

Scaling container policy management with kernel features

Joe Stringer

Cilium.io

Linux Plumbers 2019, Lisbon, Portugal



Overview

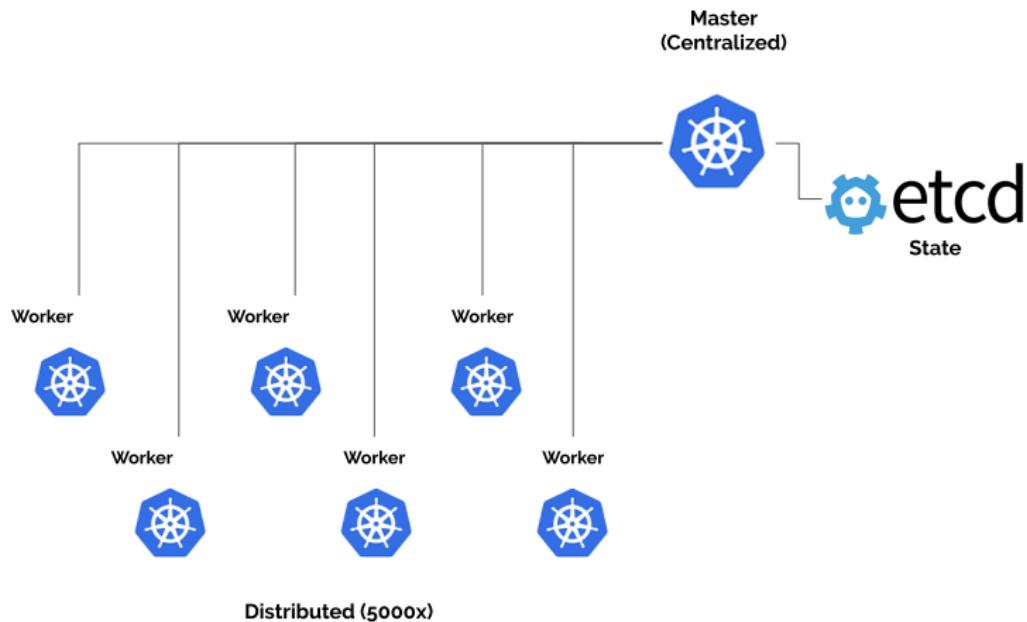
1 Background

2 Pod event processing

3 Identity-based security

4 Layer 7 security

Kubernetes Architecture 101



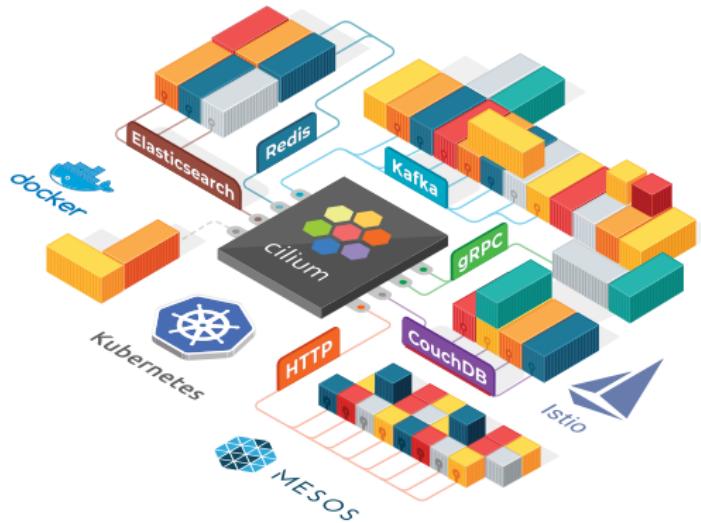
Kubernetes networking plugins

- Plumb local connectivity (CNI)
- Connect remote nodes
- Services / loadbalancing
- Network policy



Cilium

- Native BPF dataplane
- Modular architecture
- API-aware authorization
- Scalability



Cluster events

- Nodes → Fabric connectivity
- Pods → End-to-end connectivity
- Services → Logical connectivity
- Network policies → Connectivity isolation

Cluster events

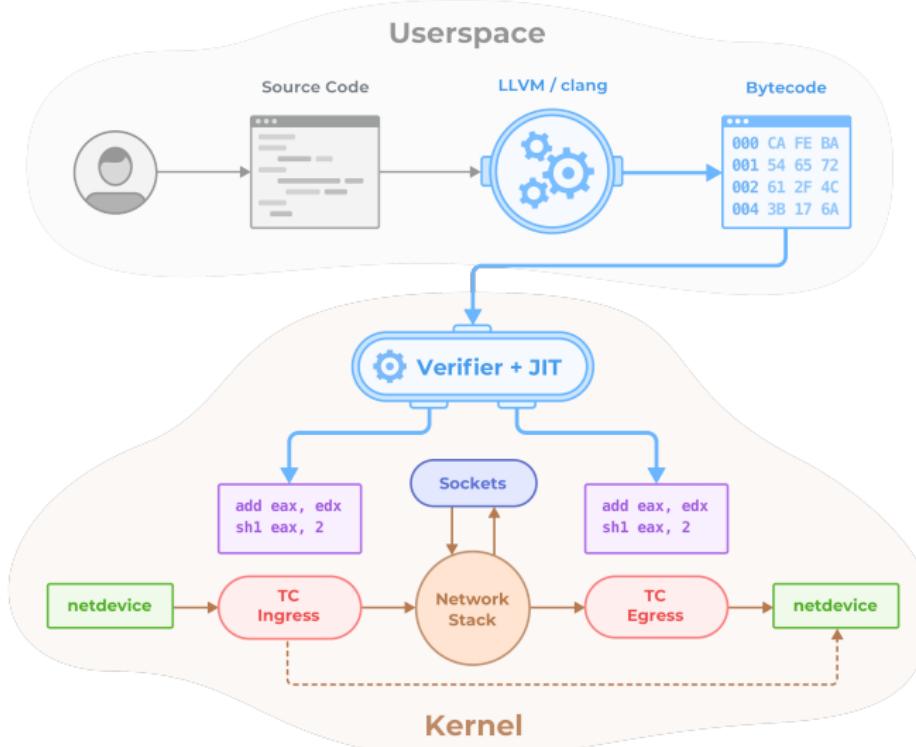
- Nodes → Fabric connectivity
- **Pods** → End-to-end connectivity
- Services → Logical connectivity
- **Network policies** → Connectivity isolation

Pod event processing

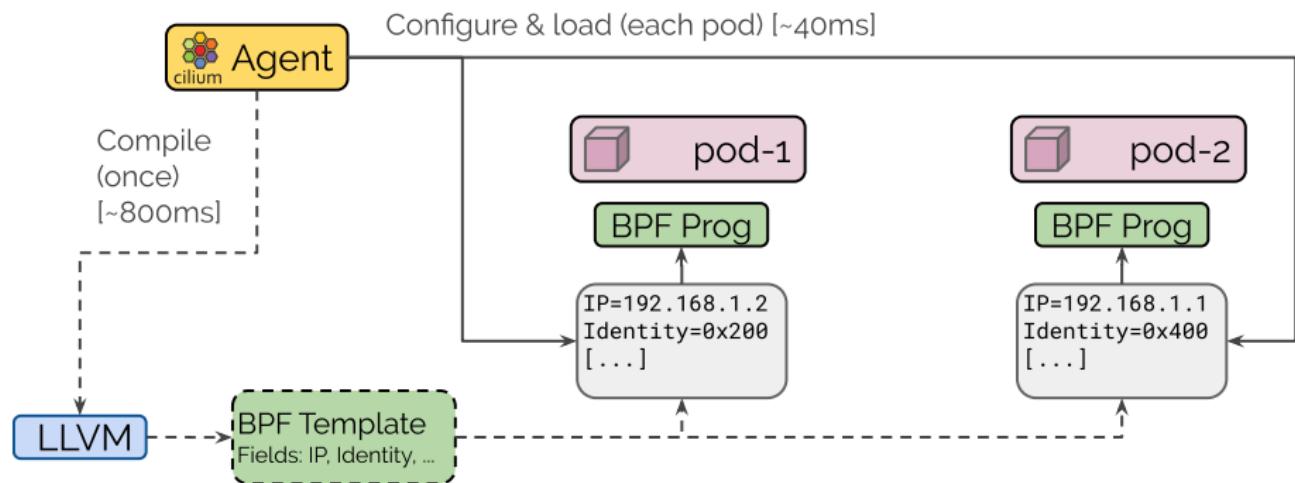
What does it mean to scale?

