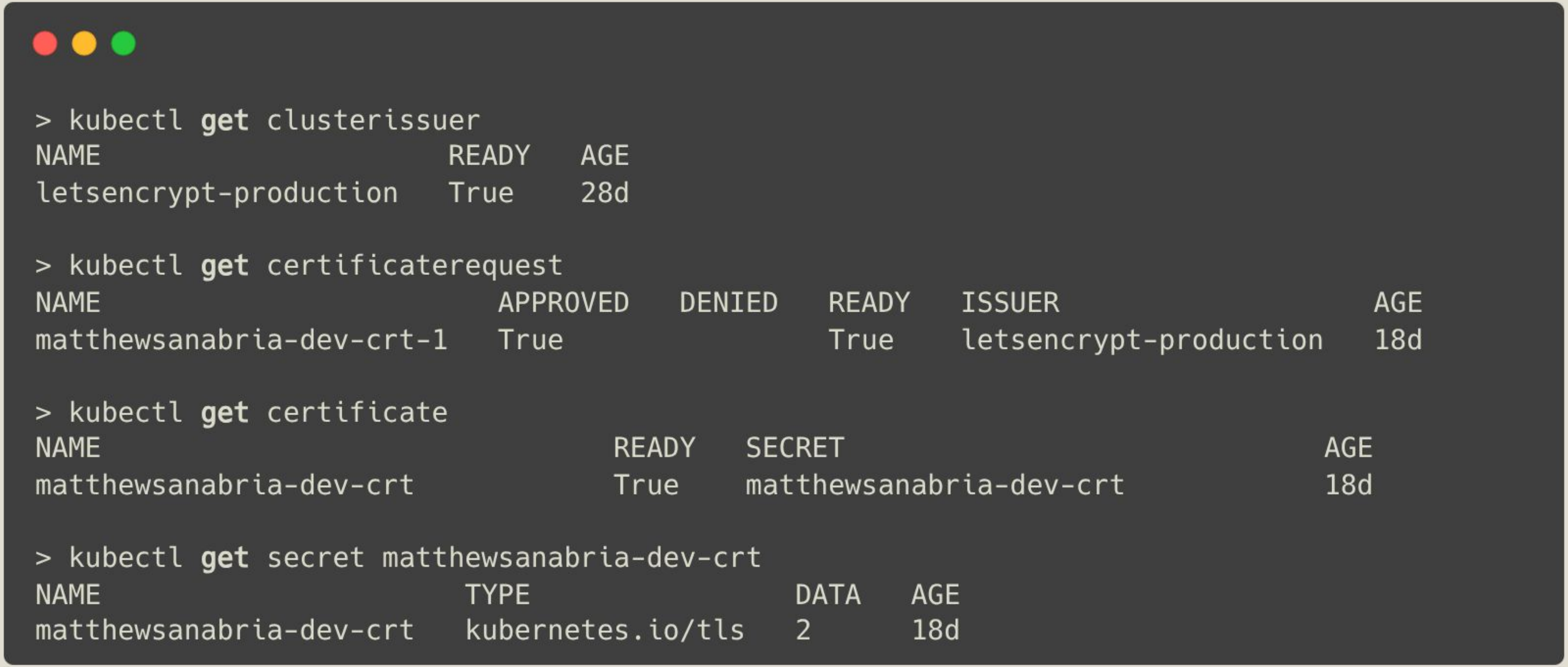
```yaml
---
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: letsencrypt-production
spec:
  acme:
    email: me@matthewsanabria.dev
    server: https://acme-v02.api.letsencrypt.org/directory
    privateKeySecretRef:
      name: letsencrypt-production
    solvers:
    - http01:
        ingress:
          name: nginx
```



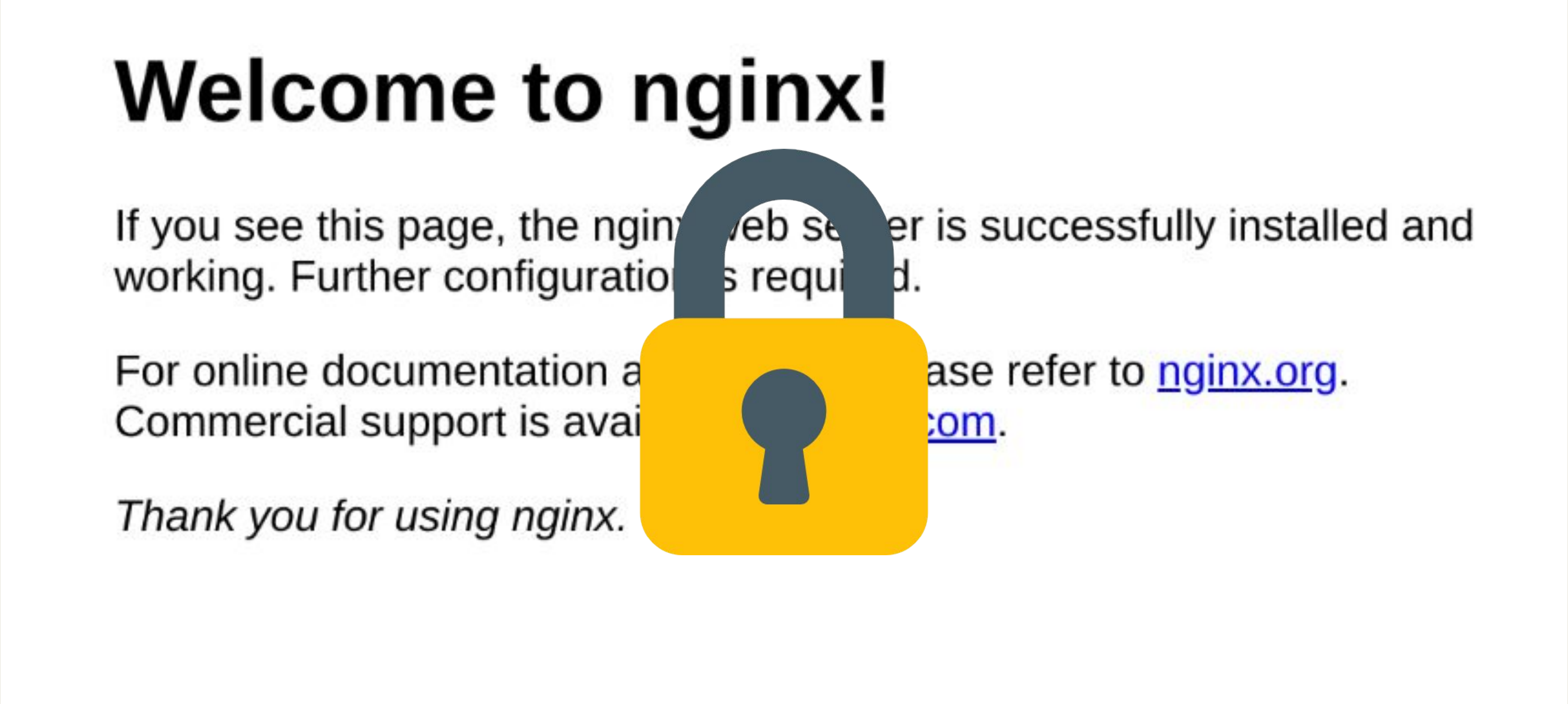
```yaml
---
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  annotations:
    cert-manager.io/cluster-issuer: letsencrypt-production
  name: external
  namespace: default
spec:
  ingressClassName: external
  rules:
  - host: matthewsanabria.dev
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: website
            port:
              number: 8080
  tls:
  - hosts:
    - matthewsanabria.dev
    secretName: matthewsanabria-dev-crt
```



```
---
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  annotations:
    ...
    ingress.kubernetes.io/ssl-redirect: "false"
  name: external
  namespace: default
```



```
> kubectl get clusterissuer
NAME                     READY   AGE
letsencrypt-production   True    28d

> kubectl get certificaterequest
NAME                        APPROVED   DENIED   READY   ISSUER                   AGE
matthewsanabria-dev-crt-1   True                True    letsencrypt-production   18d

> kubectl get certificate
NAME                               READY   SECRET                             AGE
matthewsanabria-dev-crt            True    matthewsanabria-dev-crt            18d

> kubectl get secret matthewsanabria-dev-crt
NAME                      TYPE                DATA   AGE
matthewsanabria-dev-crt   kubernetes.io/tls   2      18d
```



Welcome to nginx!

If you see this page, the ngin eb se er is successfully installed and working. Further configuratio s requi d.

For online documentation a ase refer to nginx.org. Commercial support is avai com.

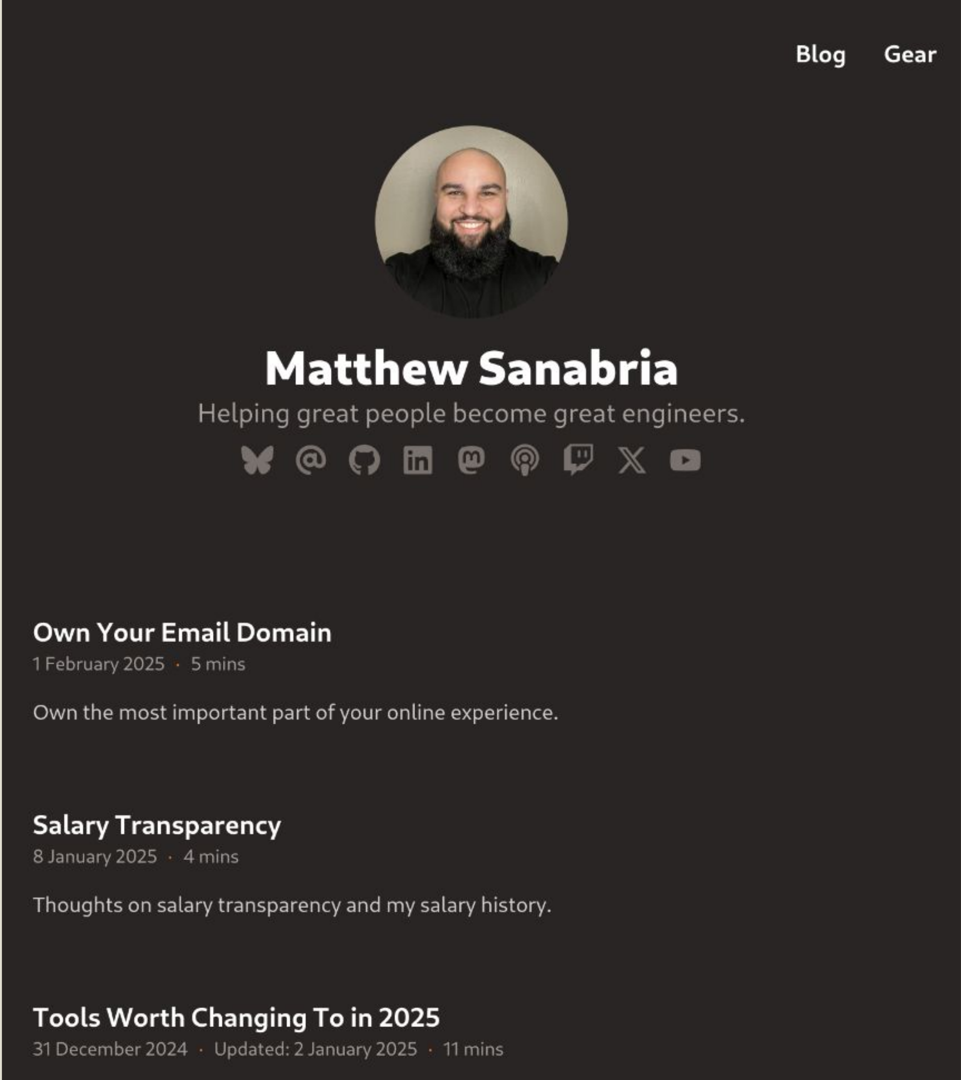
Thank you for using nginx.

WEBSITE MIGRATION

LET'S TEST THIS HOME LAB!

MATTHEWSANABRIA.DEV

- Built with Hugo
- Previously hosted on Digital Ocean
- Great test for the home lab



Matthew Sanabria

Helping great people become great engineers.

Own Your Email Domain

1 February 2025 · 5 mins

Own the most important part of your online experience.

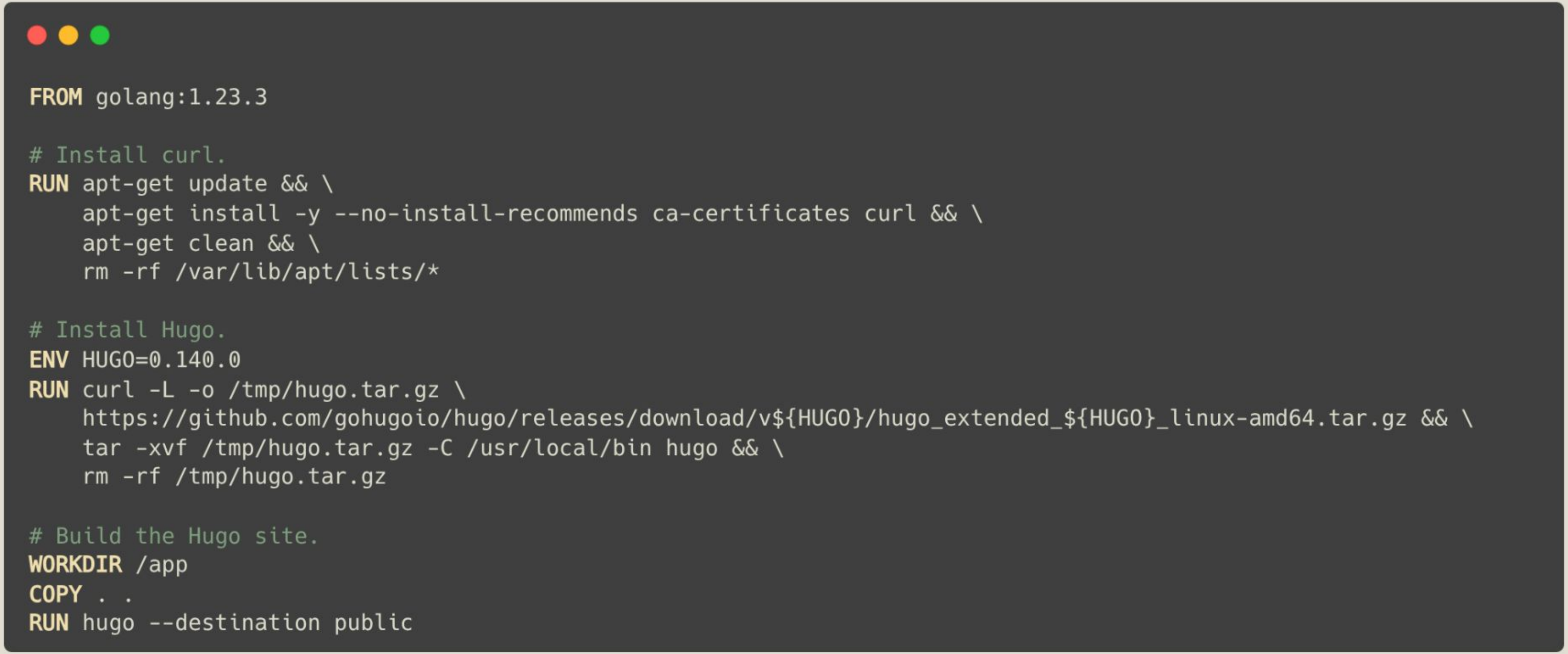
Salary Transparency

8 January 2025 · 4 mins

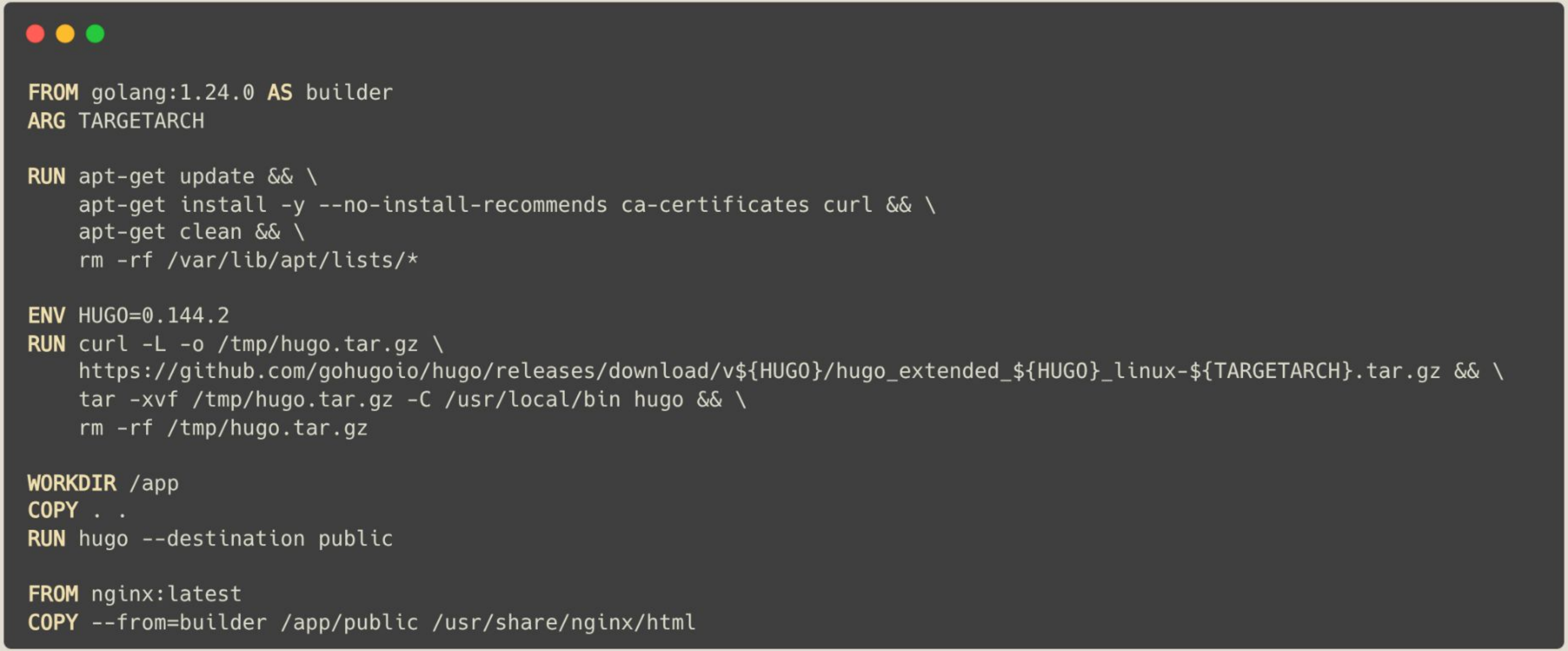
Thoughts on salary transparency and my salary history.

