AccuKnox Demo

Runtime Security for Complex Applications

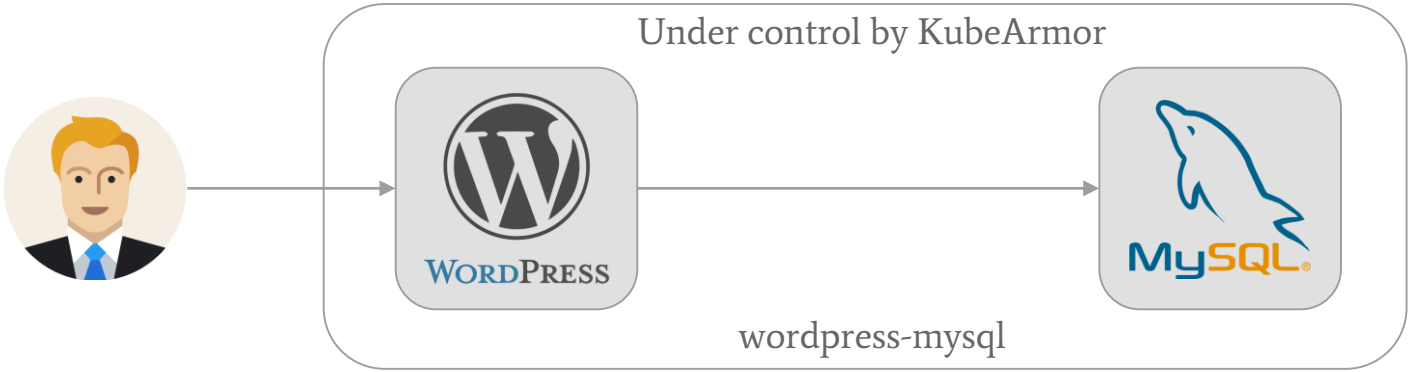
Using KubeArmor

Demo Scenario

3. Block the access of Kubernetes API server

4. Block the execution of specific processes

K8s service account mounted by default



WP credentials: /var/www/wp-config.php

2. Block any credential accesses by unknown processes

Database path: /var/lib/mysql/

1. Audit all access in this path

KubeArmor Demo Policies

```yaml
apiVersion: security.accuknox.com/v1
kind: KubeArmorPolicy
metadata:
  name: ksp-mysql-dir-audit
  namespace: wordpress-mysql
spec:
  selector:
    matchLabels:
      app: mysql
  file:
    matchDirectories:
    - dir: /var/lib/mysql/
      recursive: true
  action:
    Audit
  severity: 1
```

```yaml
apiVersion: security.accuknox.com/v1
kind: KubeArmorPolicy
metadata:
  name: ksp-wordpress-process-block
  namespace: wordpress-mysql
spec:
  severity: 3
  selector:
    matchLabels:
      app: wordpress
  process:
    matchPaths:
    - path: /usr/bin/apt
    - path: /usr/bin/apt-get
  action:
    Block
```