- Minimize unnecessary events
 - Reduce load on centralized components
 - Minimize event sizes
- Roll out bursty workloads quickly
- Reduce time from policy change to enforcement

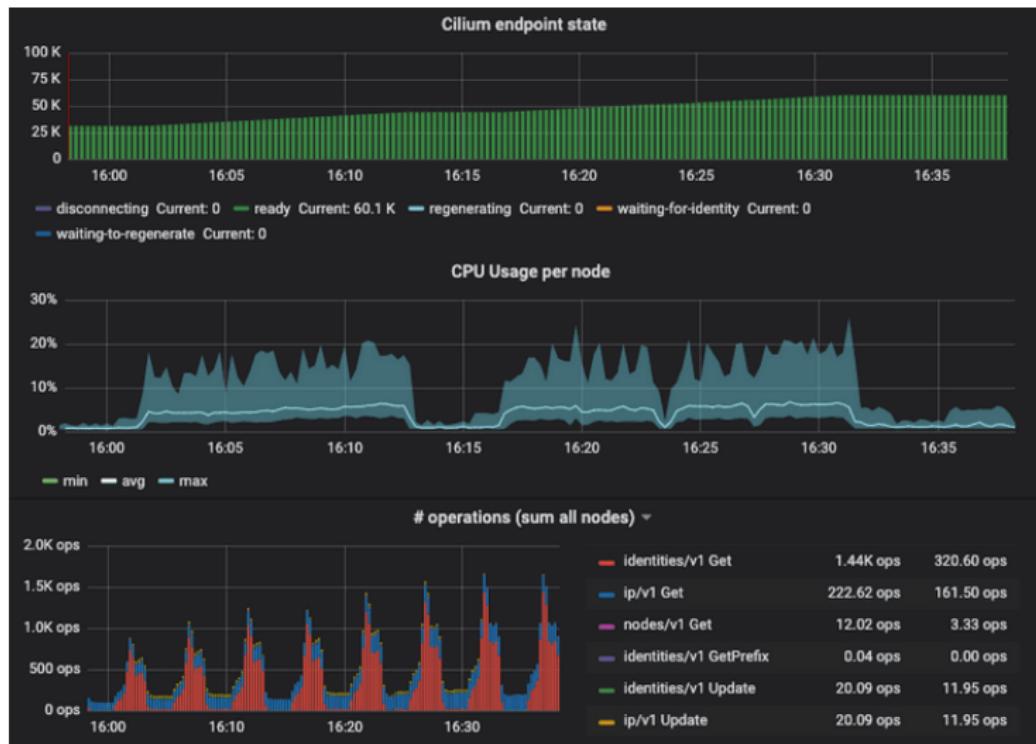
BPF plumbing



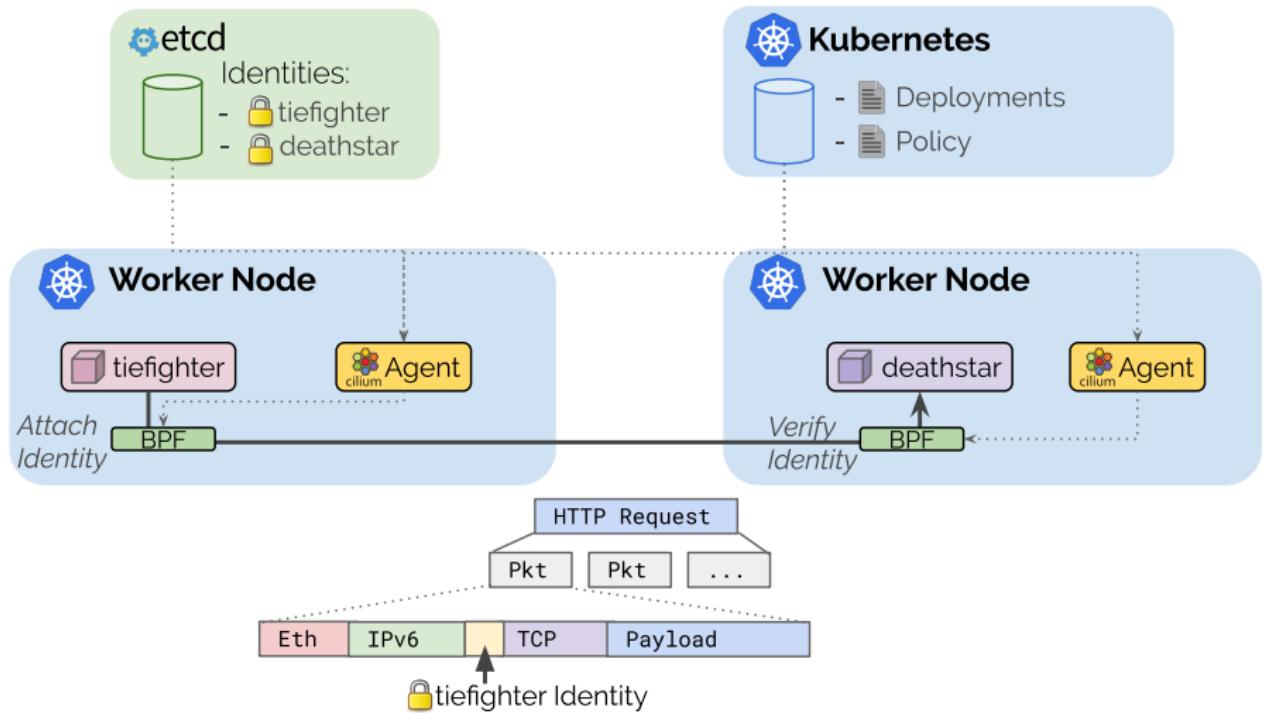
ELF Templating



1K nodes: Scaling to 60k pods



Identity-based security



Policy example

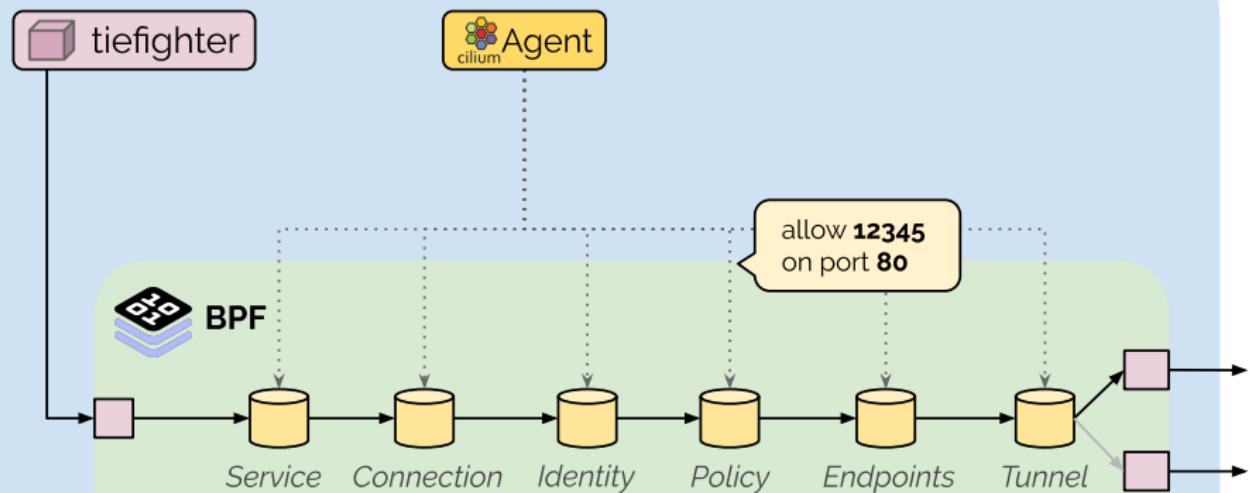
```
apiVersion: "cilium.io/v2"
kind: CiliumNetworkPolicy
description: "Restrict deathstar access to empire ships"
metadata:
  name: "deathstar-ingress"
spec:
  endpointSelector:
    matchLabels:
      org: empire
      class: deathstar
  ingress:
    - fromEndpoints:
        - matchLabels:
            org: empire
  toPorts:
    - ports:
        - port: "80"
          protocol: TCP
```

