MINISOC

SCOPO DEL PROGETTO

MONITORING

Collezionare, categorizzare e analizzare i dati ricevuti, statistiche ed eventi critici dell’ infrastruttura

ANALYSIS

Network analysis, Malware analysis, report di findings

RESPONSE

Fornire capacità di difesa attiva e analisi su tutta l’infrastruttura

SCALABLE & RELIABLE

L’intero progetto è basato su Kubernetes, cloud e soluzioni di autoscaling

MICROSERVIZIO: SOC

- Interagisce con l’utente finale
- Invia richieste agli altri microservizi
- Supporta **l’autenticazione**
- Fornisce le **dashboard** di sicurezza
- **Monitora** le applicazioni scelte dall’utente

Flask

MICROSERVIZIO: TOOL

- Utilizza tool come **YARA** e **TSHARK**
- **Network analysis** di *pcap* caricati tramite il SOC
- **Malware analysis** di *file* caricati tramite il SOC
- Raggiungibile solo dal SOC

VIRUSTOTAL

+

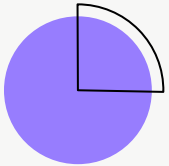
WIRESHARK

MICROSERVIZIO: APP

- Semplice servizio **web**
- Applicazione esposta su Internet
- Può essere sostituita con una vera applicazione da monitorare

+

GoAccess



KUBERNETES

Grazie a Kubernetes il deploy può essere automatizzato, esteso e facilmente configurato a runtime. La divisione delle risorse presenti è nel seguente modo

DEPLOYMENT → Microservizi che compongono il progetto

SERVICES → Networking e balancing verso internet

HPA → Autoscale dei microservizi

VOLUMES → Condivisione dei dati tra i Microservizi

CONFIGMAP → Configurazione parametrica

HELM

Helm permette di facilitare il deployment fornendo template personalizzabili.
Abbiamo utilizzato Helm per installare le seguenti risorse

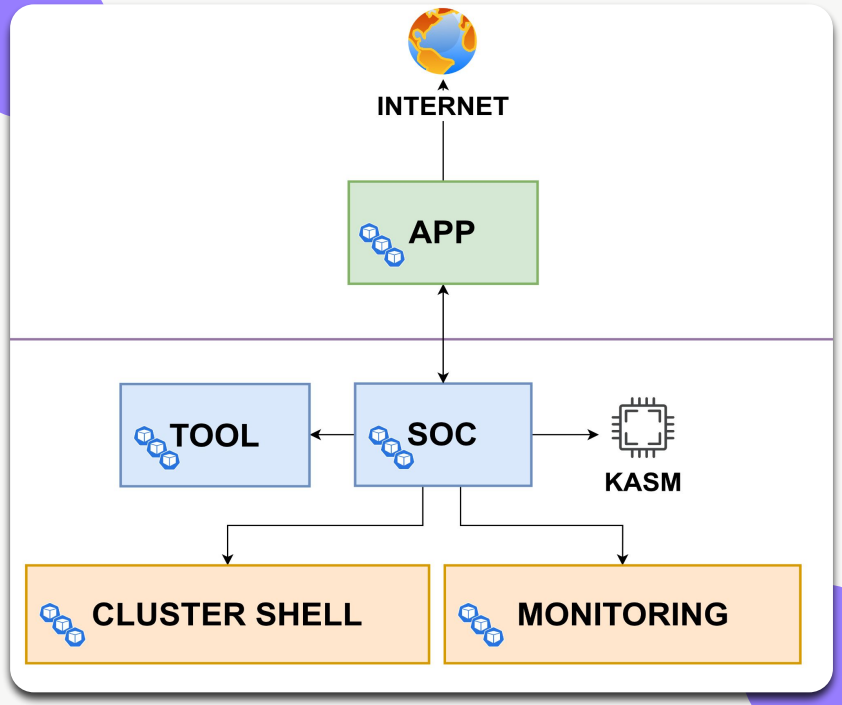
GRAFANA → Grafici delle performance dei Cluster

PROMETHEUS → Collector di Log

TRIVY → Collector di vulnerabilità dei microservizi

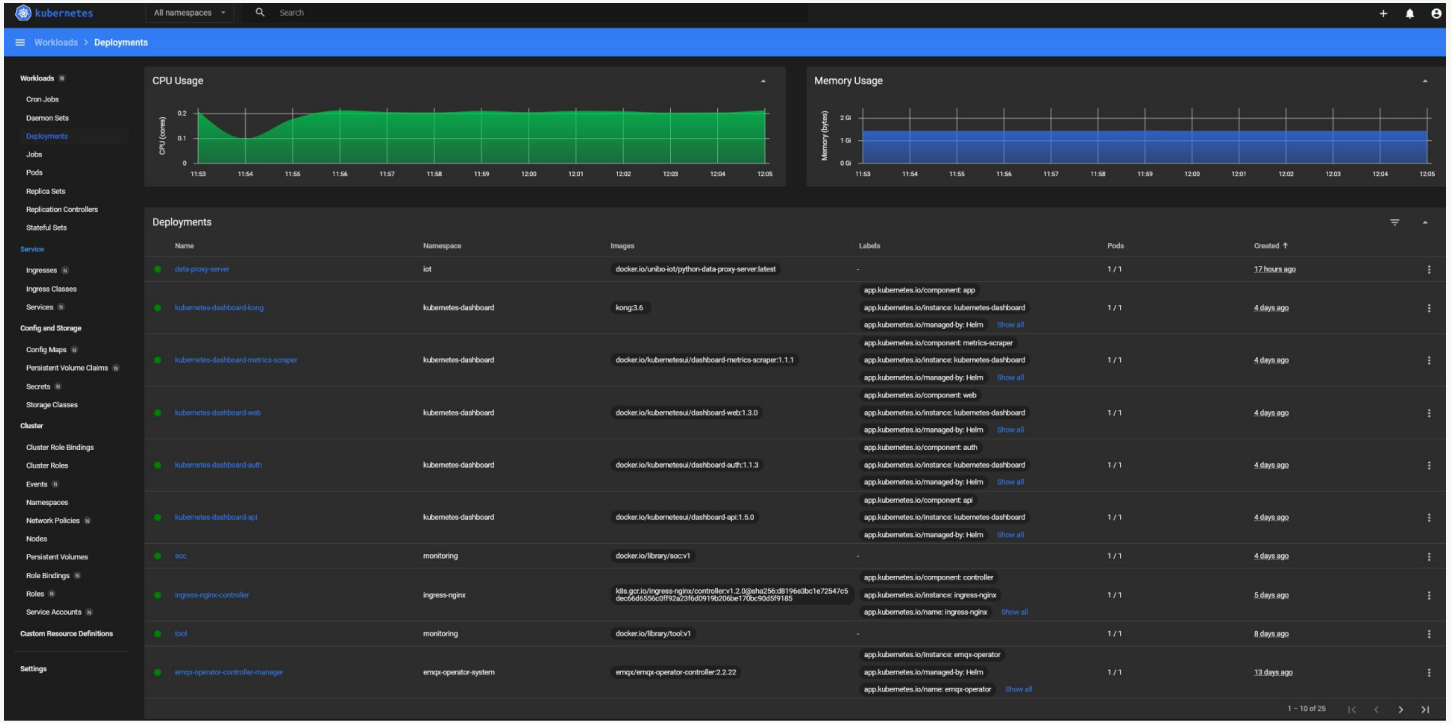
DASHBOARD → Pannello di controllo del Cluster