Tools Worth Changing To in 2025

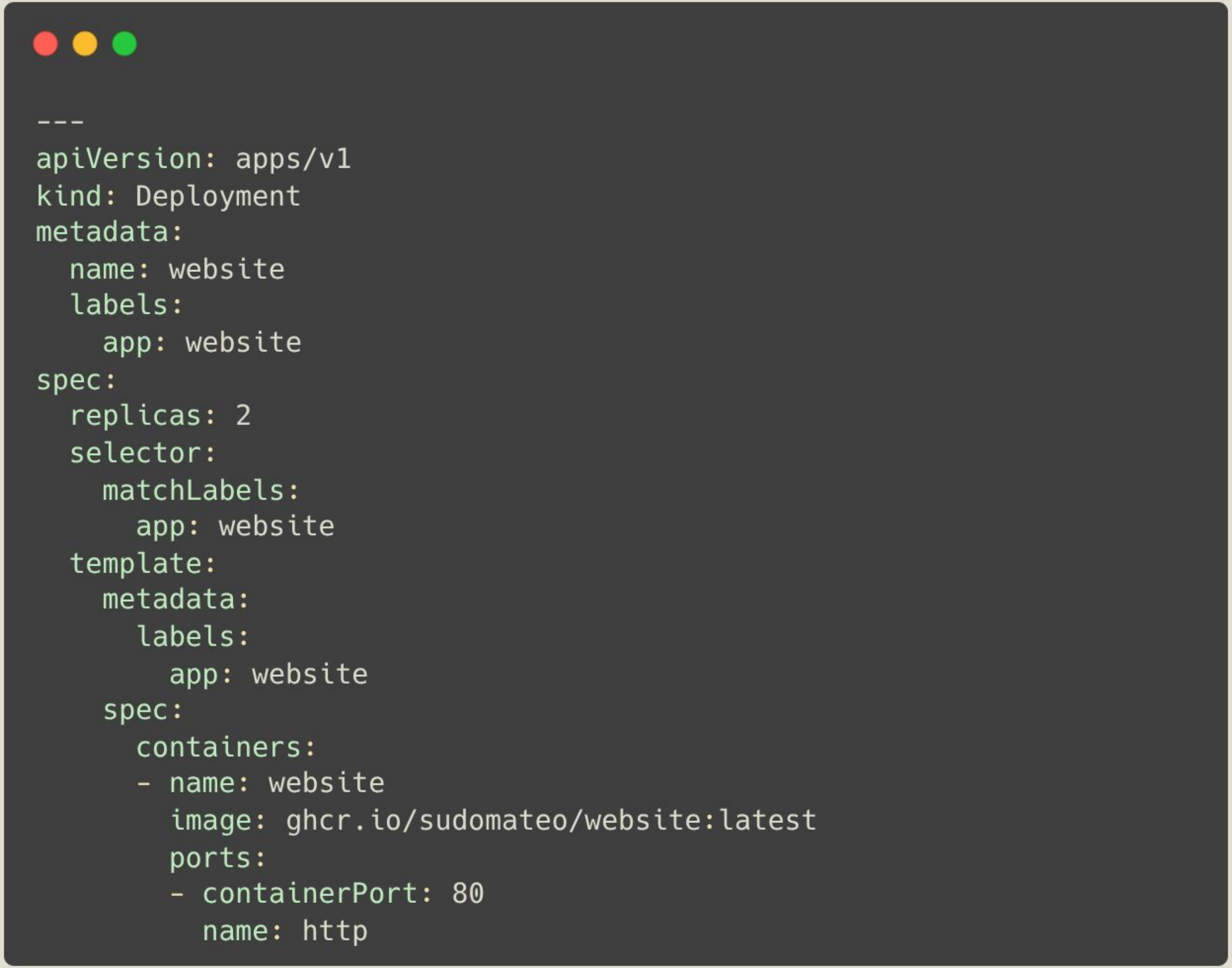
31 December 2024 · Updated: 2 January 2025 · 11 mins



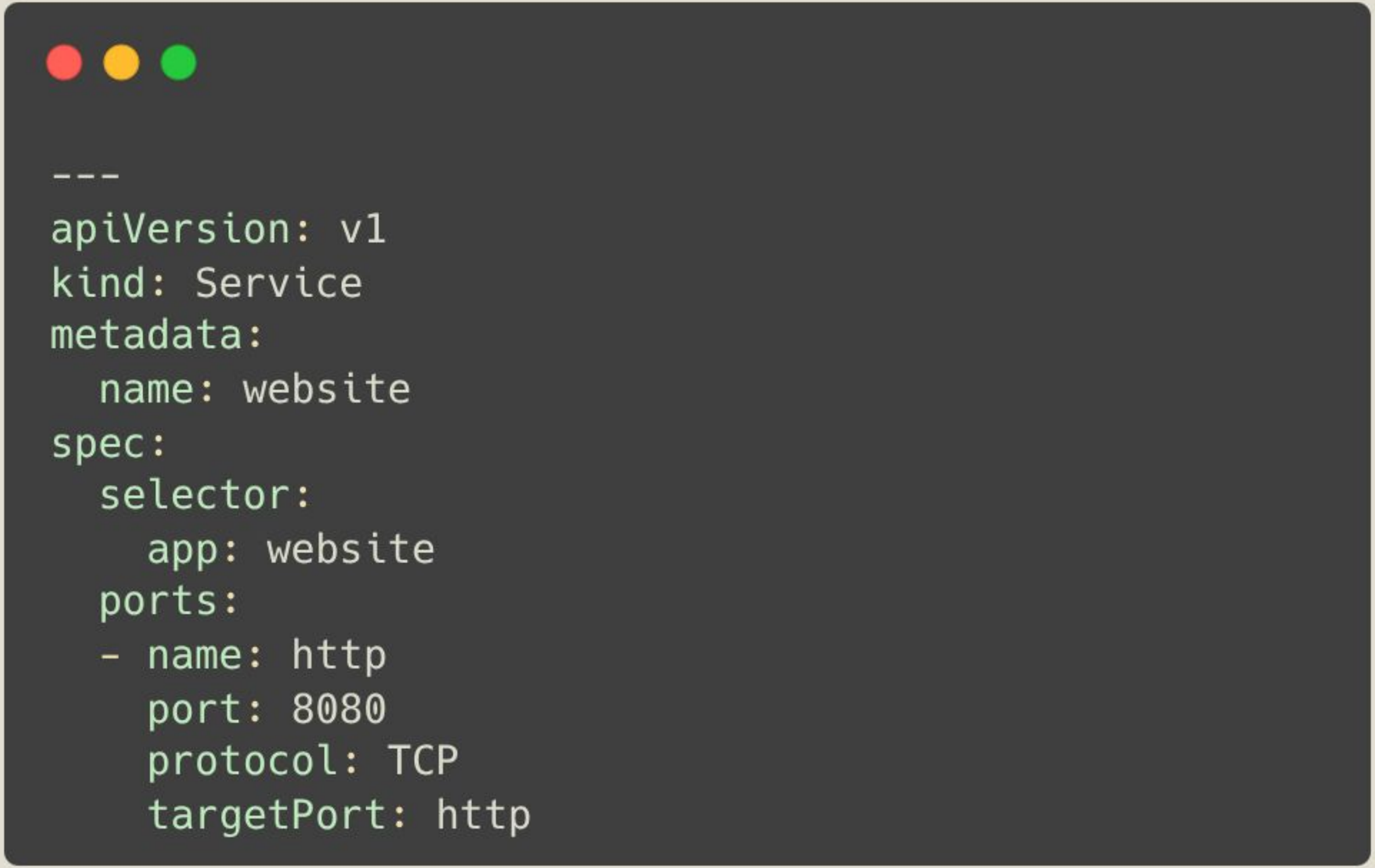
```
FROM golang:1.23.3

# Install curl.
RUN apt-get update && \
    apt-get install -y --no-install-recommends ca-certificates curl && \
    apt-get clean && \
    rm -rf /var/lib/apt/lists/*

# Install Hugo.
ENV HUGO=0.140.0
RUN curl -L -o /tmp/hugo.tar.gz \
    https://github.com/gohugoio/hugo/releases/download/v${HUGO}/hugo_extended_${HUGO}_linux-amd64.tar.gz && \
    tar -xvf /tmp/hugo.tar.gz -C /usr/local/bin hugo && \
    rm -rf /tmp/hugo.tar.gz

# Build the Hugo site.
WORKDIR /app
COPY . .
RUN hugo --destination public
```



```
FROM golang:1.24.0 AS builder
ARG TARGETARCH

RUN apt-get update && \
    apt-get install -y --no-install-recommends ca-certificates curl && \
    apt-get clean && \
    rm -rf /var/lib/apt/lists/*

ENV HUGO=0.144.2
RUN curl -L -o /tmp/hugo.tar.gz \
    https://github.com/gohugoio/hugo/releases/download/v${HUGO}/hugo_extended_${HUGO}_linux-${TARGETARCH}.tar.gz && \
    tar -xvf /tmp/hugo.tar.gz -C /usr/local/bin hugo && \
    rm -rf /tmp/hugo.tar.gz

WORKDIR /app
COPY . .
RUN hugo --destination public

FROM nginx:latest
COPY --from=builder /app/public /usr/share/nginx/html
```



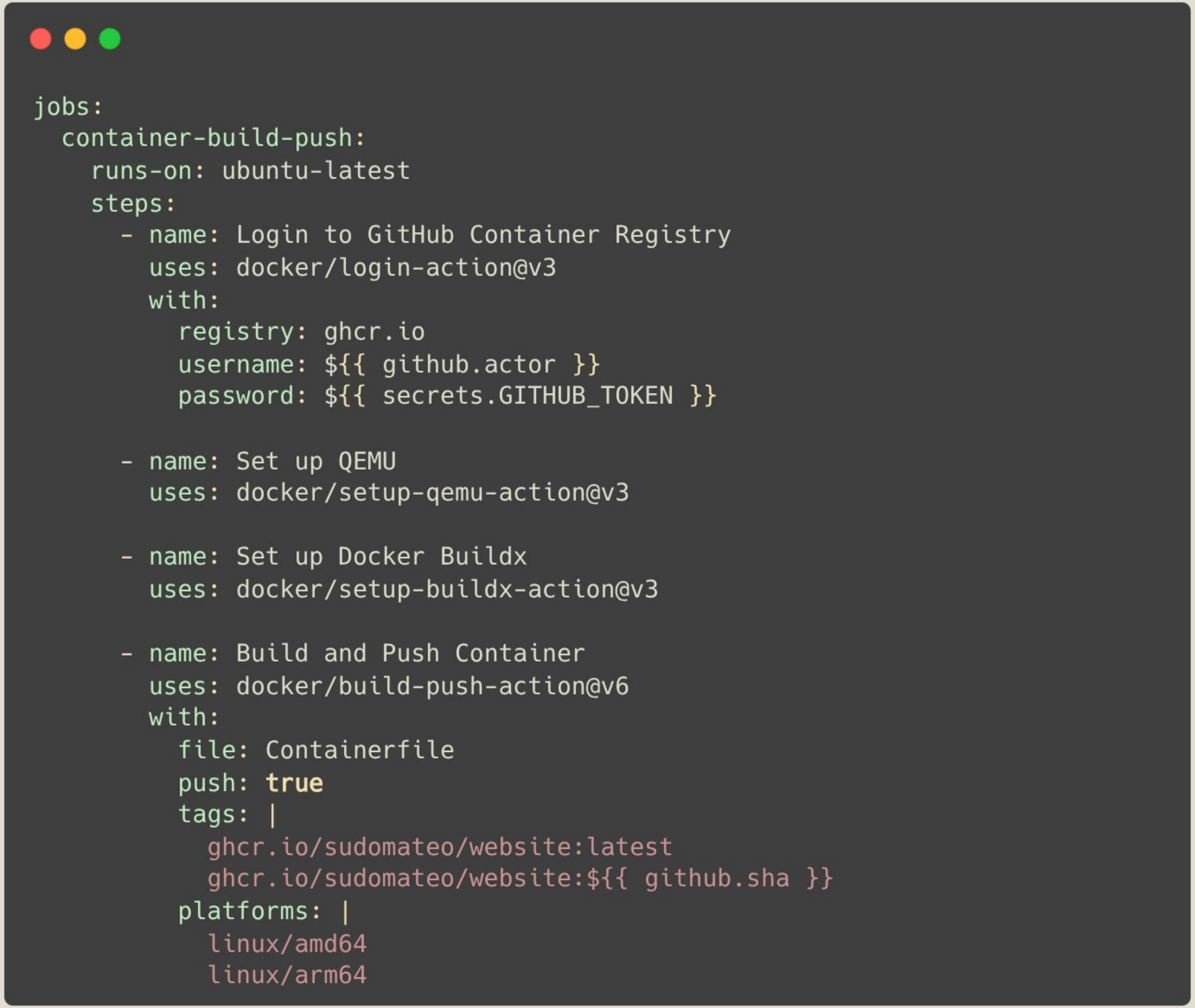
```yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: website
  labels:
    app: website
spec:
  replicas: 2
  selector:
    matchLabels:
      app: website
  template:
    metadata:
      labels:
        app: website
    spec:
      containers:
      - name: website
        image: ghcr.io/sudomateo/website:latest
        ports:
        - containerPort: 80
          name: http
```



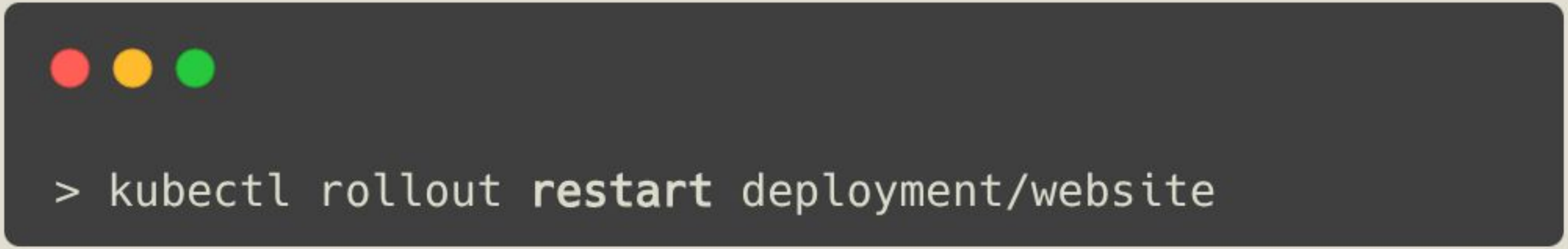
```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: website
spec:
  selector:
    app: website
  ports:
  - name: http
    port: 8080
    protocol: TCP
    targetPort: http
```

```
exec format error
```

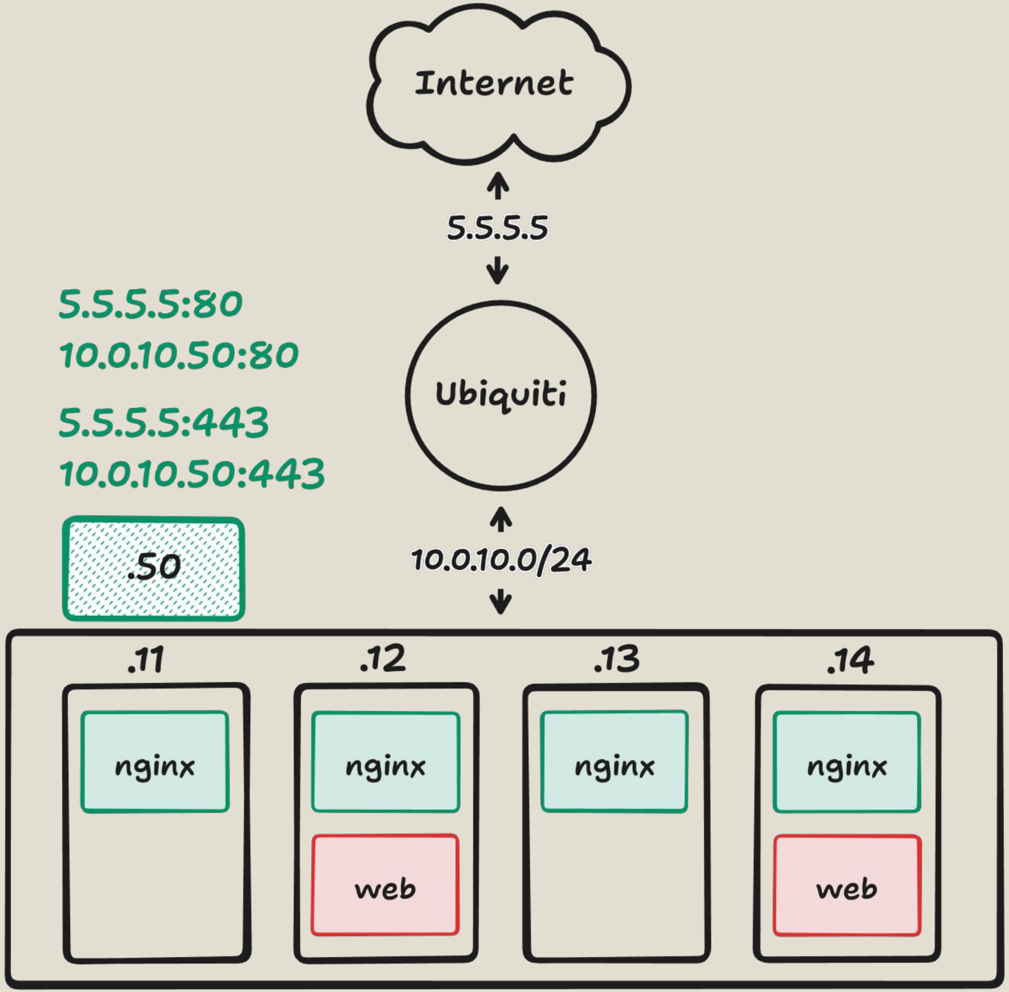
WE'RE NOT ON AMD64 ANYMORE



```yaml
jobs:
  container-build-push:
    runs-on: ubuntu-latest
    steps:
      - name: Login to GitHub Container Registry
        uses: docker/login-action@v3
        with:
          registry: ghcr.io
          username: ${{ github.actor }}
          password: ${{ secrets.GITHUB_TOKEN }}

      - name: Set up QEMU
        uses: docker/setup-qemu-action@v3

      - name: Set up Docker Buildx
        uses: docker/setup-buildx-action@v3

      - name: Build and Push Container
        uses: docker/build-push-action@v6
        with:
          file: Containerfile
          push: true
          tags: |
            ghcr.io/sudomateo/website:latest
            ghcr.io/sudomateo/website:${{ github.sha }}
          platforms: |
            linux/amd64
            linux/arm64
```



```
> kubectl rollout restart deployment/website
```

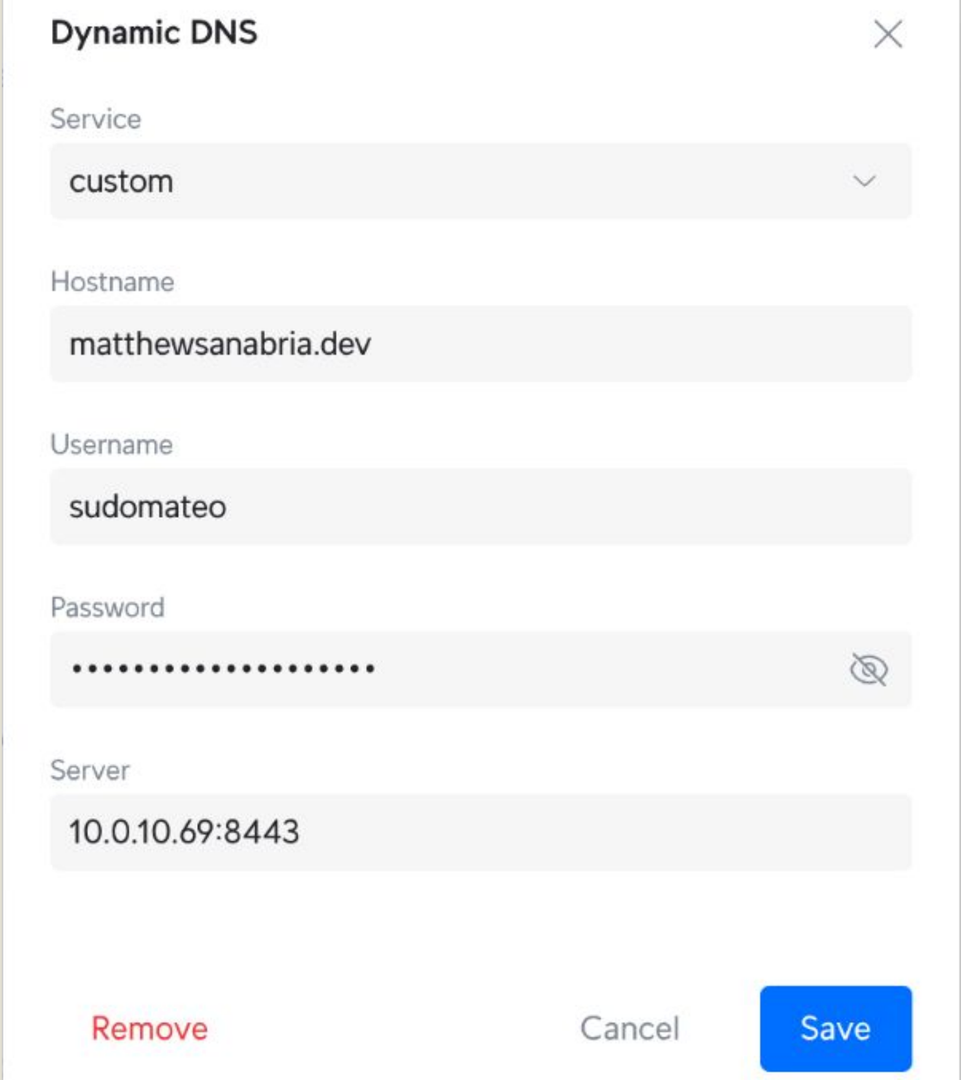

Internet
5.5.5.5
Ubiquiti
10.0.10.0/24
5.5.5.5:80
10.0.10.50:80
5.5.5.5:443
10.0.10.50:443
.50
.11
.12
.13
.14
nginx
nginx
web
nginx
nginx
web

DYNAMIC DNS

WHAT IF MY PUBLIC IP CHANGES?