Label selectors

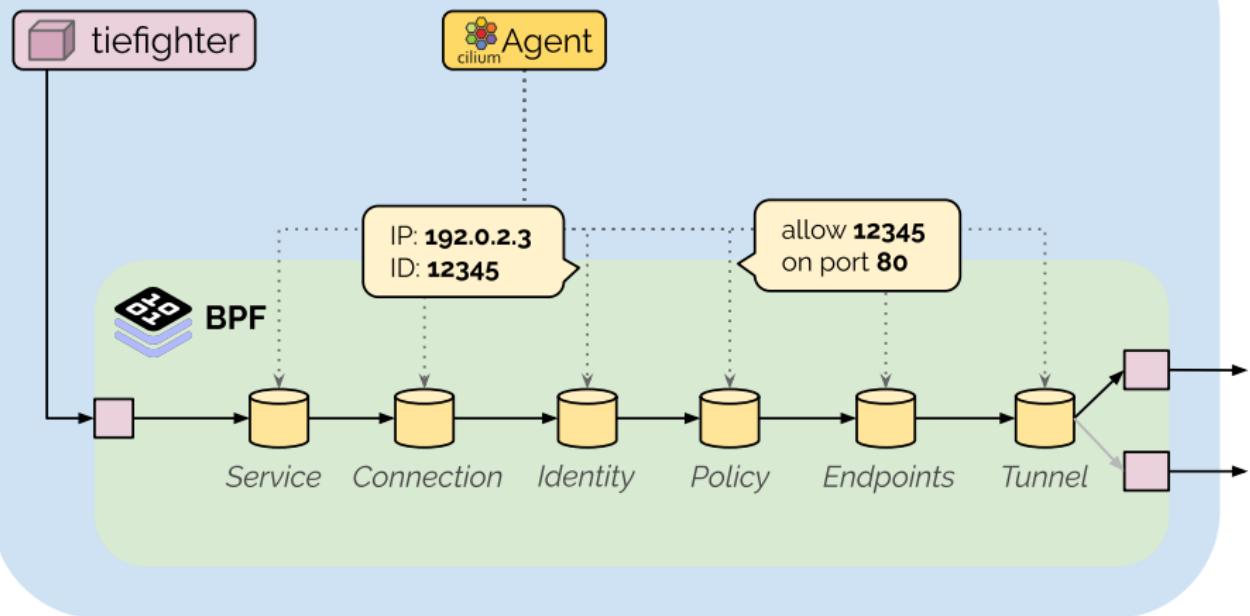
- 12345 {org:empire, class:deathstar}
- 12468 {org:empire, class:tiefighter}
- 12465 {org:alliance, class:xwing}

matchLabels: org: empire	12345 {org:empire, class:deathstar} 12468 {org:empire, class:tiefighter}
matchLabels: org: empire class: deathstar	12345 {org:empire, class:deathstar}