DIAGRAMMA APPLICATIVO

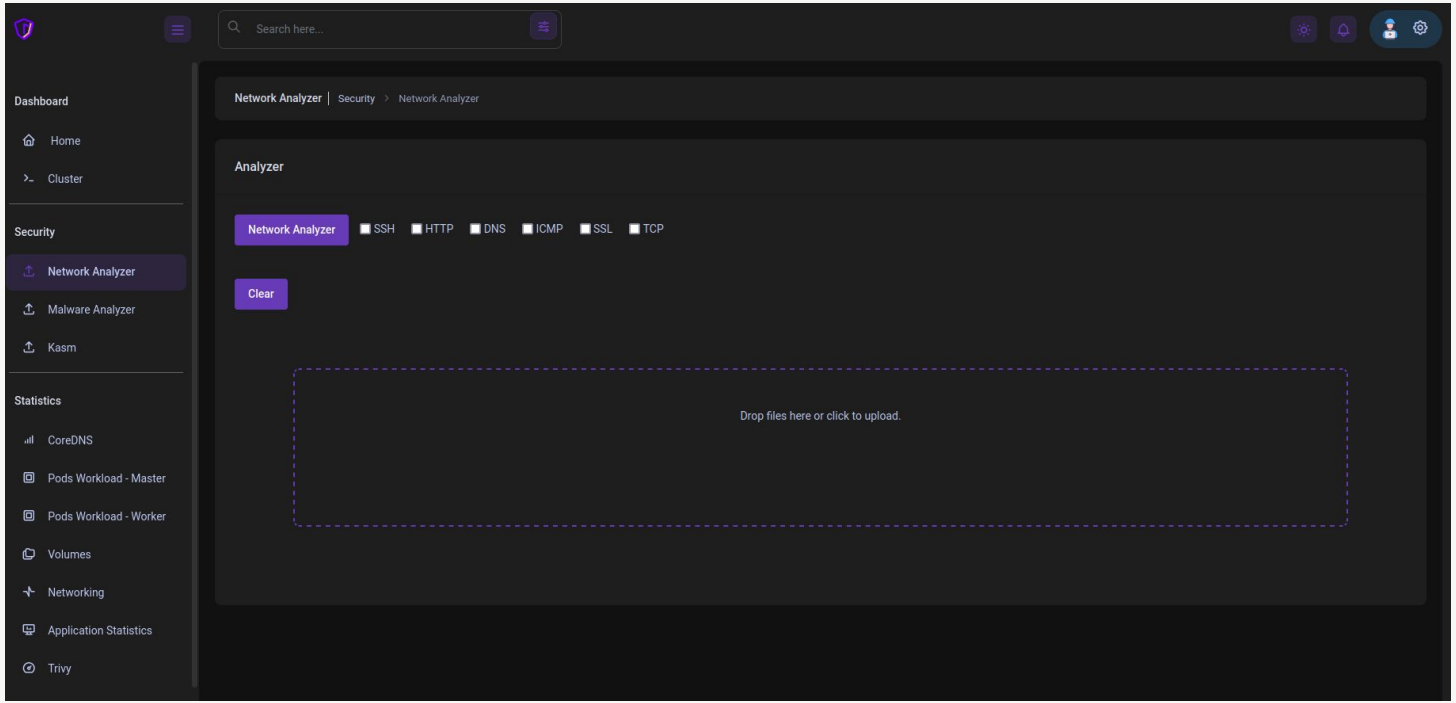


LANDING PAGE

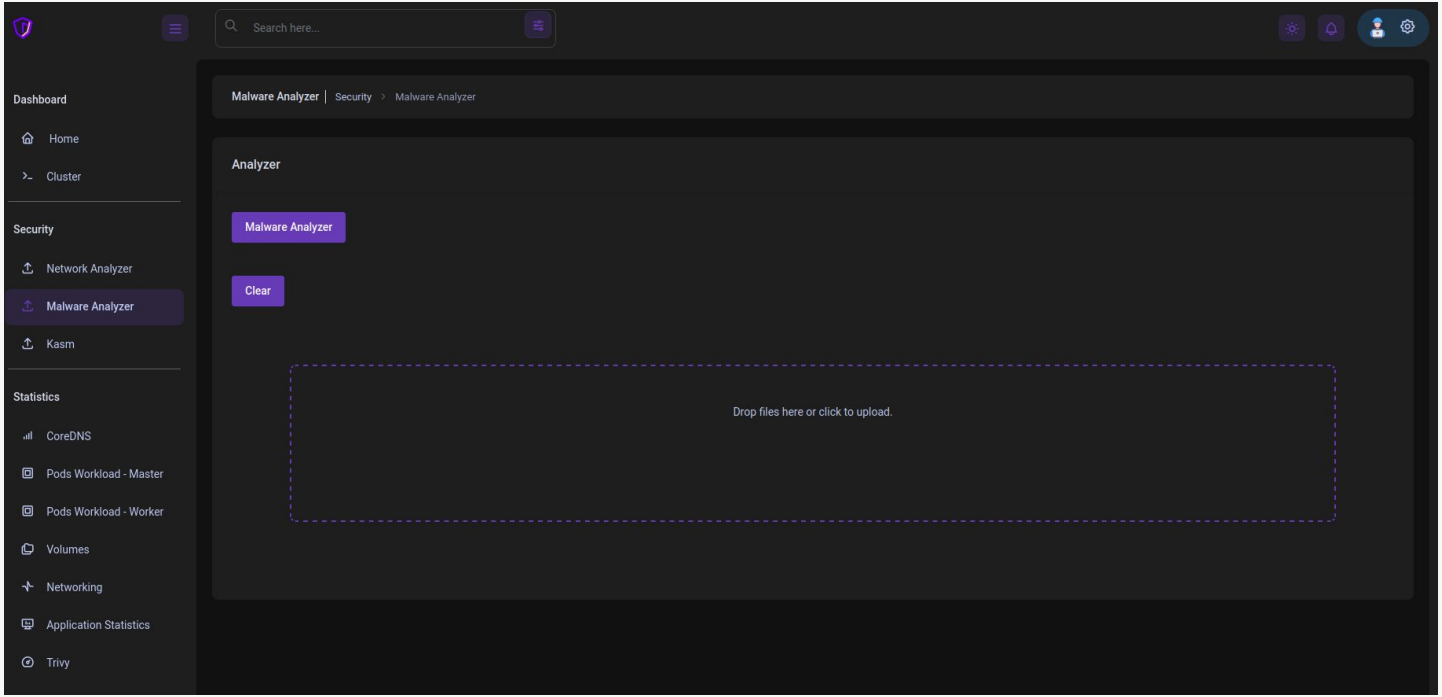
DASHBOARD


kubernetes
All namespaces
Search
Workloads > Deployments
Workloads
Cron Jobs
Daemon Sets
Deployments
Jobs
Pods
Replica Sets
Replication Controllers
Stateful Sets
Service
Ingresses
Ingress Classes
Services
Config and Storage
Config Maps
Persistent Volume Claims
Secrets
Storage Classes
Cluster
Cluster Role Bindings
Cluster Roles
Events
Namespaces
Network Policies
Nodes
Persistent Volumes
Role Bindings
Roles
Service Accounts
Custom Resource Definitions
Settings
CPU Usage
CPU (cores)
Memory Usage
Memory (bytes)
Deployments
Name
Namespace
Images
Labels
Pods
Created
data-proxy-server
iot
docker.io/unibo-iot/python-data-proxy-server:latest
1 / 1
17 hours ago
kubernetes-dashboard-kong
kubernetes-dashboard
kong:3.6
app.kubernetes.io/component: app
app.kubernetes.io/instance: kubernetes-dashboard
app.kubernetes.io/managed-by: Helm
Show all
1 / 1
4 days ago
kubernetes-dashboard-metrics-scraper
docker.io/kubernetesui/dashboard-metrics-scraper:1.1.1
app.kubernetes.io/component: metrics-scraper
app.kubernetes.io/instance: kubernetes-dashboard
app.kubernetes.io/managed-by: Helm
Show all
1 / 1
4 days ago
kubernetes-dashboard-web
docker.io/kubernetesui/dashboard-web:1.3.0
app.kubernetes.io/component: web
app.kubernetes.io/instance: kubernetes-dashboard
app.kubernetes.io/managed-by: Helm
Show all
1 / 1
4 days ago
kubernetes-dashboard-auth
docker.io/kubernetesui/dashboard-auth:1.1.3
app.kubernetes.io/component: auth
app.kubernetes.io/instance: kubernetes-dashboard
app.kubernetes.io/managed-by: Helm
Show all
1 / 1
4 days ago
kubernetes-dashboard-api
docker.io/kubernetesui/dashboard-api:1.5.0
app.kubernetes.io/component: api
app.kubernetes.io/instance: kubernetes-dashboard
app.kubernetes.io/managed-by: Helm
Show all
1 / 1
4 days ago
soc
monitoring
docker.io/library/soc:v1
1 / 1
4 days ago
ingress-nginx-controller
ingress-nginx
k8s.gcr.io/ingress-nginx/controller:v1.2.0@sha256:d8196e3bc1e72547c5dec66d6556c0ff92a23f6d0919b206be170bc90d5f9185
app.kubernetes.io/component: controller
app.kubernetes.io/instance: ingress-nginx
app.kubernetes.io/name: ingress-nginx
Show all
1 / 1
5 days ago
tool
monitoring
docker.io/library/tool:v1
1 / 1
8 days ago
emqx-operator-controller-manager
emqx-operator-system
emqx/emqx-operator-controller:2.2.22
app.kubernetes.io/instance: emqx-operator
app.kubernetes.io/managed-by: Helm
app.kubernetes.io/name: emqx-operator
Show all
1 / 1
13 days ago
1 – 10 of 25

