

Создание и применение сетевой политики к рабочей нагрузке в кластере

--

Для начала работ вам понадобится:

- *docker*
- *k3d*

Подробная установка docker рассматривалась в методических материалах к предыдущим урокам

--

Часть 1. Создание кластера и запуск рабочей нагрузки в нем

Создадим кластер k3d:

```
$ k3d cluster create mycluster
```

```
filipp@filipp-notebook:~/Desktop$ k3d cluster create mycluster
INFO[0000] Prep: Network
INFO[0000] Created network 'k3d-mycluster'
INFO[0000] Created volume 'k3d-mycluster-images'
INFO[0000] Starting new tools node...
INFO[0000] Starting Node 'k3d-mycluster-tools'
INFO[0001] Creating node 'k3d-mycluster-server-0'
INFO[0001] Creating LoadBalancer 'k3d-mycluster-serverlb'
INFO[0001] Using the k3d-tools node to gather environment information
INFO[0001] HostIP: using network gateway 172.19.0.1 address
INFO[0001] Starting cluster 'mycluster'
INFO[0001] Starting servers...
INFO[0001] Starting Node 'k3d-mycluster-server-0'
INFO[0005] All agents already running.
INFO[0005] Starting helpers...
INFO[0005] Starting Node 'k3d-mycluster-serverlb'
INFO[0012] Injecting '172.19.0.1 host.k3d.internal' into /etc/hosts of all nodes...
INFO[0012] Injecting records for host.k3d.internal and for 2 network members into CoreDNS
.
INFO[0013] Cluster 'mycluster' created successfully!
INFO[0013] You can now use it like this:
kubectl cluster-info
```

Создадим неймспейс для инструмента mobsf:

```
$ kubectl create ns mobsf
```

```
filipp@filipp-notebook:~/Desktop$ kubectl create ns mobsf
namespace/mobsf created
```

Выполним создание деплоймента mobsf:

```
$ kubectl create deployment mobsf
--image=opensecurity/mobile-security-framework-
mobsf:v3.1.1 -n mobsf
```

```
filipp@filipp-notebook:~/Desktop$ kubectl create deployment mobsf --image=opensecurity/mobile-security-framework-mobsf:v3.1.1 -n mobsf
deployment.apps/mobsf created
```

Создадим службу для деплоймента mobsf:

```
$ kubectl expose deployment mobsf --port=8000 --
type=NodePort -n mobsf
```

```
filipp@filipp-notebook:~/Desktop$ kubectl expose deployment mobsf --port=8000 --type=NodePort -n mobsf
service/mobsf exposed
```

Убедимся, что pod с инструментом mobsf успешно запущен:

```
$ kubectl get pod -n mobsf
```

```
filipp@filipp-notebook:~/Desktop$ kubectl get pod -n mobsf
NAME                                READY   STATUS    RESTARTS   AGE
mobsf-5db69649fc-mwf4q             1/1     Running   0           5m55s
```

Запросим external-ip у службы traefik в кластере и NodePort у службы mobsf:

```
$ kubectl get svc -n kube-system
$ kubectl get svc -n mobsf
```

