1. Introduction of eBPF
2. Introduction of Cilium
3. Cilium data path for DSR
4. Cilium DSR improvement
5. Q&A
What is eBPF

- An instruction set based on BPF
- RISC, JIT
- A sandboxed program in kernel
- Lightweight, safe, portable

https://ebpf.io/
Why eBPF

- Hook pre-defined kernel functions
- A programmable interface, much more powerful than procfs/sysfs/syscall
- Safer, lighter and finer-granularity than kernel modules
- Compile-Once Run-Everywhere (?)
eBPF Architecture

- eBPF instruction set
- eBPF in-kernel verifier and JIT
- eBPF syscalls via bpf()
- Kernel components: programs, maps, helper functions, iterators, BTF, XDP etc.
- LLVM support
- Libbpf, bpftools
# ./tcpconnect -t

<table>
<thead>
<tr>
<th>TIME(s)</th>
<th>PID</th>
<th>COMM</th>
<th>IP</th>
<th>SADDR</th>
<th>DADDR</th>
<th>DPORT</th>
</tr>
</thead>
<tbody>
<tr>
<td>31.871</td>
<td>2482</td>
<td>local_agent</td>
<td>4</td>
<td>10.103.219.236</td>
<td>10.251.148.38</td>
<td>7001</td>
</tr>
<tr>
<td>31.874</td>
<td>2482</td>
<td>local_agent</td>
<td>4</td>
<td>10.103.219.236</td>
<td>10.101.3.132</td>
<td>7001</td>
</tr>
<tr>
<td>31.878</td>
<td>2482</td>
<td>local_agent</td>
<td>4</td>
<td>10.103.219.236</td>
<td>10.171.133.98</td>
<td>7101</td>
</tr>
<tr>
<td>90.917</td>
<td>2482</td>
<td>local_agent</td>
<td>4</td>
<td>10.103.219.236</td>
<td>10.251.148.38</td>
<td>7001</td>
</tr>
<tr>
<td>90.928</td>
<td>2482</td>
<td>local_agent</td>
<td>4</td>
<td>10.103.219.236</td>
<td>10.102.64.230</td>
<td>7001</td>
</tr>
<tr>
<td>90.938</td>
<td>2482</td>
<td>local_agent</td>
<td>4</td>
<td>10.103.219.236</td>
<td>10.115.167.169</td>
<td>7101</td>
</tr>
</tbody>
</table>

https://github.com/iovisor/bcc/blob/master/tools/tcpconnect_example.txt
Cilium

- open source software providing, securing and observing network connectivity between container workloads
- High scale, low overhead

https://cilium.io/
Cilium at ByteDance

- Use Cilium as CNI for Kubernetes to replace kube-proxy
- Better performance, less cpu usage
- Deployed on a few edge networks.
Hubble

- Hubble is a fully distributed networking and security observability platform
- built on top of Cilium and eBPF to enable deep visibility
- Can answer the following questions:
  - Service dependencies & communication map
  - Network monitoring & alerting
  - Application monitoring
  - Security observability
Cilium Components

- Agent (Daemon)
- Client (CLI)
- Operator
- CNI Plugin
Cilium Terminologies

- **Labels**
  - E.g. `io.cilium.mykey=myvalue`
  - Label Source: Kubernetes or container

- **Endpoint**
  - Unique IP
  - By default, assign both IPv4 and IPv6

- **Identification**
  - Uint32 value
    - `0x00000001 - 0x000000FF` (1 to $2^{8} - 1$) => reserved identities
    - `0x00000100 - 0x0000FFFF` ($2^{8}$ to $2^{16} - 1$) => cluster-local identities
    - `0x00010000 - 0x00FFFFFF` ($2^{16}$ to $2^{24} - 1$) => identities for remote clusters
    - `0x01000000 - 0x0100FFFF` ($2^{24}$ to $2^{24} + 2^{16} - 1$) => identities for CIDRs (node-local)
    - `0x01010000 - 0xFFFFFFFF` ($2^{24} + 2^{16}$ to $2^{32} - 1$) => reserved for future use
Networking mode 1: Encapsulation/tunneling

- Encapsulation (or tunneling mode)
- Vxlan or Geneve
- Requirement:
  - the underlying network and firewalls must allow encapsulated packets
- Advantage:
  - Simplicity
  - Addressing space
  - Auto-configuration
  - Identity context in the pkt
- Disadvantage:
  - MTU Overhead
Networking mode 2: native routing
Masquerading
SNAT mode

Figure 1: SNAT packet flow
Direct Server Return (DSR) mode

- Benefit: less processing on LB.
- Where to store VIP?
Current Cilium DSR Support for IPv4

- Use IPv4 options

Figure 2: IP packet format in DSR with IP option
Problem with Current DSR Support for IPv4

- Problem: will go to slow path on switches
  - Network switch has a fast path and slow path
  - IP packets with option will go via slow path on many switches
  - Switches CPU usage arrived 100% for some cores. Bottleneck!

- Could we do better?
Proposal: Use IP-in-IP for DSR on IPv4

- Use IP-in-IP

- Will go to fast path on switches
- Drawbacks: smaller MTU for UDP and TCP Syn packets
Cilium eBPF DSR Data Path Changes

- On the LB:
  nodeport_lb4
    |- tail_nodeport_ipv4_dsr
      |- dsr_set_ipip4

- On real node:
  handle_to_container
    |- tail_ipv4_to_endpoint
      |- ipv4_policy
        |- handle_dsr_v4
          |- snat_v4_create_dsr
Cilium eBPF DSR Data Path

- When sending a reply to the client
  - bpf_lxc finds out that the "dsr" bit was set
  - does a lookup in the NAT table to find the mapping
  - rewrites the source addr and port to the svc addr and port.

- No changes here
Future Work

- IPv6 support? We may not need that.

PR link: https://github.com/cilium/cilium/pull/18449
We Are Hiring!

Interested in our job openings (Linux Kernel, Compiler, Networking, Virtualization)?

Please scan the QR code for JDs or contact our recruiter Fiona Mao via email: fiona.mao@bytedance.com
Thank you!