DYNAMIC DNS

- Supported by UniFi
- Open questions:
  - What's the protocol?
  - Can this stay internal?
  - What's the program?



Dynamic DNS

Service: custom

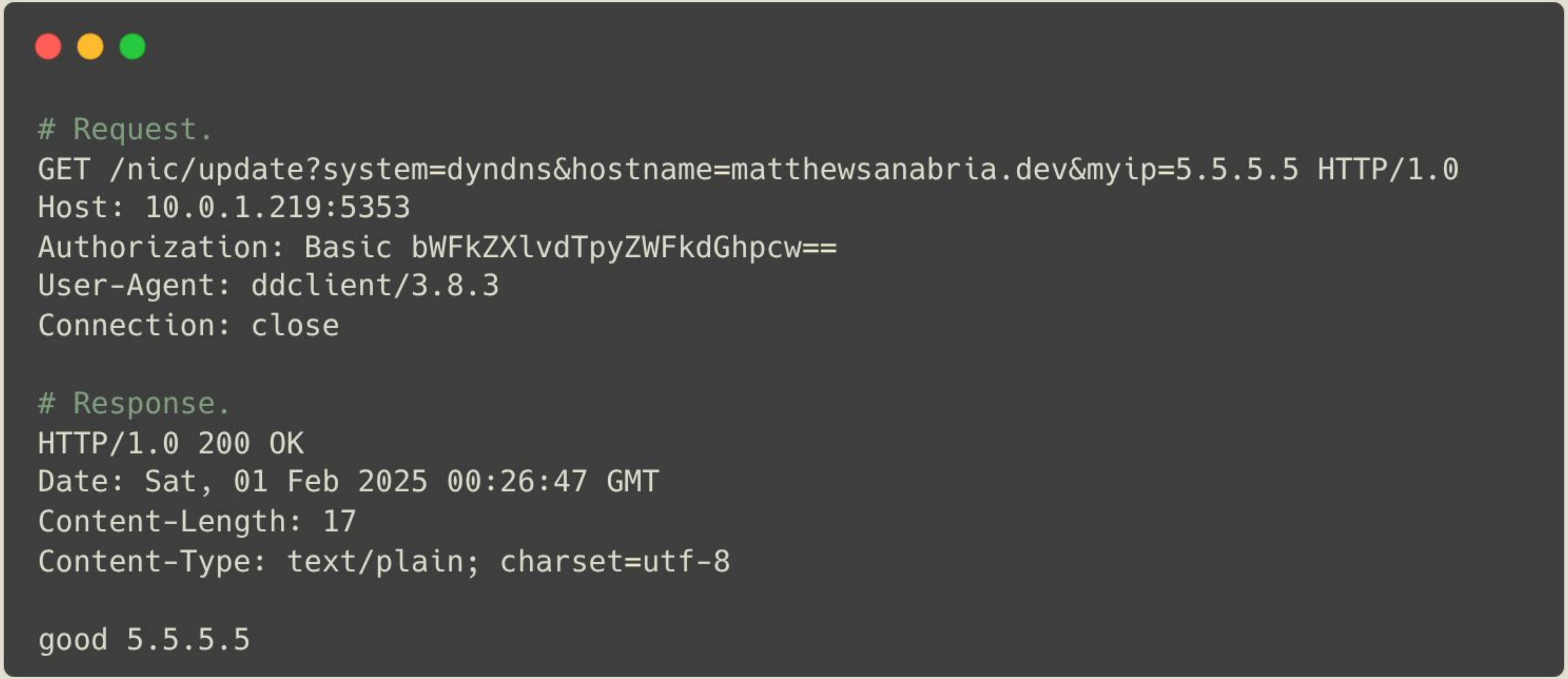
Hostname: matthewsanabria.dev

Username: sudomateo

Password: ••••••••••••••••••••

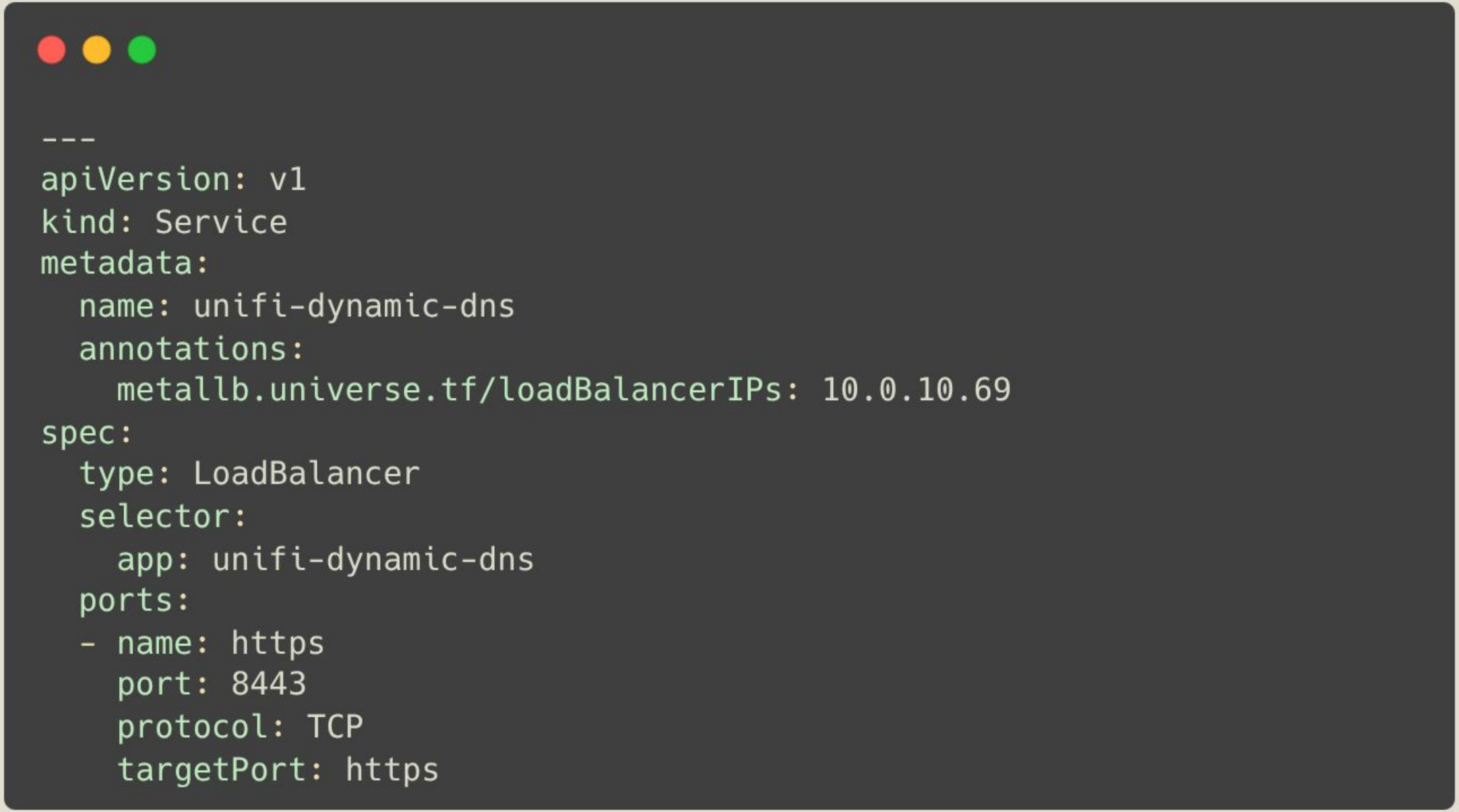
Server: 10.0.10.69:8443

WHAT'S THE PROTOCOL?



```
# Request.
GET /nic/update?system=dyndns&hostname=matthewsanabria.dev&myip=5.5.5.5 HTTP/1.0
Host: 10.0.1.219:5353
Authorization: Basic bWFkZXlvdTpyZWFkdGhpcw==
User-Agent: ddclient/3.8.3
Connection: close

# Response.
HTTP/1.0 200 OK
Date: Sat, 01 Feb 2025 00:26:47 GMT
Content-Length: 17
Content-Type: text/plain; charset=utf-8

good 5.5.5.5
```

CAN THIS STAY INTERNAL?



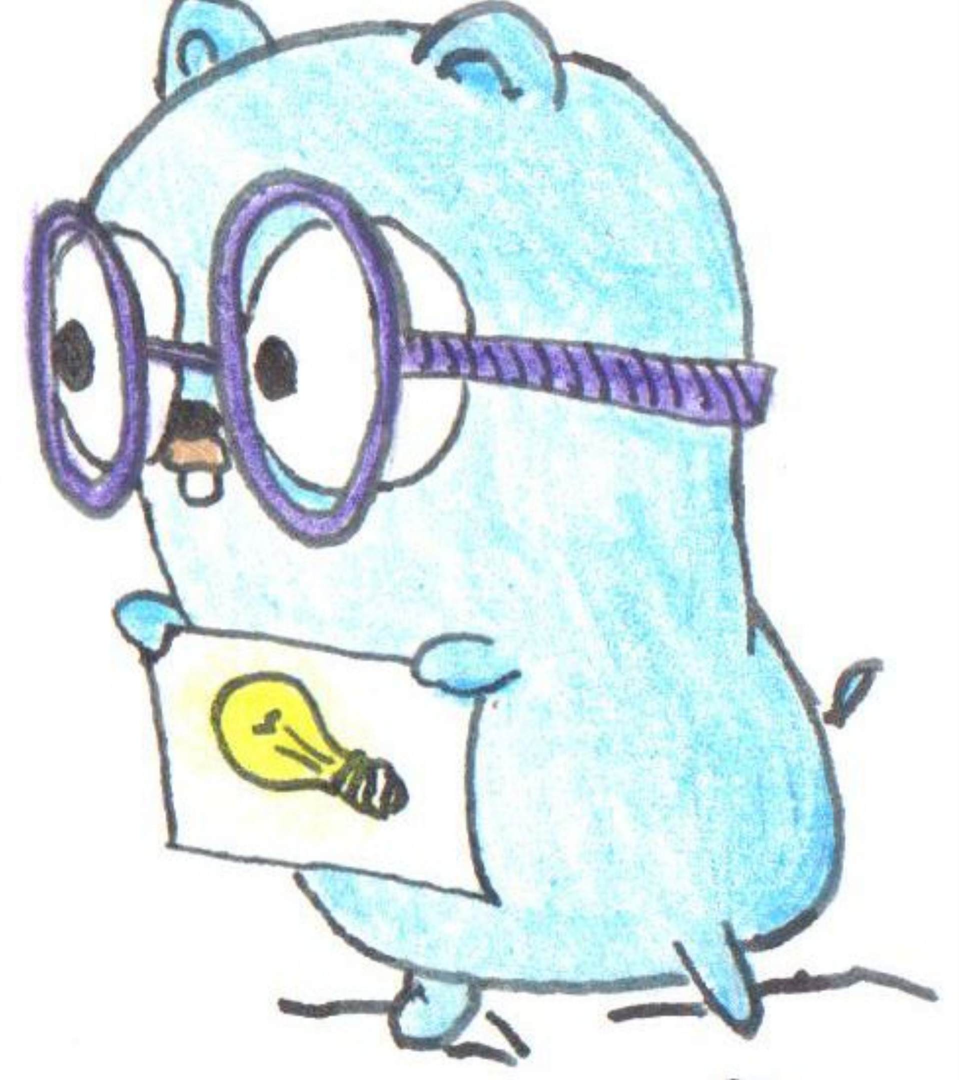
```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: unifi-dynamic-dns
  annotations:
    metallb.universe.tf/loadBalancerIPs: 10.0.10.69
spec:
  type: LoadBalancer
  selector:
    app: unifi-dynamic-dns
  ports:
  - name: https
    port: 8443
    protocol: TCP
    targetPort: https
```

WHAT'S THE PROGRAM?

UNIFI-DYNAMIC-DNS

sudomateo/unifi-dynamic-dns

1. Receives dynamic DNS request
2. Updates Terraform Cloud variable
3. Triggers Terraform Cloud run
4. Waits

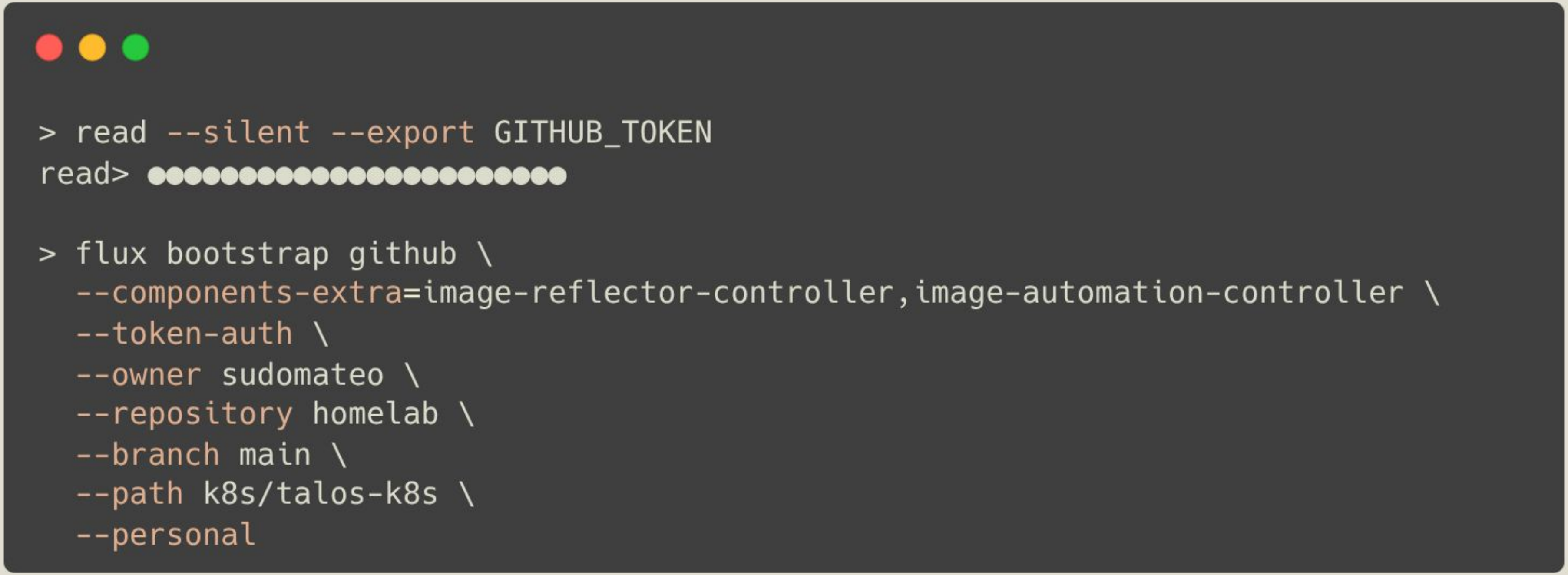


CI/CD

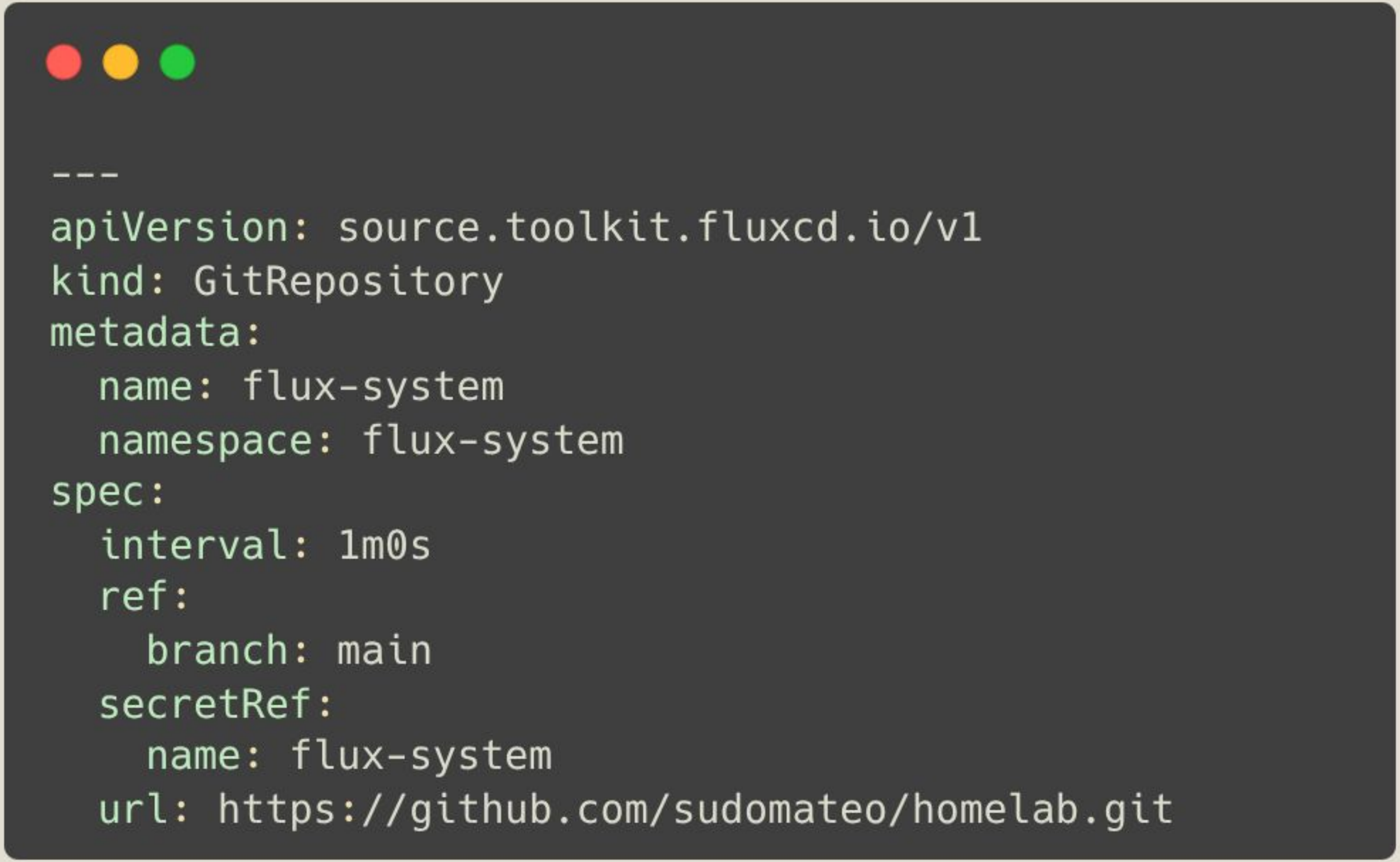
HOW CAN I AUTOMATICALLY DEPLOY THING?

