VPP Host Stack
TCP and Session Layers

Florin Coras, Dave Barach,
Keith Burns, Dave Wallace
VPP - A Universal Terabit Network Platform
For Native Cloud Network Services

- **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!
VPP – How does it work?
Compute Optimized SW Network Platform

1. Packet processing is decomposed into a directed graph of nodes ...

2. ... packets move through graph nodes in vector ...

3. ... graph nodes are optimized to fit inside the instruction cache ...

4. ... packets are pre-fetched into the data cache.

Makes use of modern Intel® Xeon® Processor micro-architectures.
Instruction cache & data cache always hot ➔ Minimized memory latency and usage.

* Each graph node implements a “micro-NF”, a “micro-NetworkFunction” processing packets.
Motivation: Container networking

- **PID 1234**
  - `send()`
  - FIFO
  - TCP
  - IP (routing)
  - Device

- **glibc**

- **Kernel**

- **PID 4321**
  - `recv()`
  - FIFO
  - TCP
  - IP (routing)
  - Device

DPDK Summit North America 2017
Motivation: Container networking

PID 1234
- send()
  - FIFO
  - TCP
  - IP (routing)
  - device

VPP
- etc etc etc
- ACL, SR, VXLAN, LISP
- IP4/6
- MPLS
- Ethernet

PID 4321
- recv()
  - FIFO
  - TCP
  - IP (routing)
  - device

DPDK Summit North America 2017
Why not this?

DPDK Summit North America 2017
VPP Host Stack

VPP

IP, DPDK

TCP

Session

Binary API

App

shm segment

rx

tx
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
Allocated within shared memory segments
- 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
VPP Host Stack: Comms Library (VCL)

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

Diagram:
- VPP
  - IP, DPDK
  - Session
  - TCP
  - Binary API
  - App
  - shm segment
  - rx, tx

DPDK Summit North America 2017
Application Attachment

attach
bind (server)
connect (client)

App

Binary API

Session

TCP

IP, DPDK

VPP

shm segment

DPDK Summit North America 2017
Session Establishment

Binary API

Session

TCP

IP, DPDK

VPP

Client

Server

Binary API

Session

TCP

IP, DPDK

VPP

attach
bind

listen
Session Establishment

Client

attach
connect

Binary API

Session

TCP

IP, DPDK

VPP

Server

attach
bind

Binary API

Session

TCP

IP, DPDK

VPP

DPDK Summit North America 2017
Session Establishment

Client

Binary API

Session

TCP

IP, DPDK

VPP

handshake

Server

Binary API

Session

TCP

IP, DPDK

VPP
Session Establishment

Client

Binary API

Session

TCP

IP, DPDK

VPP

Binary API

Session

TCP

IP, DPDK

VPP

Server

connect succeeded

new client

handshake

DPDK Summit North America 2017
Session Establishment

Client

Binary API

Session

TCP

IP, DPDK

VPP

Server

Binary API

Session

TCP

IP, DPDK

VPP

connect reply

accept notify

shm segment

rx tx

rx tx

shm segment

DPDK Summit North America 2017
Data Transfer

Client
- TCP
- IP, DPDK
- Copy to buffer
- Session
- Binary API
- TX write evt
- VPP

Server
- TCP
- IP, DPDK
- Copy to fifo
- Session
- Binary API
- RX write evt
- VPP

Congestion control
Reliable transport

DPDK Summit North America 2017
Data Transfer

Not yet part of CSIT but some rough numbers on a E2690: 200k CPS and 8Gbps/core!
Redirected Connections (Cut-through)
Redirected Connections (Cut-through)

Client --- Binary API --- Server

connect redirect

Session

TCP

IP, DPDK

VPP
Redirected Connections (Cut-through)

Throughput is memory bandwidth constrained: ~120Gbps!

Client

connect

Binary API

redirect

Session

TCP

IP, DPDK

VPP

DPDK Summit North America 2017
Ongoing work

- Overall integration with k8s
  - Istio/Envoy
- TCP
  - Rx policer/tx pacer
  - TSO
  - New congestion control algorithms
  - PMTU discovery
  - Optimization/hardening/testing
- VCL/LD_PRELOAD
  - Iperf, nginx, wget, curl
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
Multi-threading
Features: Namespaces

App

Request access to vpp ns + secret

Binary API

Session

TCP

IP

ns1

fib1

ns2

ns3

TCP

IP

VPP

DPDK Summit North America 2017
Features: Session Tables

Request access to global and/or local scope

App1

Binary API

NS Local Session Table

TCP

Global Session Table

fib1

ns1

ns2
Features: Session Tables

- Both tables 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