Positive Hack Days

NetworkPolicy: родной межсетевой экран Kubernetes

WhoAmI

Дмитрий Евдокимов

Founder, CTO Luntry

Автор Telegram-канала “k8s (in)security”

Автор курса “Cloud Native безопасность в Kubernetes”

Спикер: BlackHat, HITB, ZeroNights, HackInParis, Confidence, SAS, PHDays, DevOpsConf, KuberConf, HighLoad++ и др.

Сергей Канибор

Исследователь ИБ, Luntry

luntry.ru

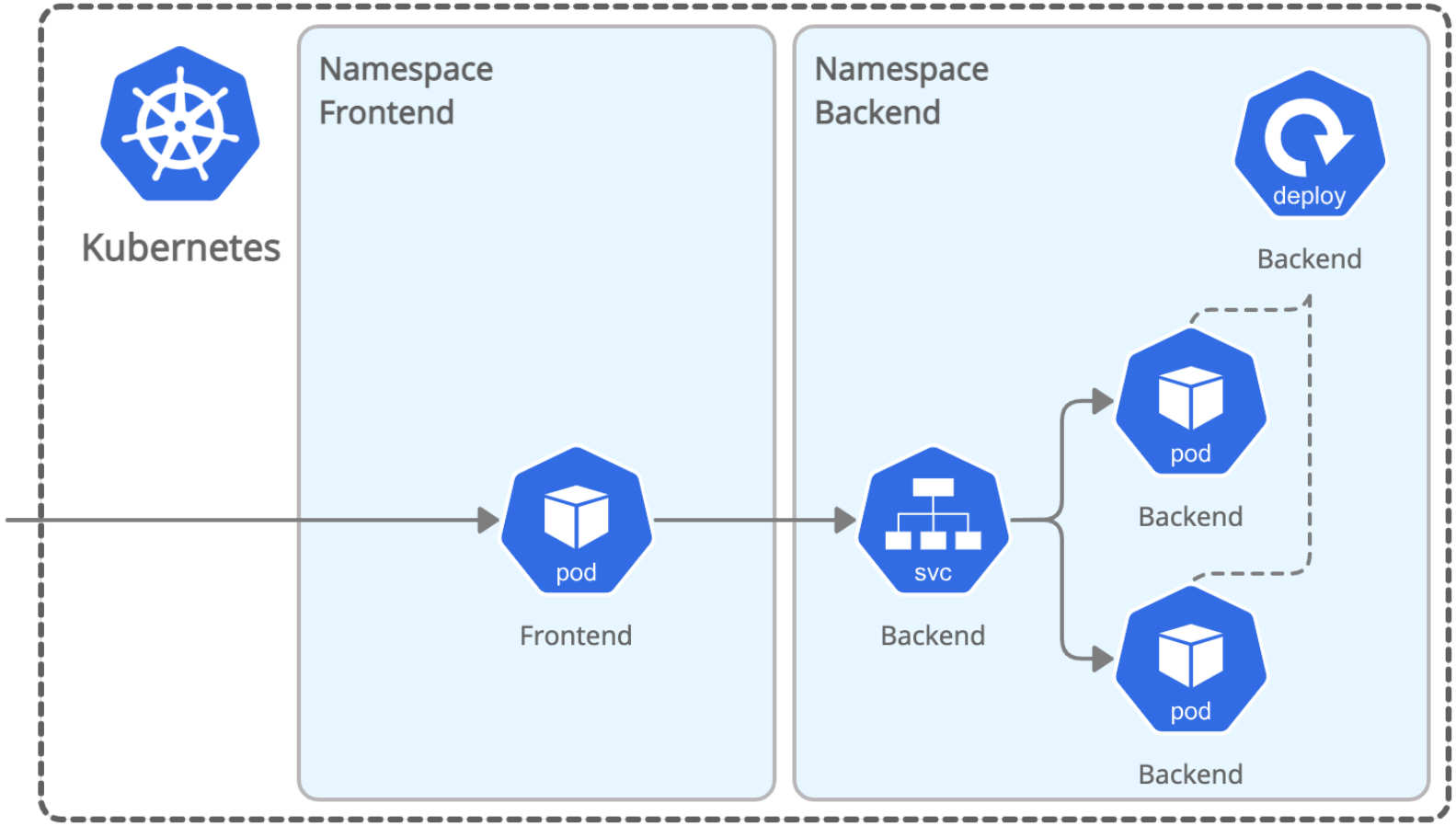
Agenda

1. Kubernetes Network 101
   - CNI
2. Контроль на уровне сети
   - Zero Trust
3. Native NetworkPolicy
   - Возможности
   - Ограничения
   - Развитие
4. Native NetworkPolicy
   - Возможности
     (Node, Clusterwide, FQDN, Action, …)
   - Особенности Calico
   - Особенности Cilium
   - Сравнение
5. Tips&Tricks
6. Conclusion

Kubernetes Network 101

Kubernetes Network 101

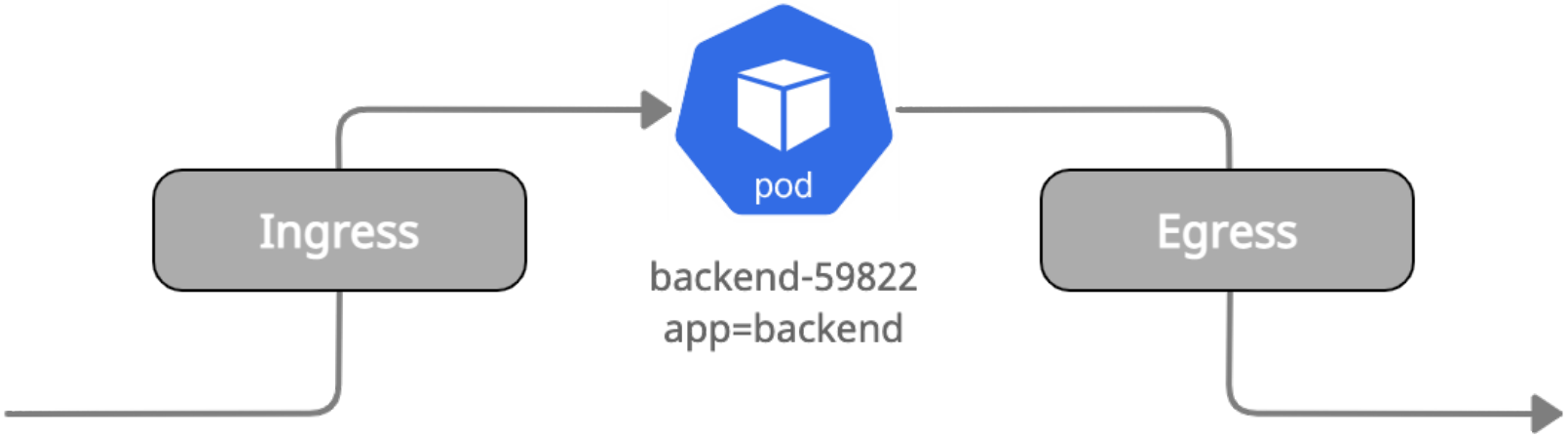
- У всех Pods есть IP
- PodCIDR на каждой Node
- Service для load-balancing
- DNS для service-discovery



