Ай-ТИ Фреш

ТЕХНИЧЕСКИЙ РАЗБОР

Cilium CNI в enterprise Kubernetes: eBPF вместо iptables

Как мы внедряем Cilium 1.20: kube-proxy replacement, политики L3–L7, Hubble и Cluster Mesh

Июль 2026

itfresh.ru · ИТ-аутсорсинг для юридических лиц

О ЧЁМ ЭТОТ РАЗБОР

Суть проблемы

Кластеру Kubernetes на iptables-CNI с ростом сервисов становится тесно: цепочки kube-proxy разрастаются до тысяч правил, латентность сервисного трафика плавает, а при инциденте никто не может сказать, кто к кому ходил и почему пакет дропнулся. Мы переводим кластеры на Cilium: датаплейн eBPF в ядре, политики по identity вместо IP, видимость потоков через Hubble.

Почему это важно бизнесу

- Деградация отклика при росте сервисов: kube-proxy на iptables перебирает правила линейно, латентность плавает в часы пик
- Без видимости L7 разбор сетевого инцидента занимает часы: неясно, какой под и какой запрос был заблокирован и кем
- Плоская сеть без политик: компрометация одного пода открывает весь кластер, включая БД и бэкенды 1С
- Sidecar-mesh ради L7 добавляет прокси в каждый под — расход CPU/RAM растёт кратно числу подов; Cilium даёт L7-контроль без sidecar'ов, экономя у…
- Cluster Mesh закрывает DR: отказ площадки не останавливает сервисы при связанном втором кластере

ЧТО МЫ ЗАКЛАДЫВАЕМ

Ключевые параметры реализации

1.20

ветка Cilium в прод; апгрейд строго по одной минорной версии через preflight-DaemonSet cilium-...

Upgrade Guide, docs.cilium.io v1.20

≥5.10

минимальное ядро Linux на узлах (эквивалент 4.18 на RHEL 8.10); netkit вместо veth включаем то...

System Requirements, docs.cilium.io v1.20

true

kubeProxyReplacement: отказ от kube-proxy, socket-LB в ядре; strict/partial упразднены

Kubernetes Without kube-proxy, docs.cilium.io v1.20

maglev

loadBalancer.algorithm: консистентное хеширование бэкендов при отказе узлов, tableSize 16381

Maglev Consistent Hashing, docs.cilium.io v1.20; наш станда...

2

operator.replicas — два cilium-operator на разных узлах, иначе IPAM встаёт при отказе узла

наш стандарт ITfresh

255

лимит кластеров Cluster Mesh (511 при maxConnectedClusters=511); cluster.id и maxConnectedClus...

Setting up Cluster Mesh, docs.cilium.io v1.20

РЕАЛИЗАЦИЯ 1 ИЗ 3

Миграция действующего кластера на Cilium без даунтайма

Что настраиваем

Кластер 3 control-plane + 6 worker, самосборный Kubernetes, исходный стек — iptables-CNI + kube-proxy

Как мы это делаем

1. Ставим Cilium вторичным overlay: routingMode=tunnel, tunnelProtocol=vxlan, нестандартный VXLAN-порт 8473, cni.customConf=true и policyEnforcementMode=never — ни кон…
2. Через CRD CiliumNodeConfig переводим узлы по одному: cordon → drain → метка миграции узла → перезапуск агента Cilium на узле → перезагрузка узла → uncordon; трафик…
3. После обхода всех узлов сносим старый CNI и DaemonSet kube-proxy, включаем kubeProxyReplacement=true, возвращаем штатный порт VXLAN 8472 и enforcement политик
4. Верификация каждого шага: cilium status --wait, cilium connectivity test, контроль дропов через hubble observe --verdict DROPPED

РЕЗУЛЬТАТ

Кластер переезжает на eBPF-дataплейн по одному узлу, рабочие нагрузки переезжают штатным drain'ом без окна простоя. После удаления kube-proxy сервисный путь короче: балансировка сервисов выполняется в ядре на syscall connect(), а не на каждом пакете.