Datapath Configuration

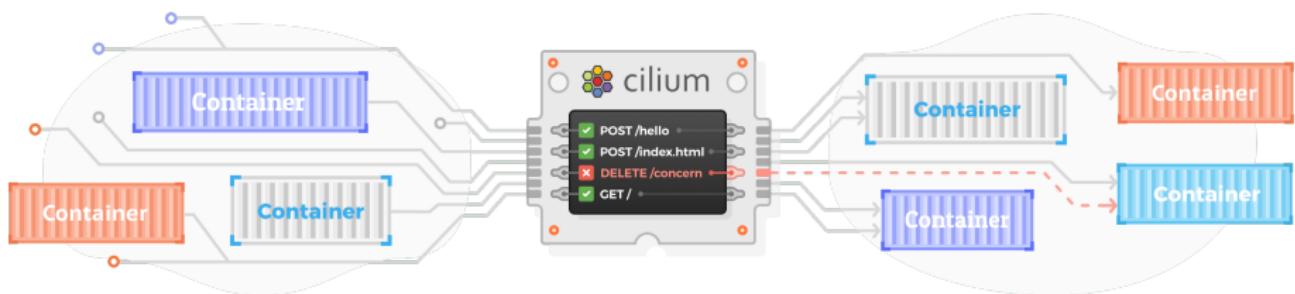


Datapath Configuration

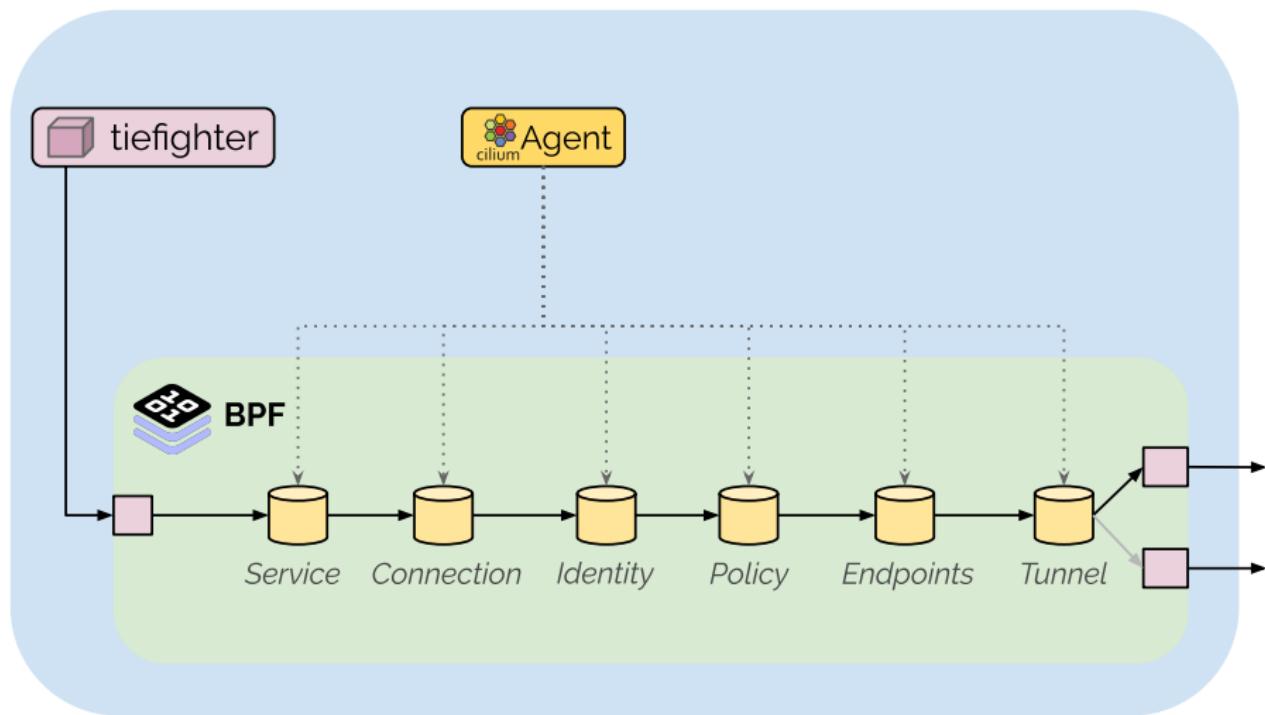


Layer 7 security

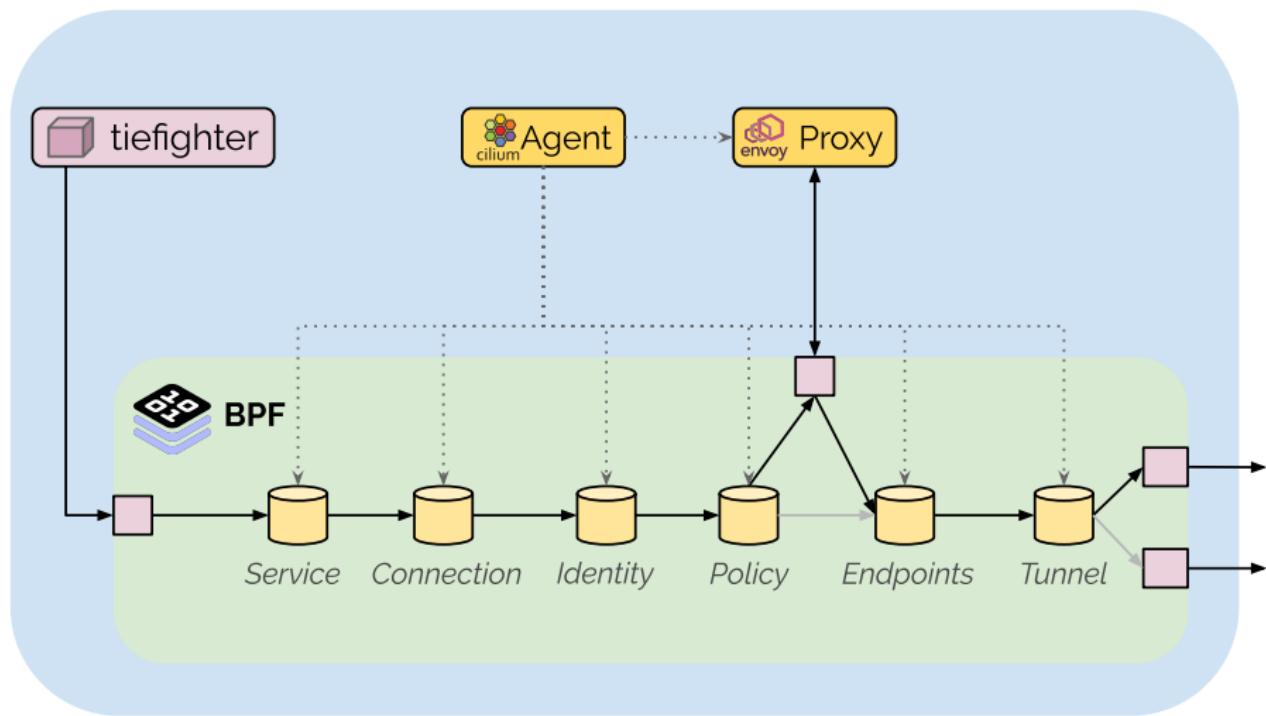
L7 is the new L4



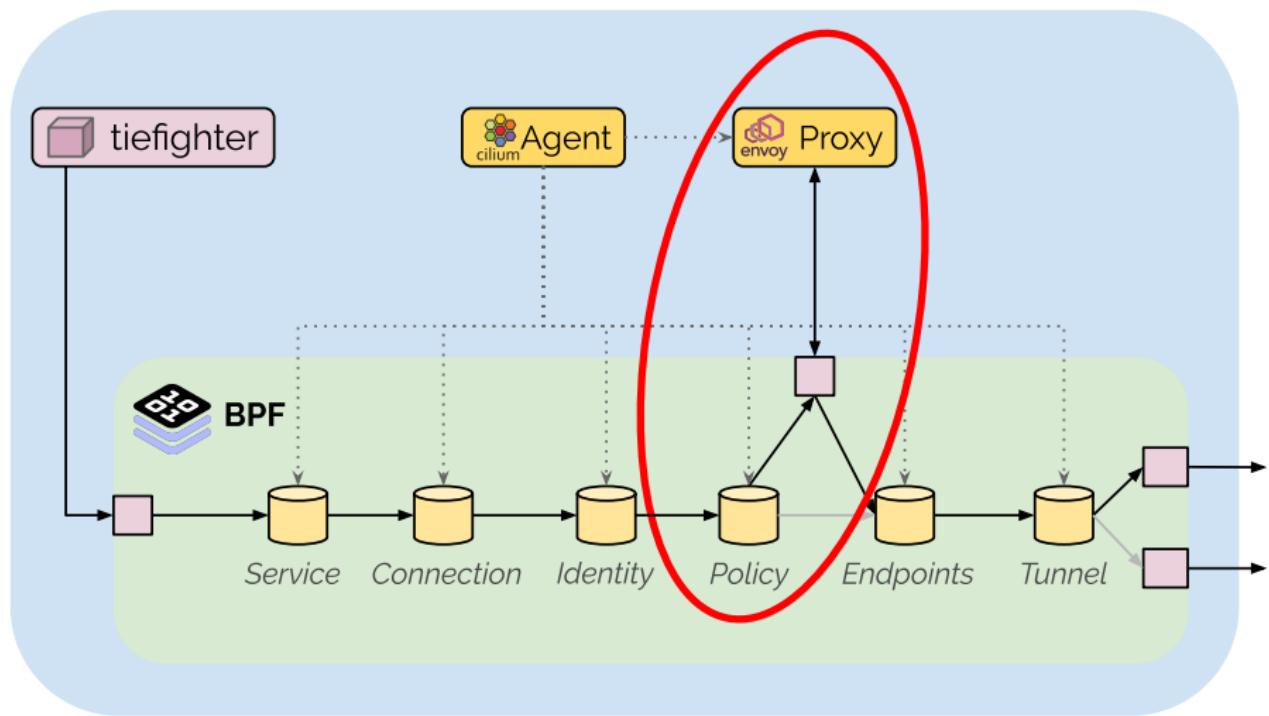
Datapath Configuration: L3 flow



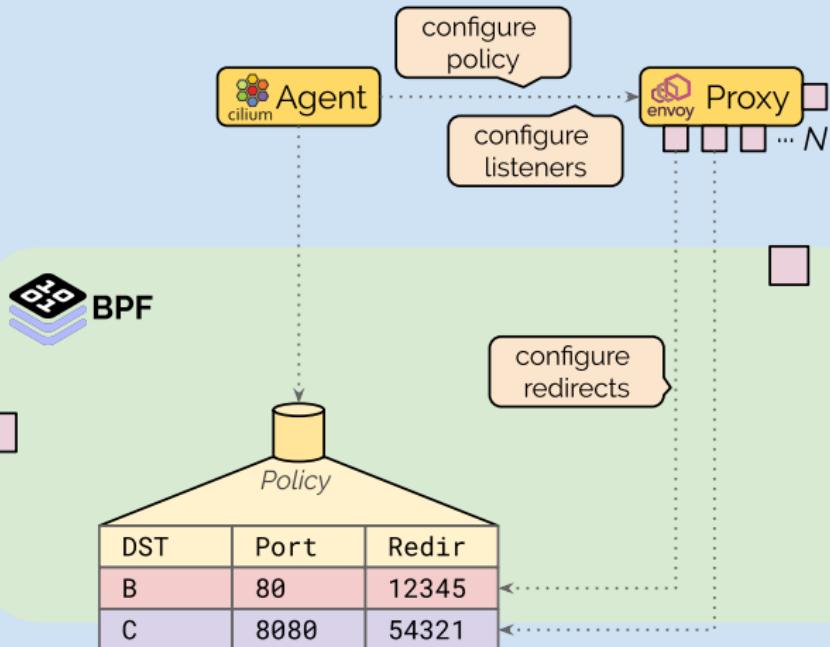
Datapath Configuration: L7 flow



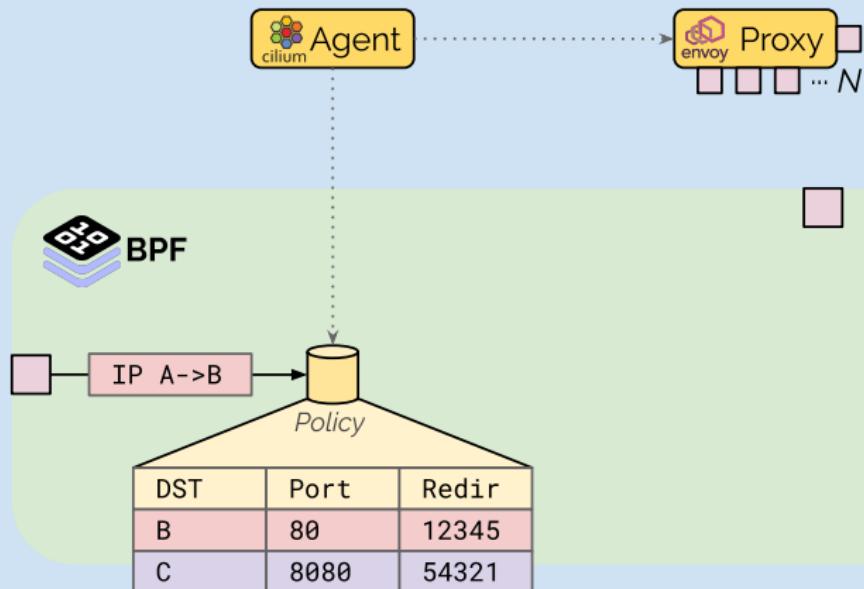
Datapath Configuration: L7 flow