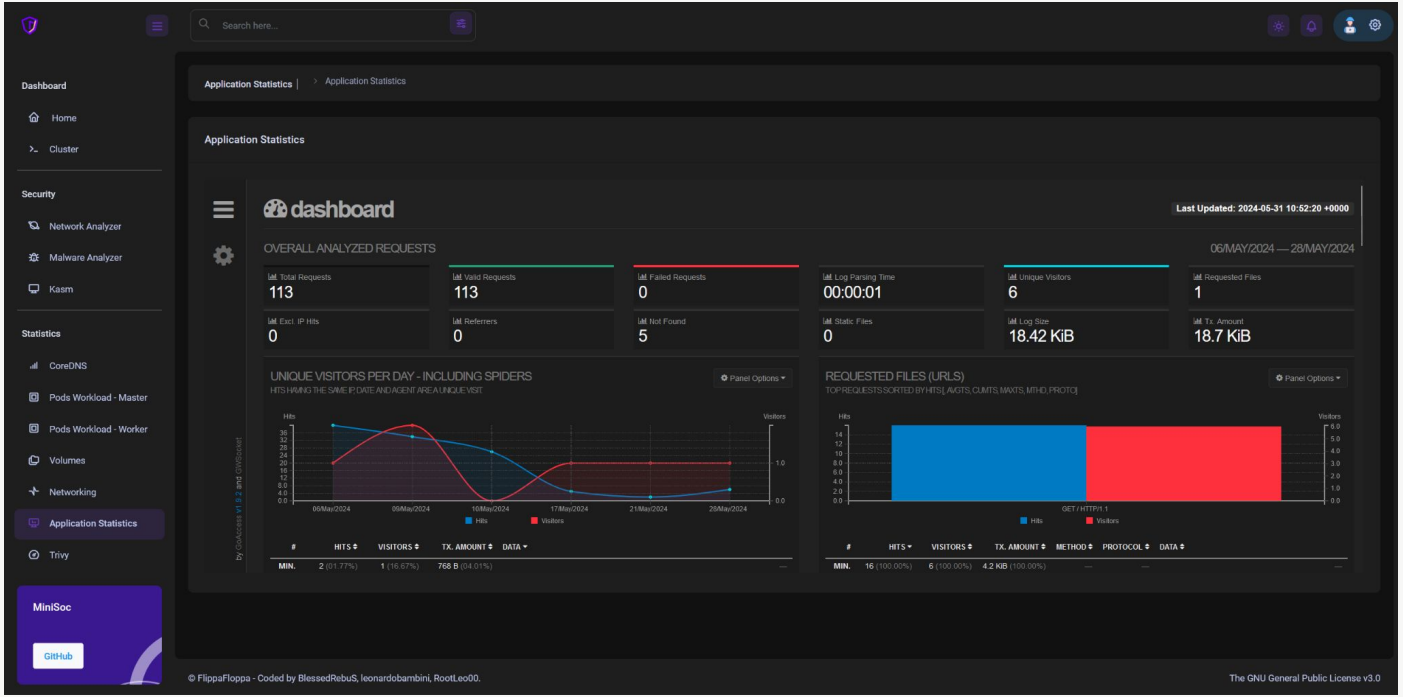
NETWORK ANALYZER



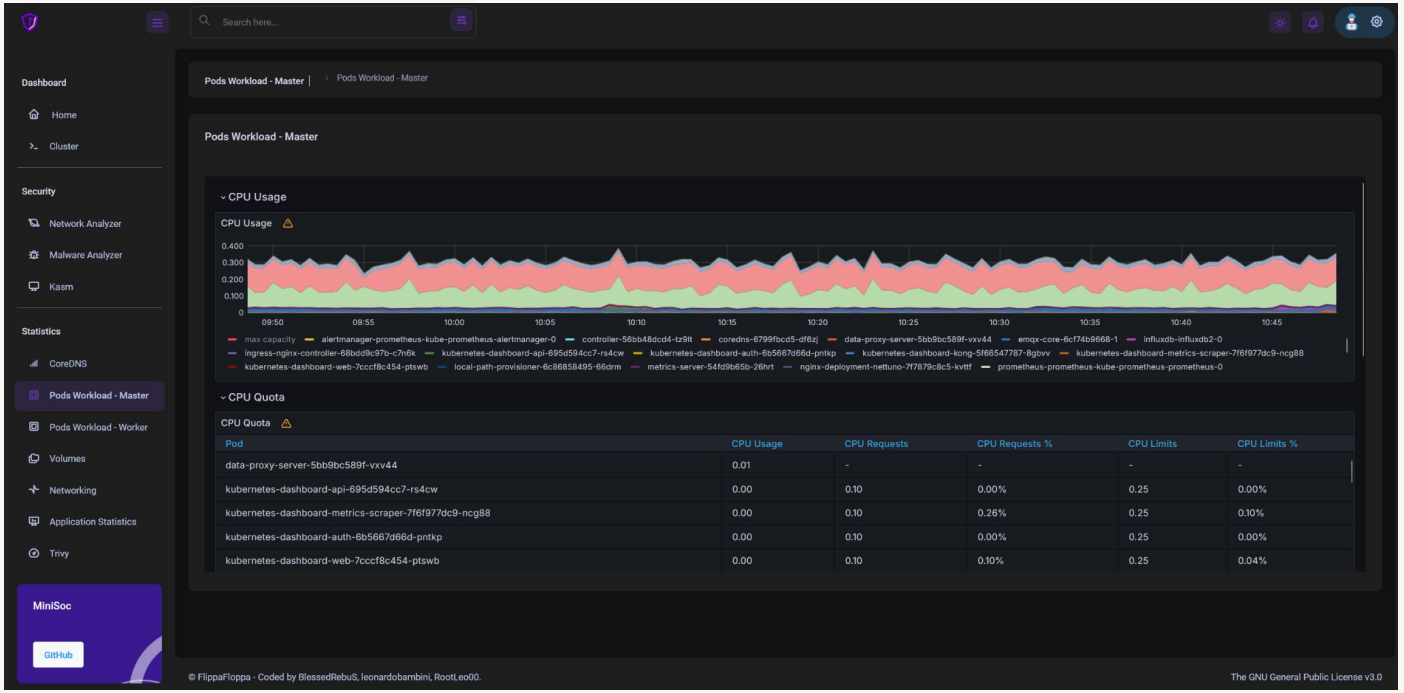
MALWARE ANALYZER



APPLICATION STATISTICS



CLUSTER PERFORMANCES



NODI & DEPLOYMENTS

- Ogni elemento dell'applicazione è un **deployment**
- In presenza di poche o nulle richieste ogni deploy ha 1 solo pod
- I deployment sono bilanciati su **due nodi**

Nodes(all)[2]

NAME↑	STATUS	ROLE	TAINTS	VERSION	PODS	CPU	MEM	%CPU	%MEM	CPU/A	MEM/A	AGE
k3s-worker	Ready	<none>	0	v1.29.4+k3s1	13	145	2024	3	25	4000	7941	27d
kubernetes	Ready	control-plane,master	0	v1.29.4+k3s1	25	561	4044	7	25	8000	15991	28d

<node>

Deployments(monitoring)[7]

NAME↑	READY	UP-TO-DATE	AVAILABLE	AGE
prometheus-grafana	1/1	1	1	24d
prometheus-kube-prometheus-operator	1/1	1	1	24d
prometheus-kube-state-metrics	1/1	1	1	24d
simple-app	1/1	1	1	24d
soc	1/1	1	1	4d4h
tool	1/1	1	1	8d
trivy-operator	1/1	1	1	24d

<deployment>

SERVICES

- I services gestiscono il **networking**
- Divisi in **LoadBalancer** e **ClusterIP**
- **External-IP** è legato alla NIC della VM, usando **MetalLB**

```yaml
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  name: soc-svc
  namespace: monitoring
  annotations:
    metallb.universe.tf/allow-shared-ip: "scalable"
spec:
  ports:
  - port: 5000
    protocol: TCP
    targetPort: 5000
  selector:
    app: soc
  sessionAffinity: ClientIP
  sessionAffinityConfig:
    clientIP:
      timeoutSeconds: 10000
  type: LoadBalancer
status:
  loadBalancer: {}
```

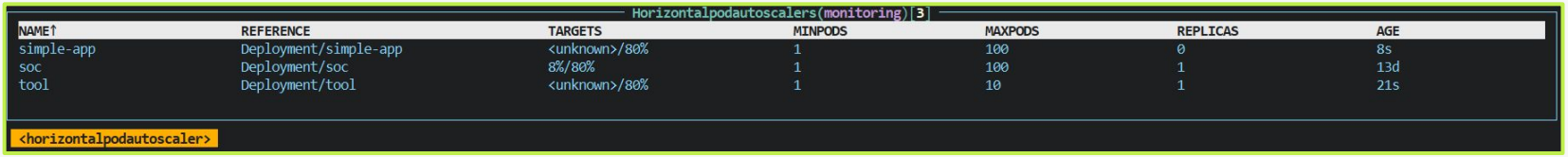
Services(monitoring)[12]

NAME↑	TYPE	CLUSTER-IP	EXTERNAL-IP	PORTS	AGE
alertmanager-operated	ClusterIP			http-web:9093►0 tcp-mesh:9094►0 udp-mesh:9094►0/UDP	24d
prometheus-grafana	LoadBalancer	10.43.196.92	192.168.1.150	http-web:3000►32323	24d
prometheus-kube-prometheus-alertmanager	ClusterIP	10.43.26.232		http-web:9093►0 reloader-web:8080►0	24d
prometheus-kube-prometheus-operator	ClusterIP	10.43.61.234		https:443►0	24d
prometheus-kube-prometheus-prometheus	ClusterIP	10.43.142.161		http-web:9090►0 reloader-web:8080►0	24d
prometheus-kube-state-metrics	ClusterIP	10.43.22.20		http:8080►0	24d
prometheus-operated	ClusterIP			http-web:9090►0	24d
prometheus-prometheus-node-exporter	ClusterIP	10.43.90.200		http-metrics:9100►0	24d
simple-app	LoadBalancer	10.43.63.40	192.168.1.150	5001►32746	24d
soc-svc	LoadBalancer	10.43.84.210	192.168.1.150	5000►32529	13d
tool-svc	ClusterIP	10.43.92.154		5001►0	8d
trivy-operator	ClusterIP			metrics:80►0	24d

<service>

SCALABILITY: HPA

- **Horizontal Pod Autoscaler** impostato su tutti i deployment
- Permette di gestire il carico in maniera **dinamica**
- Ripristina il numero di **pod** una volta passato il picco di carico



VOLUME & STORAGE 1/2

- Vengono definiti dei **Persistent Volume** come pool di storage
- Tramite i **Persistent Volume Claim** i pod si legano ai PV
- La capacity dei volumi è definibile a runtime
- Su **AWS** è definito **l'autoscaling** dei volumi

Persistentvolumeclaims(monitoring)[2]

NAME↑	STATUS	VOLUME	CAPACITY	ACCESS MODES	STORAGECLASS	AGE
ingress-pvc	Bound	pvc-00634e9f-28af-4954-aca1-1b41448b0016	200Mi	RWO	local-path	25d
shared-pvc	Bound	shared-pv	1Gi	RWO	local-path	15d

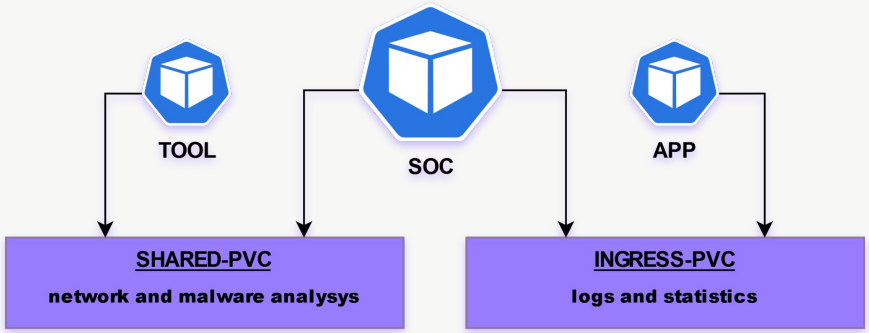
<persistentvolumeclaim>

NAME	PROVISIONER	RECLAIMPOLICY	VOLUMEBINDINGMODE	ALLOWVOLUMEEXPANSION	AGE
gp2 (default)	kubernetes.io/aws-ebs	Delete	WaitForFirstConsumer	false	82m
gp3 (default)	ebs.csi.aws.com	Retain	WaitForFirstConsumer	true	6m5s