FLUX CD

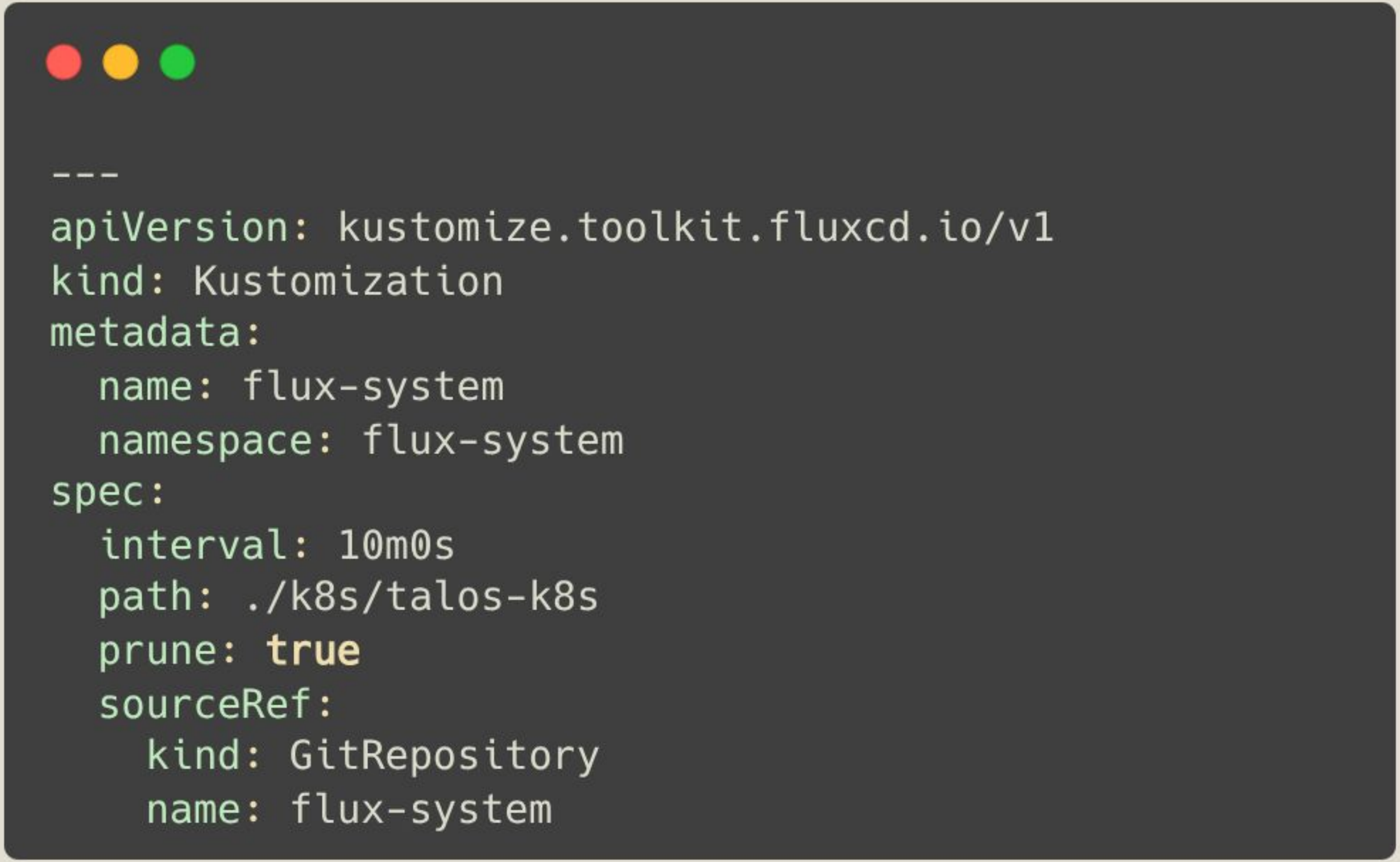
- "GitOps" for Kubernetes
- Minimal
- CNCF project



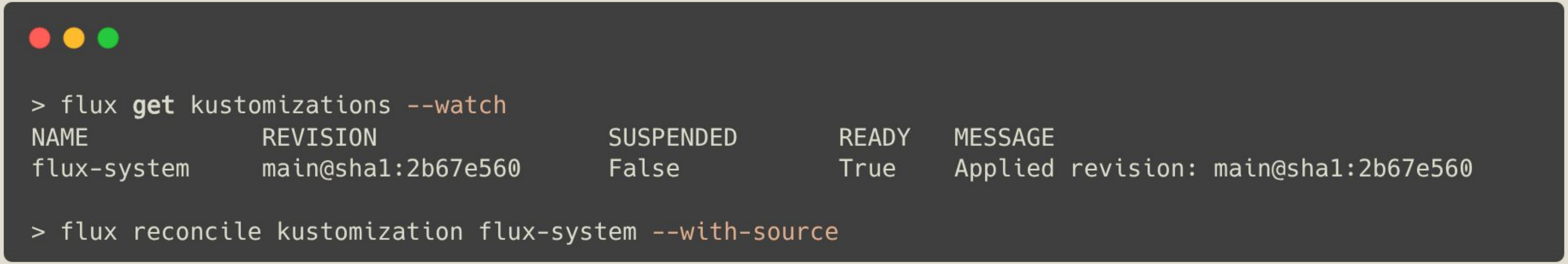
```
> read --silent --export GITHUB_TOKEN
read> ●●●●●●●●●●●●●●●●●●●●●●●

> flux bootstrap github \
  --components-extra=image-reflector-controller,image-automation-controller \
  --token-auth \
  --owner sudomateo \
  --repository homelab \
  --branch main \
  --path k8s/talos-k8s \
  --personal
```



```
---
apiVersion: source.toolkit.fluxcd.io/v1
kind: GitRepository
metadata:
  name: flux-system
  namespace: flux-system
spec:
  interval: 1m0s
  ref:
    branch: main
  secretRef:
    name: flux-system
  url: https://github.com/sudomateo/homelab.git
```

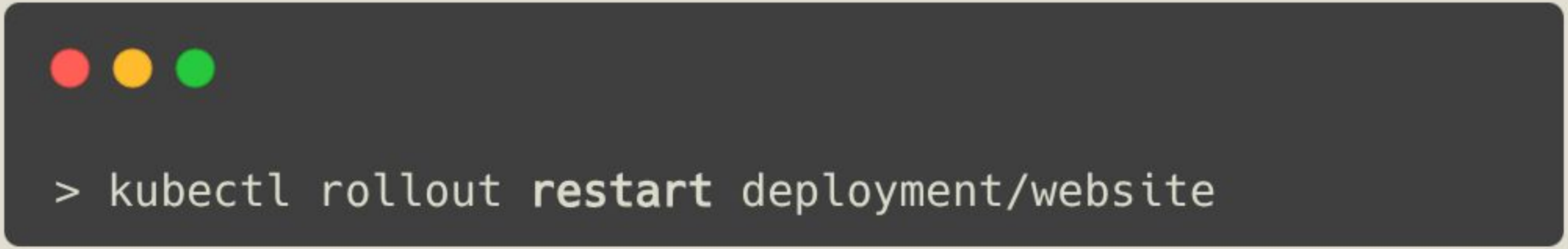


```yaml
---
apiVersion: kustomize.toolkit.fluxcd.io/v1
kind: Kustomization
metadata:
  name: flux-system
  namespace: flux-system
spec:
  interval: 10m0s
  path: ./k8s/talos-k8s
  prune: true
  sourceRef:
    kind: GitRepository
    name: flux-system
```

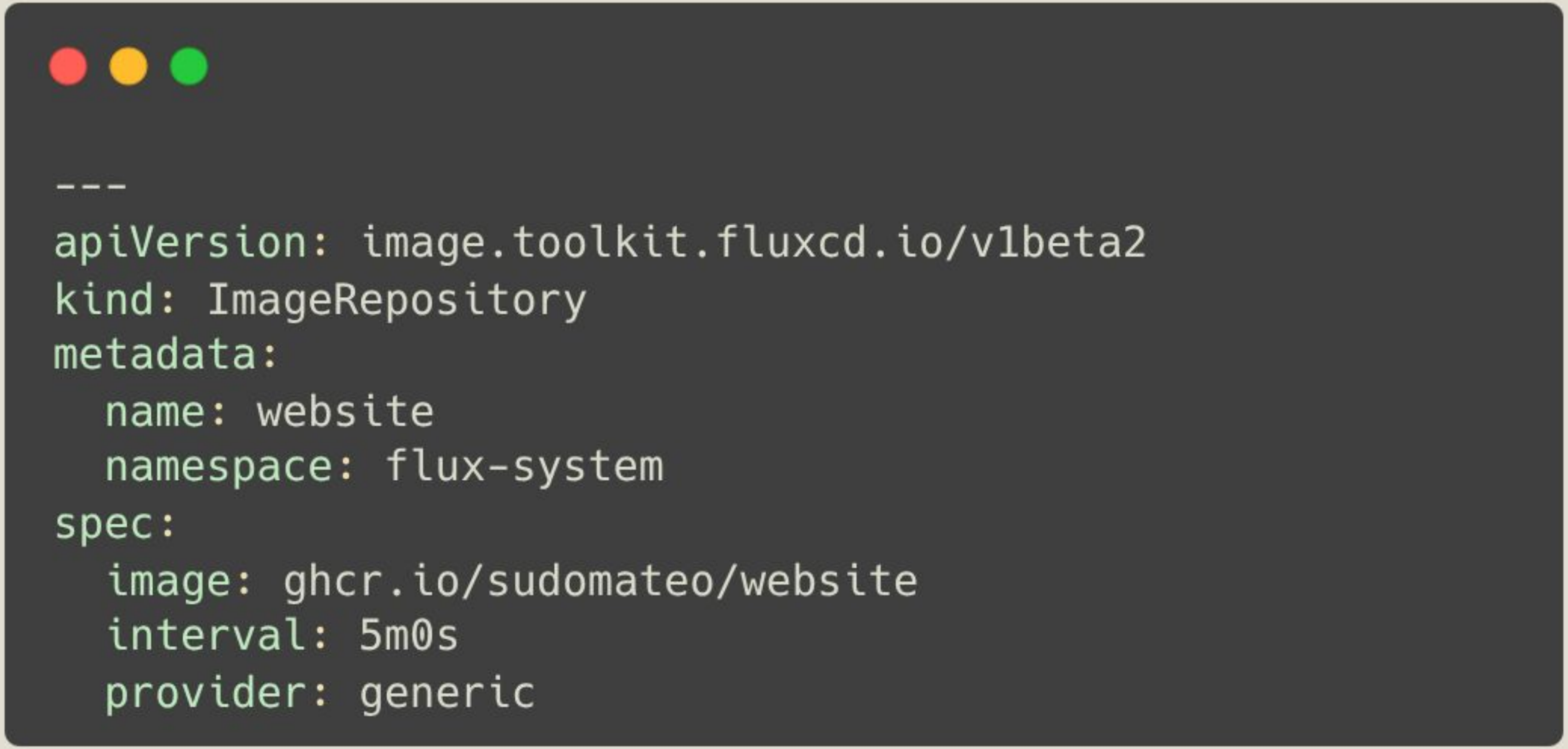


```
> flux get kustomizations --watch
NAME            REVISION                SUSPENDED       READY   MESSAGE
flux-system     main@sha1:2b67e560      False           True    Applied revision: main@sha1:2b67e560

> flux reconcile kustomization flux-system --with-source
```

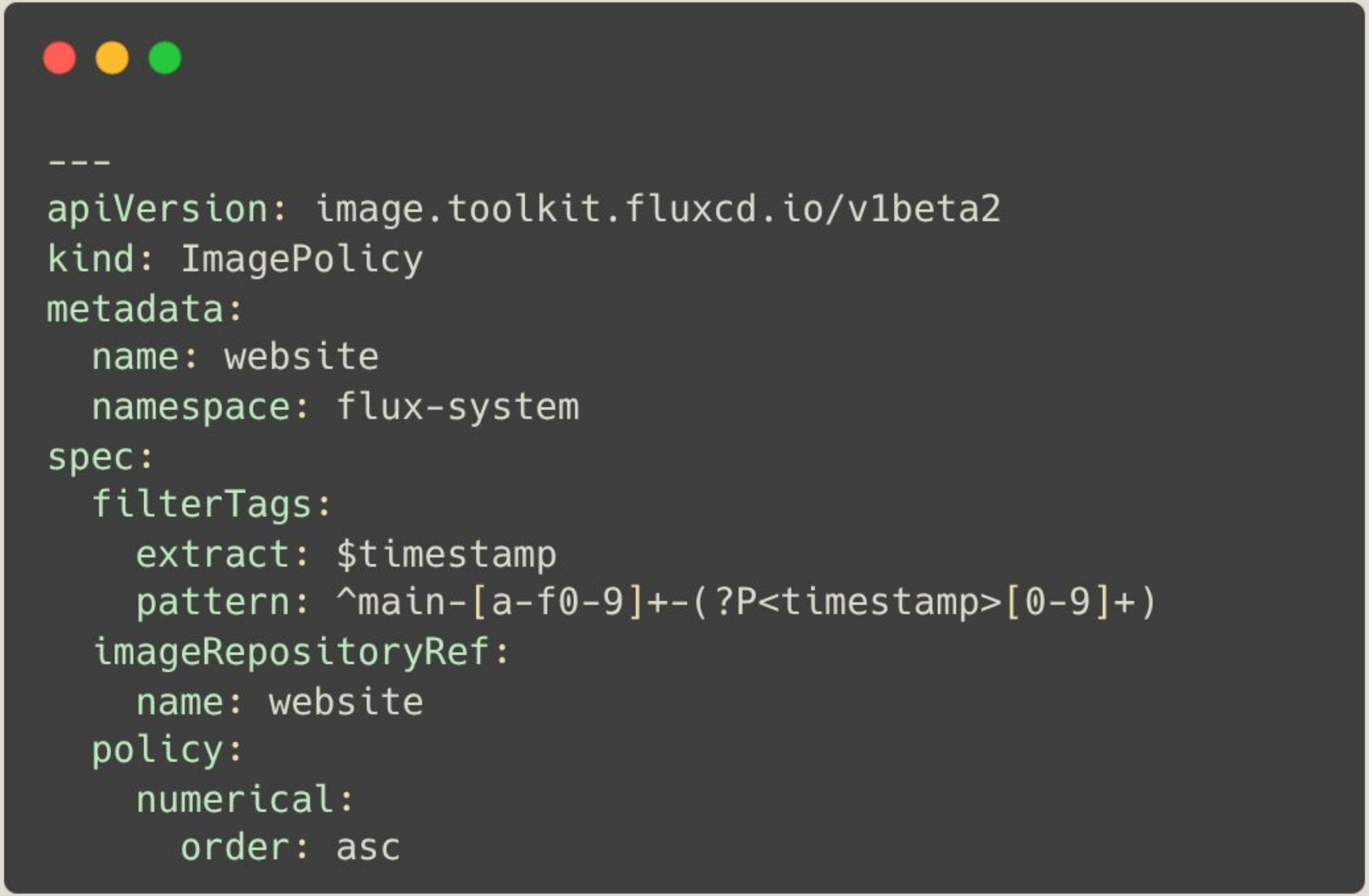
WHAT ABOUT IMAGE UPDATES?