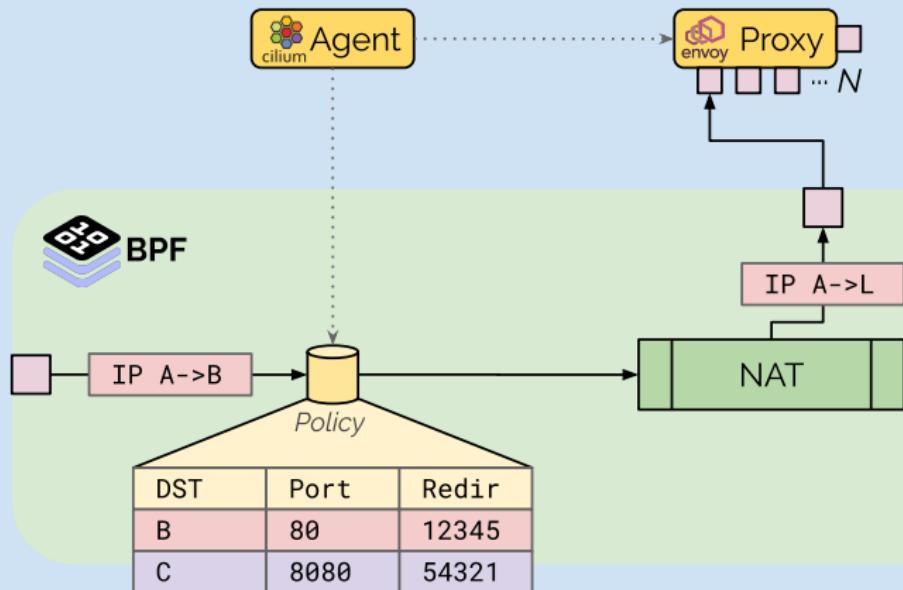
L7 Configuration: Past



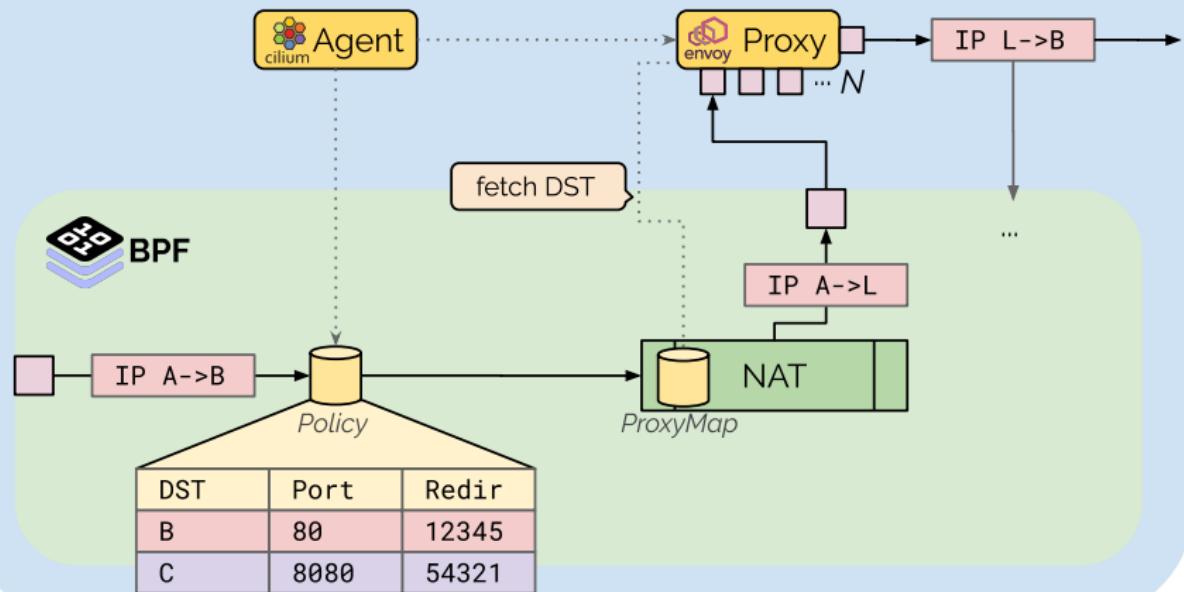
L7 Configuration: Past



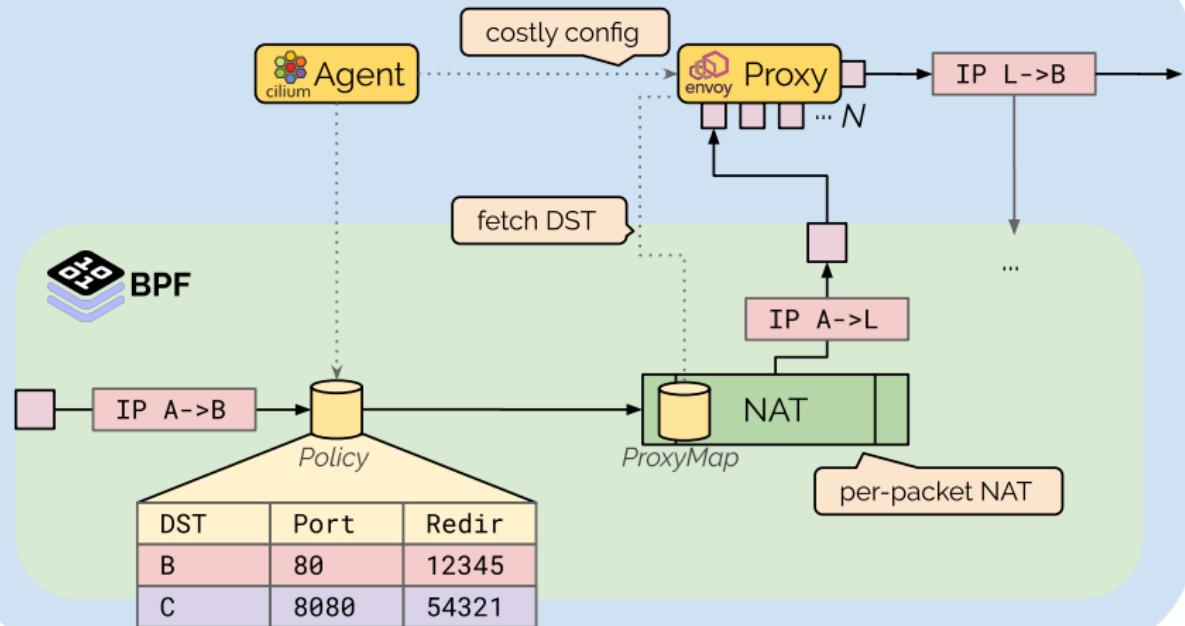
L7 Configuration: Past



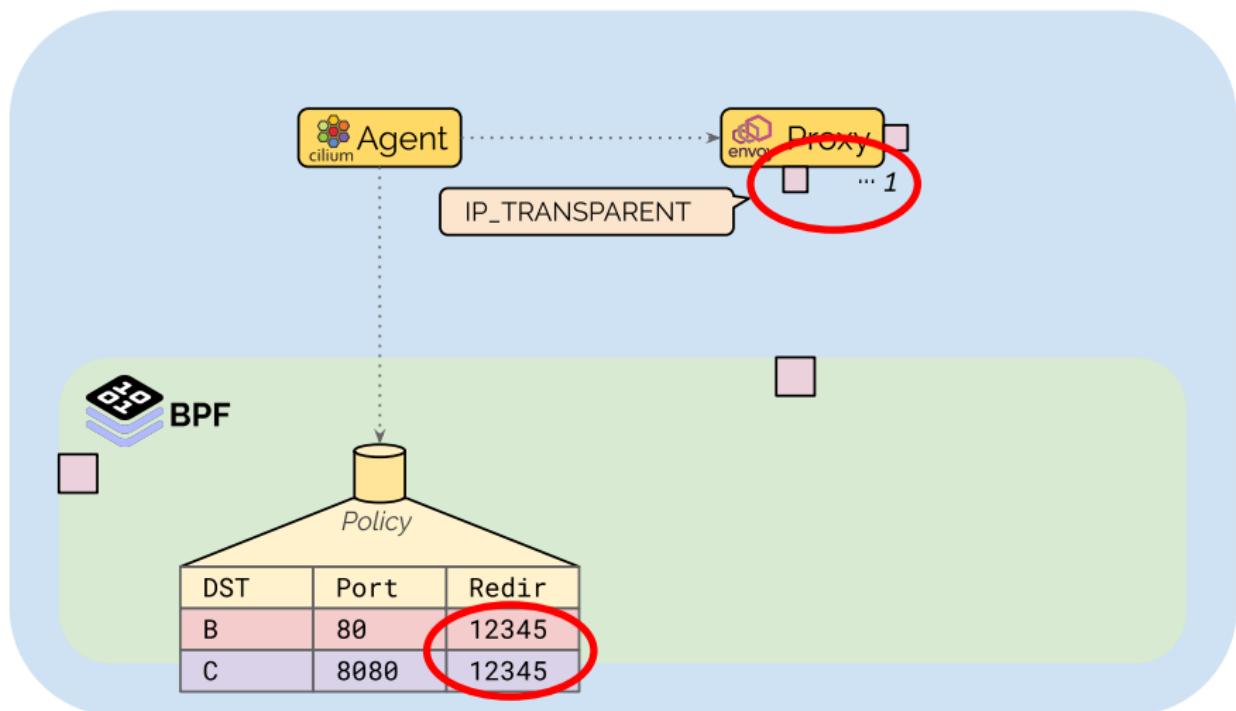
L7 Configuration: Past



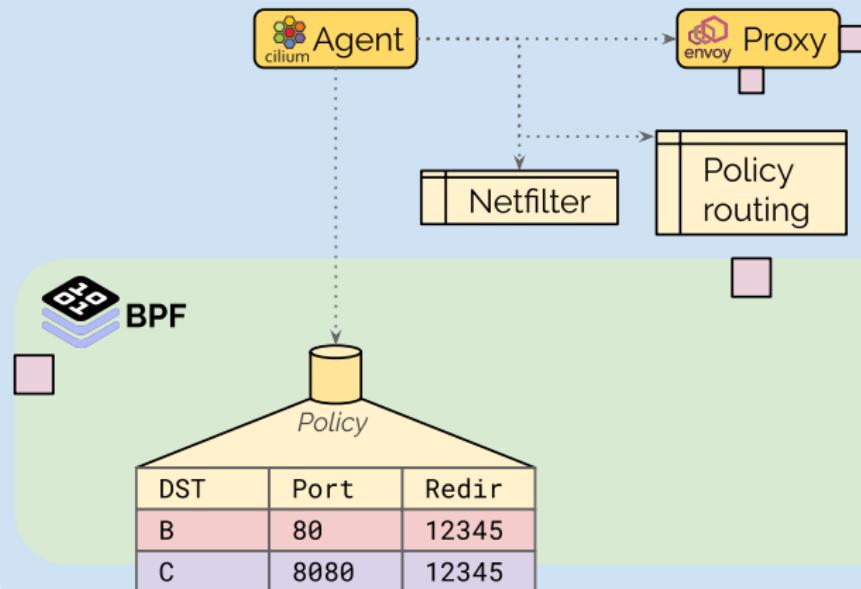
L7 Configuration: Past



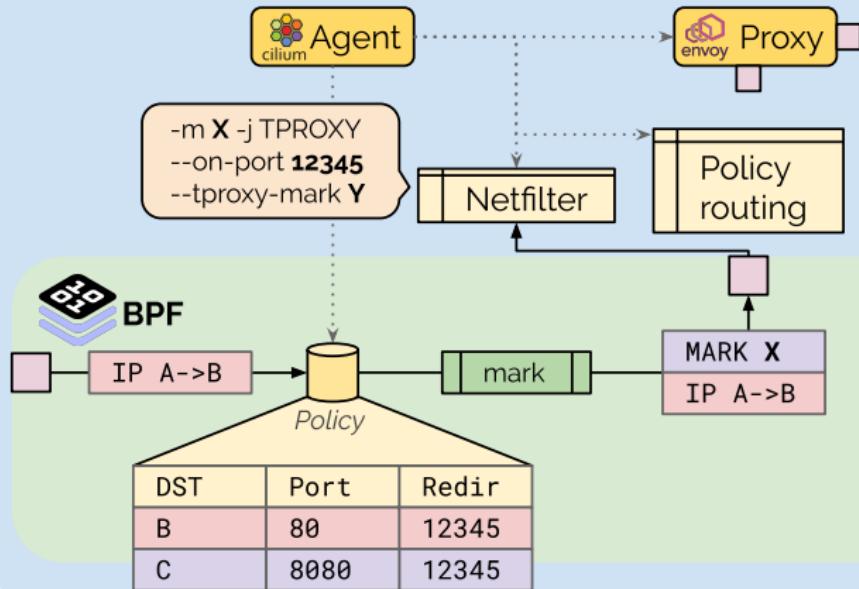
L7 Configuration: Present



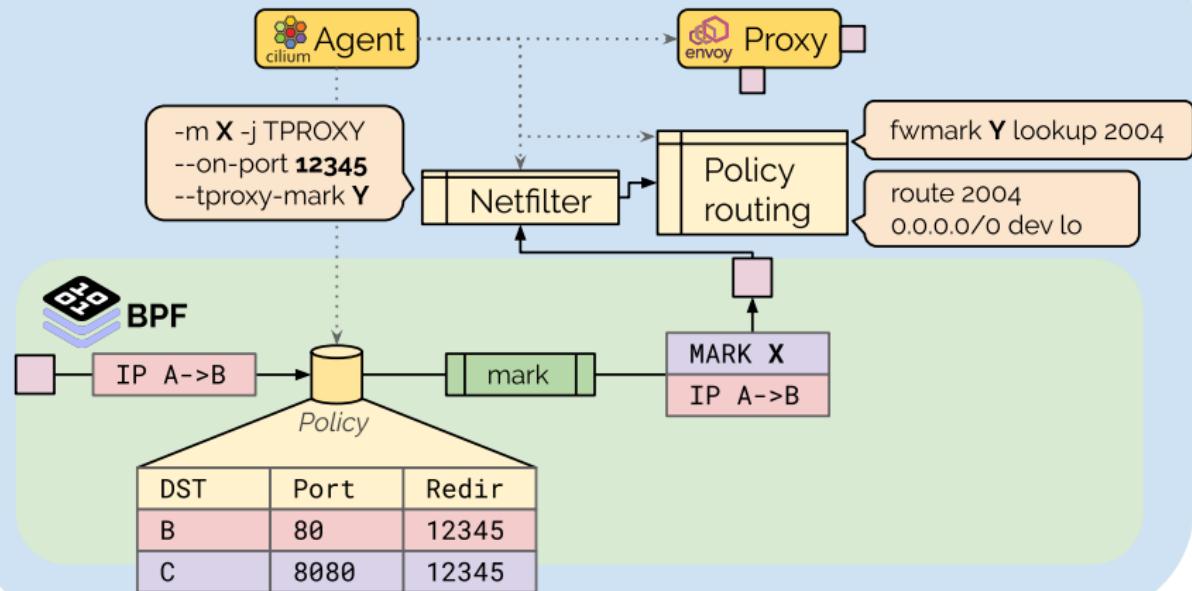