Сетевое поведение Pod’ов

По умолчанию Pods в Kubernetes могут коммуницировать со всеми Pods без ограничений:

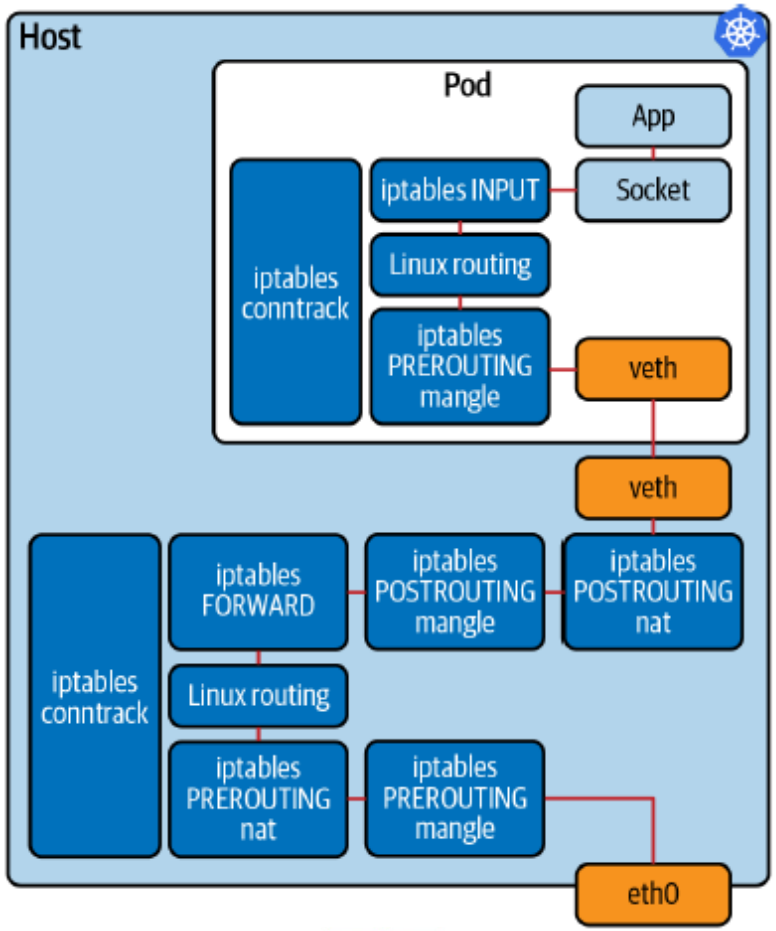
- North-south traffic – с ресурсами за пределами Kubernetes
- East-west traffic – с другими приложениями в Kubernetes
- Трафик не шифруется



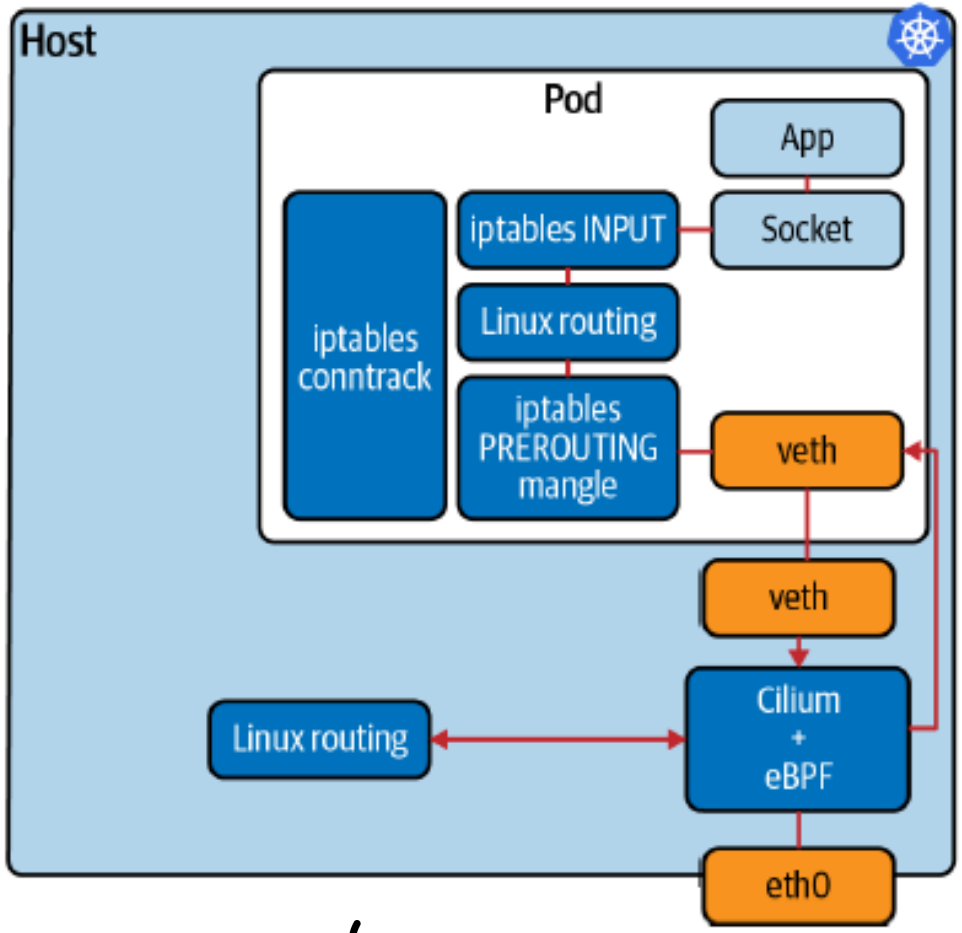
Container Networking Interface (CNI)

kubectl get all -n kube-system | egrep -i "ACI|Calico|Canal|Cilium|CNI-Genie|Contiv|Contrail|Flannel|Knitter|Multus|OVN-Kubernetes|OVN4NFV-K8S-Plugin|NSX-T|Nuage|Romana|Weave"

Под капотом: iptables, nftables или ebpf



Packets have to traverse two networking stacks.