NAMESPACE	NAME	READY	UP-TO-DATE	AVAILABLE	AGE
kube-system	coredns	2/2	2	2	79m
kube-system	ebs-csi-controller	2/2	2	2	77m
monitoring	prometheus-grafana	1/1	1	1	54m
monitoring	prometheus-kube-prometheus-operator	1/1	1	1	54m
monitoring	prometheus-kube-state-metrics	1/1	1	1	54m
monitoring	simple-app	1/1	1	1	40m
monitoring	soc	1/1	1	1	47m
monitoring	tool	1/1	1	1	46m
monitoring	trivy-operator	1/1	1	1	52m
monitoring	volume-autoscaler	1/1	1	1	9s

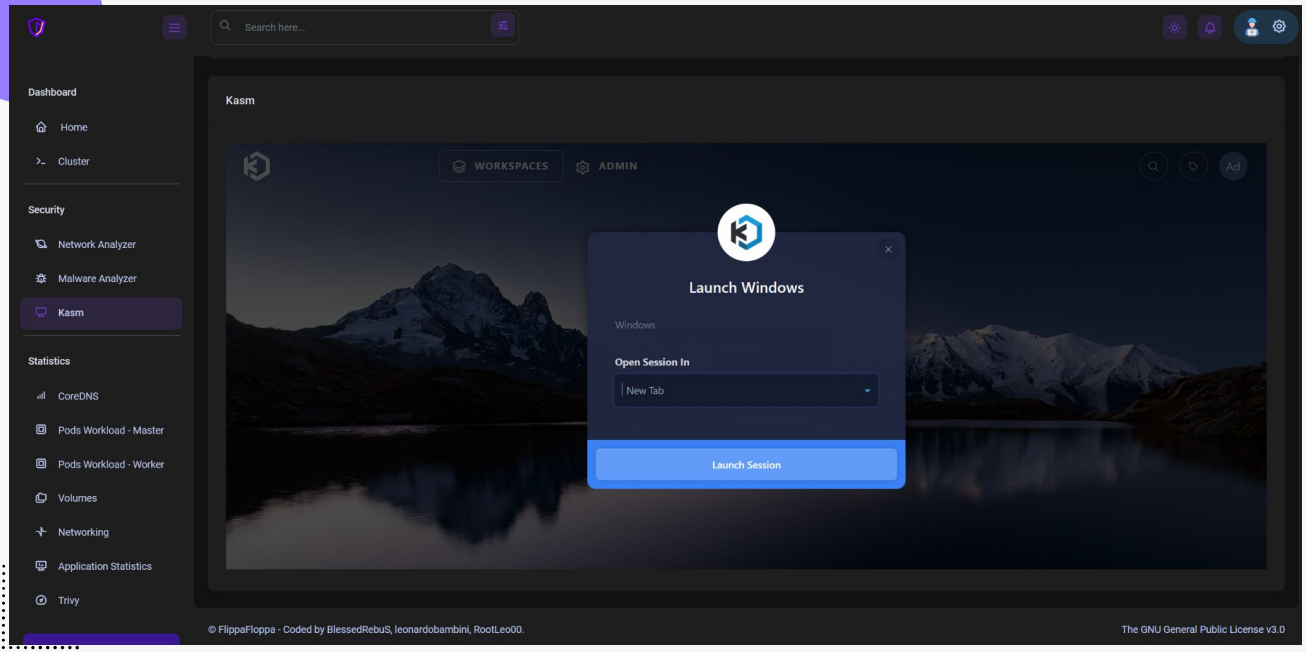
VOLUME & STORAGE 2/2

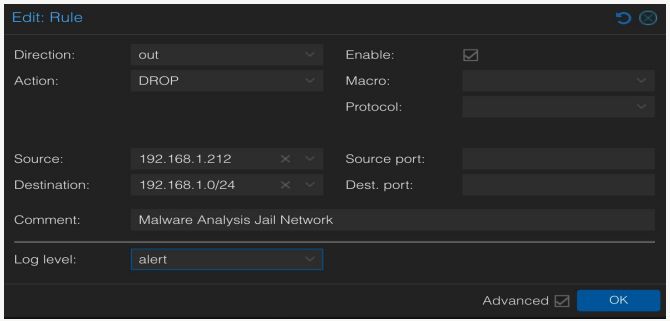
- Il **SOC** fa il mount (centralizza) tutti i volumi
- **Segregazione dei volumi** per questioni di sicurezza
- Richiesta dei dati <u>on demand</u> e non passando via rete

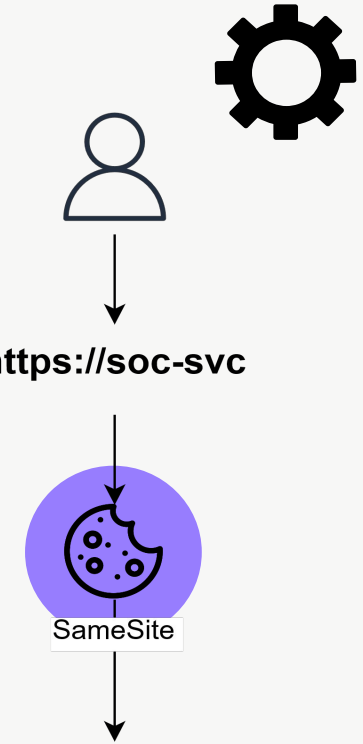



```
    containers:
     - image: nginx
       name: nginx
       ...
     - name: goaccess
       command:
       command: [ "/bin/sh", "-c", "--" ]
       args: [ "while true; do goaccess /var/log/nginx/nginx-access.log
-o /var/log/nginx/result.html --log-format=COMBINED; done;" ]
       image: allinurl/goaccess
       imagePullPolicy: Always
       resources:
         requests:
           cpu: 200m
           memory: 32Mi
       terminationMessagePath: /dev/termination-log
       terminationMessagePolicy: File
       volumeMounts:
       - mountPath: /var/log/nginx
         name: nginx-ingress-pvc
```

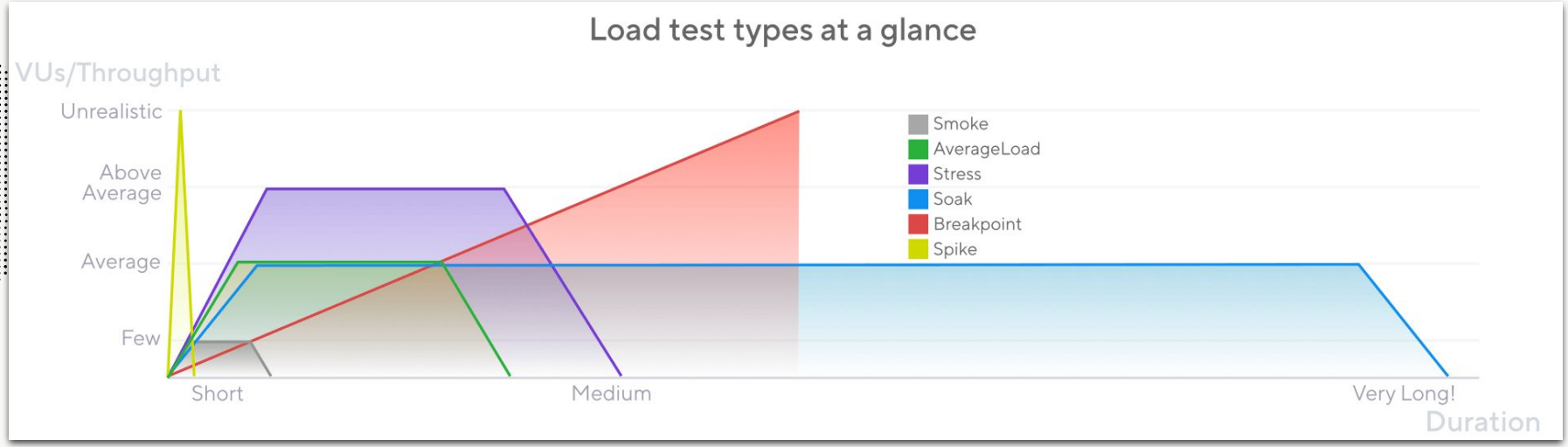
KASM


Search here...
Dashboard
Home
Cluster
Security
Network Analyzer
Malware Analyzer
Kasm
Statistics
CoreDNS
Pods Workload - Master
Pods Workload - Worker
Volumes
Networking
Application Statistics
Trivy
Kasm
WORKSPACES
ADMIN
Launch Windows
Windows
Open Session In
New Tab
Launch Session
© FlippaFloppa - Coded by BlessedRebuS, leonardobambini, RootLeo00.
The GNU General Public License v3.0