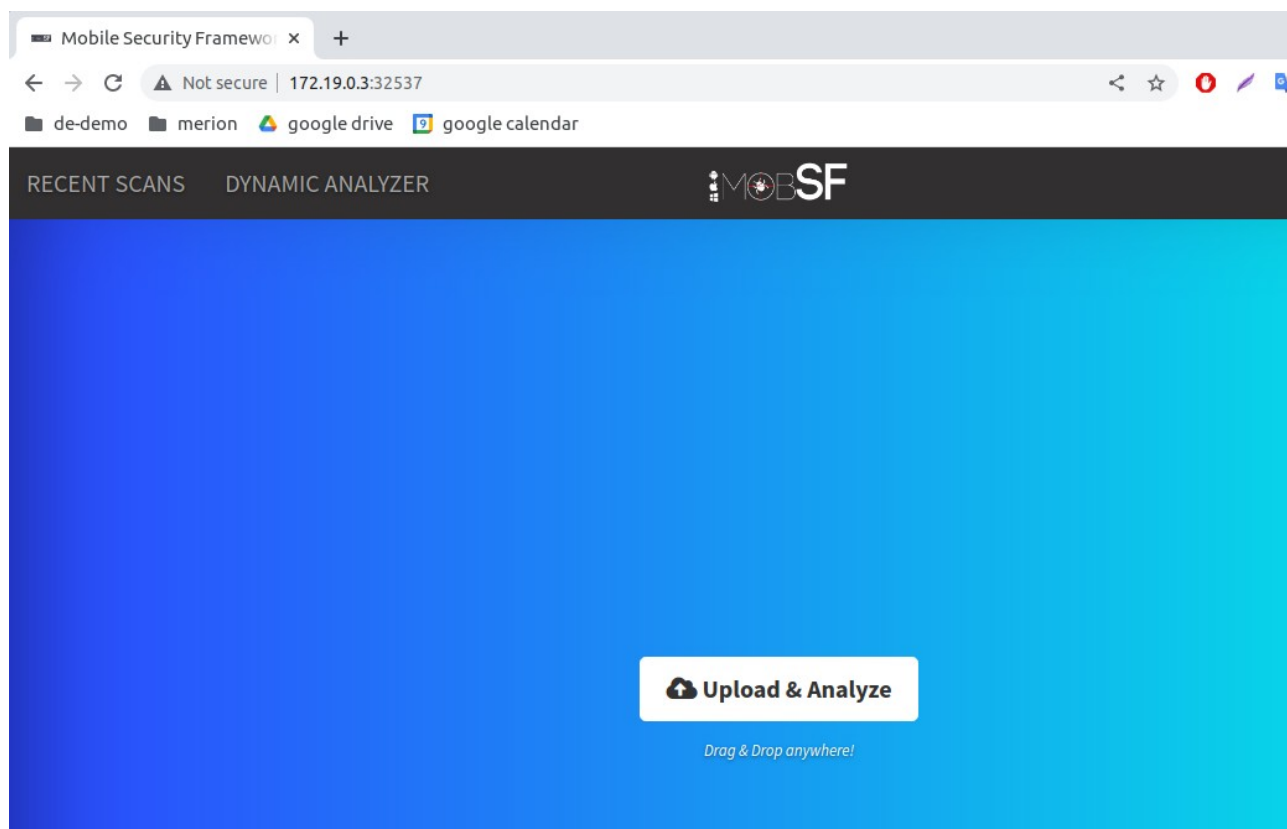
```

filipp@filipp-notebook:~/Desktop$ kubectl get svc -n kube-system
NAME                TYPE                CLUSTER-IP      EXTERNAL-IP      PORT(S)                                     AGE
kube-dns            ClusterIP           10.43.0.10      <none>           53/UDP,53/TCP,9153/TCP                    7m14s
metrics-server      ClusterIP           10.43.135.45    <none>           443/TCP                                    7m13s
traefik             LoadBalancer       10.43.15.181    172.19.0.3       80:32496/TCP,443:30728/TCP                5m11s
filipp@filipp-notebook:~/Desktop$ kubectl get svc -n mobsf
NAME    TYPE        CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
mobsf   NodePort    10.43.42.94    <none>         8000:32537/TCP   3m50s

```

Воспользуемся браузером и перейдем по адресу веб-интерфейса mobsf:

browser: `http://<traefik svc external-ip>:<mobsf svc nodeport>`



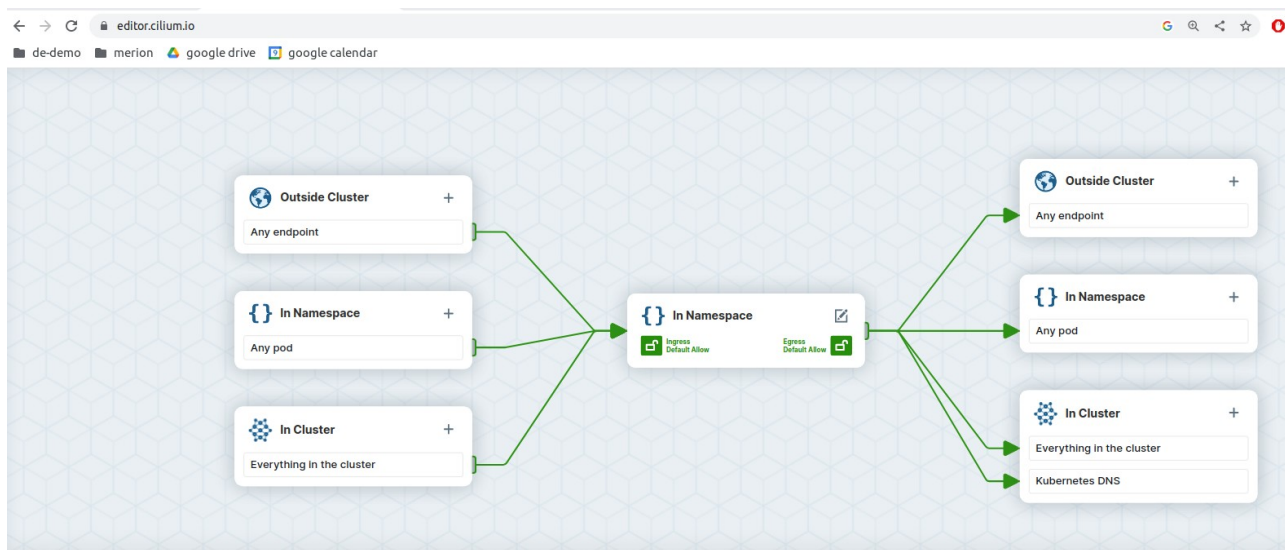
Веб-интерфейс инструмента mobsf доступен