Using eBPF, Cilium can direct packets to bypass the host networking stack.

Источник: https://www.oreilly.com/library/view/what-is-ebpf/9781492097266/

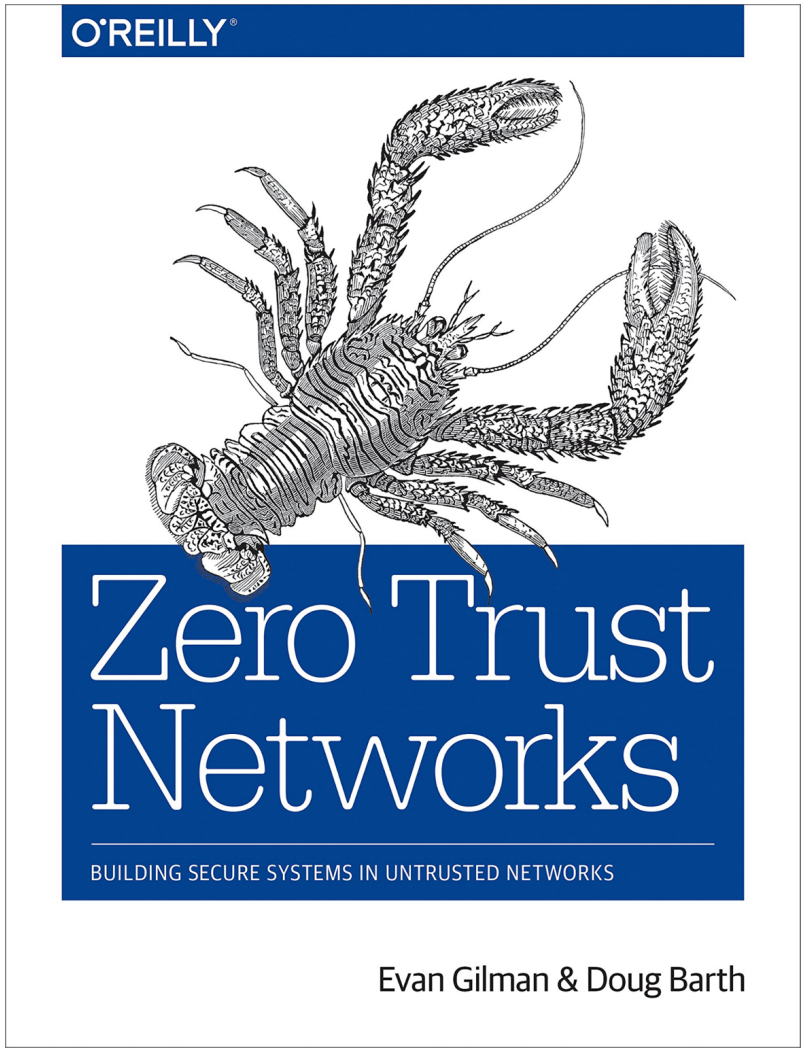
Контроль на уровне сети

Zero Trust Networking

“Zero Trust Networks are resilient even when attackers manage to breach applications or infrastructure. They make it hard for attackers to move laterally, and reconnaissance activities easier to spot.

Organizations that embrace the change control model in this How-To will be able to tightly secure their network without imposing a drag on innovation in their applications. Security teams can be enablers of business value, not roadblocks.”

Документация Calico



Контроль сетевого взаимодействия в Kubernetes

- На уровне приложений
  - Через библиотеку, добавленную при сборке, или через инжект
  - Реализовано в user space
- На уровне Service Mesh (Istio, Linkerd)
  - С помощью sidecar-контейнера
  - Обычно, это ресурс AuthorizationPolicy
  - Уровень L7
  - Реализовано в user space
- На уровне CNI
  - Native или Custom NetworkPolicy
  - Уровень L3-L4 (иногда и L7)
  - Реализовано в kernel space

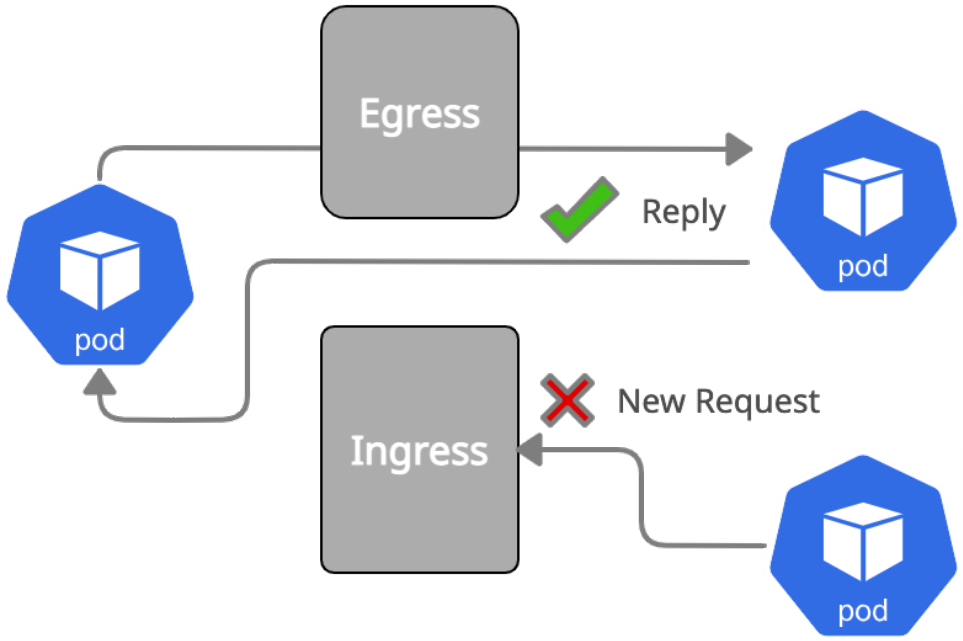
Реализации Network Policy

APIGROUP	NAMESPACED	KIND
cilium		
cilium.io	false	CiliumClusterwideNetworkPolicy
cilium.io	true	CiliumNetworkPolicy
networking.k8s.io	true	NetworkPolicy
calico		
crd.projectcalico.org	false	GlobalNetworkPolicy
crd.projectcalico.org	true	NetworkPolicy
networking.k8s.io	true	NetworkPolicy
weave		
networking.k8s.io	true	NetworkPolicy
kube-router		
networking.k8s.io	true	NetworkPolicy
flannel		
networking.k8s.io	true	NetworkPolicy (не умеет)