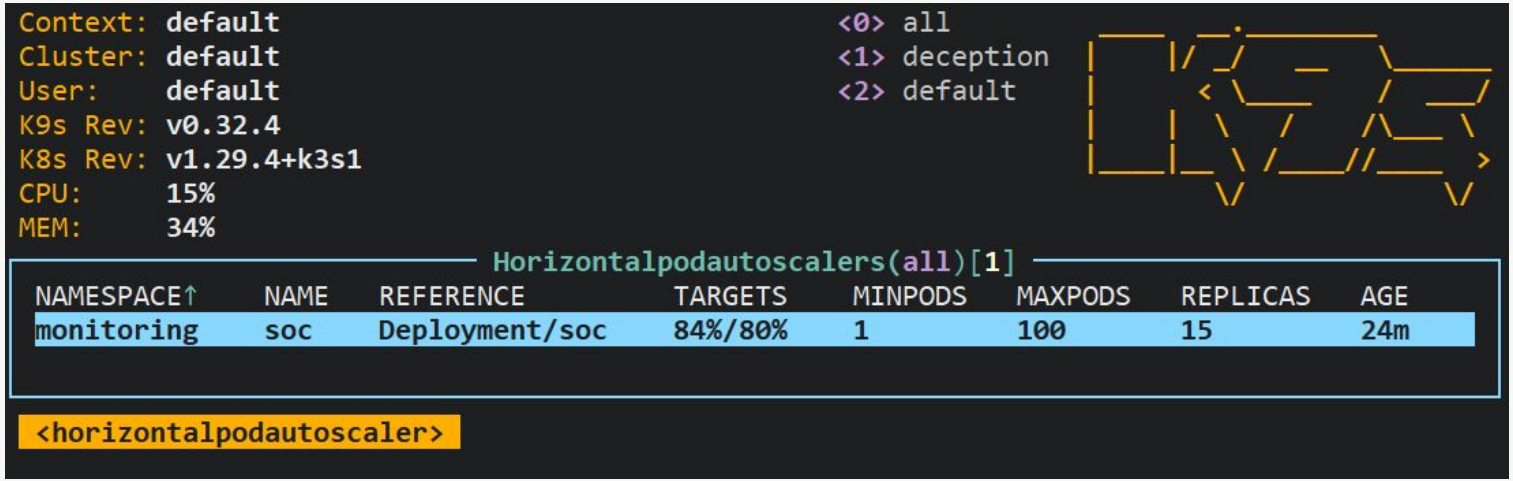

Edit: Rule
Direction: out
Action: DROP
Enable:
Macro:
Protocol:
Source: 192.168.1.212
Destination: 192.168.1.0/24
Source port:
Dest. port:
Comment: Malware Analysis Jail Network
Log level: alert
Advanced
OK


https://soc-svc
SameSite
https://kasm.soc-svc

BENCHMARKS

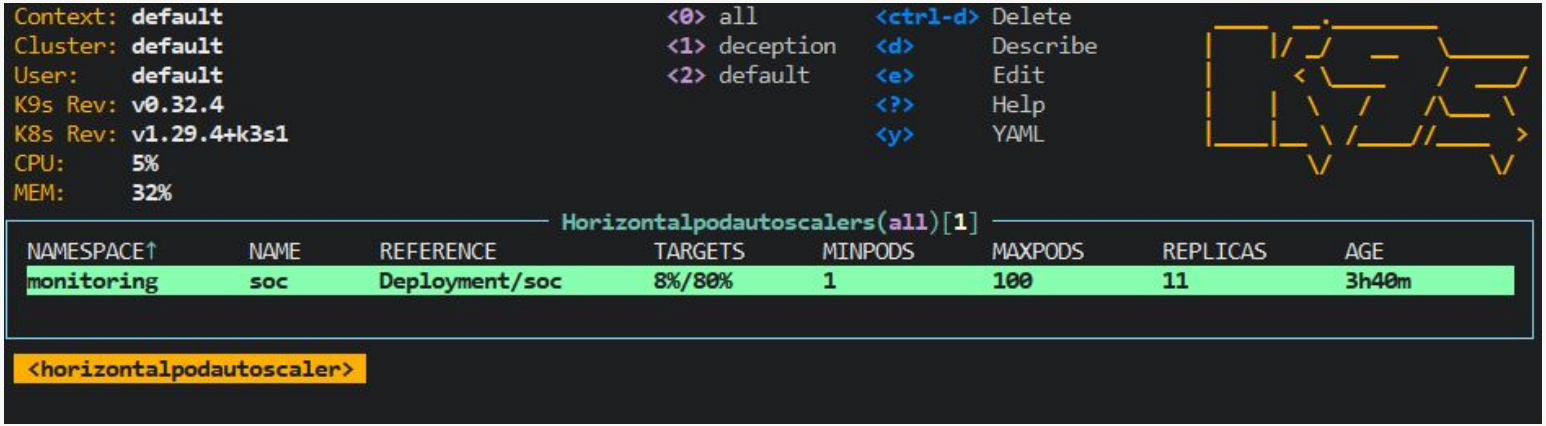

Load test types at a glance
VUs/Throughput
Unrealistic
Above Average
Average
Few
Short
Medium
Very Long!
Duration
Smoke
AverageLoad
Stress
Soak
Breakpoint
Spike

SPIKE TEST



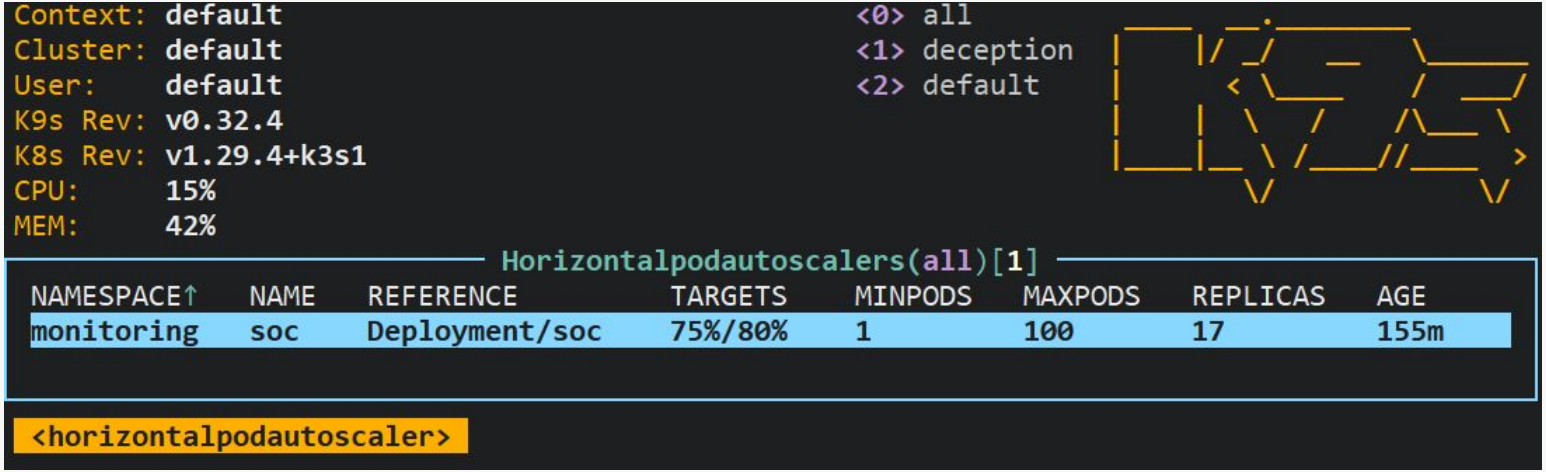

```
stages: [
    { duration: '2m', target: 2000 }, // fast ramp-up to a high point
    { duration: '1m', target: 0 }, // quick ramp-down to 0 users
  ]
```

STRESS TEST




```
stages: [
    { duration: '10m', target: 200 }, // traffic ramp-up from 1 to a higher 200 users over 10 minutes.
    { duration: '30m', target: 200 }, // stay at higher 200 users for 30 minutes
    { duration: '5m', target: 0 }, // ramp-down to 0 users
  ],
```

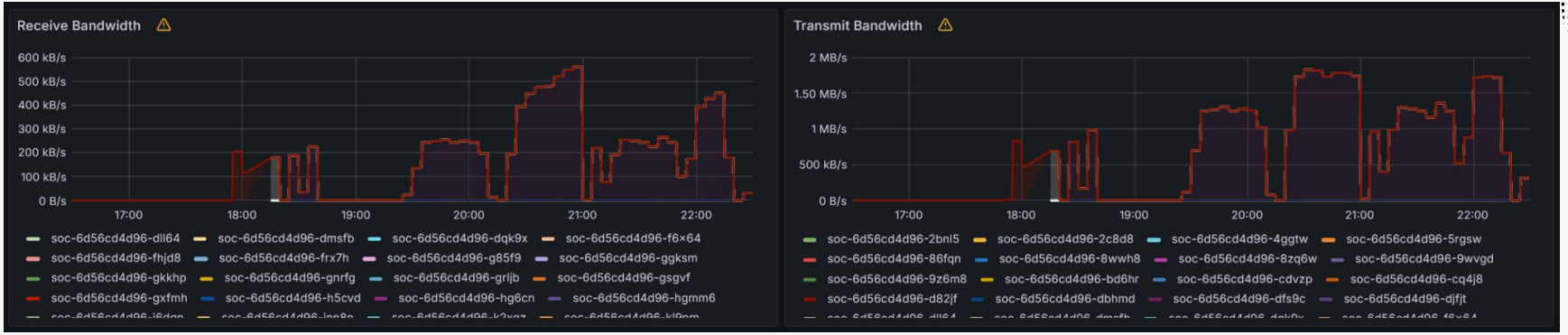
BREAKPOINT TEST

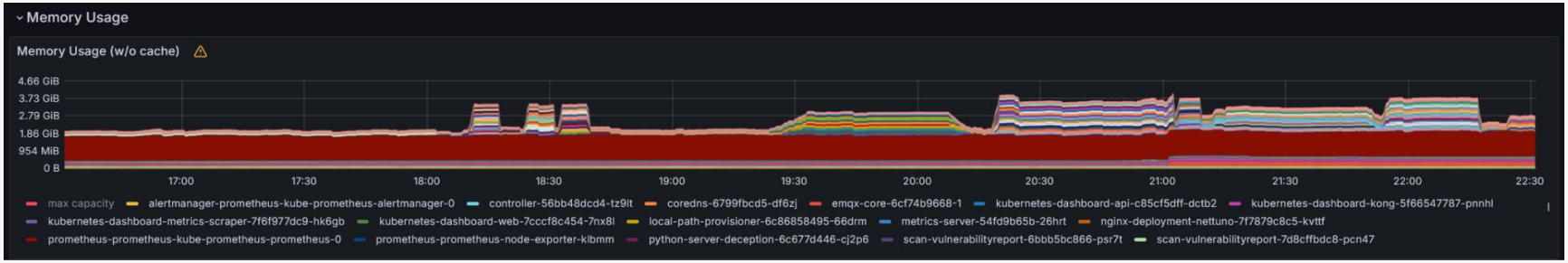



```
stages: [
  { duration: '2h', target: 20000 }, // just slowly ramp-up to a HUGE load
]
```