L7 Configuration: Present



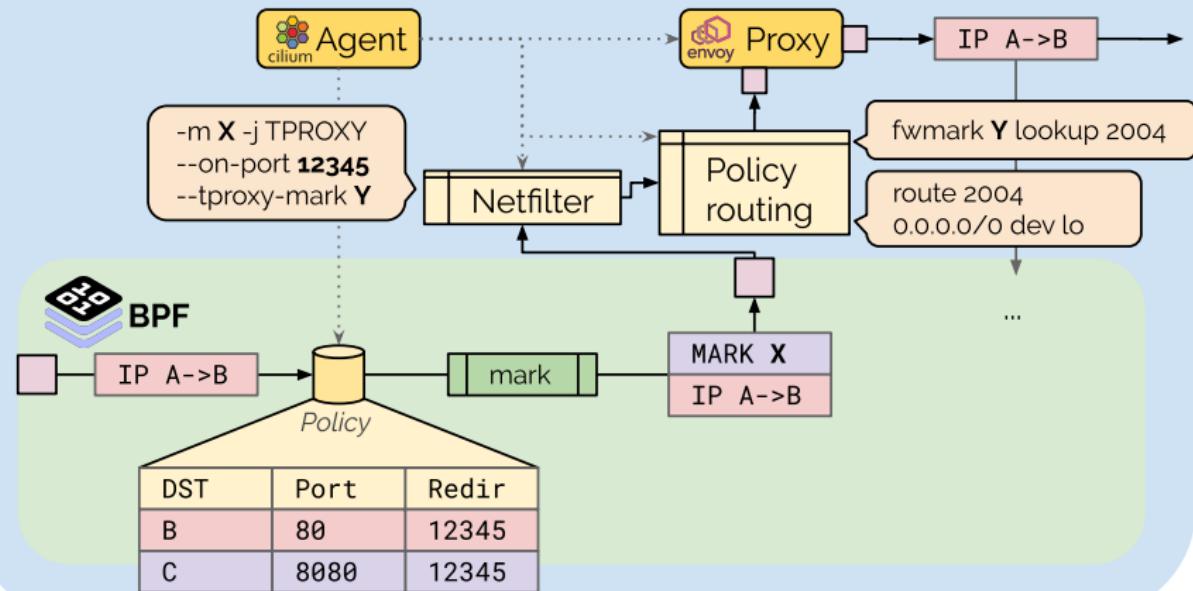
L7 Configuration: Present



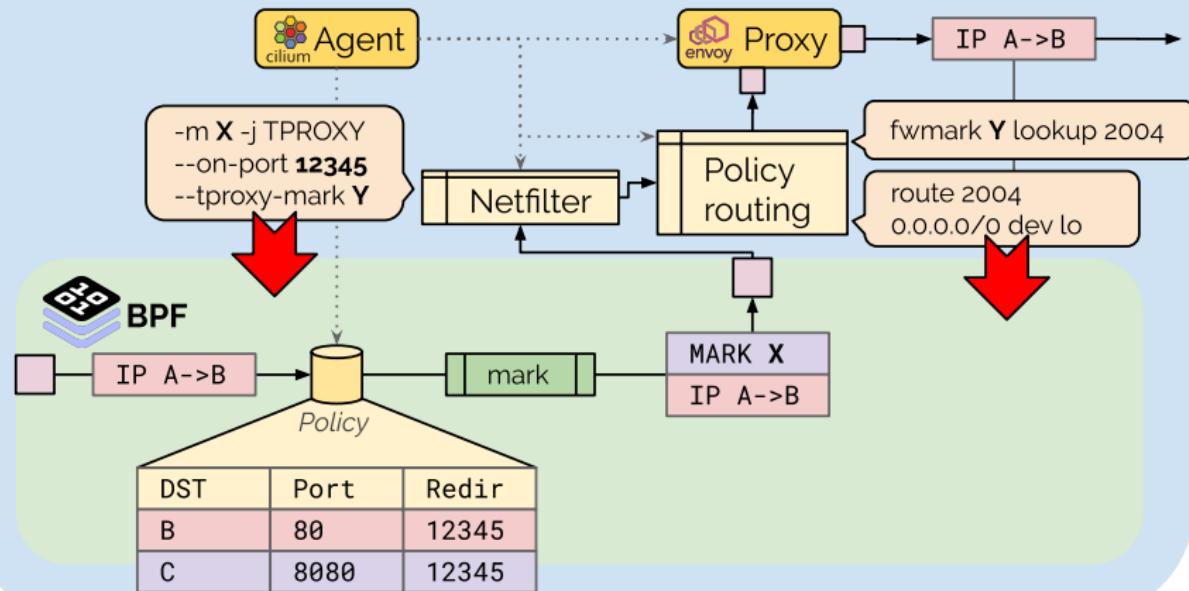
L7 Configuration: Present



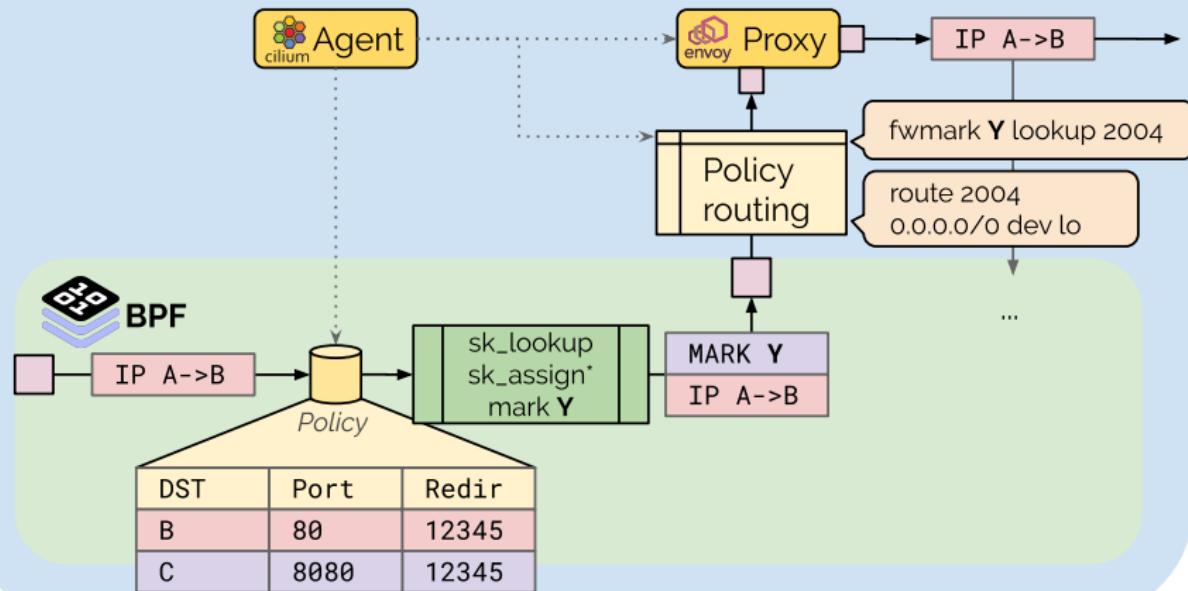
L7 Configuration: Present



L7 Configuration: Proposal



L7 Configuration: Proposal



Socket assign: RFC

```
/* int bpf_sk_assign(struct sk_buff *skb, struct bpf_sock *sk,
 *                     u64 flags)
 *   Description
 *     Assign the *sk* to the *skb*.
 *
 *   This operation is only valid from TC ingress path.
 *
 *   The *flags* argument must be zero.
 *
 * Return
 *   0 on success, or a negative errno in case of failure.
 *
 *   * **-EINVAL**: Unsupported flags specified.
 *   * **-EOPNOTSUPP**: Unsupported operation, for example
 *     a call from outside of TC ingress.
 *   * **-ENOENT**: The socket cannot be assigned.
 */
```