--

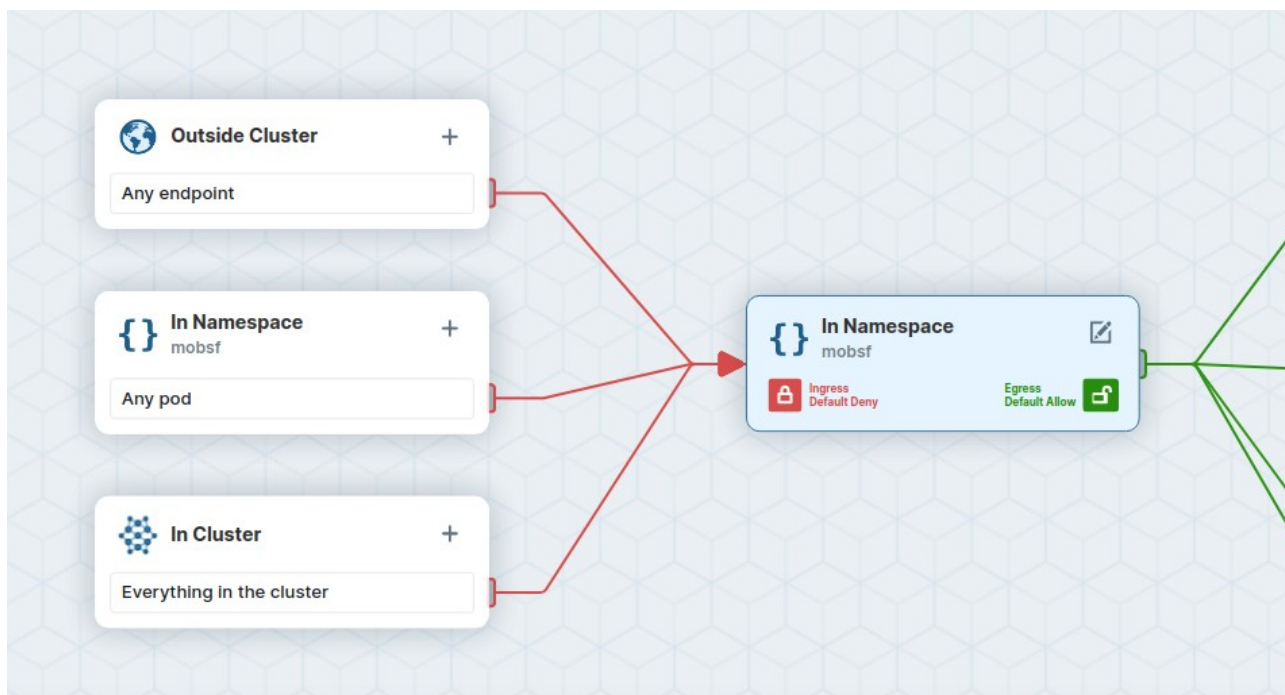
Часть 2. Создание и применение сетевой политики в отношении рабочей нагрузки в кластере

Воспользуемся онлайн редактором сетевых политик от сообщества cilium:

browser: <https://editor.cilium.io/>



Укажем неймспейс `mobsf` в объекте "In Namespace" и запретим входящие соединения (Ingress Default Deny)