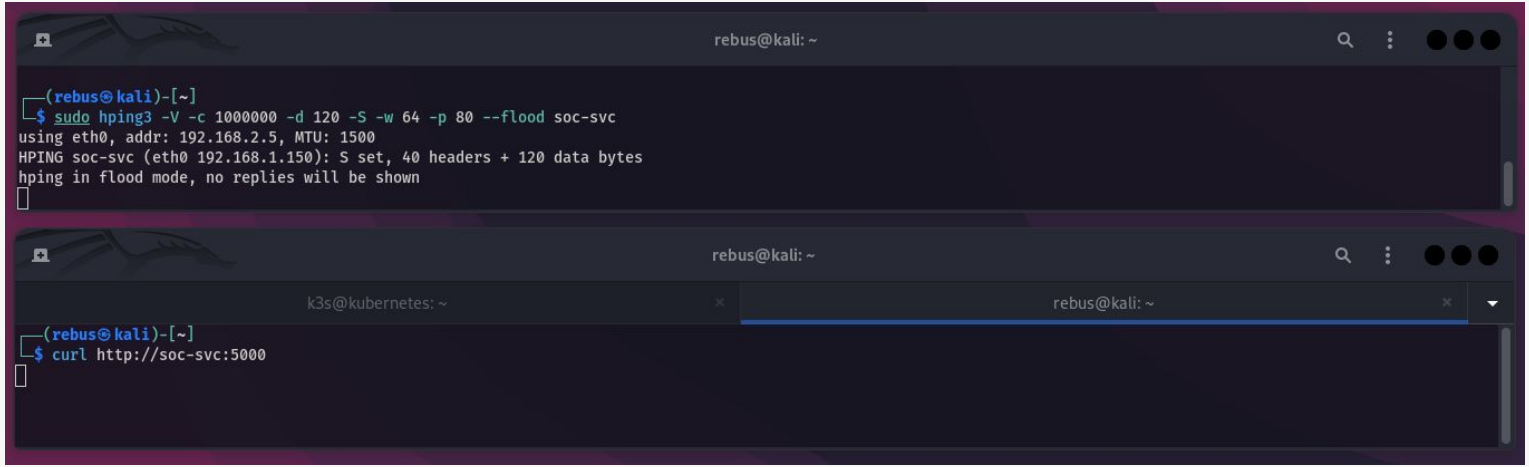
RESOURCES: CPU & MEMORY

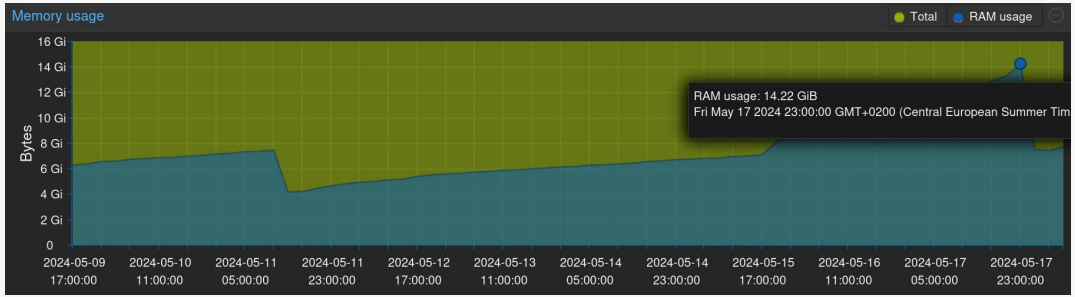
RESOURCES: NETWORK & PODS

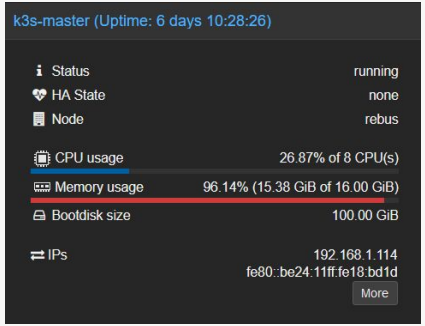




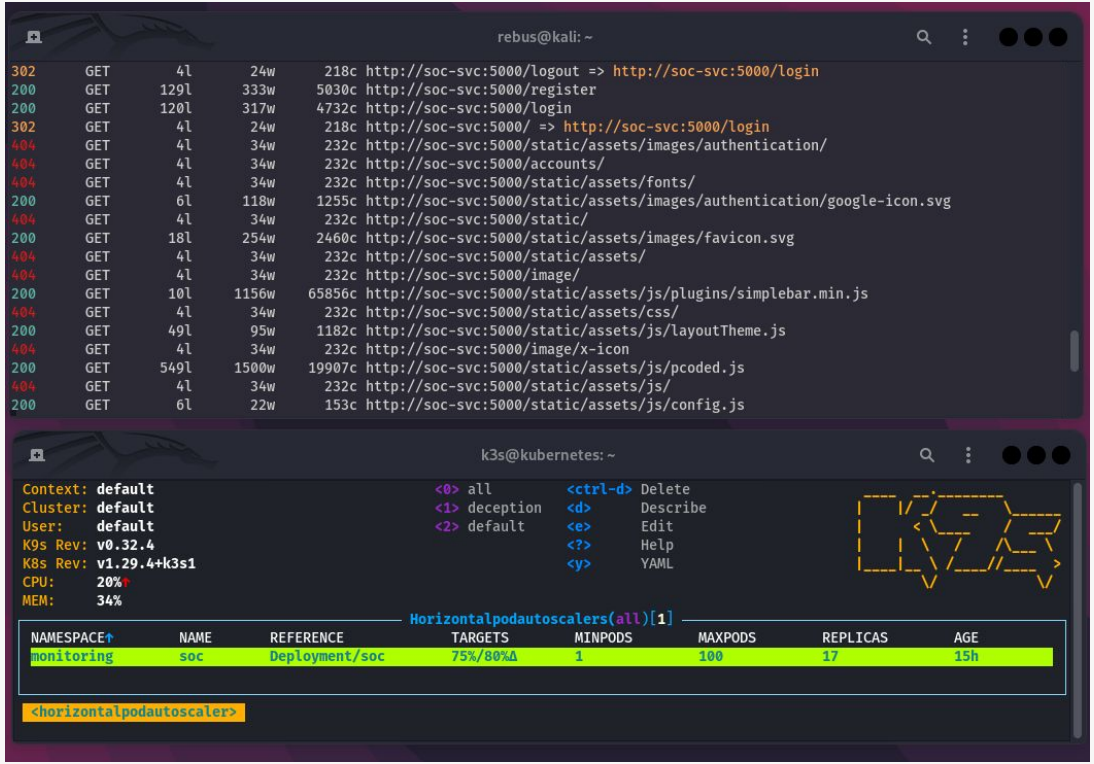
ATTACKS: DoS


rebus@kali: ~
(rebus㉿kali)-[~]
$ sudo hping3 -V -c 1000000 -d 120 -S -w 64 -p 80 --flood soc-svc
using eth0, addr: 192.168.2.5, MTU: 1500
HPING soc-svc (eth0 192.168.1.150): S set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown
rebus@kali: ~
k3s@kubernetes: ~
rebus@kali: ~
(rebus㉿kali)-[~]
$ curl http://soc-svc:5000


Memory usage
Total
RAM usage
16 Gi
14 Gi
12 Gi
10 Gi
8 Gi
6 Gi
4 Gi
2 Gi
0
Bytes
RAM usage: 14.22 GiB
Fri May 17 2024 23:00:00 GMT+0200 (Central European Summer Tim
2024-05-09 17:00:00
2024-05-10 11:00:00
2024-05-11 05:00:00
2024-05-11 23:00:00
2024-05-12 17:00:00
2024-05-13 11:00:00
2024-05-14 05:00:00
2024-05-14 23:00:00
2024-05-15 17:00:00
2024-05-16 11:00:00
2024-05-17 05:00:00
2024-05-17 23:00:00


k3s-master (Uptime: 6 days 10:28:26)
Status
running
HA State
none
Node
rebus
CPU usage
26.87% of 8 CPU(s)
Memory usage
96.14% (15.38 GiB of 16.00 GiB)
Bootdisk size
100.00 GiB
IPs
192.168.1.114
fe80::be24:11ff:fe18:bd1d
More