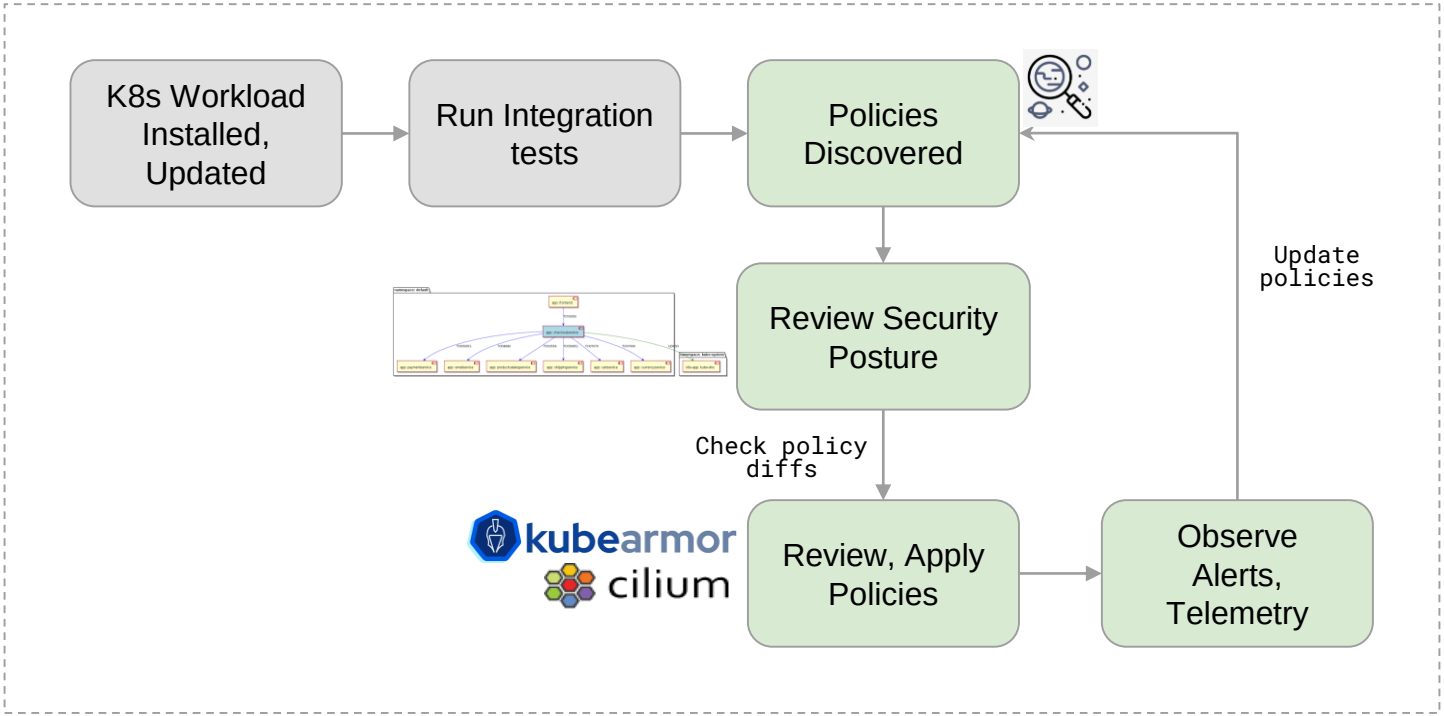
```yaml
apiVersion: security.accuknox.com/v1
kind: KubeArmorPolicy
metadata:
  name: ksp-wordpress-config-block
  namespace: wordpress-mysql
spec:
  severity: 10
  selector:
    matchLabels:
      app: wordpress
  file:
    matchPaths:
    - path: /var/www/html/wp-config.php
      fromSource:
        path: /bin/cat

  # cd /var/www/html
  # cat wp-config.php

  action:
    Block
```

```yaml
apiVersion: security.accuknox.com/v1
kind: KubeArmorPolicy
metadata:
  name: ksp-wordpress-sa-block
  namespace: wordpress-mysql
spec:
  severity: 7
  selector:
    matchLabels:
      app: wordpress
  file:
    matchDirectories:
    - dir: /run/secrets/kubernetes.io/serviceaccount/
      recursive: true

# cat /run/secrets/kubernetes.io/serviceaccount/token
# curl https://$KUBERNETES_PORT_443_TCP_ADDR/api --insecure --header \
  "Authorization: Bearer $(cat /run/secrets/kubernetes.io/serviceaccount/token)"

  action:
    Block
```