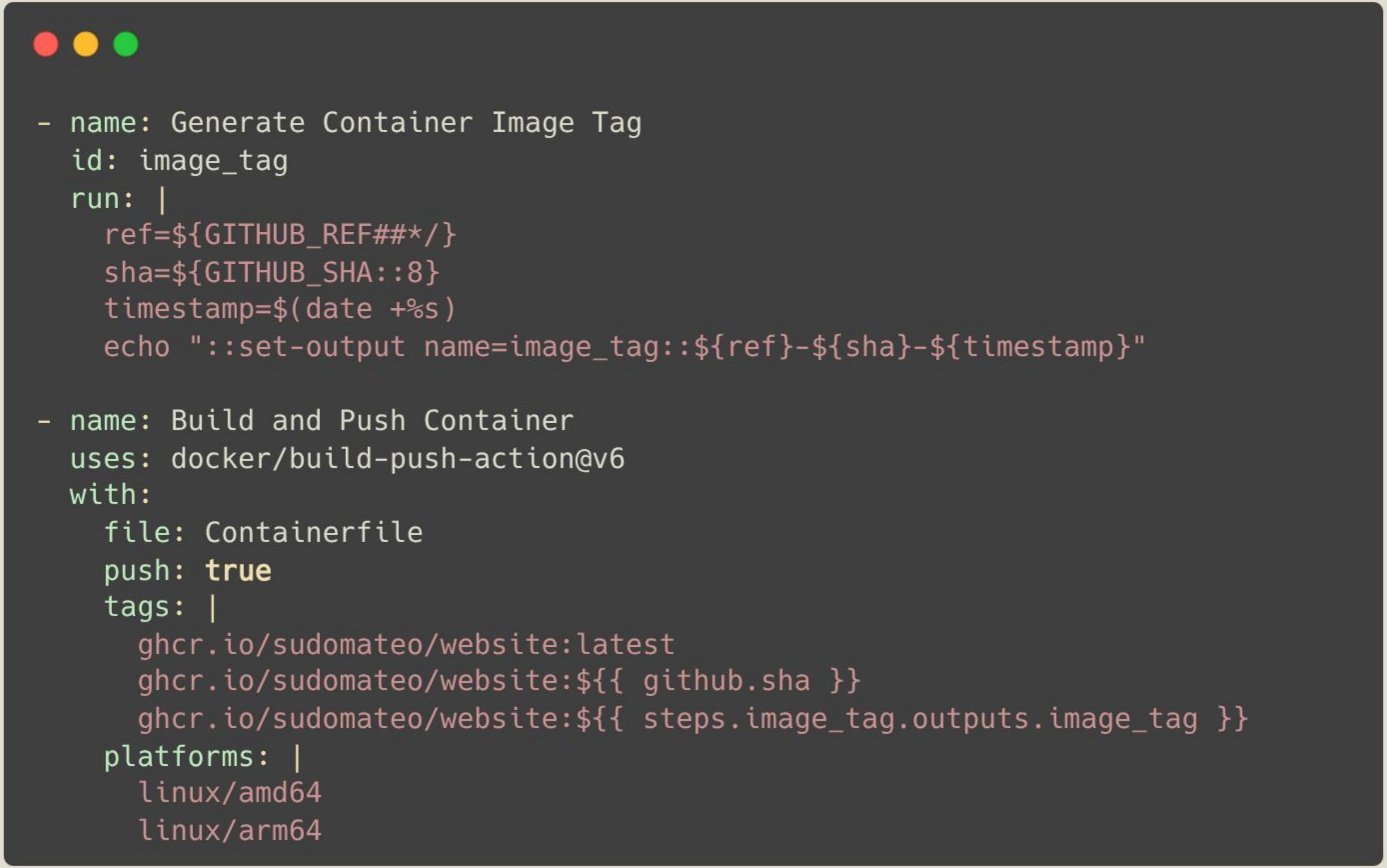
```
> kubectl rollout restart deployment/website
```



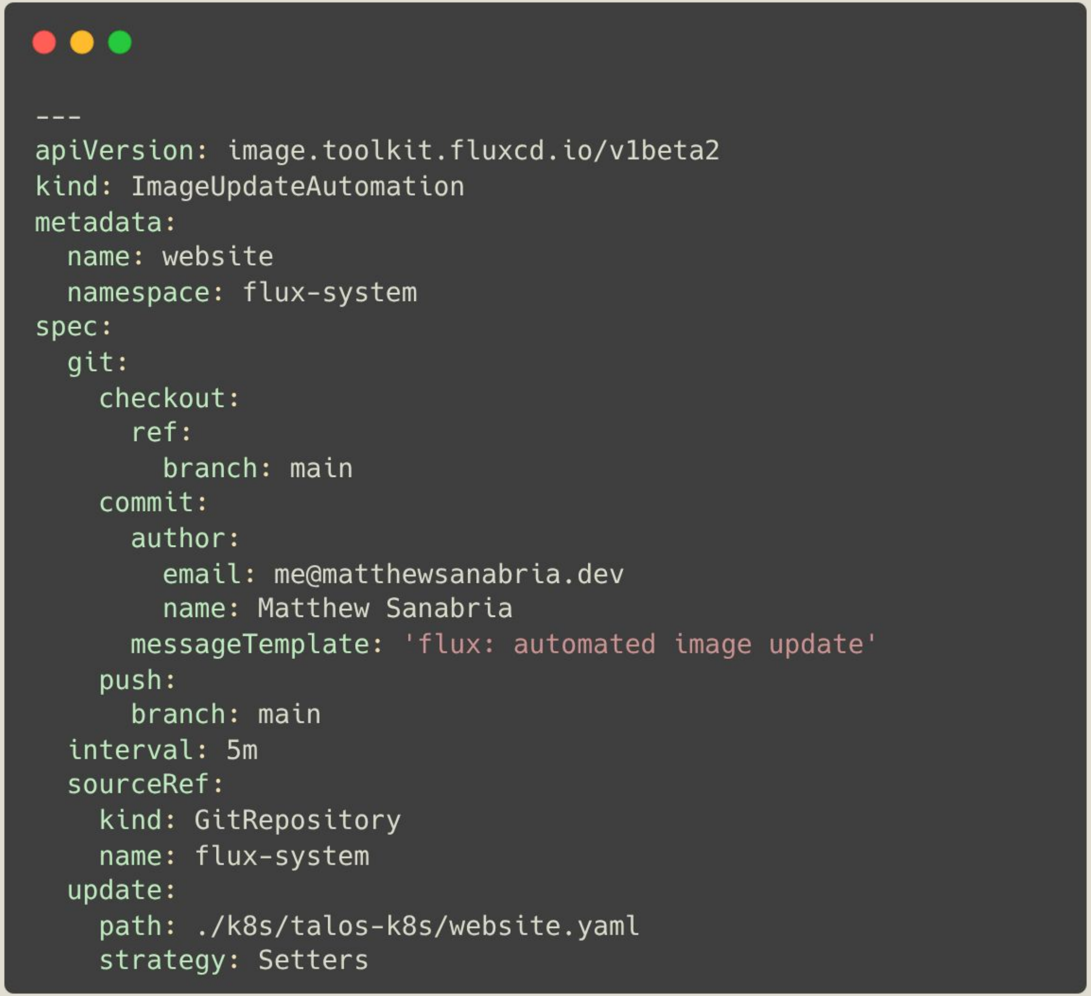
```yaml
---
apiVersion: image.toolkit.fluxcd.io/v1beta2
kind: ImageRepository
metadata:
  name: website
  namespace: flux-system
spec:
  image: ghcr.io/sudomateo/website
  interval: 5m0s
  provider: generic
```



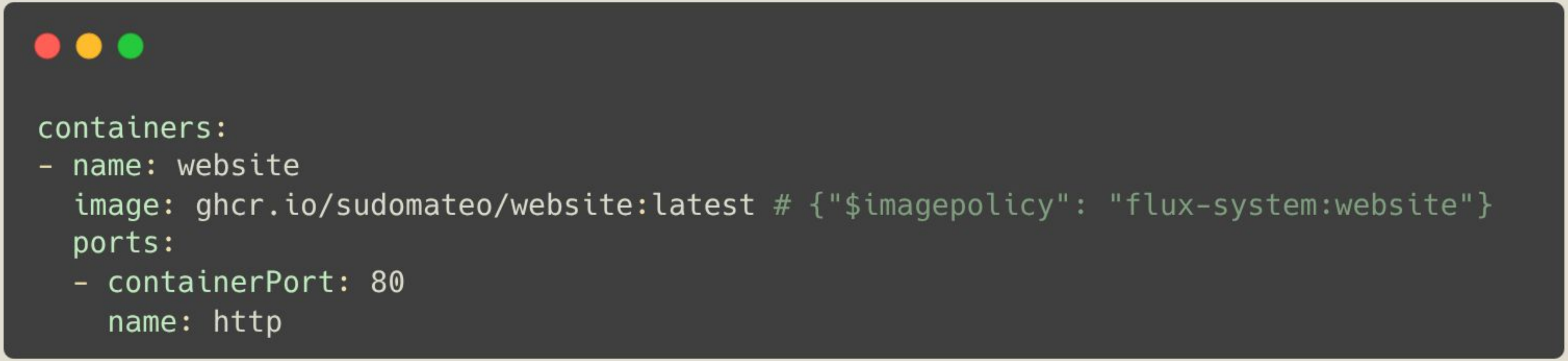
```yaml
---
apiVersion: image.toolkit.fluxcd.io/v1beta2
kind: ImagePolicy
metadata:
  name: website
  namespace: flux-system
spec:
  filterTags:
    extract: $timestamp
    pattern: ^main-[a-f0-9]+-(?P<timestamp>[0-9]+)
  imageRepositoryRef:
    name: website
  policy:
    numerical:
      order: asc
```



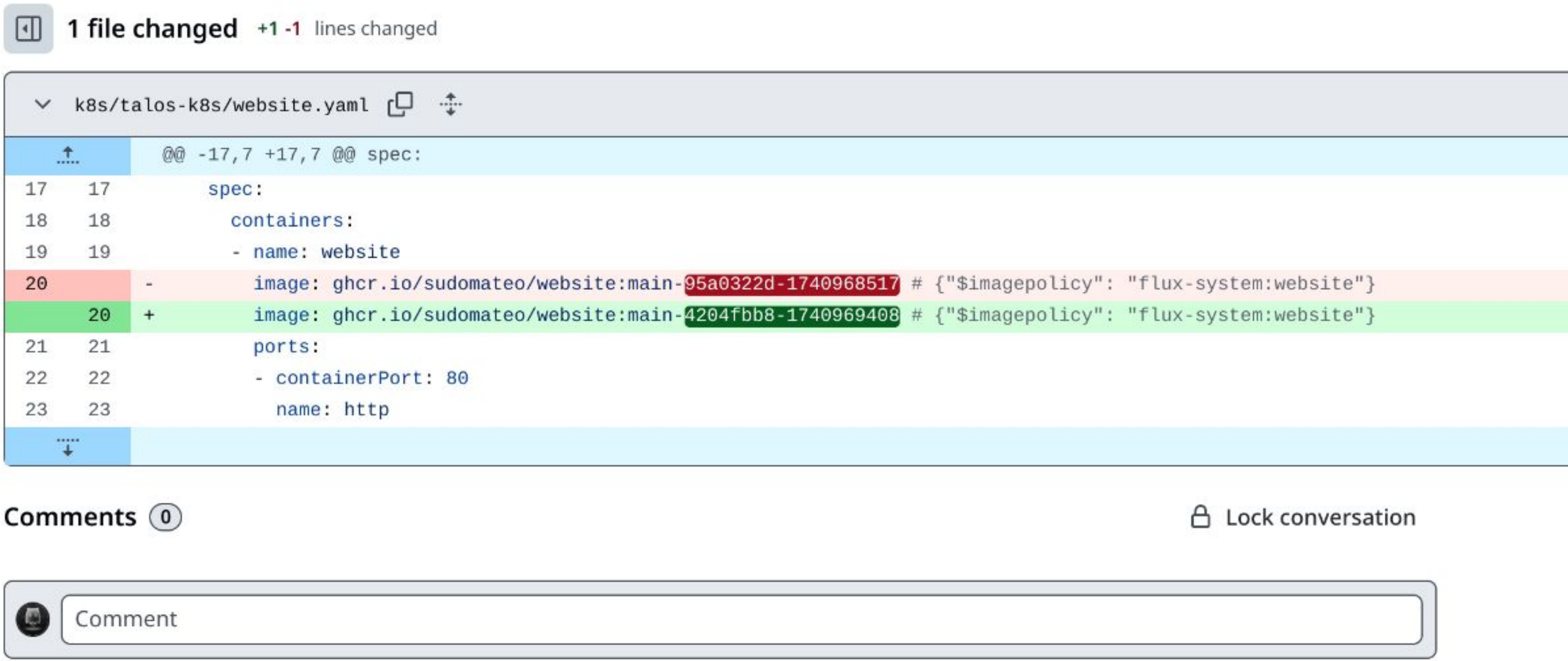
```yaml
- name: Generate Container Image Tag
  id: image_tag
  run: |
    ref=${GITHUB_REF##*/}
    sha=${GITHUB_SHA::8}
    timestamp=$(date +%s)
    echo "::set-output name=image_tag::${ref}-${sha}-${timestamp}"

- name: Build and Push Container
  uses: docker/build-push-action@v6
  with:
    file: Containerfile
    push: true
    tags: |
      ghcr.io/sudomateo/website:latest
      ghcr.io/sudomateo/website:${{ github.sha }}
      ghcr.io/sudomateo/website:${{ steps.image_tag.outputs.image_tag }}
    platforms: |
      linux/amd64
      linux/arm64
```



```yaml
---
apiVersion: image.toolkit.fluxcd.io/v1beta2
kind: ImageUpdateAutomation
metadata:
  name: website
  namespace: flux-system
spec:
  git:
    checkout:
      ref:
        branch: main
    commit:
      author:
        email: me@matthewsanabria.dev
        name: Matthew Sanabria
      messageTemplate: 'flux: automated image update'
    push:
      branch: main
  interval: 5m
  sourceRef:
    kind: GitRepository
    name: flux-system
  update:
    path: ./k8s/talos-k8s/website.yaml
    strategy: Setters
```



```yaml
containers:
- name: website
  image: ghcr.io/sudomateo/website:latest # {"$imagepolicy": "flux-system:website"}
  ports:
  - containerPort: 80
    name: http
```



1 file changed +1 -1 lines changed

k8s/talos-k8s/website.yaml

```diff
@@ -17,7 +17,7 @@ spec:
       spec:
         containers:
         - name: website
-          image: ghcr.io/sudomateo/website:main-95a0322d-1740968517 # {"$imagepolicy": "flux-system:website"}
+          image: ghcr.io/sudomateo/website:main-4204fbb8-1740969408 # {"$imagepolicy": "flux-system:website"}
           ports:
           - containerPort: 80
             name: http
```

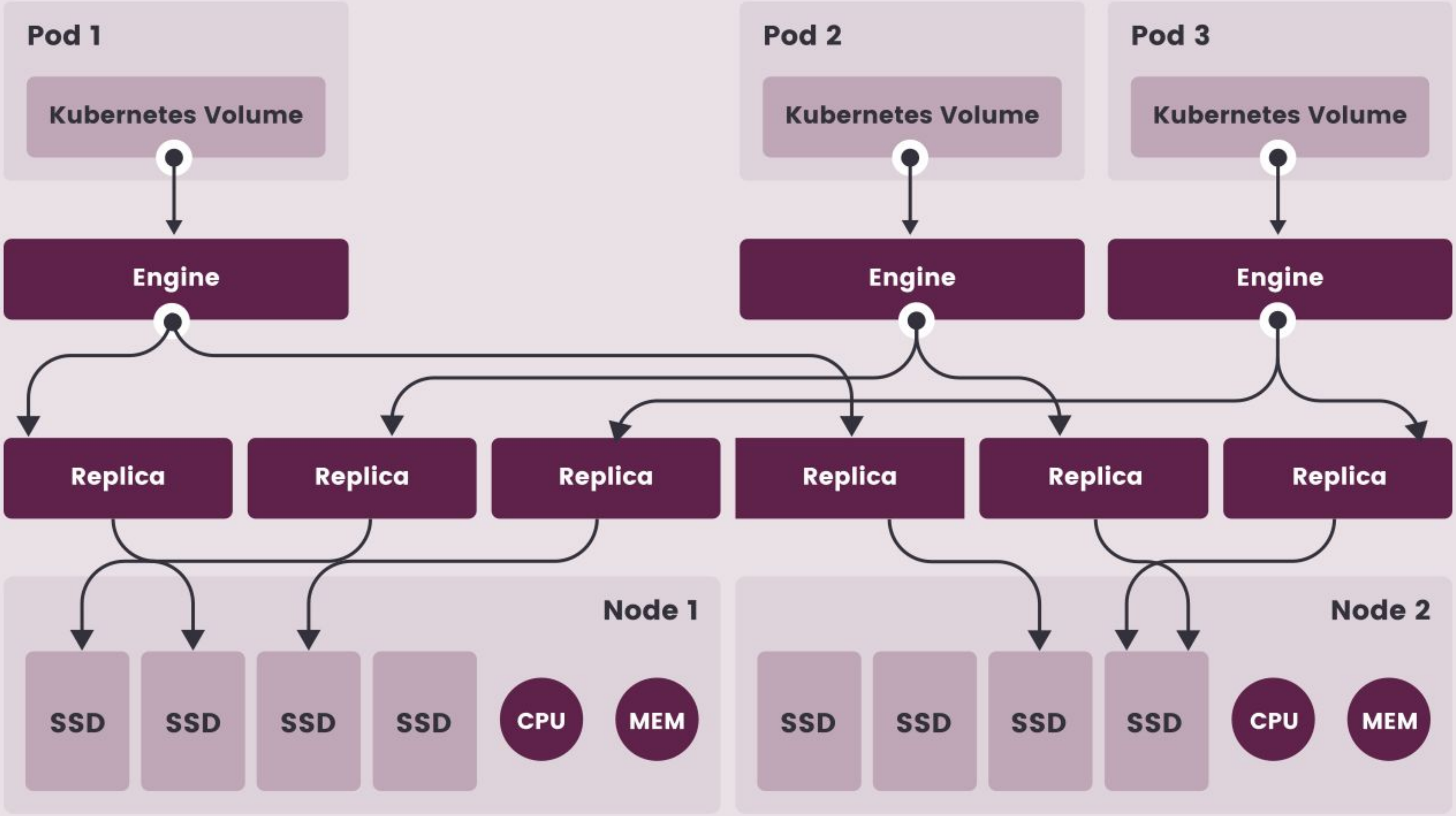
YAY, GITOPS!

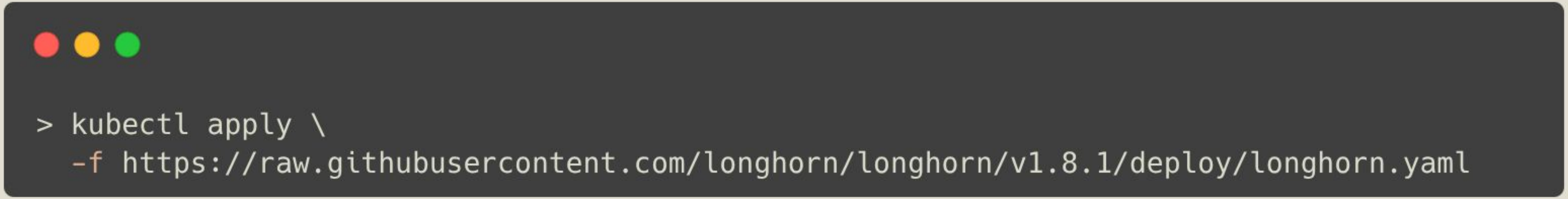
PERSISTENT STORAGE

WHAT ABOUT PERSISTENT VOLUMES?

