EFFICIENCY
The most efficient software data plane Packet Processing on the planet

PERFORMANCE
FD.io on x86 servers outperforms specialized packet processing HW

SOFTWARE DEFINED NETWORKING
Software programmable, extendable and flexible

CLOUD NETWORK SERVICES
Foundation for cloud native network services

LINUX FOUNDATION
Open source collaborative project in Linux Foundation

Breaking the Barrier of Software Defined Network Services
1 Terabit Services on a Single Intel® Xeon® Server!
Motivation: Container networking

FD.io Mini-Summit at KubeCon Europe 2018
Motivation: Container networking
VPP Host Stack

App

Binary API

Session

TCP

IP, DPDK

shm segment

FD.io Mini-Summit at KubeCon Europe 2018
VPP Host Stack: Session Layer

- Maintains per app state and conveys to/from session events
- Allocates and manages sessions/segments/fifos
- Isolates network resources via namespacing
- Session lookup tables (5-tuple) and local/global session rule tables (filters)
- Support for pluggable transport protocols
- Binary/native C API for external/builtin applications
VPP Host Stack: SVM FIFOs

- Allocated within shared memory segments with or without file backing (ssvm/memfd)
- Fixed position and size
- Lock free enqueue/dequeue but atomic size increment
- Option to dequeue/peek data
- Support for out-of-order data enqueues
VPP Host Stack: TCP

- Clean-slate implementation
- “Complete” state machine implementation
- Connection management and flow control (window management)
- Timers and retransmission, fast retransmit, SACK
- NewReno congestion control, SACK based fast recovery
- Checksum offloading
- Linux compatibility tested with IWL TCP protocol tester

FD.io Mini-Summit at KubeCon Europe 2018
VPP Host Stack: more transports

- Binary API
  - Session
  - TCP
  - IP, DPDK

- App

- shm segment
  - SCTP
  - UDP
  - TLS
VPP Host Stack: Comms Library (VCL)

- Comms library (VCL) apps can link against
- LD_PRELOAD library for legacy apps
- epoll

FD.io Mini-Summit at KubeCon Europe 2018
Application Attachment

attach
bind (server)
connect (client)

App
Binary API
Session
TCP
IP, DPDK
VPP

shm segment
Session Establishment

Client

Binary API

Session

TCP

IP, DPDK

VPP

Server

Binary API

Session

TCP

IP, DPDK

VPP

attach
bind

listen

FD.io Mini-Summit at KubeCon Europe 2018
Session Establishment

Client
- Binary API
  - Session
    - TCP
      - IP, DPDK
- attach
- connect

Server
- Binary API
  - Session
    - TCP
      - IP, DPDK
- attach
- bind
- listen
Session Establishment

Client

Binary API

Session

TCP

IP, DPDK

VPP

handshake

Server

Binary API

Session

TCP

IP, DPDK

VPP
Session Establishment

Client

TCP

IP, DPDK

VPP

Binary API

Session

new client

handshake

Server

TCP

IP, DPDK

VPP

connect succeeded

FD.io Mini-Summit at KubeCon Europe 2018
Session Establishment

Client
Session
TCP
IP, DPDK
VPP

Server
Session
TCP
IP, DPDK
VPP

Binary API
connect reply
accept notify

shm segment
rx tx

shm segment
rx tx
Data Transfer

FD.io Mini-Summit at KubeCon Europe 2018
Data Transfer

Some rough numbers on a E2699: ~12Gbps/core (1.5k MTU), ~20Gbps/core (9k MTU), ~185k CPS!
Data Transfer: Dgram Transports

Data and dgram header dequeued from fifo

Data enqueued in fifo with dgram hdr
Redirected Connections (Cut-through)

Client

Server

Binary API

Session

TCP

IP, DPDK

VPP

bind
Redirected Connections (Cut-through)

Client

connect

Binary API

Session

TCP

IP, DPDK

redirect

Server

VPP
Redirected Connections (Cut-through)

VPP tracks these sessions, allocates ssvm segments and asks both peers to map them.
Redirected Connections (Cut-through)

Throughput is memory bandwidth constrained: ~120Gbps!
Multi-threading for stream connections

Connections/sessions ‘pinned’ to a thread
Per-thread data structures/state
Features: Namespaces

Namespaces are configured independently and associate applications to network layer resources like interfaces and fib tables.

Request access to vpp ns + secret
Features: Session Tables

Request access to global and/or local scope

Binary API

NS Local Session Table
TCP
ns1
Global Session Table
fib1
ns2
NS Local Session Table
TCP
App1
Features: Session Tables

- Both table have “rules table” that can be used for filtering
- Local tables are namespace specific and can be used for egress filtering
- Global tables are fib table specific and can be used for ingress filtering
TLS App

- TLS App registers as transport at VPP init time
- TLS protocol implementation handled by plugin “engines”. We support openssl and mbedtls
- Client app registers key and certificate via api and requests tls as session transport
- CA certs read at TLS app init time. Defaults to reading /etc/ssl/certs/ca-certificates.crt
- Ping and Ray from Intel working on accelerating the openssl engine with QAT cards
TLS App

- TLS App registers as transport at VPP init time
- TLS protocol implementation handled by plugin “engines”. We support openssl and mbedtls
- Client app registers key and certificate via api and requests tls as session transport
- CA certs read at TLS app init time. Defaults to reading /etc/ssl/certs/ca-certificates.crt
- Ping and Ray from Intel working on accelerating the openssl engine with QAT cards

Some rough OpenSSL numbers on a E2699: ~1Gbps/core (no hw accel)
Ongoing work

• Overall integration with k8s
  • Istio/Envoy
• TCP
  • Rx policer/tx pacer
  • TSO
  • New congestion control algorithms
  • PMTU discovery
  • Optimization/hardening/testing
Next steps – Get involved

- Get the Code, Build the Code, Run the Code
  - Session layer: src/vnet/session
  - TCP: src/vnet/tcp
  - SVM: src/svm
  - VCL: src/vcl
- Read/Watch the Tutorials
- Read/Watch VPP Tutorials
- Join the Mailing Lists
Thank you!

Florin Coras
email: fcoras@cisco.com
irc: florinc