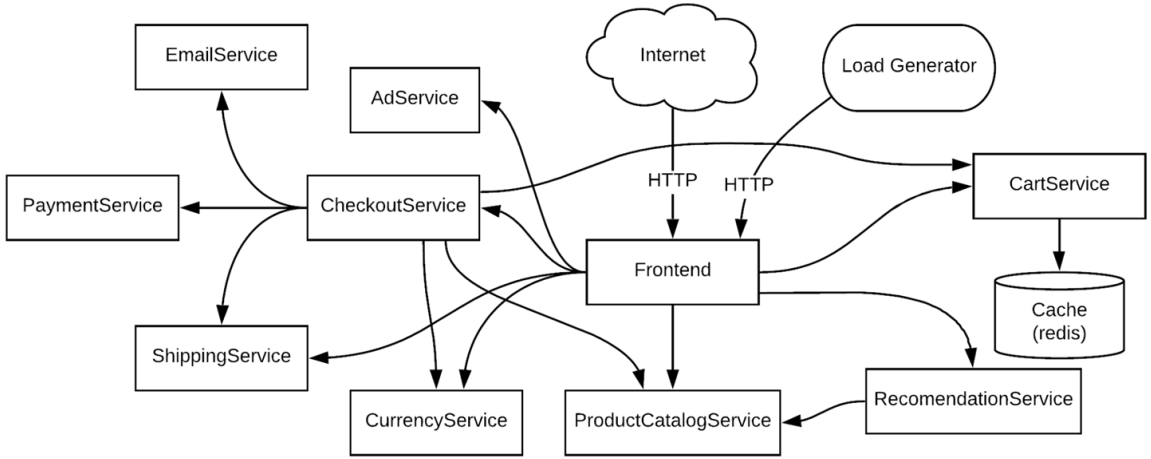
MITRE | ATT&CK®

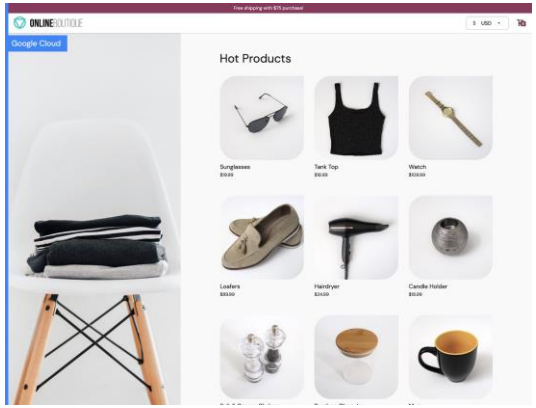
Lateral Movement	Credential Access	Execution
Access cloud resources	App credentials in config files	bash/cmd inside container
App credentials in config files	Access container service account	

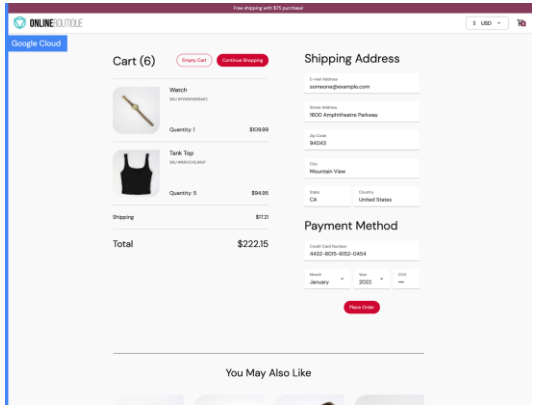
LifeCycle


K8s Workload Installed, Updated
Run Integration tests
Policies Discovered
Review Security Posture
Check policy diffs
Review, Apply Policies
Observe Alerts, Telemetry
Update policies
kubearmor
cilium

Demo Application: Google Microservice Demo

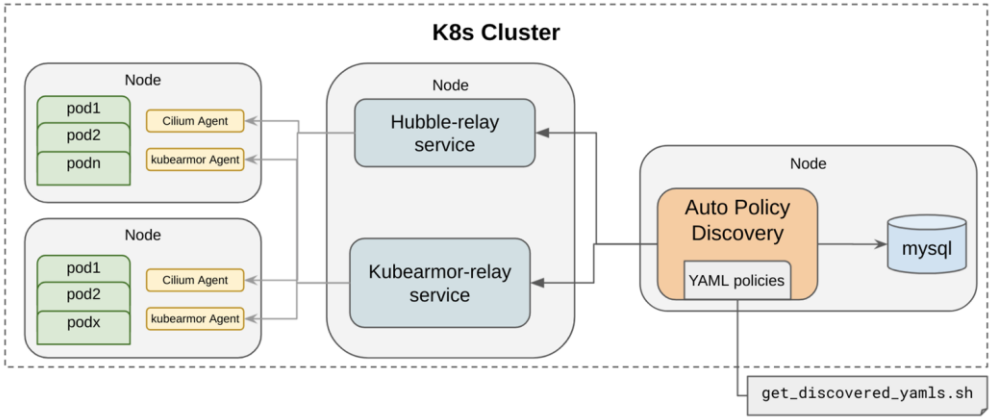






- Online Boutique Ecommerce application
- Internal Load Generator that generates traffic

