Contribution ID: **149** Type: **not specified**

eBPF / XDP Based Firewall and Packet Filtering

Tuesday, November 13, 2018 4:05 PM (35 minutes)

iptables have been the typical tool to create firewall for linux hosts. We have used them at Facebook for setting up host firewalls on our servers across a variety of tiers. In this proposal, we introduce a eBPF / XDP based firewall solution which we use for packet filtering and has parity to our iptables implementation. We discuss various aspects of this. Following is a brief summary of these aspects, which we will detail further in the paper / presentation.

- Design and Implementation:
  - We use BPF Tables (maps, lpm tries, and arrays) to match for appropriate packet header contents
  - The heart of a firewall is a eBPF filter which parses a packet and does lookups against all relevant maps collecting the matching values. A logical rule set is applied to these collected values. This logical set reads similar to a human-readable high level firewall policy. With iptable rules, amidst all the verbose matching criteria inline every rule, such a policy level representation is hard to infer.
- Performance benefits and comparisons with iptables
  - iptables does packet matching linearly against each rule until a match is found. In our proposal, we use BPF Tables (maps) containing keys for all rules, making packet matching highly efficient. We then apply the policy using the collected results, which results in a considerable speedup over iptables.
- Ease of policy / config updates and maintenance
  - The network administrator owns the firewall while the app developers typically require opening ports for their applications to work. With our approach of using a eBPF filter, we create a logical separation between the filter which enforces the policy and the contents of the associated maps which represent the specific ports and prefixes that need to be filtered. The policy is owned by the network administrator (Example: ports open to the internet, ports open from within specific prefixes, drop everything else). The data (port numbers, prefixes, etc) can now belong to a separate logical section which presents application developers a predetermined destination to update their data (Example: File containing port opened to internal subnets, etc). This reduces friction between the 2 different functions and reduces human errors.
- Deployment experience:
  - We deploy this solution in our edge infrastructure to implement our firewall policy.
  - We update configuration, reload filters and contents of the various maps containing keys and values for filtering
- BPF Program array
  - We use the power of BPF program array to chain different programs like rate limiter, firewall, load balancers, etc. These are building blocks to create a rich, high performant networking solution
- Proposal for a completely generic firewall solution to migrate existing iptables rules to eBPF / XDP based filtering
  - We present a proposal which can translate existing iptables rules to a better performant eBPF program with mostly user space processing and validation.

Presenters: DEEPAK, Anant (Facebook); MEHRA, Puneet (Facebook); HUANG, Richard (Facebook)

Session Classification: Networking Track