ATTACKS: Path Traversal

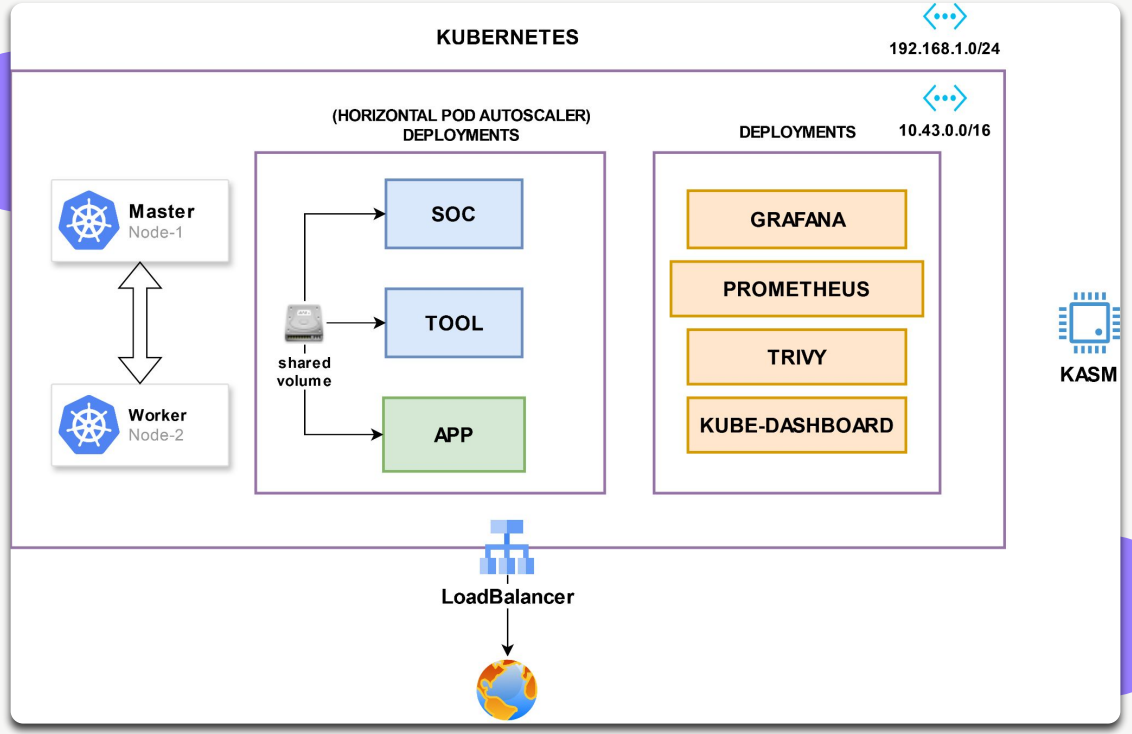


Risultati:

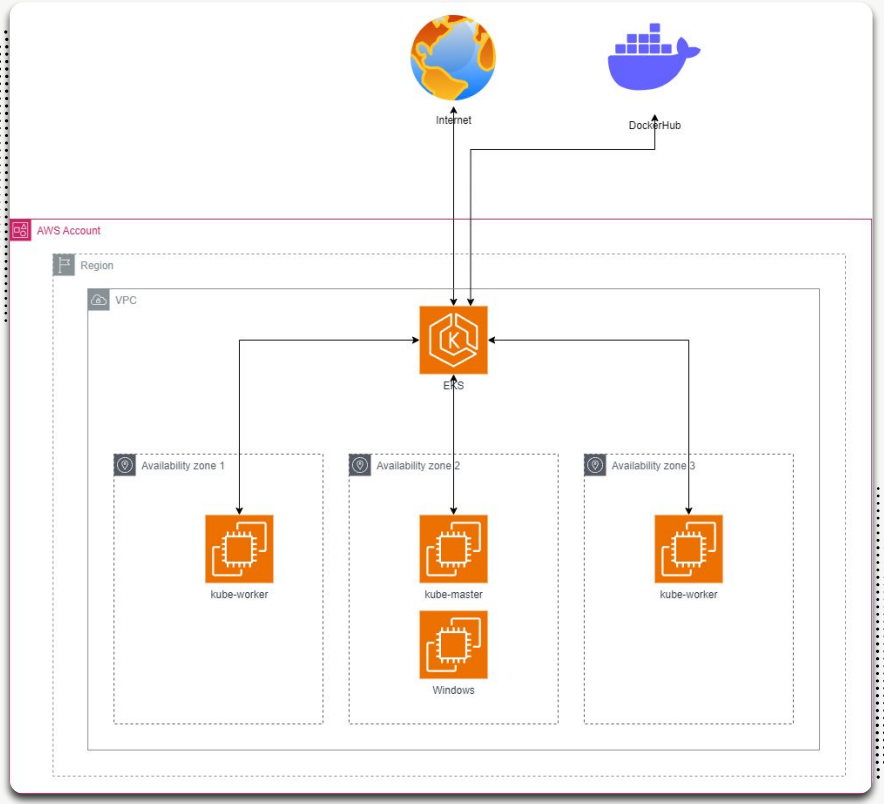
- Limite di 17 Pods per MS
- 1 Nodo con **16 GB** Ram, CPU 12 core **3.7 GHz**
- NIC 1 **Gb/s**

```
requests:
  memory: "64Mi"
  cpu: "100m"
```

DIAGRAMMA INFRASTRUTTURA



DISEGNO ARCHITETTURALE (AWS)



AUTOSCALING: NODI

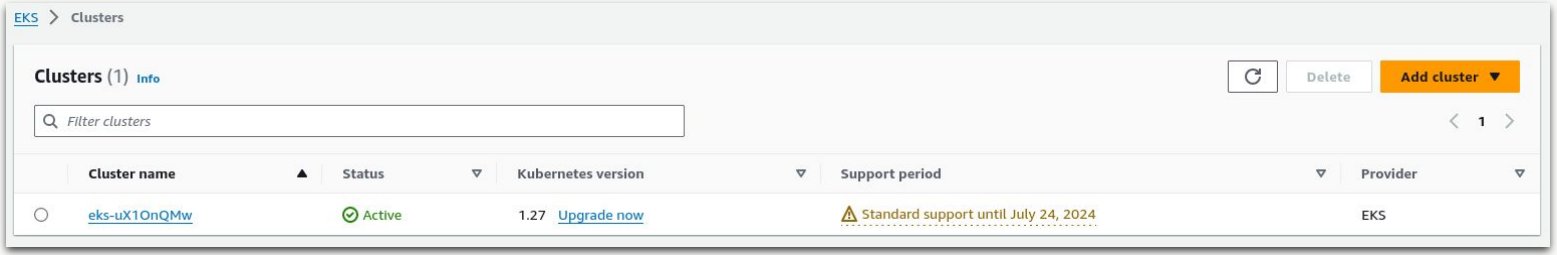
EKS offre lo scaling orizzontale dei nodi master e worker

Autoscaling basato su metriche (es: CPU media utilizzata)

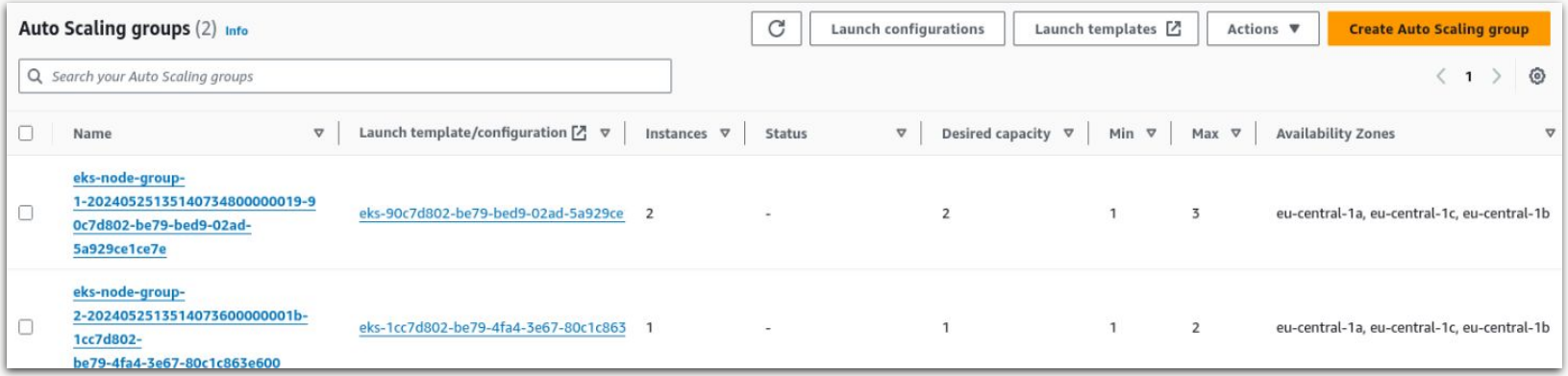
Nodi aggiunti o rimossi quando si supera una certa soglia di una certa metrica

Definizione di **minimi e massimi** desiderati

Efficienza economica grazie agli sprechi limitati



AUTOSCALING: NODI



AUTOSCALING: VOLUMI

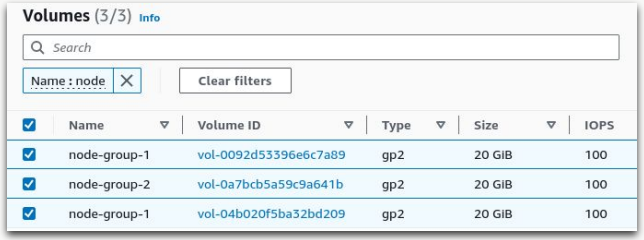
EBS come fonte di storage per Kubernetes

Driver CSI per connettere Kubernetes a EBS

Scaling automatico quando si raggiunge una certa quantità di spazio utilizzato

```
NAME              PROVISIONER             RECLAIMPOLICY   VOLUMEBINDINGMODE      ALLOWVOLUMEEXPANSION   AGE
gp2 (default)     kubernetes.io/aws-ebs   Delete          WaitForFirstConsumer   false                  82m
gp3 (default)     ebs.csi.aws.com         Retain          WaitForFirstConsumer   true                   6m5s
```

```
NAMESPACE     NAME                                    READY   UP-TO-DATE   AVAILABLE   AGE
kube-system   coredns                                 2/2     2            2           79m
kube-system   ebs-csi-controller                      2/2     2            2           77m
monitoring    prometheus-grafana                      1/1     1            1           54m
monitoring    prometheus-kube-prometheus-operator     1/1     1            1           54m
monitoring    prometheus-kube-state-metrics           1/1     1            1           54m
monitoring    simple-app                              1/1     1            1           40m
monitoring    soc                                     1/1     1            1           47m
monitoring    tool                                    1/1     1            1           46m
monitoring    trivy-operator                          1/1     1            1           52m
monitoring    volume-autoscaler                       1/1     1            1           9s
```