Accuknox Daemonsets and services



- KubeArmor Daemonset
- Cilium Daemonset
- Relay services
- Auto Policy Discovery service

NAMESPACE	NAME	READY	STATUS
kube-system	coredns-96cc4f57d-xw7j6	1/1	Running
kube-system	hubble-relay-57bb755b8-6kj79	1/1	Running
default	recommendationservice-77bdd78d96-hl968	1/1	Running
default	svclb-frontend-external-knh8d	1/1	Running
default	currencyservice-b89649bf6-ct8mf	1/1	Running
default	shippingservice-5b887b455b-l9zdd	1/1	Running
default	frontend-6dd766ff95-j28bt	1/1	Running
kube-system	cilium-operator-67df8d4fc7-w8g7c	1/1	Running
default	cartservice-5dff477f54-cpzrk	1/1	Running
default	loadgenerator-7fb546d89-qghd6	1/1	Running
default	paymentservice-8657cc6794-rhtq6	1/1	Running
default	redis-cart-77d5f5577-k8n25	1/1	Running
kube-system	local-path-provisioner-5bd75fdd7f-9r8qh	1/1	Running
default	checkoutservice-8f859666-qjp82	1/1	Running
explorer	knoxautopolicy-6fbf6f6c76-tvhph	1/1	Running
default	emailservice-7b4d9776-679kk	1/1	Running
default	productcatalogservice-64cc47d648-pn2f5	1/1	Running
default	adservice-ccfc858d4-mk5wr	1/1	Running
explorer	mysql-0	1/1	Running
kube-system	metrics-server-ff9dbcb6c-94hfk	1/1	Running
kube-system	cilium-4rnq6	1/1	Running
kube-system	kubearmor-4lfbs	1/1	Running
kube-system	kubearmor-policy-manager-54ffc4dc56-x2l5q	2/2	Running
kube-system	kubearmor-host-policy-manager-766447b4d7-gtth8	2/2	Running
kube-system	kubearmor-relay-645667c695-6kglt	1/1	Running
default	attacker-7f99cbd557-pjgkg	1/1	Running

Retrieve Auto Discovered Policies

- Retrieve discovered policies from auto-policy-discovery service
  - `get_discovered_yamls.sh`: sends a GRPC request to fetch discovered policies
- KubeArmor Policies
  - Yaml policy per deployment
- Cilium Network Policy
  - Single yaml across the cluster
  - *filter-policy* tool to split up on per deployment basis, if needed