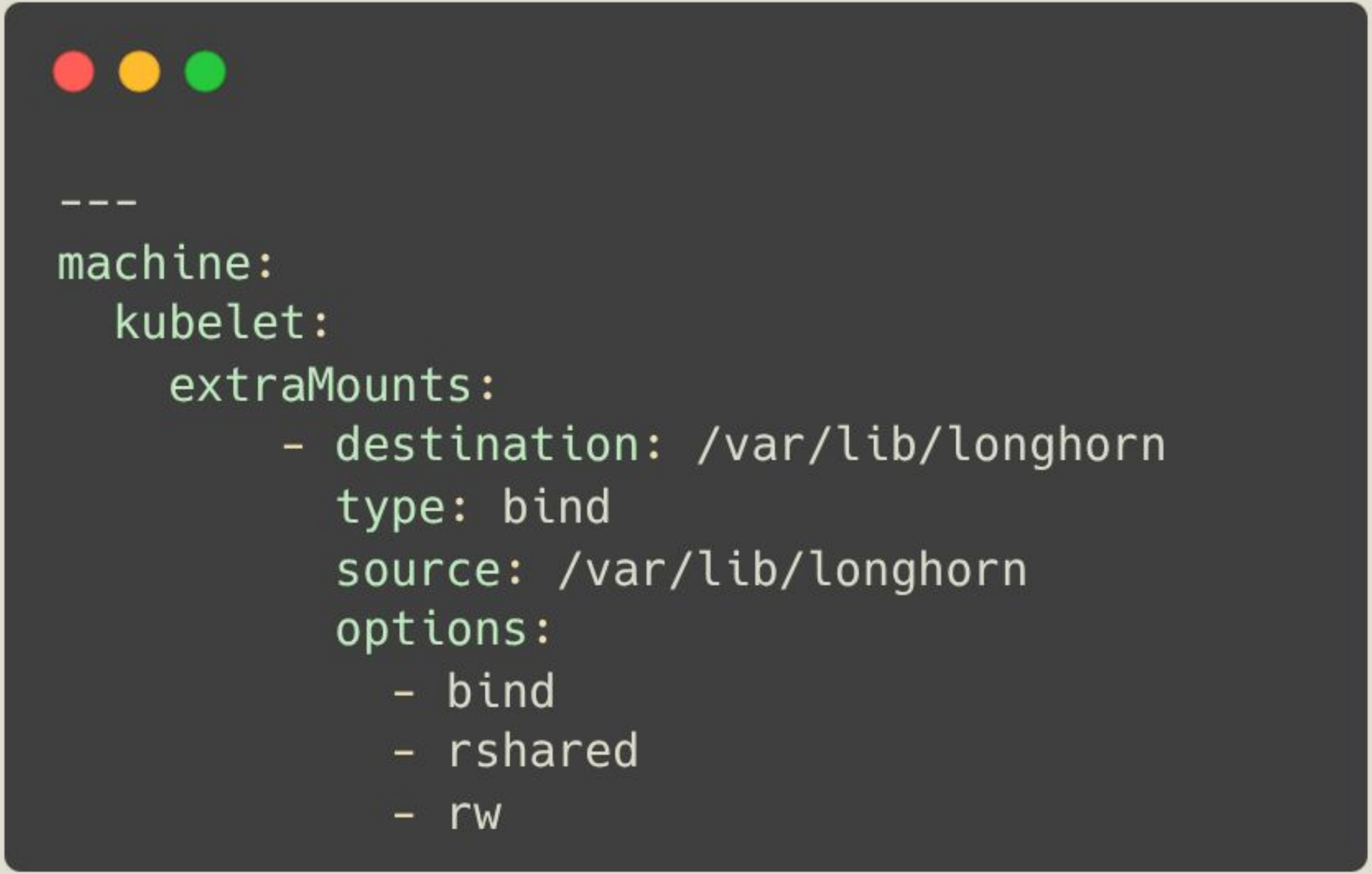
LONGHORN

- Distributed block storage
- Open source
- Part of CNCF

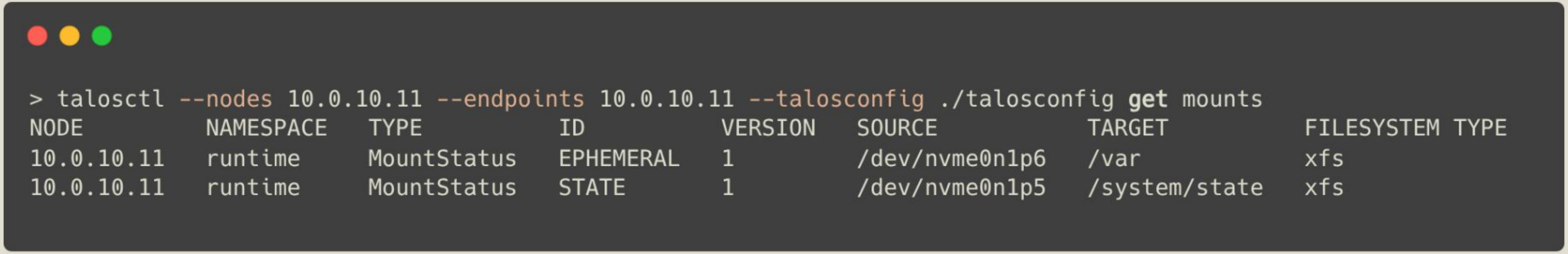

Pod 1
Kubernetes Volume
Pod 2
Kubernetes Volume
Pod 3
Kubernetes Volume
Engine
Engine
Engine
Replica
Replica
Replica
Replica
Replica
Replica
Node 1
SSD
SSD
SSD
SSD
CPU
MEM
Node 2
SSD
SSD
SSD
SSD
CPU
MEM



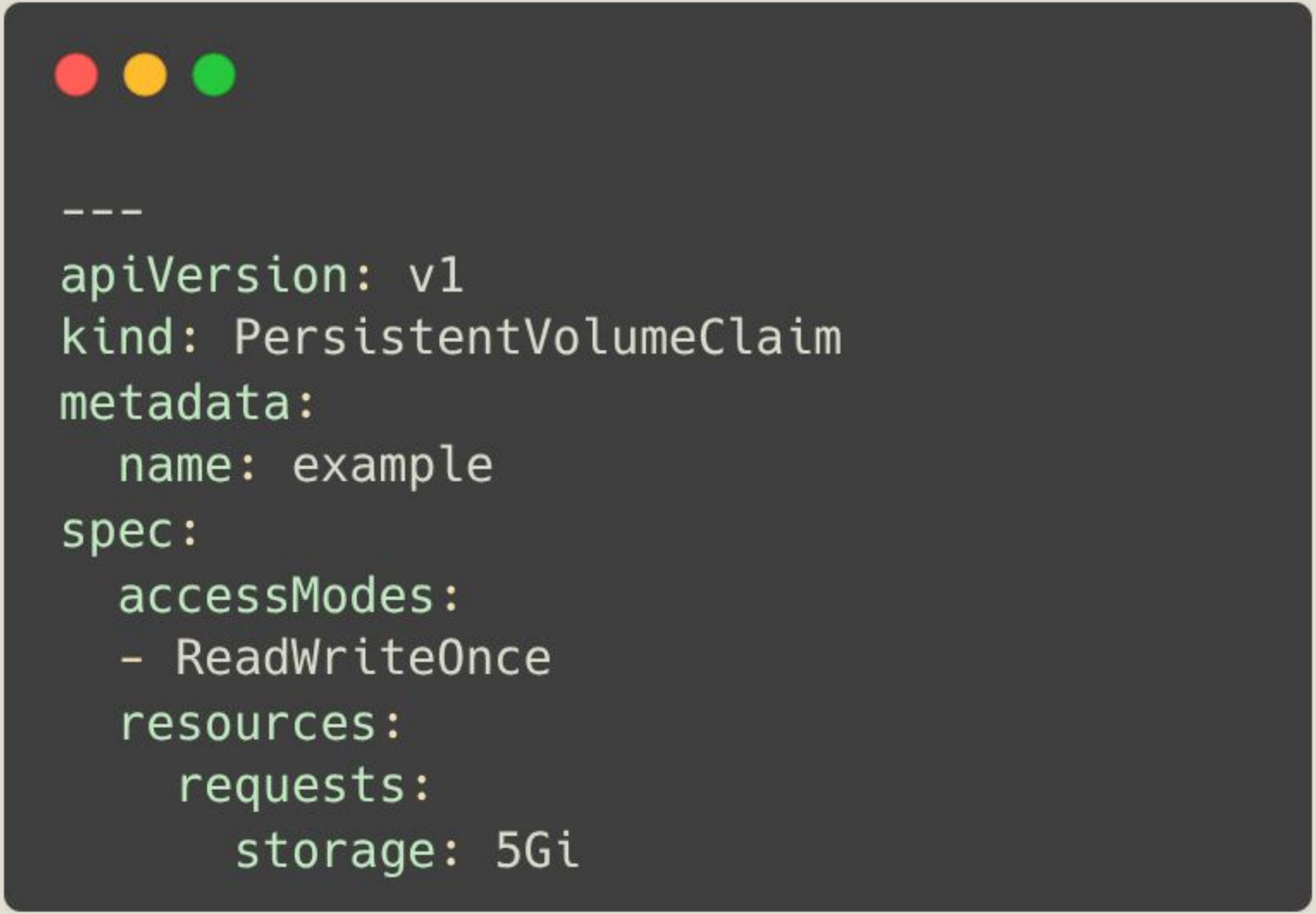
```
> kubectl apply \
  -f https://raw.githubusercontent.com/longhorn/longhorn/v1.8.1/deploy/longhorn.yaml
```



```yaml
---
machine:
  kubelet:
    extraMounts:
        - destination: /var/lib/longhorn
          type: bind
          source: /var/lib/longhorn
          options:
            - bind
            - rshared
            - rw
```



```
> talosctl --nodes 10.0.10.11 --endpoints 10.0.10.11 --talosconfig ./talosconfig get mounts
NODE         NAMESPACE   TYPE          ID          VERSION   SOURCE           TARGET          FILESYSTEM TYPE
10.0.10.11   runtime     MountStatus   EPHEMERAL   1         /dev/nvme0n1p6   /var            xfs
10.0.10.11   runtime     MountStatus   STATE       1         /dev/nvme0n1p5   /system/state   xfs
```



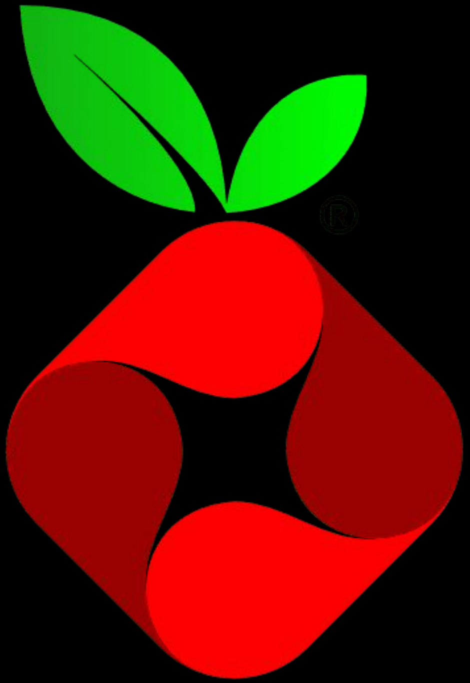
```yaml
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: example
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 5Gi
```

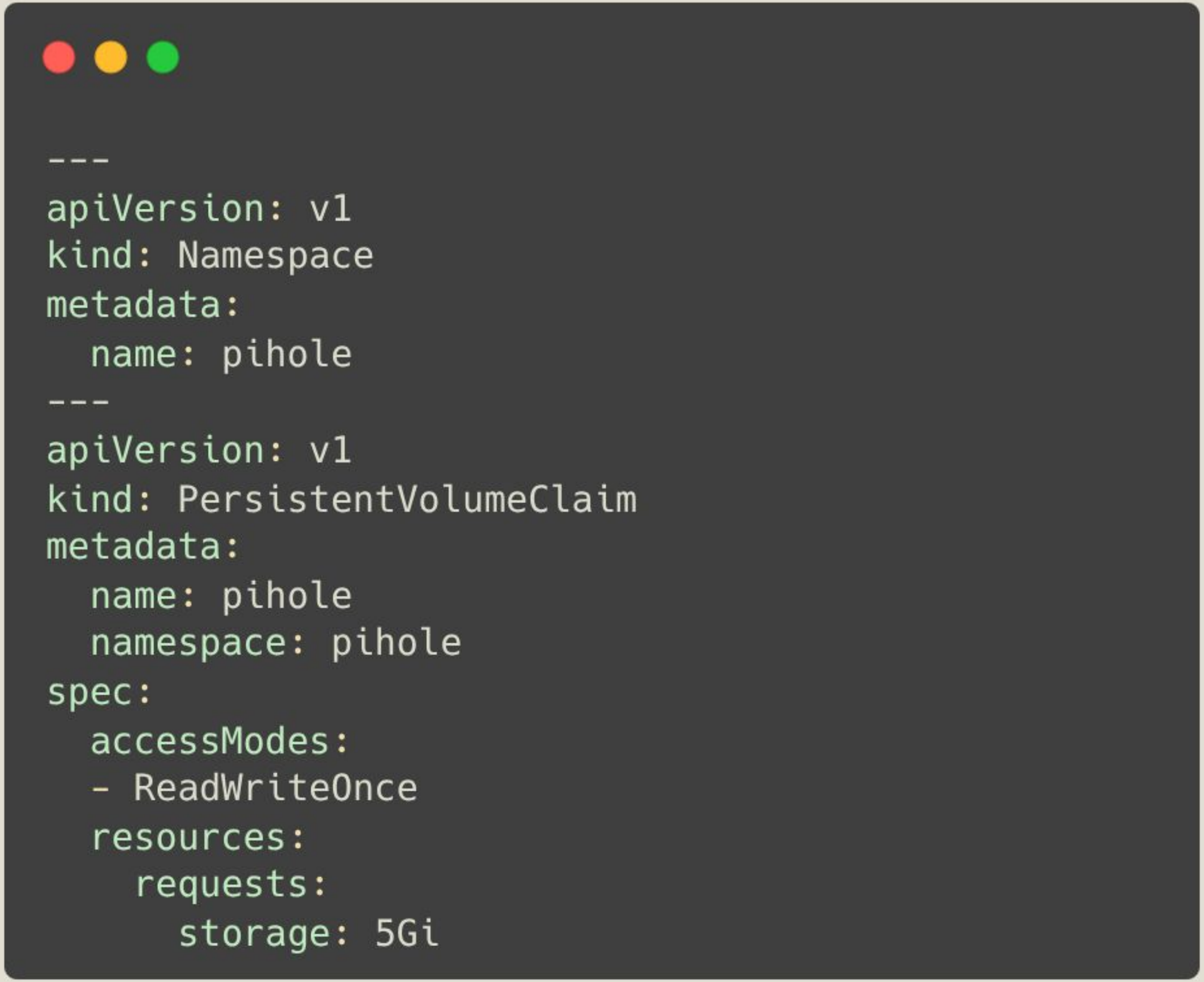
A STATEFUL APP

LET'S TEST LONGHORN WITH AN APPLICATION!