Native NetworkPolicy

Родной NetworkPolicy

- Это YAML
  - Декларативный подход (Policy-as-Code)
- Stable, начиная с Kubernetes версии 1.7
  - Последняя текущая версия Kubernetes 1.24 ;)
- Реализует сегментацию сети
  - По принципу white list
- Работает в Pod сети
  - Ограничение Pod-to-Pod трафика
- Уровень гранулярности – Pod
  - Неважно, какой процесс и какой контейнер



YAML NetworkPolicy

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: test-network-policy
  namespace: default
spec:
  podSelector:
    matchLabels:
      role: db
  policyTypes:
  - Ingress
  - Egress
  ingress:
  - from:
    - ipBlock:
        cidr: 172.17.0.0/16
        except:
        - 172.17.1.0/24
    - namespaceSelector:
        matchLabels:
          project: myproject
    - podSelector:
        matchLabels:
          role: frontend
    ports:
    - protocol: TCP
      port: 6379
  egress:
  - to:
    - ipBlock:
        cidr: 10.0.0.0/24
    ports:
    - protocol: TCP
      port: 5978
```

имя политики

в каком namespace действует

то к каким Pods будет применена политика

типы правил в политике. По умолчанию, Ingress.

блок ingress правил

Откуда трафик

1 источник

2 источник

3 источник

На какой порт

блок egress правил

Куда трафик

На какой порт

Развитие NetworkPolicy

SIG Network (WG-Network Policy API) работает над улучшениями для NetworkPolicy:

- Поддержка диапазона портов в v1.21 в alpha
  - NetworkPolicyEndPort
- NetworkPolicyStatus – в 1.24 бесполезен, но ждем 1.25 с поддержкой от CNI
- Поддержка Fully Qualified Domain Names (FQDNs) в политиках
  - Есть прототип от Google, и его можно уже попробовать как CRD FQDNNetworkPolicies.
- Поддержка политик для всего кластера

⟶ Custom NetworkPolicy

Network policies могут применяться к …

- Pods
- Host interfaces
- VM’s
- entities (k8s api, hosts, …)

```
apiVersion: "cilium.io/v2"
kind: CiliumClusterwideNetworkPolicy
metadata:
  name: "lock-down-ingress-worker-node"
spec:
  nodeSelector:
    matchLabels:
      type: ingress-worker
  ingress:
  - fromEntities:
    - remote-node
    - health
  - toPorts:
    - ports:
      - port: "6443"
        protocol: TCP
```

```
apiVersion: projectcalico.org/v3
kind: GlobalNetworkPolicy
metadata:
  name: ingress-k8s-workers
spec:
  selector: has(kubernetes-worker)
    ingress:
  - action: Allow
    destination:
      nets:
      - 127.0.0.0/8
    - action: Allow
    protocol: TCP
    source:
      selector: has(node-role.kubernetes.io/master)
    destination:
      ports:
        - 10250
```

```
kind: NetworkPolicy
apiVersion:
networking.k8s.io/v1
metadata:
  name: frontend-egress-deny
spec:
  podSelector:
    matchLabels:
      app: frontend
  policyTypes:
  - Egress
```

Selectors: Calico

Logical operators	
`( <expression> )`	Matches if and only if `<expression>` matches. (Parentheses are used for grouping expressions.)
`! <expression>`	Matches if and only if `<expression>` does not match. **Tip:** `!` is a special character at the start of a YAML string, if you need to use `!` at the start of a YAML string, enclose the string in quotes.
`<expression 1> && <expression 2>`	"And": matches if and only if both `<expression 1>`, and, `<expression 2>` matches
`<expression 1> || <expression 2>`	"Or": matches if and only if either `<expression 1>`, or, `<expression 2>` matches.
Match operators	
`all()`	Match all in-scope resources. To match *no* resources, combine this operator with `!` to form `!all()`.
`global()`	Match all non-namespaced resources. Useful in a `namespaceSelector` to select global resources such as global network sets.
`k == 'v'`	Matches resources with the label 'k' and value 'v'.
`k != 'v'`	Matches resources without label 'k' or with label 'k' and value *not* equal to `v`
`has(k)`	Matches resources with label 'k', independent of value. To match pods that do not have label `k`, combine this operator with `!` to form `!has(k)`
`k in { 'v1', 'v2' }`	Matches resources with label 'k' and value in the given set
`k not in { 'v1', 'v2' }`	Matches resources without label 'k' or with label 'k' and value *not* in the given set
`k contains 's'`	Matches resources with label 'k' and value containing the substring 's'
`k starts with 's'`	Matches resources with label 'k' and value starting with the substring 's'
`k ends with 's'`	Matches resources with label 'k' and value ending with the substring 's'

Действия в политиках

Calico: `Allow, Deny, Log, Pass`

- Log пишет отладочную инфу в syslog
- Pass пропускает текущее правило и переходит к следующему. Если дальнейшие правила не указаны, то применяется Deny
- Allow по умолчанию

```
ingress:
 - action: Allow
     protocol: TCP
```

Calico: `Allow, Deny`

- Deny (beta-feature) не поддерживается в L7-политиках
- Allow по умолчанию

```
ingressDeny:
 - fromEntities:
    - "world"
```

Контроль доступа: Calico

- Порты: нумерованные, диапазон портов, и Kubernetes named ports
- Протоколы: TCP, UDP, ICMP, SCTP, UDPlite, ICMPv6, protocol numbers (1-255)
- HTTP attributes (при использовании Istio)
- ICMP attributes
- IP версия (IPv4, IPv6)
- IP или CIDR
- Endpoint selectors
- Namespace selectors
- Service account selectors