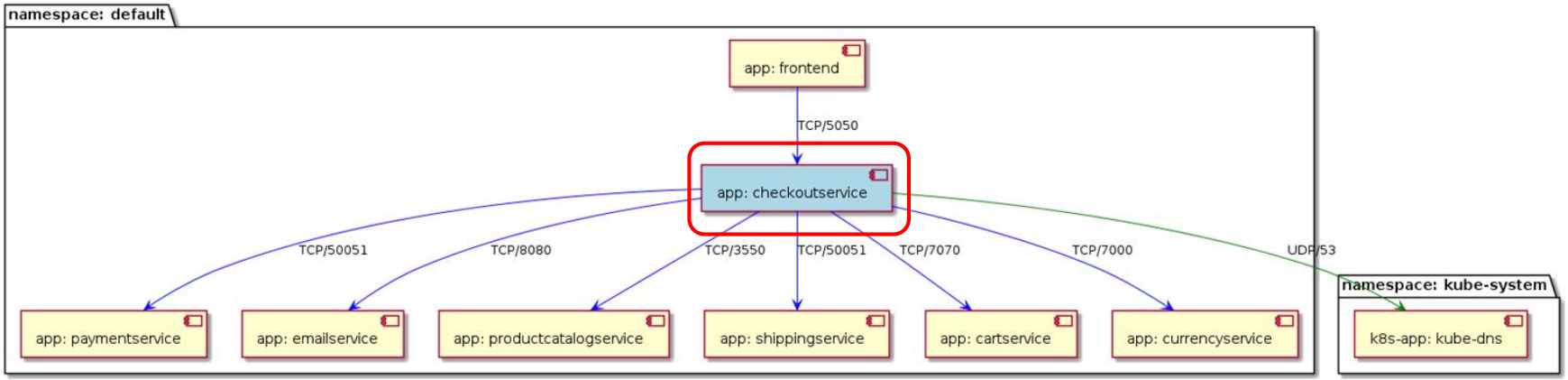
```
❯ ./get_discovered_yamls.sh
{
  "res": "ok"
}
Got 161 cilium policies in file cilium_policies.yaml
{
  "res": "ok"
}
Got 1 kubearmor policies in file kubearmor_policies_default_default__uorfjumc.yaml
Got 1 kubearmor policies in file kubearmor_policies_default_default_attacker_gtikozjp.yaml
Got 1 kubearmor policies in file kubearmor_policies_default_default_kabuntu_sooljeye.yaml
Got 1 kubearmor policies in file kubearmor_policies_default_default_main_lkoehrsg.yaml
Got 1 kubearmor policies in file kubearmor_policies_default_default_redis_gfxkociy.yaml
Got 1 kubearmor policies in file kubearmor_policies_default_default_server_bxuhwhxx.yaml
Got 1 kubearmor policies in file kubearmor_policies_default_default_server_feuzmcnu.yaml
Got 1 kubearmor policies in file kubearmor_policies_default_default_server_forjhldw.yaml
Got 1 kubearmor policies in file kubearmor_policies_default_default_server_hgaraquf.yaml
Got 1 kubearmor policies in file kubearmor_policies_default_default_server_hhziwxkq.yaml
Got 1 kubearmor policies in file kubearmor_policies_default_default_server_mnykhxfh.yaml
Got 1 kubearmor policies in file kubearmor_policies_default_default_server_nsqicjut.yaml
Got 1 kubearmor policies in file kubearmor_policies_default_default_server_sovrhiqt.yaml
Got 1 kubearmor policies in file kubearmor_policies_default_default_server_tnoufwqq.yaml
Got 1 kubearmor policies in file kubearmor_policies_default_default_server_vzcswcbc.yaml
Got 1 kubearmor policies in file kubearmor_policies_default_explorer_knoxautopolicy_rsumqalb.yaml
Got 1 kubearmor policies in file kubearmor_policies_default_explorer_mysql_qrrufwnd.yaml
```

Discovered Network (Cilium) Policies

Visualize discovered policies: Network

- Lets see what “app: checkoutservice” network activity is

```
./filter-policy -f cilium_policies.yaml -l '^app: checkoutservice' -g checkoutservice.png -o checkoutservice.yaml
```