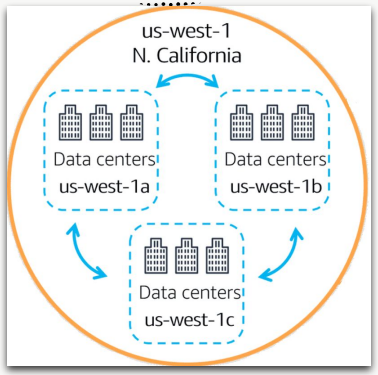
https://github.com/DevOps-Nirvana/Kubernetes-Volume-Autoscaler

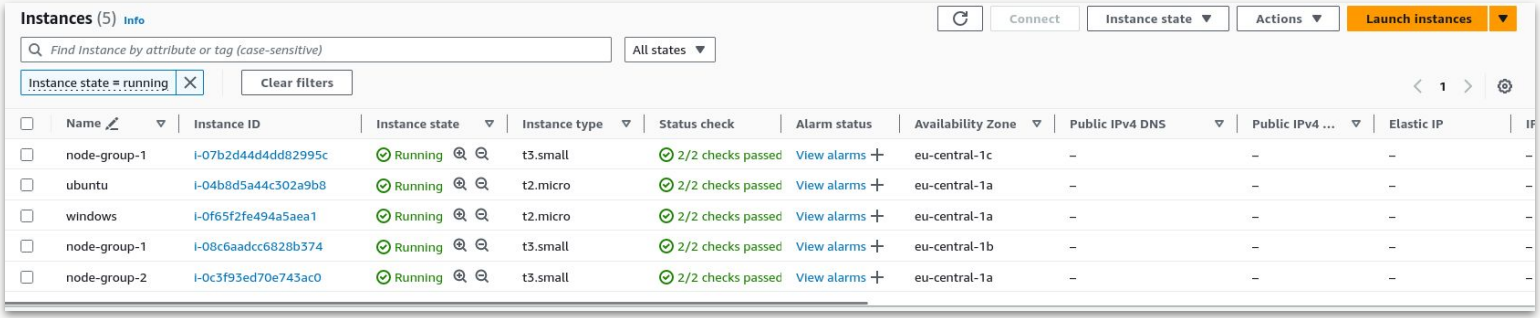
ALTA AFFIDABILITÀ: NODI

Separazione dei nodi in più **Availability Zones**

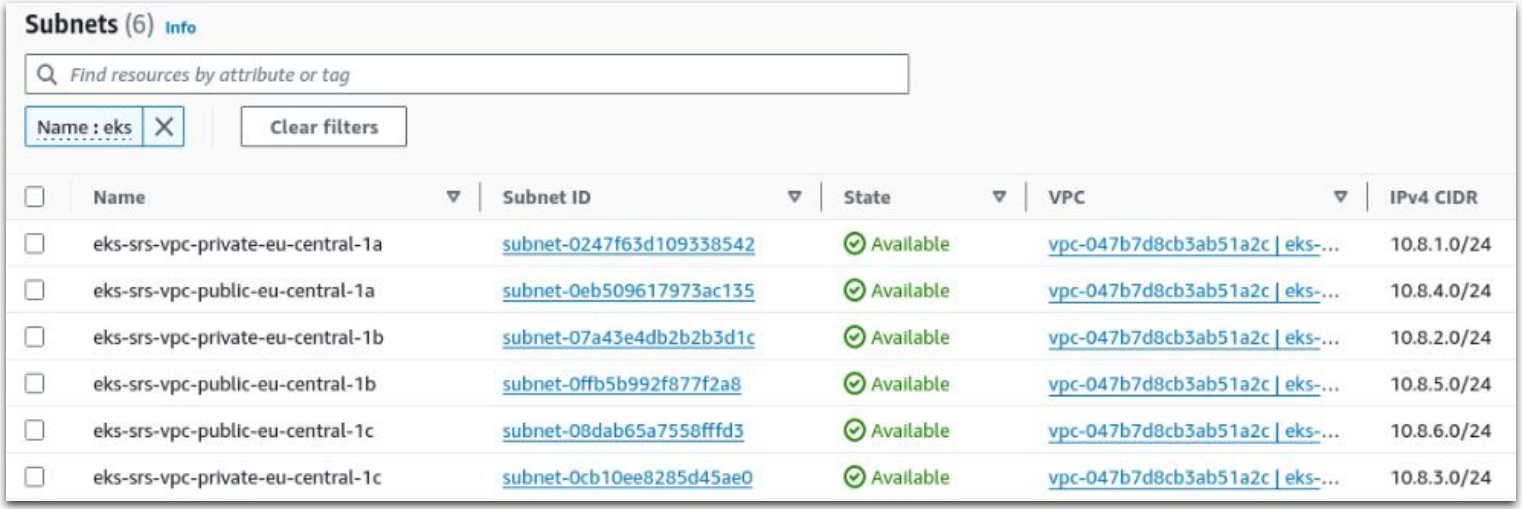
Availability Zone e Region: fault domain diversi

Separazione su più region non possibile con EKS





ALTA AFFIDABILITÀ: RETE



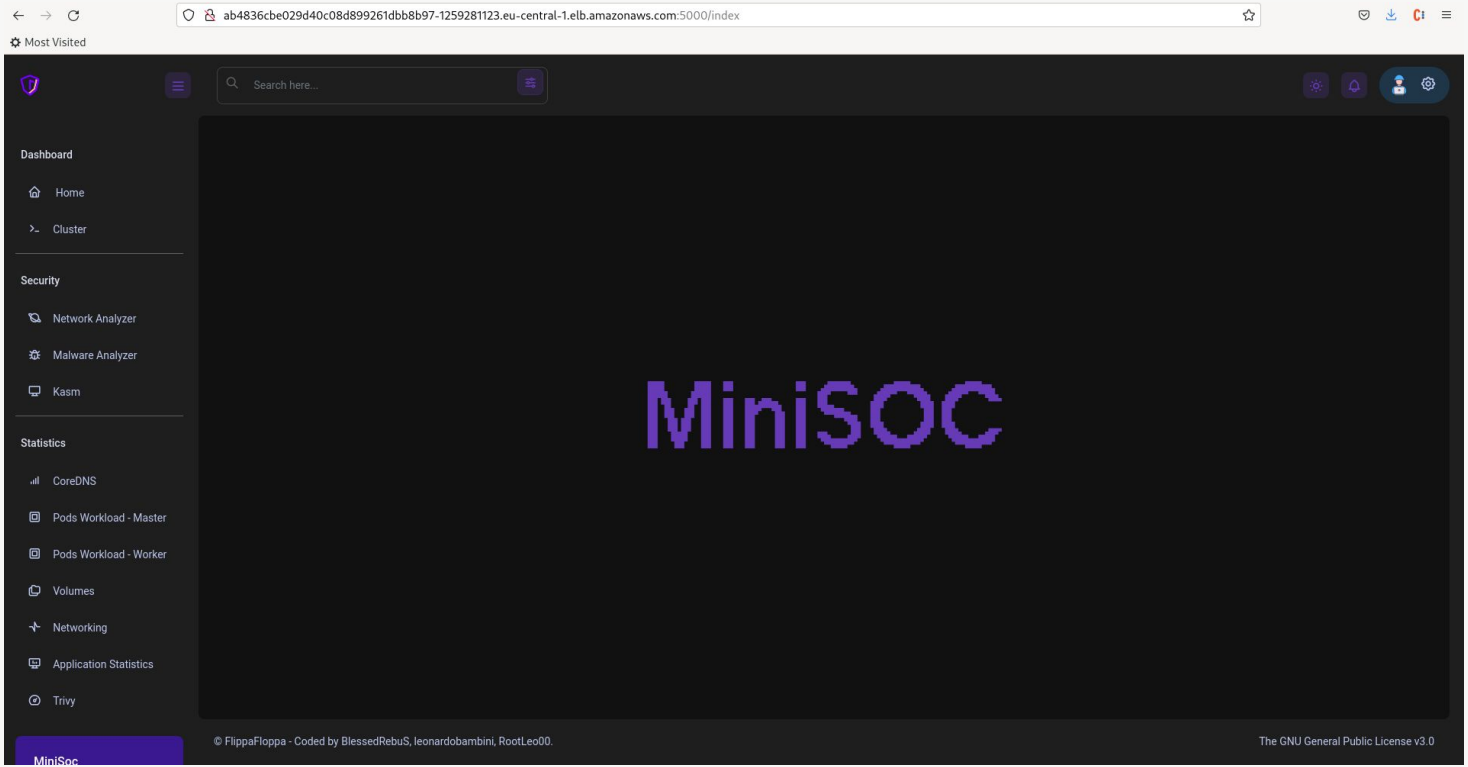
Subnets (6) Info

Find resources by attribute or tag

Name : eks | Clear filters

Name	Subnet ID	State	VPC	IPv4 CIDR
eks-srs-vpc-private-eu-central-1a	subnet-0247f63d109338542	Available	vpc-047b7d8cb3ab51a2c | eks-...	10.8.1.0/24
eks-srs-vpc-public-eu-central-1a	subnet-0eb509617973ac135	Available	vpc-047b7d8cb3ab51a2c | eks-...	10.8.4.0/24
eks-srs-vpc-private-eu-central-1b	subnet-07a43e4db2b2b3d1c	Available	vpc-047b7d8cb3ab51a2c | eks-...	10.8.2.0/24
eks-srs-vpc-public-eu-central-1b	subnet-0ffb5b992f877f2a8	Available	vpc-047b7d8cb3ab51a2c | eks-...	10.8.5.0/24
eks-srs-vpc-public-eu-central-1c	subnet-08dab65a7558fffd3	Available	vpc-047b7d8cb3ab51a2c | eks-...	10.8.6.0/24
eks-srs-vpc-private-eu-central-1c	subnet-0cb10ee8285d45ae0	Available	vpc-047b7d8cb3ab51a2c | eks-...	10.8.3.0/24

MINISOC on AWS



DEMO

https://github.com/FlippaFloppa/MINISOC

GRAZIE PER L'ATTENZIONE

Bambini Leonardo
Leonelli Caterina
Di Fazio Patrick