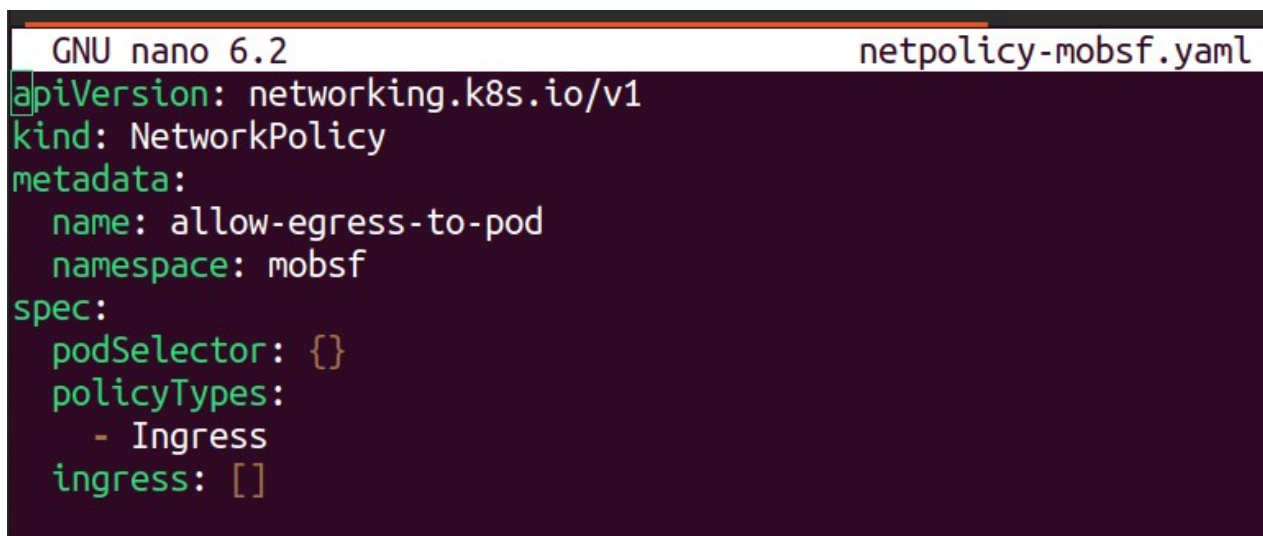
Сгенерирования сетевая политика должна иметь следующий вид:

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-egress-to-pod
```

```
namespace: mobsf
spec:
  podSelector: {}
  policyTypes:
    - Ingress
  ingress: []
```

Вернемся в терминал и создадим манифест сетевой политики, добавив в него сгенерированное в cilium editor содержимое:

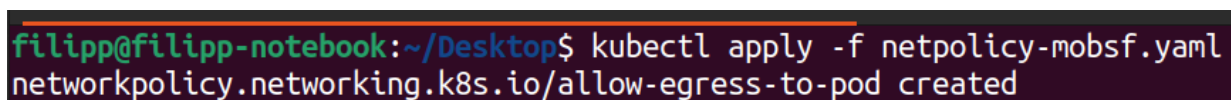
```
$ nano netpolicy-mobsf.yaml
```



```
GNU nano 6.2 netpolicy-mobsf.yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-egress-to-pod
  namespace: mobsf
spec:
  podSelector: {}
  policyTypes:
    - Ingress
  ingress: []
```

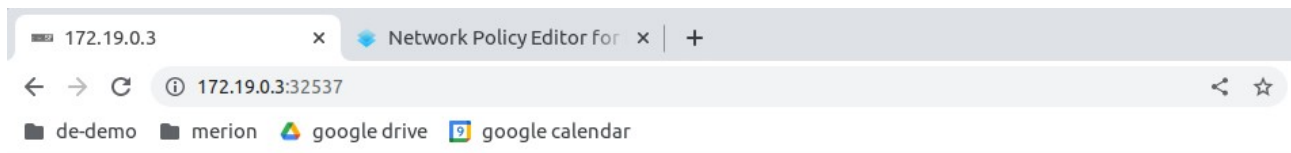
Применим манифест сетевой политики в кластере:

```
$ kubectl apply -f netpolicy-mobsf.yaml
```



```
filipp@filipp-notebook:~/Desktop$ kubectl apply -f netpolicy-mobsf.yaml
networkpolicy.networking.k8s.io/allow-egress-to-pod created
```

Перейдем снова в браузер и обновим страницу с веб-интерфейсом инструмента mobsf:



This site can't be reached

172.19.0.3 refused to connect.

Try:

- Checking the connection
- [Checking the proxy and the firewall](#)

ERR_CONNECTION_REFUSED

Details

Reload

Сетевая политика в отношении рабочей нагрузки `mobsf` была успешно применена в кластере - обработка входящих соединений для этого инструмента запрещена, что подтверждается невозможностью отобразить веб-интерфейс

--