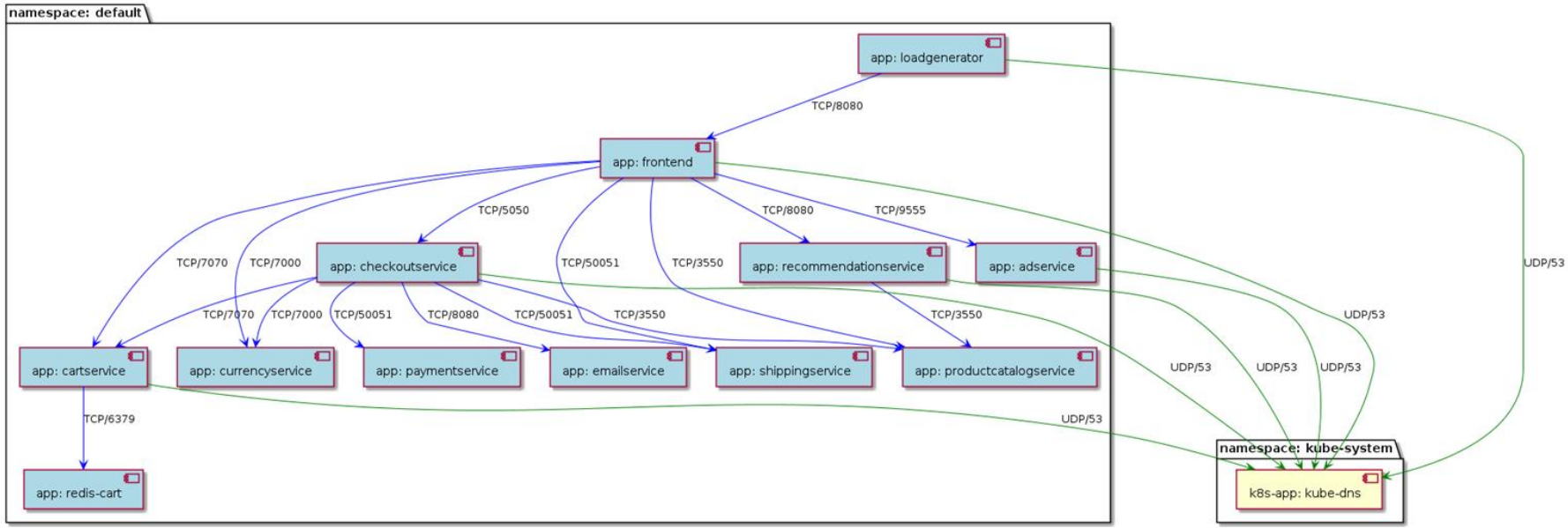


`checkoutservice.yaml` contains Cilium network policies only for ‘`app: checkoutservice`’. Helps in applying policies one deployment/namespace at a time.

Visualize discovered policies: Network

- Lets see what “overall” network activity is

```
./filter-policy -f cilium_policies.yaml -l '^app: .*' -g full.png -o full.yaml
```



Apply discovered policies

- Let's apply the *checkoutservice.yaml* discovered policies ...
  - *checkoutservice.yaml* can be found here.

```
❯ kubectl apply -f checkoutservice.yaml
ciliumnetworkpolicy.cilium.io/autopol-egress-cssntheiergviet created
ciliumnetworkpolicy.cilium.io/autopol-ingress-pvmiotdtzzmkgig created
ciliumnetworkpolicy.cilium.io/autopol-egress-xsceyezggglldut created
ciliumnetworkpolicy.cilium.io/autopol-egress-zhrwbscrnenkfft created
ciliumnetworkpolicy.cilium.io/autopol-egress-zmtkekrmhujcrqq created
ciliumnetworkpolicy.cilium.io/autopol-egress-xxgpyrqdjypojsa created
ciliumnetworkpolicy.cilium.io/autopol-egress-nsohskpduzmyssf created
ciliumnetworkpolicy.cilium.io/autopol-egress-efbrnlbtnrlouvg created
```

Verify that the online boutique app works unhindered...

```
apiVersion: cilium.io/v2
kind: CiliumNetworkPolicy
metadata:
  name: autopol-egress-cssntheiergviet
  namespace: default
spec:
  endpointSelector:
    matchLabels:
      app: checkoutservice
  egress:
  - toEndpoints:
    - matchLabels:
        app: paymentservice
        k8s:io.kubernetes.pod.namespace: default
    toPorts:
    - ports:
      - port: "50051"
        protocol: TCP
---
apiVersion: cilium.io/v2
kind: CiliumNetworkPolicy
metadata:
  name: autopol-ingress-pvmiotdtzzmkgig
  namespace: default
spec:
  endpointSelector:
    matchLabels:
      app: checkoutservice
  ingress:
  - fromEndpoints:
    - matchLabels:
        app: frontend
        k8s:io.kubernetes.pod.namespace: default
    toPorts:
    - ports:
      - port: "5050"
        protocol: TCP
:::::
```

Lets attack the checkoutservice deployment...

- We have an attacker pod installed in the cluster...
  - Attacker pod tries to nmap to the checkoutservice...

```
root@attacker-7f99cbd557-pjgkg:/# nmap -p 5050 checkoutservice.default.svc.cluster.local

Starting Nmap 7.60 ( https://nmap.org ) at 2022-03-06 15:47 UTC
Nmap scan report for checkoutservice.default.svc.cluster.local (10.43.10.74)
Host is up (0.00018s latency).

PORT     STATE SERVICE
5050/tcp open  mmcc

Nmap done: 1 IP address (1 host up) scanned in 0.58 seconds
root@attacker-7f99cbd557-pjgkg:/#
```

Before applying discovered policies

```
root@attacker-7f99cbd557-pjgkg:/# nmap -p 5050 checkoutservice.default.svc.cluster.local

Starting Nmap 7.60 ( https://nmap.org ) at 2022-03-06 15:48 UTC
Nmap scan report for checkoutservice.default.svc.cluster.local (10.43.10.74)
Host is up (0.00037s latency).

PORT     STATE    SERVICE
5050/tcp filtered mmcc

Nmap done: 1 IP address (1 host up) scanned in 0.60 seconds
root@attacker-7f99cbd557-pjgkg:/#
```

After applying discovered policies

Attacker not able to access the checkoutservice port after policy enforcement..