```
ingress:
   - action: Allow
     source:
       serviceAccounts:
         names:
           - api-service
            - user-auth-service
```

```
http:
  methods: ["GET", "PUT"]
  paths:
    - exact: "/projects/calico"
     - prefix: "/users"
```

```
egress:
   - action: Allow
     destination:
       services:
         name: kubernetes
          namespace: default
```

```
ingress:
  - action: Allow
    source:
      namespaceSelector: 'environment =="development"'
```

```
ports: [8080, "1234:5678", "named-port"]
```

```
ingress:
  - action: Deny
    protocol: ICMP
```

Контроль доступа: Cilium

- Label-based
- Service-based
- Entity-based
- IP/CIDR-based
- DNS-based

```
endpointSelector:
   matchLabels:
     io.cilium.k8s.policy.serviceaccount: leia
 ingress:
 - fromEndpoints:
   - matchLabels:
       io.cilium.k8s.policy.serviceaccount: luke
```

```
egress:
   - toEndpoints:
     - matchLabels:
         "k8s:io.kubernetes.pod.namespace": kube-system
         "k8s:k8s-app": kube-dns
     toPorts:
       - ports:
          - port: "53"
            protocol: ANY
         rules:
           dns:
             - matchPattern: "*"
 - toFQDNs:
     - matchName: "my-remote-service.com"
```

```
egress:
  - toEntities:
      - kube-apiserver
```

```
ingress:
 - fromEndpoints:
   - matchLabels:
       role: frontend
```

```
egress:
 - toServices:
   - k8sService:
       serviceName: myservice
       namespace: default
```

Network Set

```yaml
apiVersion: projectcalico.org/v3
kind: NetworkSet
metadata:
  name: external-database
  namespace: staging
  labels:
    role: db
spec:
  nets:
  - 198.51.100.0/28
  - 203.0.113.0/24
```

```yaml
apiVersion: "cilium.io/v2"
kind: CiliumNetworkPolicy
metadata:
  name: "cidr-rule"
spec:
  endpointSelector:
    matchLabels:
      app: myService
  egress:
  - toCIDR:
    - 20.1.1.1/32
  - toCIDRSet:
    - cidr: 10.0.0.0/8
      except:
      - 10.96.0.0/12
```

Clusterwide-политики

```
apiVersion: crd.projectcalico.org/v1
kind: GlobalNetworkPolicy
metadata:
  name: allow-coredns
spec:
  types:
  - Ingress
  - Egress
  selector: k8s-app == 'kube-dns'
  egress:
  - action: Allow
  ingress:
 - action: Allow
```

```
apiVersion: "cilium.io/v2"
kind: CiliumClusterwideNetworkPolicy
metadata:
  name: "wildcard-from-endpoints"
spec:
  nodeSelector:
    matchLabels:
      k8s:io.kubernetes.pod.namespace:
kube-system
      k8s-app: kube-dns
  ingress:
  - fromEndpoints:
    - {}
    toPorts:
    - ports:
      - port: "53"
        protocol: UDP
```

NetworkPolicy для Host

- Дают возможность ограничить не только сеть Pods, но и сами хосты
- Применяются к нодам (хостам)
- Только L3/L4

```yaml
apiVersion: projectcalico.org/v3
kind: GlobalNetworkPolicy
metadata:
  name: k8s-worker
spec:
  selector: "role == 'k8s-worker'"
  order: 0
  ingress:
  - action: Allow
    protocol: TCP
    source:
      nets:
      - "<your management CIDR>"
    destination:
      ports: [22]
  - action: Allow
    protocol: ICMP
```

Port	Protocol	CIDR	Direction	Purpose
22	TCP	0.0.0.0/0	Inbound	SSH access
53	UDP	0.0.0.0/0	Outbound	DNS queries
67	UDP	0.0.0.0/0	Outbound	DHCP access
68	UDP	0.0.0.0/0	Inbound	DHCP access
179	TCP	0.0.0.0/0	Inbound & Outbound	BGP access (Calico networking)
2379	TCP	0.0.0.0/0	Inbound & Outbound	etcd access
2380	TCP	0.0.0.0/0	Inbound & Outbound	etcd access
6443	TCP	0.0.0.0/0	Inbound & Outbound	Kubernetes API server access
6666	TCP	0.0.0.0/0	Inbound & Outbound	etcd self-hosted service access
6667	TCP	0.0.0.0/0	Inbound & Outbound	etcd self-hosted service access

```yaml
apiVersion: "cilium.io/v2"
kind: CiliumClusterwideNetworkPolicy
metadata:
  name: "lock-down-ingress-worker-node"
spec:
  nodeSelector:
    matchLabels:
      type: ingress-worker
  ingress:
  - fromEntities:
    - remote-node
    - health
```

L7-политики

Calico:

- Поддерживаются только Ingress-правила
- Политики должны иметь Allow action
- В стадии Beta

```
http:
  methods: ["GET", "PUT"]
  paths:
    - exact: "/projects/calico"
     - prefix: "/users"
```

Cilium:

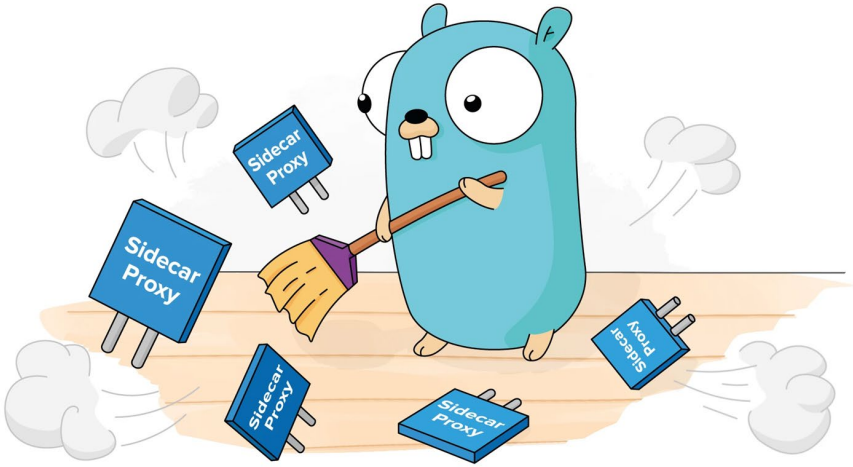
- При нарушении политики пакеты не дропаются
- Не поддерживаются в Host Policies, если используется Node Selector

