Contribution ID: **247** Type: **not specified**

Socket termination for policy enforcement and load-balancing

Tuesday 13 September 2022 17:00 (30 minutes)

Socket termination for policy enforcement and load-balancing

Cloud-native environments see a lot of churn where containers can come and go. We have compelling use cases like eBPF enabled policy enforcements and socket load-balancing, where we need an effective way to identify and terminate sockets with active as well as idle connections so that they can reconnect when the remote containers go away. Cilium [1] provides eBPF based socket load-balancing for containerized workloads, whereby service virtual ip to service backend address translation happens only once at the socket connect calls for TCP and connected UDP workloads. Client applications are likely to be unaware of the remote containers that they are connected to getting deleted. Particularly, long running connected UDP applications are prone to such network connectivity issues as there are no TCP RST like signals that the client containers can rely on in order to terminate their sockets. This is especially critical for Envoy-like proxies [2] that intercept all container traffic, and fail to resolve DNS requests over long-lived connections established to stale DNS server containers. The other use case for forcefully terminating sockets is around policy enforcement. Administrators may want to enforce policies on-the-fly whereby they want active client applications traffic to be redirected to a subset of containers, or optimize DNS traffic to be sent to node-local DNS cache containers [3] for JVM-like applications that cache DNS entries.

As we researched ways to filter and forcefully terminate sockets with active as well idle connections, we considered various solutions involving the recently introduced BPF iterator, sock_destroy API, and VRFs that we plan to present in this talk. Some of these APIs are network namespace aware, which need some bookkeeping in terms of storing containers metadata, and we plan to send kernel patches upstream in order to adapt them for container environments. Moreover, sock_destroy API was originally introduced to solve similar problems on Android, but it's behind a special config that's disabled by default. With the VRF approach to terminate sockets, we faced issues with sockets ignoring certain error codes. We hope our experiences, and discussion around the proposed BPF kernel extensions to address these problems help the community.

[1] https://github.com/cilium/cilium
[2] https://github.com/envoyproxy/envoy
[3] https://kubernetes.io/docs/tasks/administer-cluster/nodelocaldns/

I agree to abide by the anti-harassment policy

Yes

Primary author: GHAG, Aditi (Isovalent)

Presenter: GHAG, Aditi (Isovalent)

Session Classification: eBPF & Networking

Track Classification: eBPF & Networking Track