Verify the alerts during attack...

- Lets see the alerts generated during attack...
  - Full json output here.

```
> hubble observe -f -o json --verdict DROPPED
{"time":"2022-03-
06T16:06:28.351280999Z","verdict":"DROPPED","drop_reason":133,"ethernet":{"source":"4e:a8:54:72:e6:1c","dest
ination":"3a:bc:48:b5:2a:d2"},"IP":{"source":"10.0.0.40","destination":"10.0.0.83","ipVersion":"IPv4"},"l4":
{"TCP":{"source_port":38191,"destination_port":5050,"flags":{"SYN":true}}},"source":{"ID":3348,"identity":20
366,"namespace":"default","labels":["k8s:app=attacker","k8s:io.cilium.k8s.namespace.labels.kubernetes.io/met
adata.name=default","k8s:io.cilium.k8s.policy.cluster=default","k8s:io.cilium.k8s.policy.serviceaccount=defa
ult","k8s:io.kubernetes.pod.namespace=default"],"pod_name":"attacker-7f99cbd557-
pjgkg","workloads":[{"name":"attacker-
7f99cbd557","kind":"ReplicaSet"}]},"destination":{"ID":417,"identity":38995,"namespace":"default","labels":[
"k8s:app=checkoutservice","k8s:io.cilium.k8s.namespace.labels.kubernetes.io/metadata.name=default","k8s:io.c
ilium.k8s.policy.cluster=default","k8s:io.cilium.k8s.policy.serviceaccount=default","k8s:io.kubernetes.pod.n
amespace=default"],"pod_name":"checkoutservice-8f859666-qjp82","workloads":[{"name":"checkoutservice-
8f859666","kind":"ReplicaSet"}]},"Type":"L3_L4","node_name":"ubuntu2004-
vagrants","event_type":{"type":5},"traffic_direction":"INGRESS","drop_reason_desc":"POLICY_DENIED","Summary"
:"TCP Flags: SYN"}
```

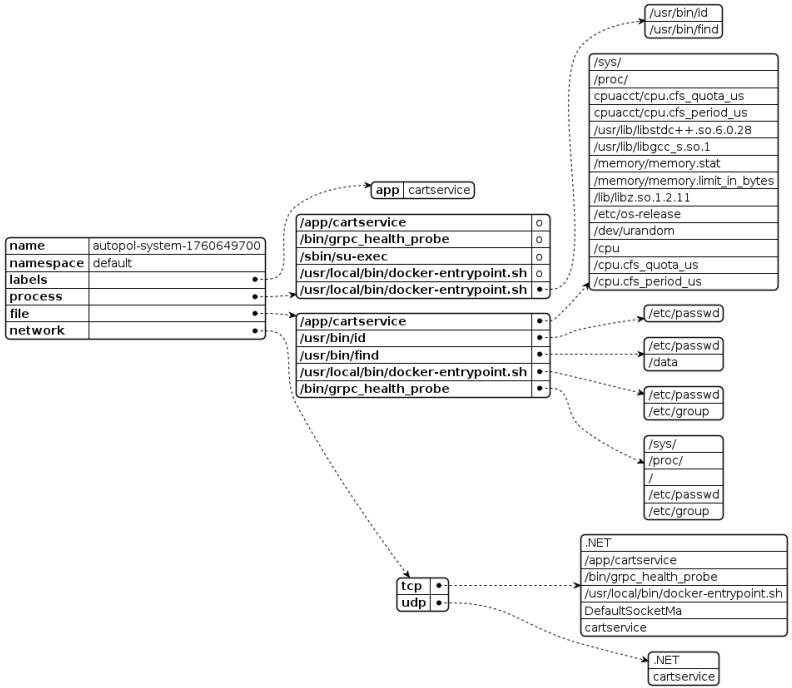

```
{
  "time": "2022-03-06T16:06:28.351280999Z",
  "verdict": "DROPPED",
  "drop_reason": 133,
  "IP": {
    "source": "10.0.0.40",
    "destination": "10.0.0.83",
    "ipVersion": "IPv4"
  },
  "l4": {
    "TCP": {
      "source_port": 38191,
      "destination_port": 5050,
      "flags": {
        "SYN": true
      }
    }
  },
  "source": {
    "ID": 3348,
    "identity": 20366,
    "namespace": "default",
    "labels": [
      "k8s:app=attacker",
      "k8s:io.cilium.k8s.namespace.labels.kubernetes.io/metadata.name=default",
      "k8s:io.cilium.k8s.policy.cluster=default",
      "k8s:io.cilium.k8s.policy.serviceaccount=default",
      "k8s:io.kubernetes.pod.namespace=default"
    ],
    "pod_name": "attacker-7f99cbd557-pjgkg",
    "workloads": [
      {
        "name": "attacker-7f99cbd557",
        "kind": "ReplicaSet"
      }
    ]
  },
  "destination": {
    "ID": 417,
    "identity": 38995,
    "namespace": "default",
    "labels": [
      "k8s:app=checkoutservice",
      "k8s:io.cilium.k8s.namespace.labels.kubernetes.io/metadata.name=default",
      "k8s:io.cilium.k8s.policy.cluster=default",
      "k8s:io.cilium.k8s.policy.serviceaccount=default",
      "k8s:io.kubernetes.pod.namespace=default"
    ],
    "pod_name": "checkoutservice-8f859666-qjp82",
    "workloads": [
      {
        "name": "checkoutservice-8f859666",
        "kind": "ReplicaSet"
      }
    ]
  },
  "Type": "L3_L4",
  "node_name": "ubuntu2004-vagrants",
  "event_type": {
    "type": 5
  },
  "traffic_direction": "INGRESS",
  "drop_reason_desc": "POLICY_DENIED",
  "Summary": "TCP Flags: SYN"
}
```