```
toPorts:
 - ports:
   - port: "80"
     protocol: TCP
   rules:
     http:
     - method: "GET"
        path: "/public"
```

Дружба с Service Mesh

- Интеграция с Istio для mTLS в L7-политиках
- Cilium Service Mesh

Istio



Алерты при срабатывании политики

Cilium – cilium monitor / Hubble

- Уведомления о дропнутых пакетах
- Уведомления о вердиктах политик
- Отладочная информация

3550	forwarded	less than 20 seconds
5050	dropped	less than 20 seconds
5050	dropped	less than 20 seconds
8080	forwarded	less than 20 seconds
6379	forwarded	less than 20 seconds
8080	forwarded	less than 20 seconds
8080	forwarded	less than 20 seconds
8080	forwarded	less than 20 seconds

Calico – Log в action

- На самом деле не дает инфы о нарушении
- Пишет отладочную информацию в syslog

```
May 4 18:26:49 nodename kernel: [686879.577344]
calico-packet: IN=eth0 OUT=azv204d22f3118
MAC=00:22:48:44:62:15:c0:d6:82:94:e6:49:08:00
SRC=172.16.208.33 DST=172.16.208.28 LEN=60 TOS=0x00
PREC=0x00 TTL=62 ID=25858 DF PROTO=TCP SPT=33374
DPT=8443 WINDOW=64240 RES=0x00 SYN URGP=0
```

Итоговое сравнение

Features	Calico	Cilium	Native
Implementation	iptables, eBPF	iptables, eBPF	
Supported OCI layers	L3/L4, L7	L3/L4, L7	L3/L4
Can apply policies to	pods/containers, VMs, and/or to host interfaces	pods/containers, and/or to host interfaces, VM	pod
Policies can define rules for	ingress, egress, or both	ingress, egress, or both	ingress, egress, or both
Policy actions	allow, deny, log, pass	allow, deny	allow
Source and destination match criteria	Ports, protocols, HTTP attributes, ICMP attributes, IP version, IP or CIDR, Endpoint selector, Namespace selector, Service account selector.	Labels based, services based, entities based, IP/CIDR based, DNS based Ports, Protocol, Host Policies, Service account	Labels, Ports, Protocols, namespaceSelector, IP/CIDR based
Network Set	NetworkSet & GlobalNetworkSet resources	fromCIDRSet/toCIDRSet in Network Policy	–
Clusterwide	GlobalNetworkPolicy	CiliumClusterwideNetworkPolicy	–
Alert	–	Hubble, Cilium Monitor (+/-)	NetworkPolicyStatus (???)
Service mesh	Istio integration	Cilium Service Mesh, Istio integration	–

Tips&Tricks

Tips&Tricks: policyTypes

Native: если никакой policyTypes не указан, то по умолчанию будет выставлен Ingress.

Cilium: Нужно указать Ingress, Egress или оба. Если типы явно не указаны, то политика не будет работать.

Calico:

Ingress rule present?	Egress rule present?	Value
No	No	Ingress
Yes	No	Ingress
No	Yes	Egress
Yes	Yes	Ingress, Egress

Tips&Tricks: Default policies

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: default-deny-ingress
spec:
  podSelector: {}
  policyTypes:
  - Ingress
```

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-all-ingress
spec:
  podSelector: {}
  ingress:
  - {}
  policyTypes:
  - Ingress
```

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: default-deny-all
spec:
  podSelector: {}
  policyTypes:
  - Ingress
  - Egress
```

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: default-deny-egress
spec:
  podSelector: {}
  policyTypes:
  - Egress
```

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-all-egress
spec:
  podSelector: {}
  egress:
  - {}
  policyTypes:
  - Egress
```

Tips&Tricks: приоритет политик

- Управляет порядком приоритета
- Calico первой применяет политику с наименьшим значением order

```
apiVersion: crd.projectcalico.org/v1
kind: NetworkPolicy
metadata:
  name: deny-busybox-egress
  namespace: advanced-policy-demo
spec:
  order: 20
  selector: run == 'access'
  types:
  - Egress
  egress:
  - action: Deny
```

```
apiVersion: crd.projectcalico.org/v1
kind: NetworkPolicy
metadata:
  name: allow-busybox-egress
  namespace: advanced-policy-demo
spec:
  order: 100
  selector: run == 'access'
  types:
  - Egress
  egress:
  - action: Allow
```

Tips&Tricks: Allow DNS


```
kind: NetworkPolicy
metadata:
  name: allow-dns-access
  namespace: advanced-policy-demo
spec:
  podSelector:
    matchLabels: {}
  policyTypes:
  - Egress
  egress:
  - to:
    - namespaceSelector:
        matchLabels:
          name: kube-system
    ports:
    - protocol: UDP
      port: 53
```

Tips&Tricks: Network Policy для Kube-API

- Native – endpoint master node
- Calico – Service selector
- Cilium – Entities или Service selector

```
apiVersion: crd.projectcalico.org/v1
kind: NetworkPolicy
metadata:
  name: allow-api-access
  namespace: my-app
spec:
  selector: app == “backend”
  egress:
    - action: Allow
      destination:
        services:
          name: kubernetes
          namespace: default
```

```
apiVersion: "cilium.io/v2"
kind: CiliumNetworkPolicy
metadata:
  name: "dev-to-host"
spec:
  endpointSelector:
    matchLabels:
      env: dev
  egress:
  - toEntities:
    - kube-apiserver
```

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: egress-apiserver
spec:
  podSelector: {}
  policyTypes:
    - Egress
  egress:
    - to:
        - ipBlock:
            cidr: 34.76.197.27/32
      ports:
        - protocol: TCP
          port: 443
```

Tips&Tricks: Cross namespace взаимодействие