Socket assign: Hiccup

- BPF progs are attached at TC ingress
- `skb_orphan()` invoked directly before PREROUTING

¹ <https://www.mail-archive.com/netdev@vger.kernel.org/msg303851.html>

² <https://www.mail-archive.com/netdev@vger.kernel.org/msg304057.html>

Socket assign: Hiccup

- BPF progs are attached at TC ingress
- `skb_orphan()` invoked directly before PREROUTING
- TC folks are already carrying hacks for this¹

¹ <https://www.mail-archive.com/netdev@vger.kernel.org/msg303851.html>

² <https://www.mail-archive.com/netdev@vger.kernel.org/msg304057.html>

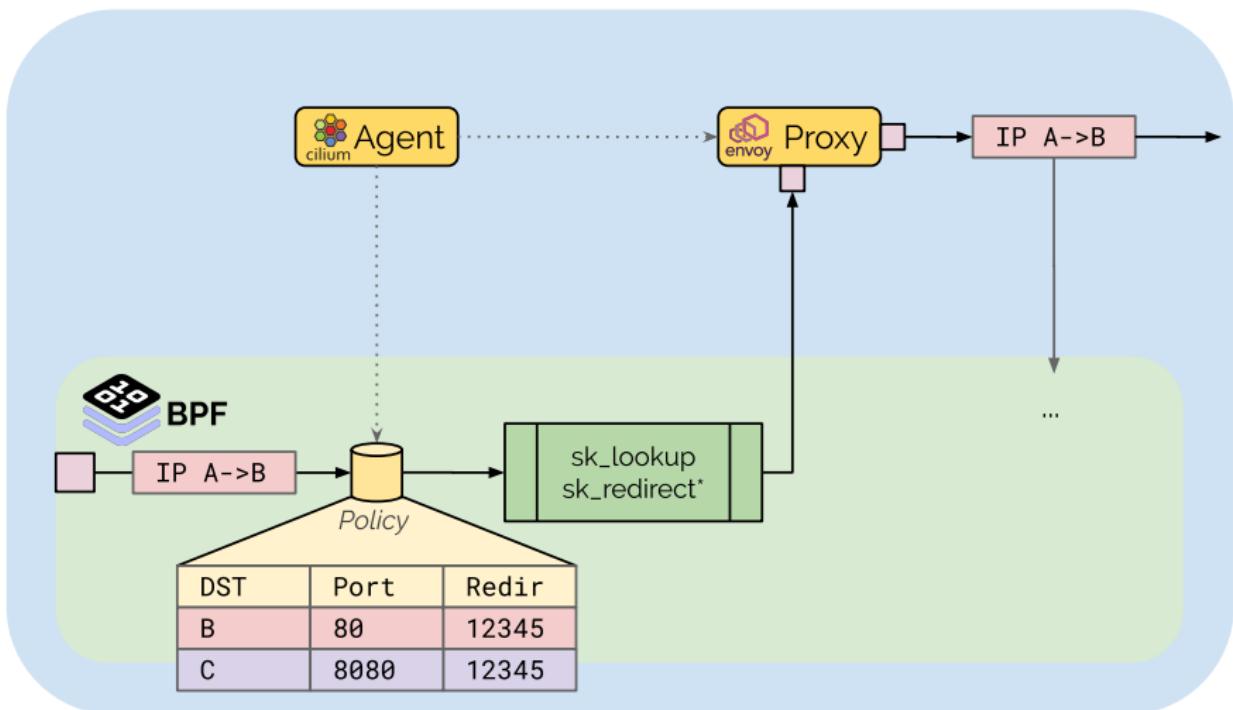
Socket assign: Hiccup

- BPF progs are attached at TC ingress
- `skb_orphan()` invoked directly before PREROUTING
- TC folks are already carrying hacks for this¹
- Just move to `__dev_forward_skb()`²?

¹ <https://www.mail-archive.com/netdev@vger.kernel.org/msg303851.html>

² <https://www.mail-archive.com/netdev@vger.kernel.org/msg304057.html>

L7 Configuration: Socket redirect



Summary

- Minimize the number of events
- Separate concerns: Policy vs addressing
- Frontload expensive operations
- ... while keeping runtime costs low

Thank you

More information

-  <https://cilium.io>
-  <https://cilium.io/slack>
-  <https://github.com/cilium/cilium>
-  <https://twitter.com/ciliumproject>



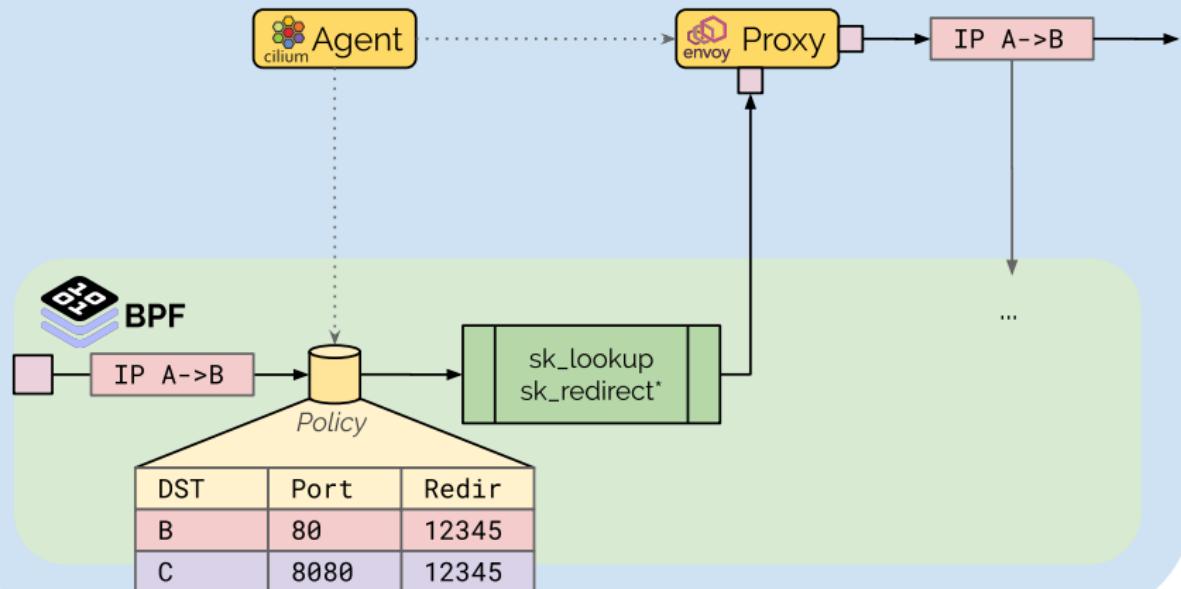
Thank you

More information

-  <https://cilium.io>
-  <https://cilium.io/slack>
-  <https://github.com/cilium/cilium>
-  <https://twitter.com/ciliumproject>



Socket redirect



Socket assign: API quirks

- Nit: Need to use `sock_common` socket lookups
- Add `bpf_skc_lookup_udp()`
- Optimization: Add lookup flags to `bpf_sk*_lookup_*`()