Discovered Application (KubeArmor) Policies

Visualize discovered policies: KubeArmor

- Lets see what “app: cartservice” application activity is upto...

```
./filter-policy -f kubearmor_policies_default_default_server_tnoufwqq.yaml -l '^app: .*' -g cartservice.png
```




```
apiVersion: security.kubearmor.com/v1
kind: KubeArmorPolicy
metadata:
  name: autopol-system-1760649700
  namespace: default
spec:
  severity: 1
  selector:
    matchLabels:
      app: cartservice
  process:
    matchPaths:
    - path: /app/cartservice
    - path: /bin/grpc_health_probe
    - path: /sbin/su-exec
    - path: /usr/bin/find
      fromSource:
      - path: /usr/local/bin/docker-entrypoint.sh
    - path: /usr/bin/id
      fromSource:
      - path: /usr/local/bin/docker-entrypoint.sh
  file:
    matchPaths:
    - path: /cpu.cfs_period_us
      fromSource:
      - path: /app/cartservice
:::::
    - path: /etc/group
      fromSource:
      - path: /bin/grpc_health_probe
      - path: /usr/local/bin/docker-entrypoint.sh
    - path: /etc/passwd
      fromSource:
      - path: /bin/grpc_health_probe
      - path: /usr/bin/find
      - path: /usr/bin/id
      - path: /usr/local/bin/docker-entrypoint.sh
    - path: /lib/libz.so.1.2.11
      fromSource:
      - path: /app/cartservice
::::
    - path: /usr/lib/libstdc++.so.6.0.28
      fromSource:
      - path: /app/cartservice
    - path: cpuacct/cpu.cfs_period_us
      fromSource:
      - path: /app/cartservice
    - path: cpuacct/cpu.cfs_quota_us
      fromSource:
      - path: /app/cartservice
    matchDirectories:
    - dir: /proc/
      fromSource:
      - path: /app/cartservice
      - path: /bin/grpc_health_probe
:::::
    - dir: /sys/
      fromSource:
      - path: /app/cartservice
      - path: /bin/grpc_health_probe
  network:
    matchProtocols:
    - protocol: tcp
      fromSource:
      - path: .NET
      - path: /app/cartservice
      - path: /bin/grpc_health_probe
      - path: /usr/local/bin/docker-entrypoint.sh
      - path: DefaultSocketMa
      - path: cartservice
    - protocol: udp
      fromSource:
      - path: .NET
      - path: cartservice
  action: Allow
```

Apply Auto Discovered KubeArmor policy

<todo>

Attack the cartservice pod...

<todo>