```
for ns in $(kubectl get namespace -A -ojsonpath='{.items[*].metadata.name}');
do kubectl label namespace $ns luntry.com/namespace=$ns; done
```

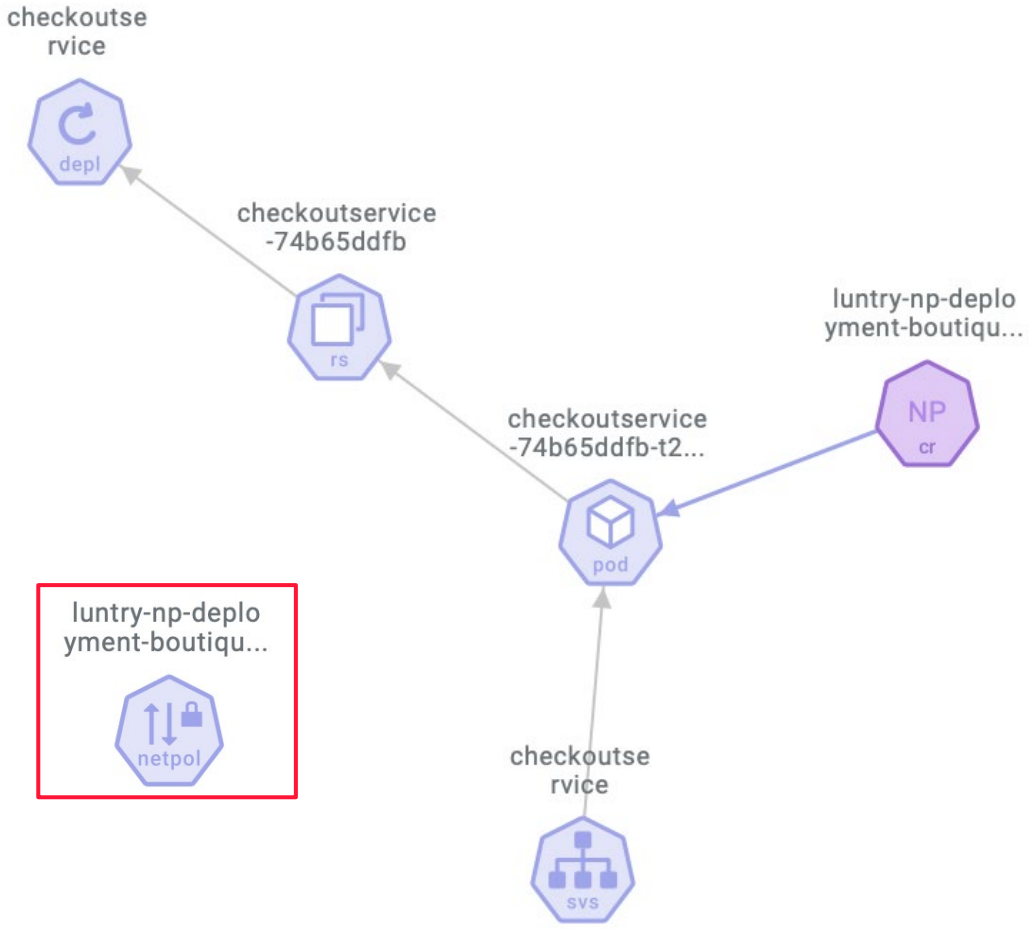
```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: database.postgres
  namespace: database
spec:
  podSelector:
    matchLabels:
      app: postgres
  ingress:
  - from:
    - namespaceSelector:
        matchLabels:
          luntry.com/namespaces: bookstore
      podSelector:
        matchLabels:
          app: admin
  policyTypes:
  - Ingress
```

Автоматический лейблинг для namespace [в v1.21 beta]

- Необходимо и удобно для inter-namespace политик

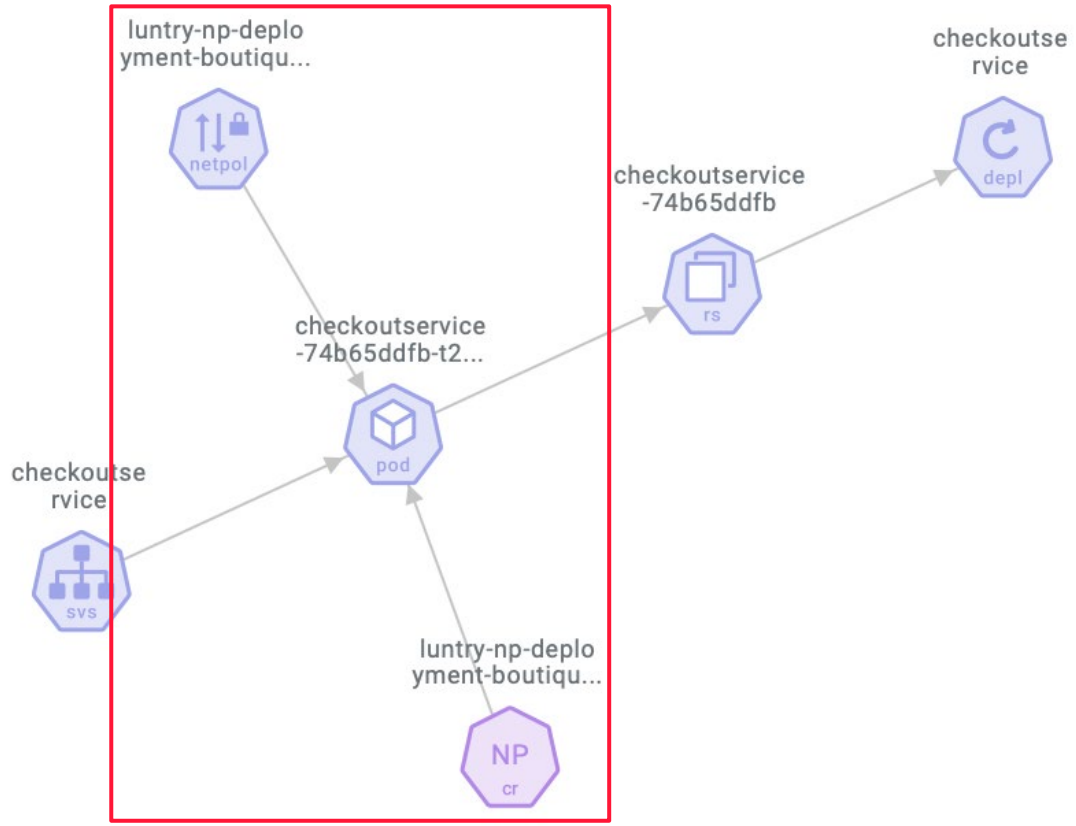
Tips&Tricks: NetworkPolicy без связи

Привязка политик. Политика может быть в namespace, но никакой роли не играть



Tips&Tricks: Native + Custom NetworkPolicy

Можно одновременно использовать и Native, и Custom политики



Tips&Tricks: Проверка с помощью PolicyEngine

Можно проверять с помощью Kyverno и/или OPA Gatekeeper!

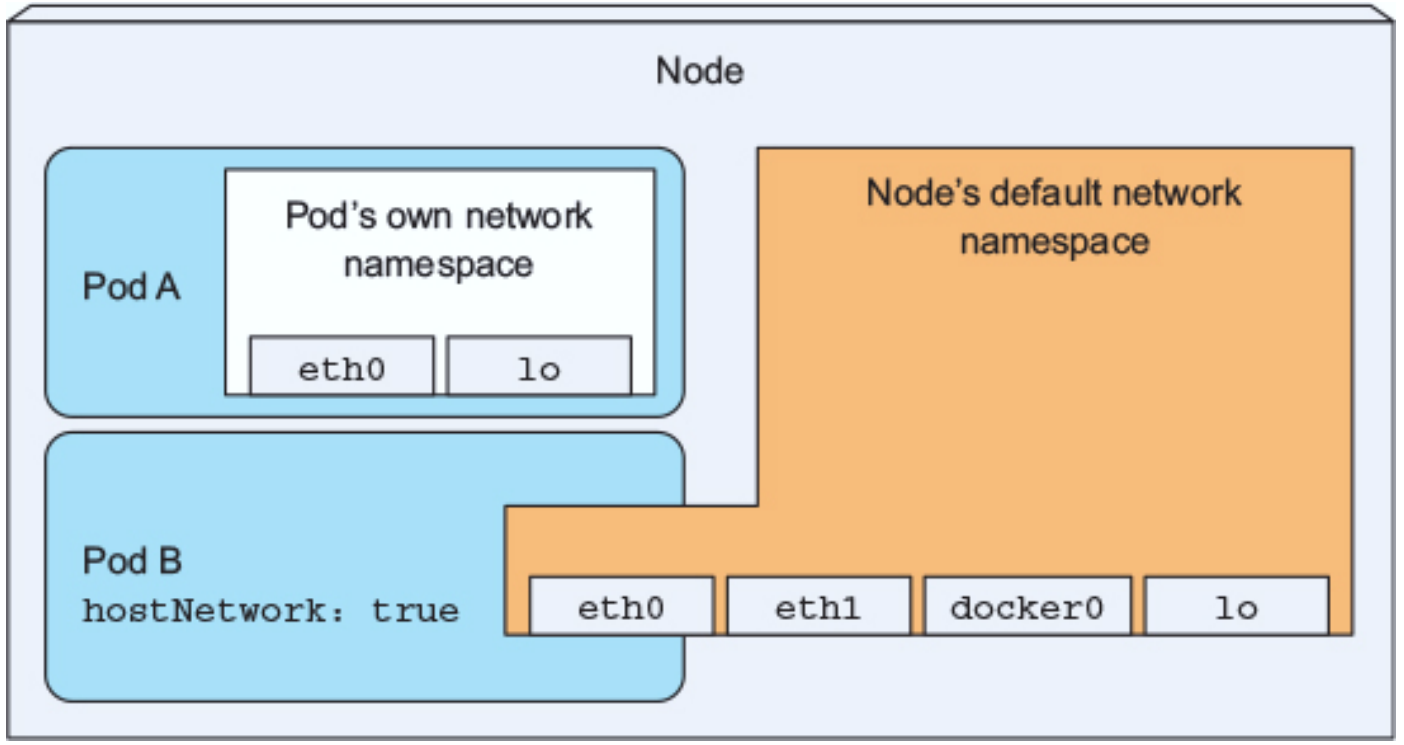
```
spec:
  rules:
  - name: no-LoadBalancer
    match:
      any:
    - resources:
        kinds:
        - networking.k8s.io/v1/NetworkPolicy
        - projectcalico.org/v3/NetworkPolicy
```