КЛЮЧЕВОЙ НЮАНС

Пока в кластере два CNI, держим policyEnforcementMode=never — политики не работают вовсе; default-deny включаем только после обхода всех узлов и возврата enforcement в default. Остатки iptables-правил kube-pro…

РЕАЛИЗАЦИЯ 2 ИЗ 3

Сегментация: CiliumNetworkPolicy L3/L4/L7 и Hubble

Что настраиваем

Прод-namespace'ы бэкендов и веб-фронтов; enforcement на каждом узле, наблюдаемость через hubble-relay

Как мы это делаем

1. Включаем Hubble: hubble.relay.enabled=true, hubble.ui.enabled=true, метрики hubble.metrics.enabled={dns,drop,tcp,flow,port-distribution,icmp,httpV2}
2. 1–2 недели снимаем фактическую матрицу трафика через hubble observe и Hubble UI — политики генерим из реальных потоков, а не «по памяти» разработчиков
3. Пишем CiliumNetworkPolicy: ingress fromEndpoints по label'ам + toPorts; L7-правила rules.http (method/path) вешаем точечно на критичные API
4. Default-deny вводим поэтапно: сначала policy-audit-mode на endpoint'ах и разбор вердиктов через hubble observe --type policy-verdict, затем enforce
5. Алертинг: rate(hubble_drop_total{reason="POLICY_DENIED"}[5m]) — всплеск дропов после релиза ловим до жалоб пользователей

РЕЗУЛЬТАТ

Сегментация переживает пересоздание подов и смену IP: правила привязаны к security identity (label'ам), а не адресам. Дежурный видит по Hubble, какой запрос кем заблокирован, с точностью до HTTP-метода и пути — разбор инцидента сжимается с часов до минут.

КЛЮЧЕВОЙ НЮАНС

Каждое L7-правило заворачивает трафик endpoint'а в Envoy-прокси (DaemonSet cilium-envoy): закладываем +1–2 мс на hop и ресурсы под прокси. Поэтому L7 — точечно, базовая сегментация — на L3/L4 в eBPF без прокси.

РЕАЛИЗАЦИЯ 3 ИЗ 3

Тюнинг датаплейна и мониторинг eBPF-карт

Что настраиваем

Прод-узлы bare-metal и BM; профиль фиксируем в helm values, метрики — в Prometheus/Grafana

Как мы это делаем

1. В плоской L2-сети включаем routingMode=native + autoDirectNodeRoutes=true и bpf.masquerade=true — маскарадинг в eBPF вместо iptables-MASQUERADE
2. bandwidthManager.enabled=true (EDT-шейпинг) + bandwidthManager.bbr.enabled=true (нужно ядро ≥5.18); лимиты подам — аннотацией kubernetes.io/egress-bandwidth
3. Для внешних LB на bare-metal: loadBalancer.mode=hybrid (DSR для TCP, SNAT для UDP), loadBalancer.acceleration=native на NIC с XDP-драйвером
4. Следим за cilium_bpf_map_pressure: порог алерта 0.9; ёмкость карт масштабируем от RAM узла через bpf.mapDynamicSizeRatio=0.0025
5. prometheus.enabled=true и operator.prometheus.enabled=true; дашборды агента, оператора и Hubble — в общий Grafana клиента

РЕЗУЛЬТАТ

Сервисный путь без туннельной инкапсуляции и без iptables: меньше латентность и CPU на узел под тем же трафиком. Переполнение conntrack-карт (обрывы соединений «без причины») ловим алертом по map_pressure заранее, а не постфактум.

КЛЮЧЕВОЙ НЮАНС

XDP работает не на всех драйверах NIC: сверяем карту со списком в доке, на BM держим acceleration=disabled. DSR требует, чтобы сеть пропускала ответы бэкендов мимо LB-узла.

ЧТО МЫ ВЫУЧИЛИ НА ПРАКТИКЕ

Подводные камни

- ✕ **kubeProxyReplacement=strict из старых гайдов**
  Значения probe/partial/strict упразднены — свежие версии принимают только true/false. Скопированный старый helm-набор валит установку.

- ✕ **Старое ядро на части узлов**
  Ниже 5.10 агент не стартует или фичи тихо отключаются. Проверяем uname -r на всех узлах до внедрения; для RHEL учитываем бэкпорты (4.18 в 8.10).

- ✕ **Пересекающиеся PodCIDR при Cluster Mesh**
  Mesh требует уникальных PodCIDR и cluster.id. Разные ipam.operator.clusterPoolIPv4PodCIDRList закладываем заранее — потом не поменять без боли.

- ✕ **L7-политика на весь namespace**
  Любое rules.http гонит трафик через Envoy — латентность и CPU растут на весь namespace. L7 вешаем только там, где нужен контроль API.

- ✕ **Апгрейд через минорную версию**
  Прыжок 1.18→1.20 не поддерживается. Идём по одной минорке, перед каждой — preflight-DaemonSet cilium-pre-flight-check (прогрев образов, валидация CNP…

- ✕ **Забытый kube-proxy после перехода**
  Оставленный DaemonSet продолжает писать iptables-правила и конфликтует с socket-LB. Удаляем kube-proxy и вычищаем его цепочки на каждом узле.

- ✕ **Смешение NetworkPolicy и CNP вслепую**
  Семантика default-deny у NetworkPolicy и CNP различается — вперемешку получаются дыры. Итог аудитим по policy-verdict-событиям в Hubble.