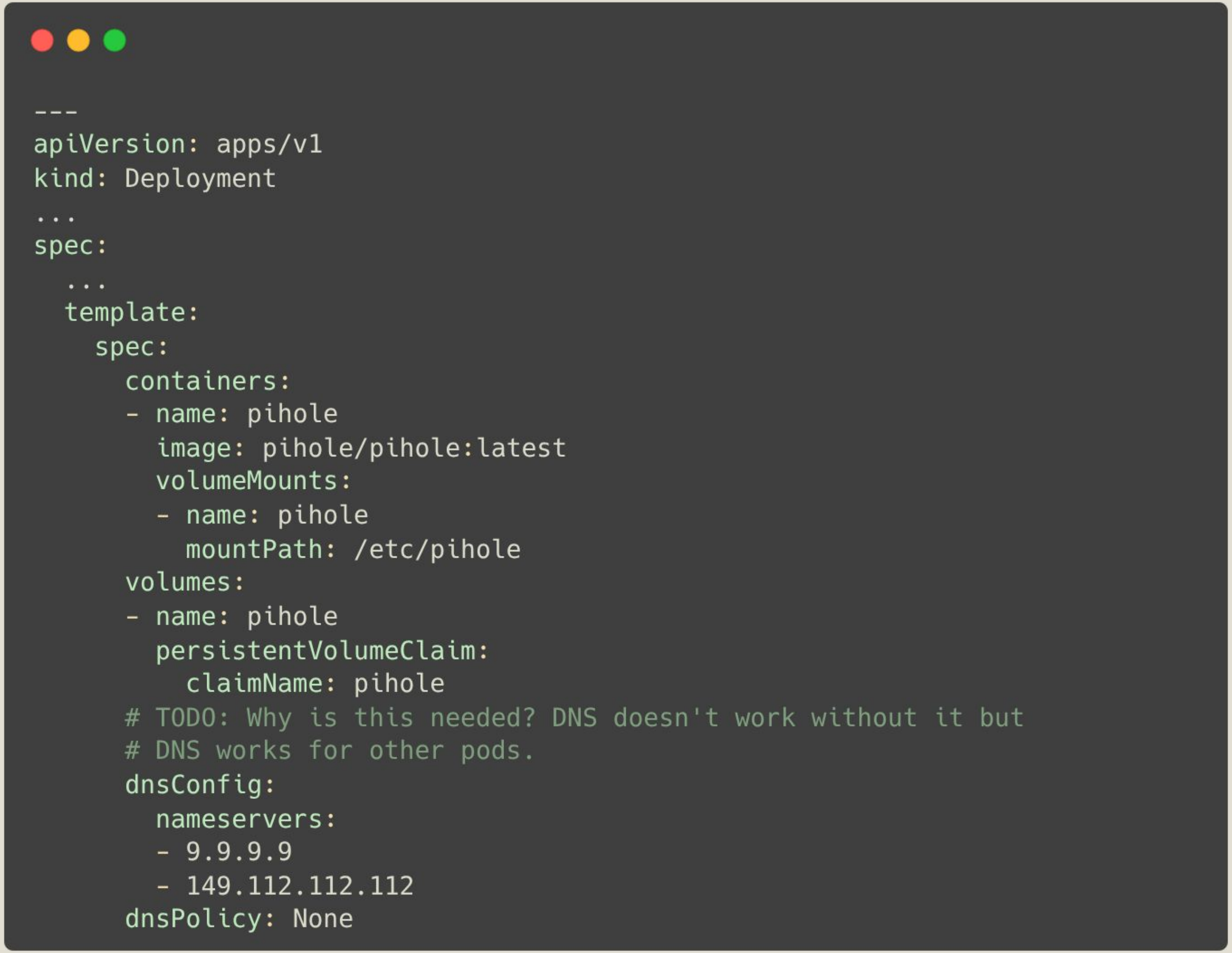
PI-HOLE

- Privacy-focused DNS server
- Built-in DHCP (optional)
- Low-risk deployment

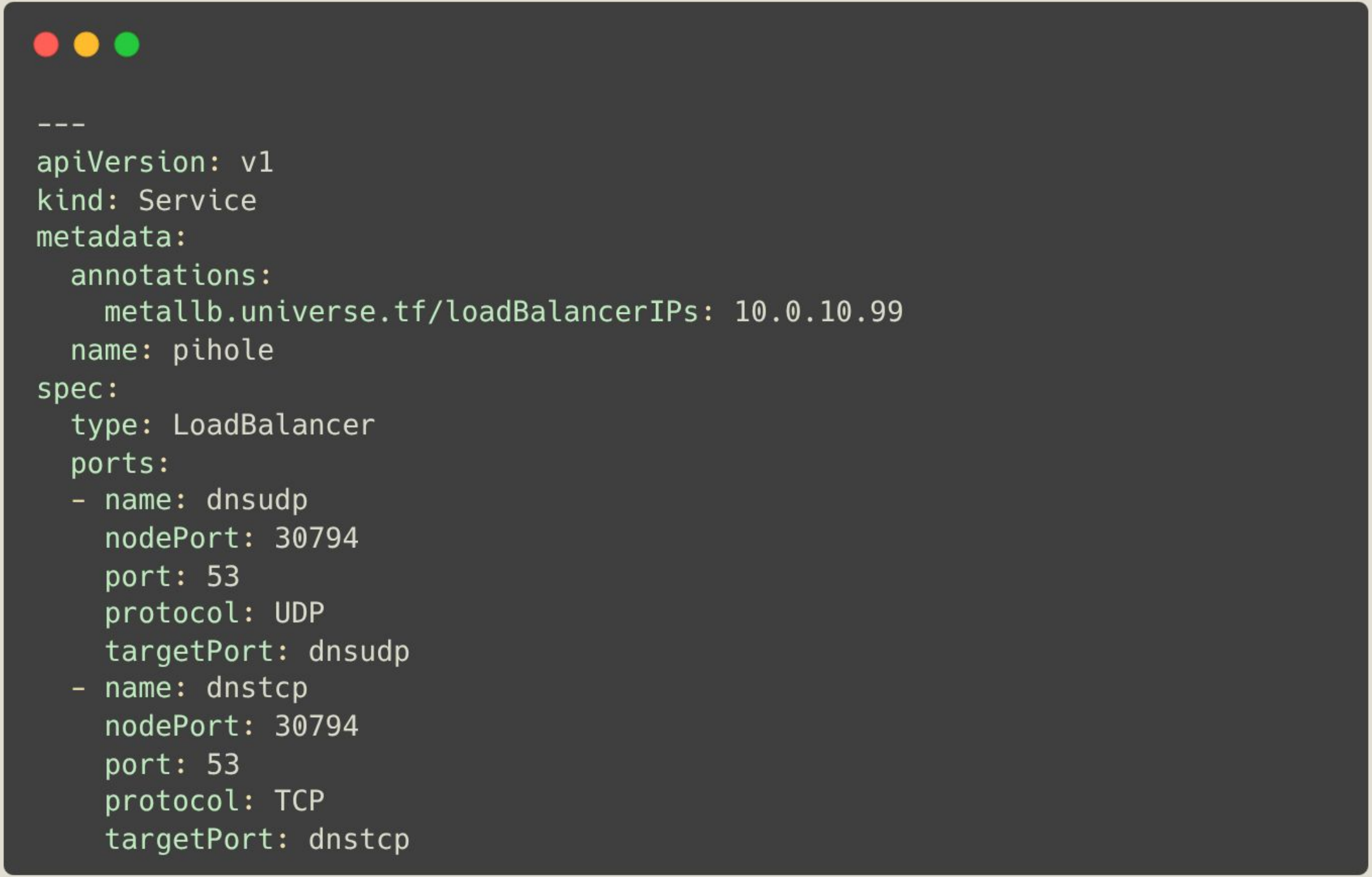




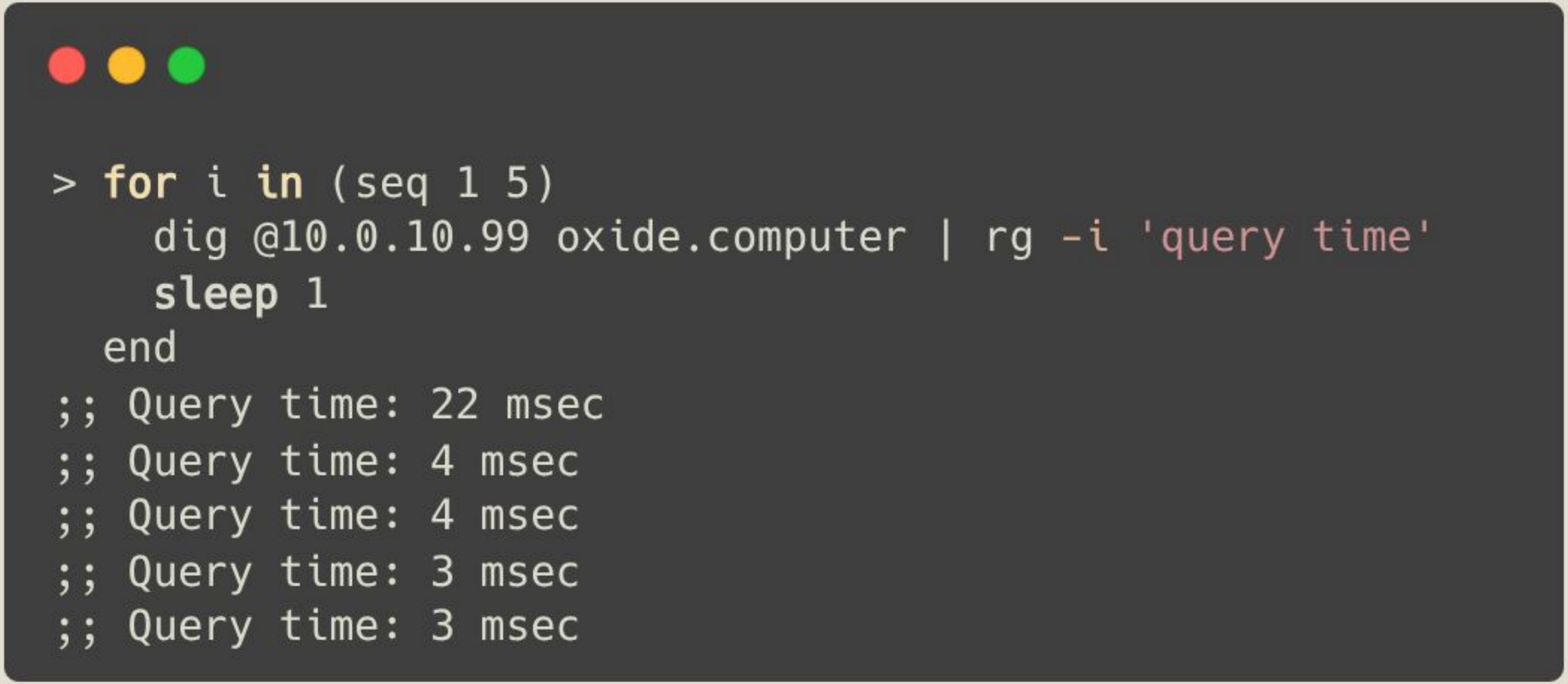
```yaml
---
apiVersion: v1
kind: Namespace
metadata:
  name: pihole
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pihole
  namespace: pihole
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 5Gi
```



```yaml
---
apiVersion: apps/v1
kind: Deployment
...
spec:
  ...
  template:
    spec:
      containers:
      - name: pihole
        image: pihole/pihole:latest
        volumeMounts:
        - name: pihole
          mountPath: /etc/pihole
      volumes:
      - name: pihole
        persistentVolumeClaim:
          claimName: pihole
      # TODO: Why is this needed? DNS doesn't work without it but
      # DNS works for other pods.
      dnsConfig:
        nameservers:
        - 9.9.9.9
        - 149.112.112.112
      dnsPolicy: None
```



```yaml
---
apiVersion: v1
kind: Service
metadata:
  annotations:
    metallb.universe.tf/loadBalancerIPs: 10.0.10.99
  name: pihole
spec:
  type: LoadBalancer
  ports:
  - name: dnsudp
    nodePort: 30794
    port: 53
    protocol: UDP
    targetPort: dnsudp
  - name: dnstcp
    nodePort: 30794
    port: 53
    protocol: TCP
    targetPort: dnstcp
```



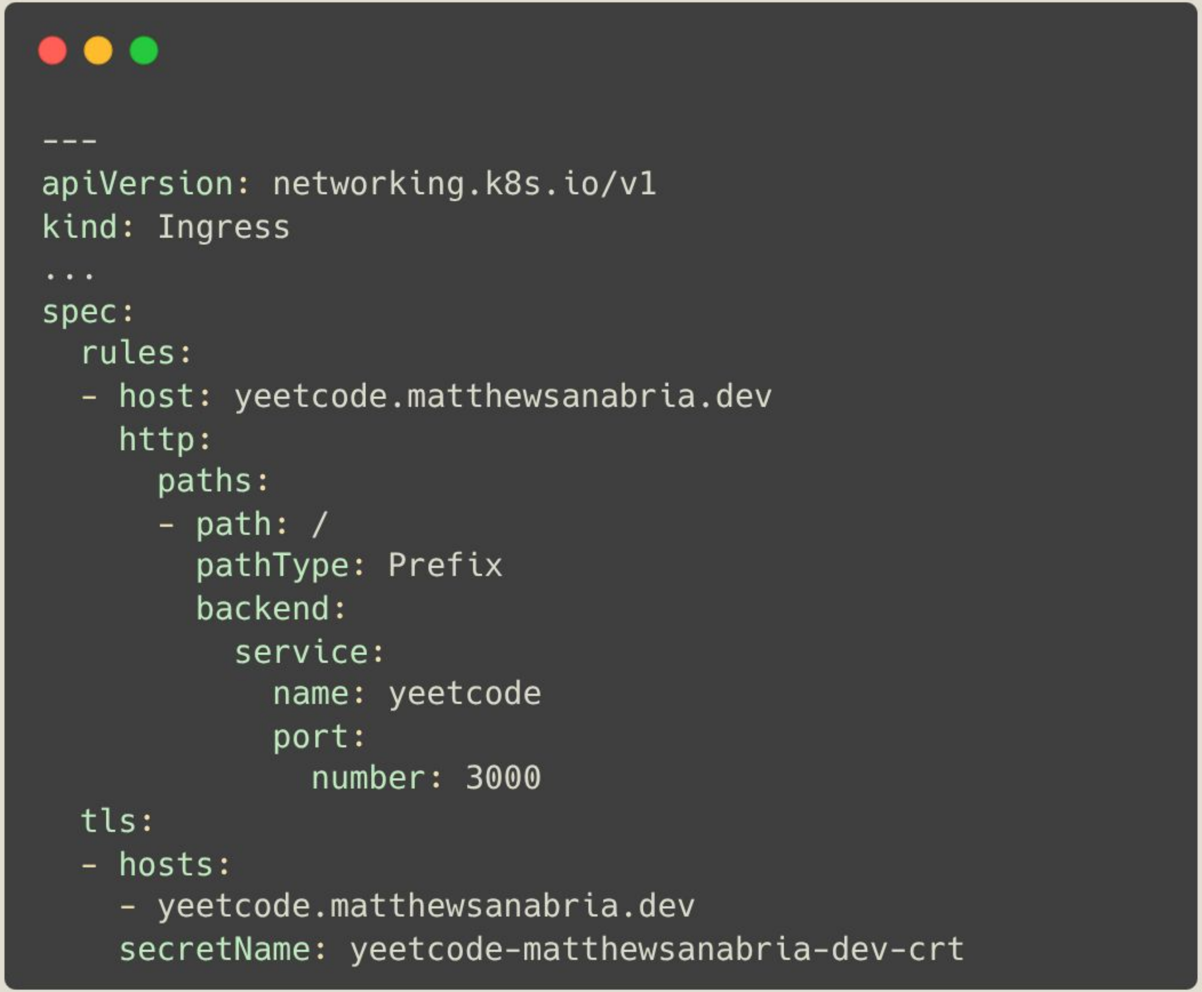
```
> for i in (seq 1 5)
    dig @10.0.10.99 oxide.computer | rg -i 'query time'
    sleep 1
  end
;; Query time: 22 msec
;; Query time: 4 msec
;; Query time: 4 msec
;; Query time: 3 msec
;; Query time: 3 msec
```

A STATELESS APP

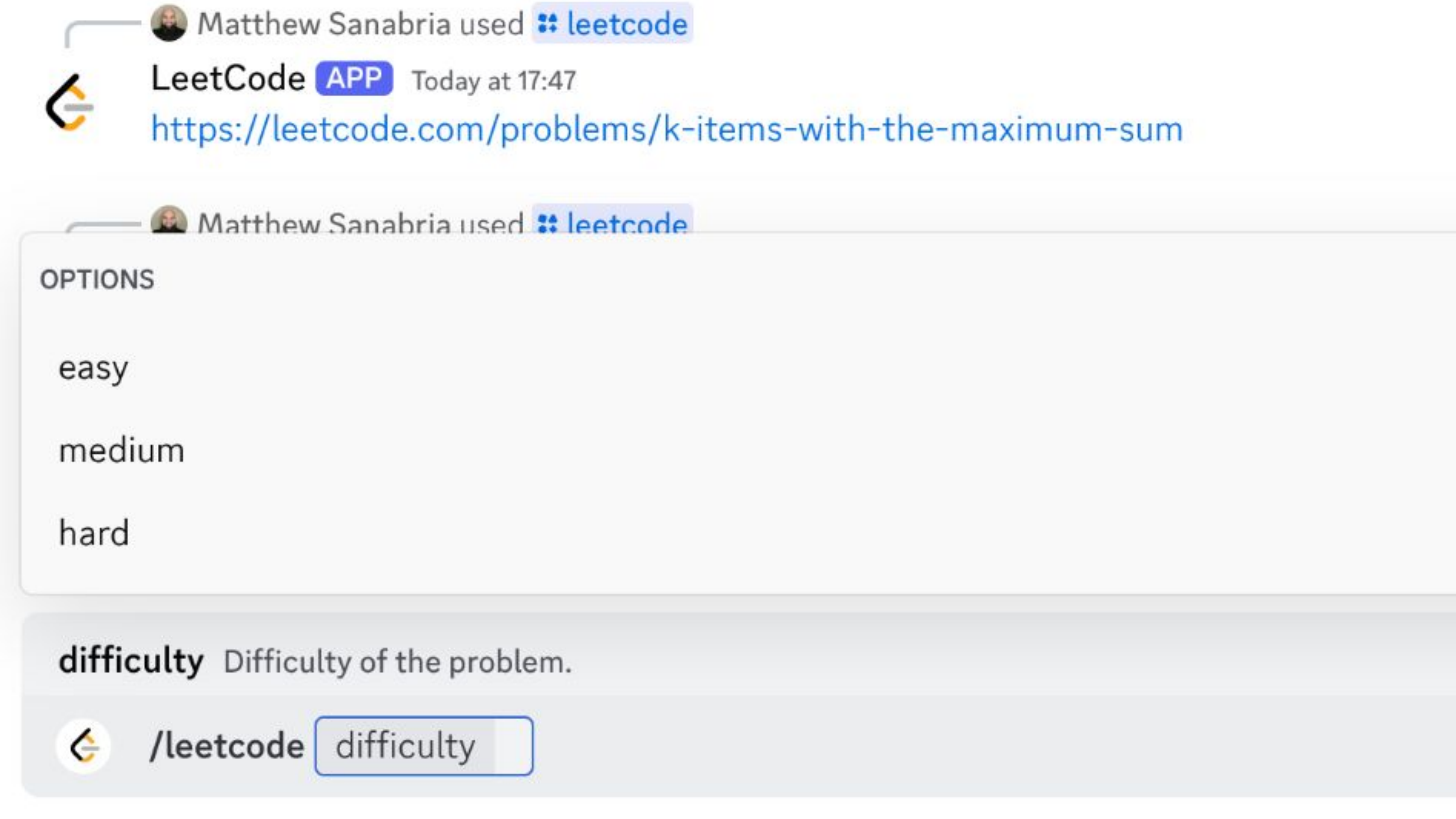
LET'S BUILD A DISCORD SLASH COMMAND!

YEETCODE

- sudomateo/yeetcode
- Discord slash command
  - `/leetcode`
- Retrieves a random LeetCode question of a specific difficulty



```yaml
---
apiVersion: networking.k8s.io/v1
kind: Ingress
...
spec:
  rules:
  - host: yeetcode.matthewsanabria.dev
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: yeetcode
            port:
              number: 3000
  tls:
  - hosts:
    - yeetcode.matthewsanabria.dev
    secretName: yeetcode-matthewsanabria-dev-crt
```



Matthew Sanabria used leetcode

LeetCode APP Today at 17:47

https://leetcode.com/problems/k-items-with-the-maximum-sum

Matthew Sanabria used leetcode

OPTIONS

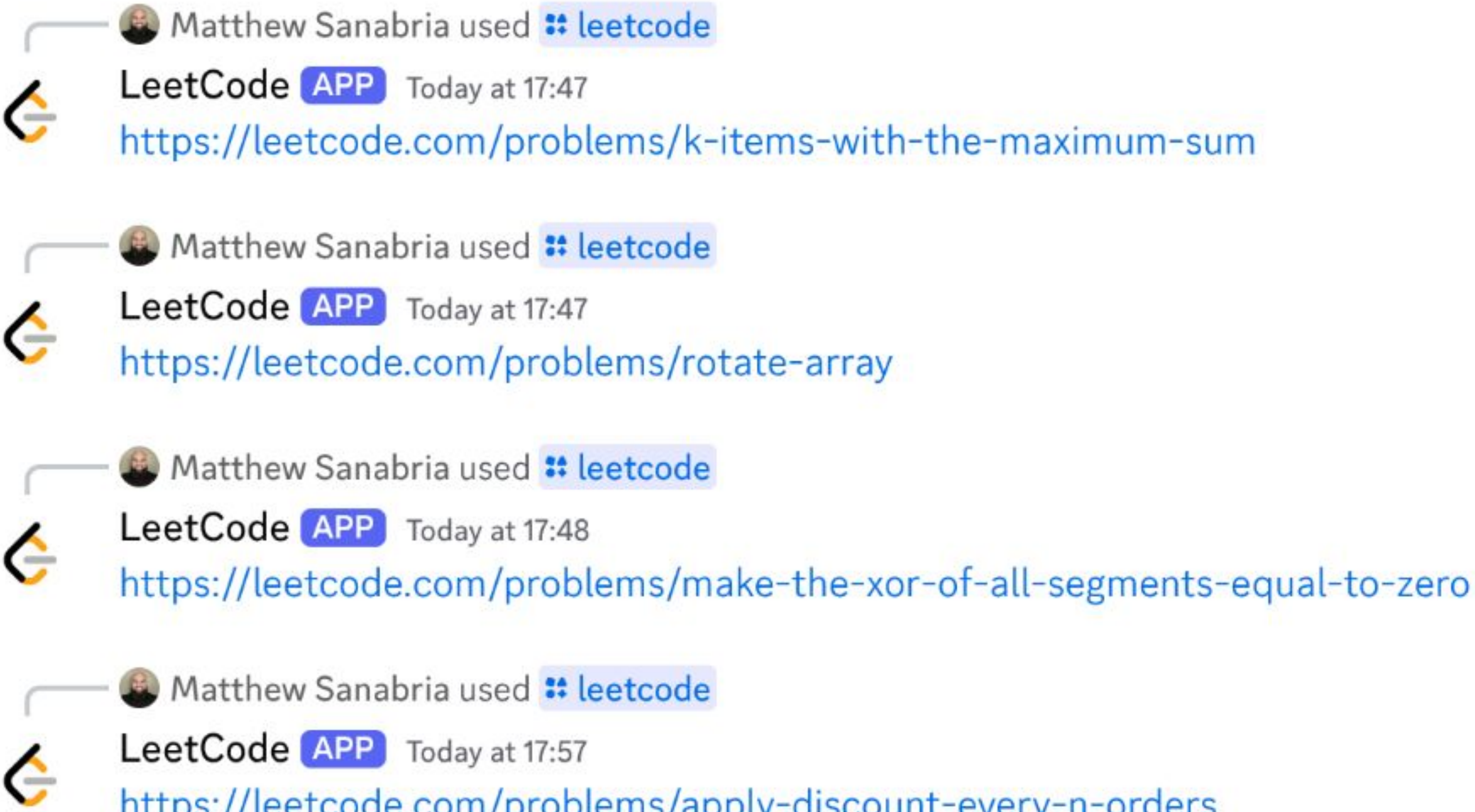
easy

medium

hard

difficulty Difficulty of the problem.