```
apiVersion:
constraints.gatekeeper.sh/v1beta1
kind: K8sDenyEgress
metadata:
  name: deny-egress
spec:
  match:
    kinds:
      - apiGroups: ["networking.k8s.io"]
        kinds: ["NetworkPolicy"]
```

Tips&Tricks: Хитрый обход NetworkPolicy

Kubernetes запускает Pods в своей собственной изолированной сети, однако можно сделать так, чтобы они работали в хостовой сети, для этого в манифесте нужно указать:

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  hostNetwork: true
  containers:
  - name: nginx
    image: nginx
```



Tips&Tricks: Cilium identitites

IP Interconnectivity

When thinking about base IP connectivity with Cilium, its useful to consider two different types of connectivity:

- Container-to-Container Connectivity
- Container Communication with External Hosts

Container-to-Container Connectivity

In the case of connectivity between two containers inside the same cluster, Cilium is in full control over both ends of the connection. It can thus transmit state and security context information between two container hosts by embedding the information in encapsulation headers or even unused bits of the IPv6 packet header. This allows Cilium to transmit the security context of where the packet origins from which allows tracing back which container labels are assigned to the origin container.

Note

As the packet headers contain security sensitive information, it is higly recommended to either encrypt all traffic or run Cilium in a trusted network environment.

⟶ Conclusion

Что нужно знать и учитывать при создании NetworkPolicy

- Используются ли default policy
- Знать labels у взаимодействующих namespaces
- Используется ли Service Mesh, и где ее Control Plane
- Какая реализация Ingress используется, и где она находится
- Используется ли отдача Logs, metrics, traces в коде приложения сторонним сервисам
- ???

Conclusion

- Kubernetes-ресурс NetworkPolicy позволит реализовать сетевую сегментацию любой сложности
- Native и Custom NetworkPolicy на текущий момент сильно отличаются по возможностям
- Декларативный подход (Policy-as-Code) к обеспечению безопасности упрощает жизнь Dev-, Ops- и Sec-командам

Полезные ссылки

1. Network Policies
2. NetworkPolicy Tutorial
3. Network Special Interest Group
4. Kubernetes Network Policy Recipes
5. Network Policy Editor
6. Kubernetes Network Policies Viewer

Спасибо за внимание!

Email: de@luntry.ru
Twitter: @evdokimovds
Telegram: @Qu3b3c

luntry.ru