- ✕ **XDP на неподдержанном драйвере NIC**
  loadBalancer.acceleration=native вне списка драйверов деградирует или ломает bonding. Сверяем драйвер с докой; на BM акселерацию не включаем.

УРОВНИ РЕАЛИЗАЦИИ

Как правильно

МИНИМУМ

- Cilium 1.19/1.20, ядро ≥5.10 на всех узлах, kubeProxyReplacement=true
- operator.replicas=2, Hubble Relay + UI, метрики dns/drop/tcp/flow в Prometheus
- cilium status --wait и cilium connectivity test после каждого изменения values

НОРМАЛЬНО

- Default-deny по namespace: CiliumNetworkPolicy из фактических потоков Hubble
- routingMode=native + bpf.masquerade=true, bandwidthManager с BBR (ядро ≥5.18)
- Алерты: hubble_drop_total (POLICY_DENIED), cilium_bpf_map_pressure > 0.9
- Апгрейды по одной минорке с preflight-проверкой и планом отката

ХОРОШО

- L7-политики на критичные API, шифрование узлов: encryption.type=wireguard
- Cluster Mesh на DR-кластер: уникальные cluster.id, непересекающиеся PodCIDR
- Gateway API вместо Ingress: gatewayAPI.enabled=true на cilium-envoy
- DSR/hybrid + Maglev + XDP-акселерация для bare-metal балансировки

ПРОВЕРЬТЕ У СЕБЯ

Чек-лист самопроверки

- [ ] Ядро на всех узлах ≥5.10 и bpf-файловая система смонтирована в /sys/fs/bpf?
- [ ] kube-proxy удалён, kubeProxyReplacement=true и socket-LB подтверждён в cilium status?
- [ ] cilium connectivity test проходит в прод-кластере без FAIL?
- [ ] В прод-namespace действует default-deny, а не отдельные разрозненные политики?
- [ ] Алерт на hubble_drop_total{reason=POLICY_DENIED} заведён и уходит дежурному?
- [ ] cilium_bpf_map_pressure на всех узлах ниже 0.9 и под алертом?
- [ ] cilium-operator в двух репликах на разных узлах?
- [ ] PodCIDR кластеров не пересекаются и cluster.id уникальны (если планируем Cluster Mesh)?
- [ ] План апгрейда по одной минорной версии с preflight-проверкой задокументирован?
- [ ] Трафик между узлами через недоверенную сеть зашифрован (WireGuard)?

Если хотя бы на два вопроса ответ «нет» или «не знаю» — тема требует внимания.

ЧТО МЫ ДЕЛАЕМ

Как поможет ITFresh

ITFresh — ИТ-аутсорсинг для юридических лиц до 50 рабочих мест в Москве и области. 15+ лет практики, собственная инфраструктура в дата-центре МТС (8 серверов Dell Xeon Platinum).

- Миграция действующего кластера с kube-proxy/iptables-CNI на Cilium без остановки сервисов
- Проектирование политик L3–L7 по фактическим потокам Hubble и поэтапный ввод default-deny
- Наблюдаемость под ключ: Hubble Relay/UI, метрики в Prometheus/Grafana, алертинг на дропы и BPF-карты
- Cluster Mesh и WireGuard-шифрование для гео-резерва и DR-площадки
- Регламентные апгрейды Cilium с preflight и аудит датаплейна (карты, XDP, DSR, bandwidth manager)

15+
лет в ИТ-поддержке

50
рабочих мест — наш профиль

МТС
дата-центр, Москва

КОНТАКТЫ

Обсудить вашу задачу

Сайт	**itfresh.ru**
Телефон	**+7 903 729-62-41**
Telegram	**@ITfresh_Boss**

Бесплатно посмотрим вашу инфраструктуру по этому чек-листу и скажем, где тонко — без обязательств.

itfresh.ru

ДОКУМЕНТАЦИЯ И НАШ СТЕК

Техническая база

01 System Requirements (docs.cilium.io — v1.20)

02 Kubernetes Without kube-proxy (docs.cilium.io — v1.20)

03 Migrating a cluster to Cilium (docs.cilium.io — v1.20)

04 Network Policy (CiliumNetworkPolicy) (docs.cilium.io — v1.20)

05 Setting up Cluster Mesh (docs.cilium.io — v1.20)

06 Hubble Observability & Metrics (docs.cilium.io — v1.20)

07 Upgrade Guide (preflight) (docs.cilium.io — v1.20)

08 Bandwidth Manager (EDT, BBR) (docs.cilium.io — v1.20)

09 Шаблон helm values ITfresh cilium-base.yaml (наш стандарт — 2026)

Основано на официальной документации продуктов и нашей практике внедрения.