/leetcode difficulty



Matthew Sanabria used leetcode

LeetCode APP Today at 17:47

https://leetcode.com/problems/k-items-with-the-maximum-sum

Matthew Sanabria used leetcode

LeetCode APP Today at 17:47

https://leetcode.com/problems/rotate-array

Matthew Sanabria used leetcode

LeetCode APP Today at 17:48

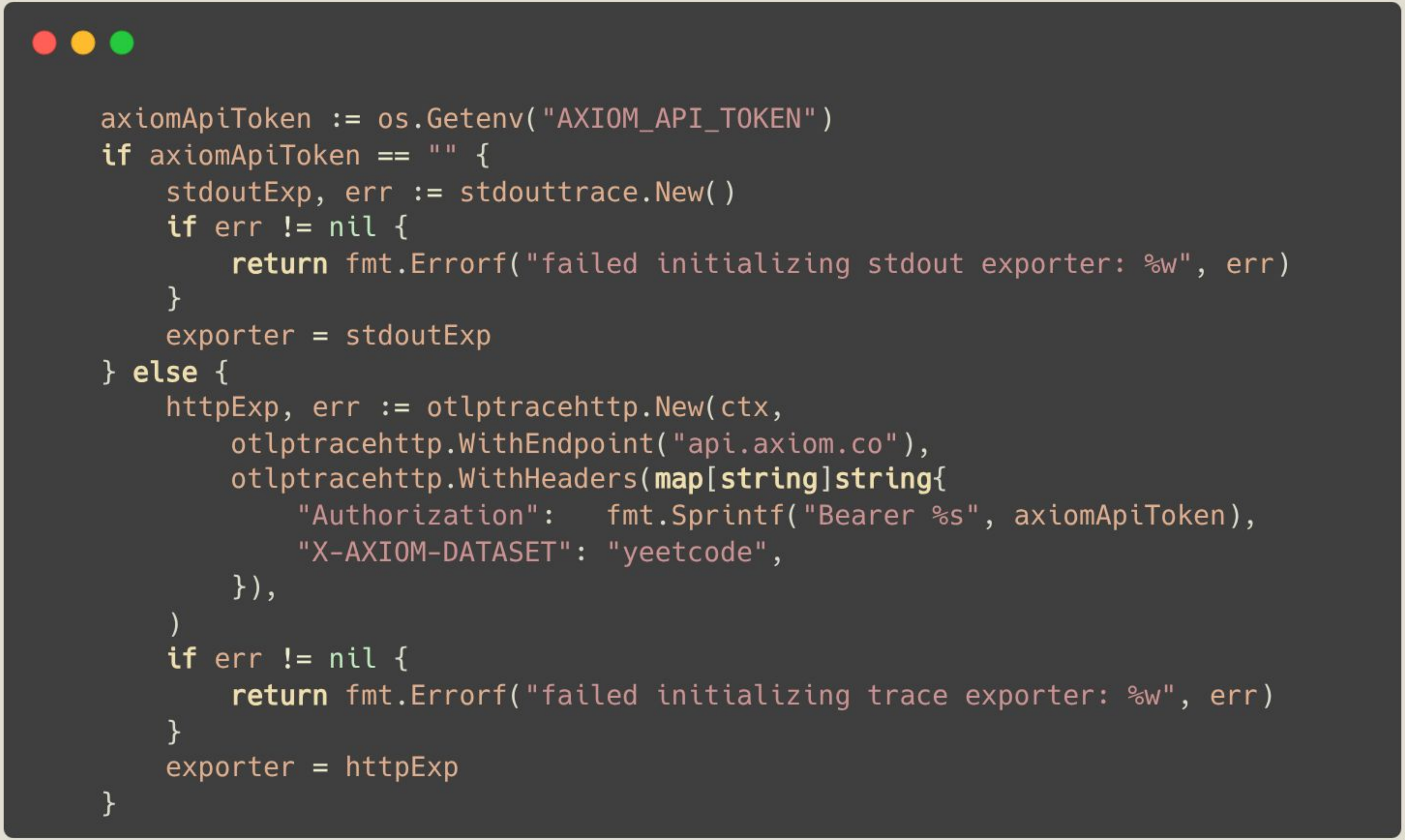
https://leetcode.com/problems/make-the-xor-of-all-segments-equal-to-zero

Matthew Sanabria used leetcode

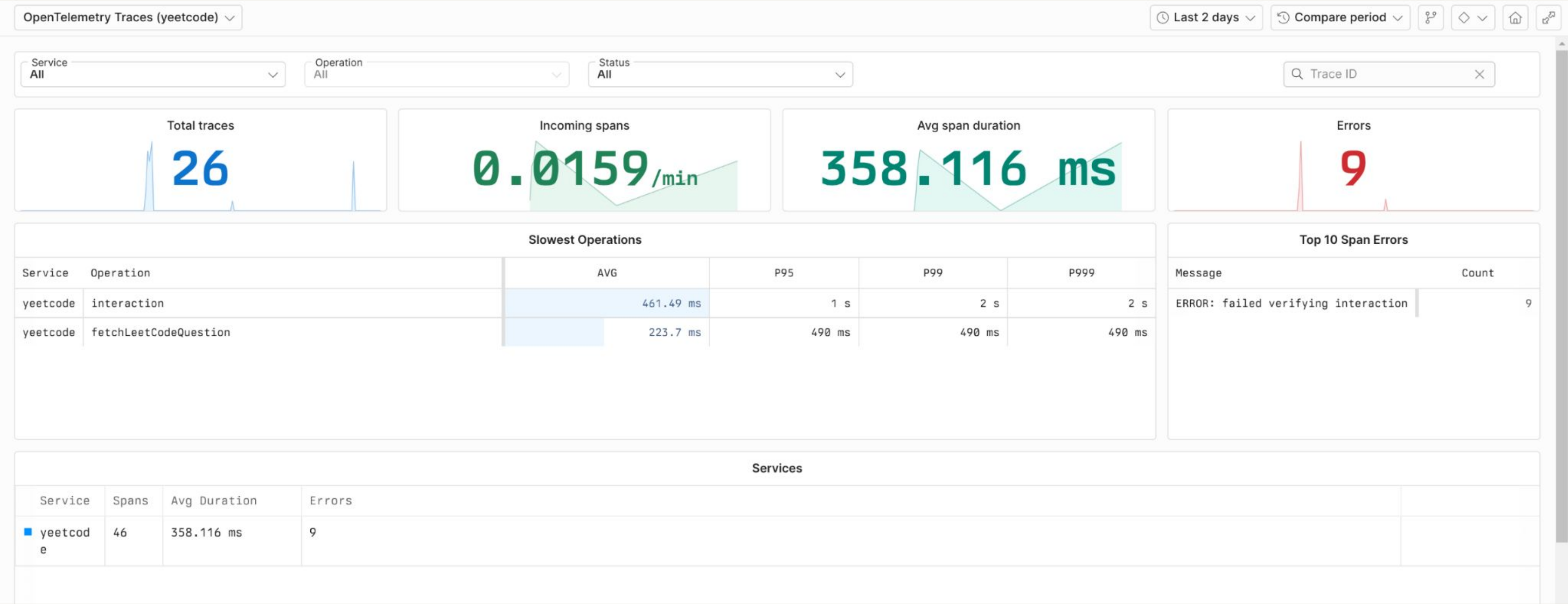
LeetCode APP Today at 17:57

https://leetcode.com/problems/apply-discount-every-n-orders

WHAT ABOUT TELEMETRY?



```
axiomApiToken := os.Getenv("AXIOM_API_TOKEN")
if axiomApiToken == "" {
    stdoutExp, err := stdouttrace.New()
    if err != nil {
        return fmt.Errorf("failed initializing stdout exporter: %w", err)
    }
    exporter = stdoutExp
} else {
    httpExp, err := otlptracehttp.New(ctx,
        otlptracehttp.WithEndpoint("api.axiom.co"),
        otlptracehttp.WithHeaders(map[string]string{
            "Authorization":   fmt.Sprintf("Bearer %s", axiomApiToken),
            "X-AXIOM-DATASET": "yeetcode",
        }),
    )
    if err != nil {
        return fmt.Errorf("failed initializing trace exporter: %w", err)
    }
    exporter = httpExp
}
```



Service: All
Operation: All
Status: All

Total traces: 26

Incoming spans: 0.0159/min

Avg span duration: 358.116 ms

Errors: 9

Slowest Operations

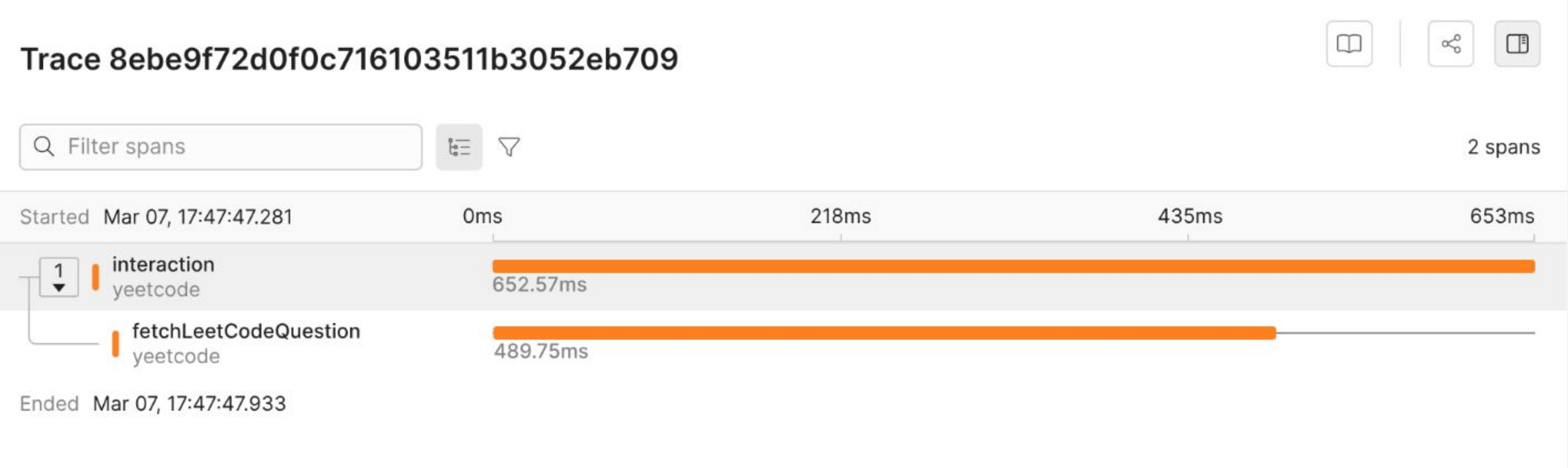
Service	Operation	AVG	P95	P99	P999
yeetcode	interaction	461.49 ms	1 s	2 s	2 s
yeetcode	fetchLeetCodeQuestion	223.7 ms	490 ms	490 ms	490 ms

Top 10 Span Errors

Message	Count
ERROR: failed verifying interaction	9

Services

Service	Spans	Avg Duration	Errors
yeetcode	46	358.116 ms	9



Trace 8ebe9f72d0f0c716103511b3052eb709

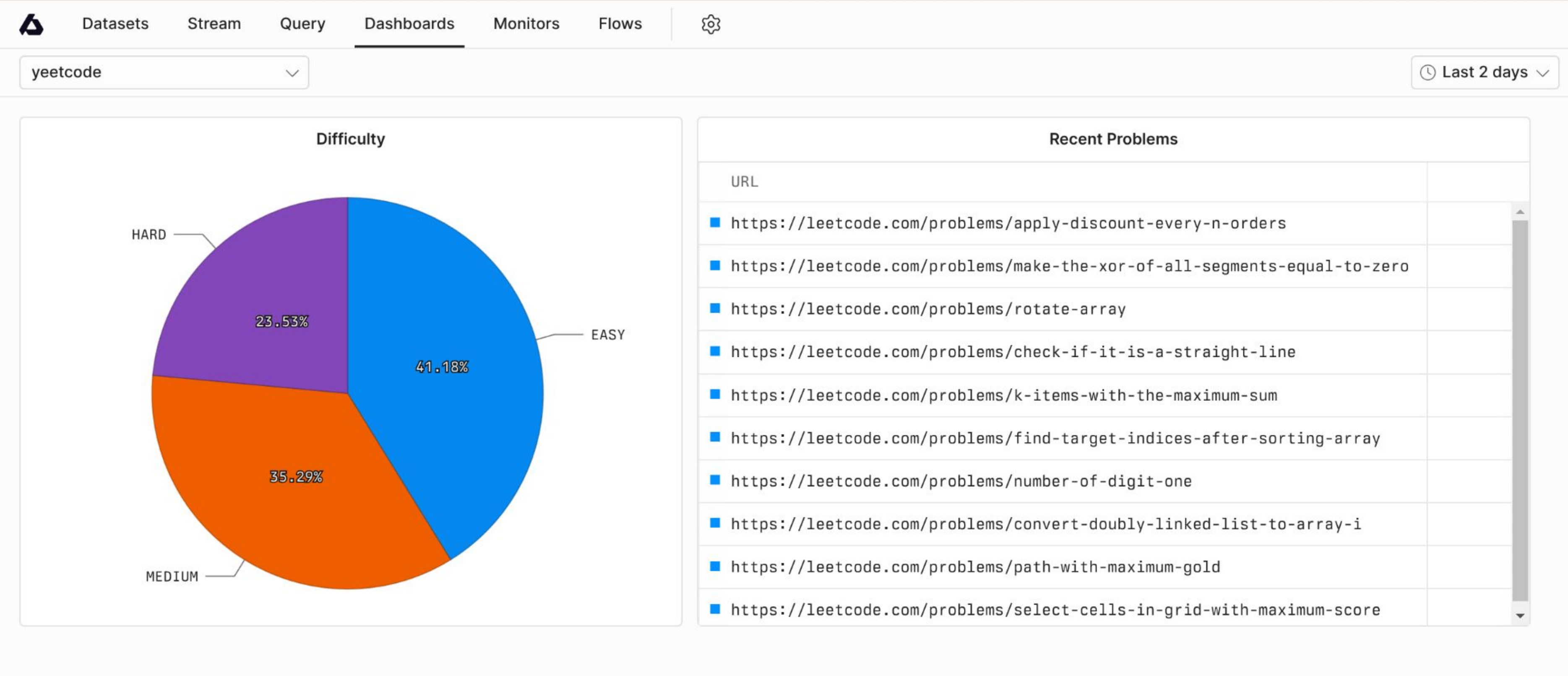
2 spans

Started Mar 07, 17:47:47.281

interaction
yeetcode
652.57ms

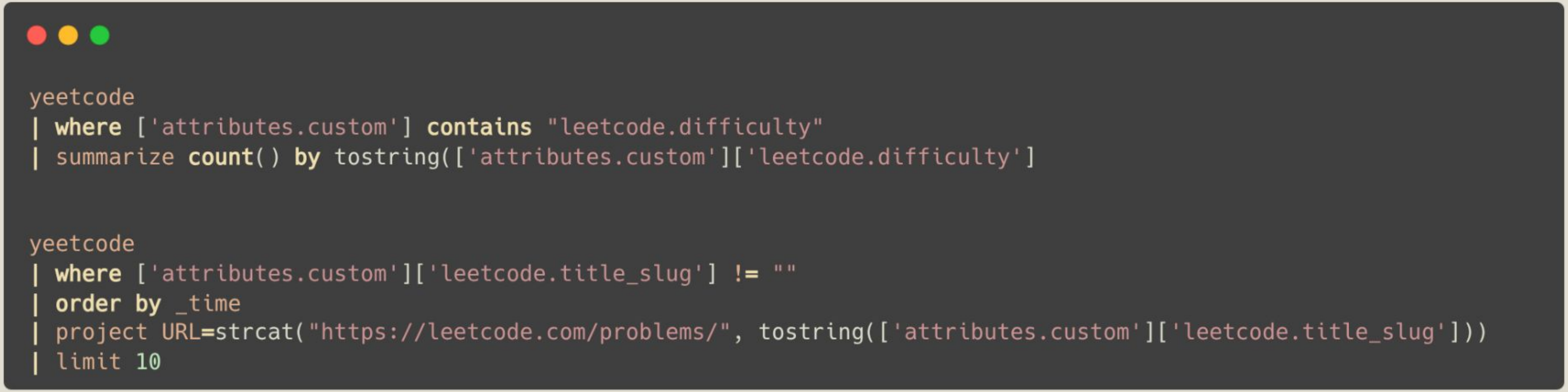
fetchLeetCodeQuestion
yeetcode
489.75ms

Ended Mar 07, 17:47:47.933



Recent Problems

URL
https://leetcode.com/problems/apply-discount-every-n-orders
https://leetcode.com/problems/make-the-xor-of-all-segments-equal-to-zero
https://leetcode.com/problems/rotate-array
https://leetcode.com/problems/check-if-it-is-a-straight-line
https://leetcode.com/problems/k-items-with-the-maximum-sum
https://leetcode.com/problems/find-target-indices-after-sorting-array
https://leetcode.com/problems/number-of-digit-one
https://leetcode.com/problems/convert-doubly-linked-list-to-array-i
https://leetcode.com/problems/path-with-maximum-gold
https://leetcode.com/problems/select-cells-in-grid-with-maximum-score



```
yeetcode
| where ['attributes.custom'] contains "leetcode.difficulty"
| summarize count() by tostring(['attributes.custom']['leetcode.difficulty']


yeetcode
| where ['attributes.custom']['leetcode.title_slug'] != ""
| order by _time
| project URL=strcat("https://leetcode.com/problems/", tostring(['attributes.custom']['leetcode.title_slug']))
| limit 10
```

LESSONS LEARNED

THE CLOUD...

THE CLOUD...
HAS RUINED ME

LEAVE TIME FOR DEBUGGING

ANNOTATIONS ARE WONDERFUL/SCARY

DOCUMENTATION MAY NOT EXIST

ASK QUESTIONS!

THE FUTURE

USE MULTIPLE NAMESPACES

USE SECRETS INTEGRATIONS

DEPLOY AND DOCUMENT MORE

WRITE SOME CONTROLLERS

BETTER OBSERVABILITY

DISASTER RECOVERY

GO HOME LAB!

THANK YOU

Go home, and lab!
https://matthewsanabria.dev