VPP Host Stack: Session Layer

- App-interface sub-layer maintains per app state
- Allocates and manages segments, message queues and fifos
- Exposes APIs for conveying session events between applications and transports
- Binary/native C API for external/builtin applications
VPP Host Stack: Session Layer

- Allocates and manages sessions
- Session lookup tables (5-tuple) and local/global session rule tables (filters)
- Support for pluggable transport protocols
- Isolates network resources via namespaces
VPP Host Stack: Session Layer

- Exposes APIs transports can use for enqueueing data to apps
- Handles segmentation of app data into buffers before sending it to transport protocols
- Can enforce tx-pacing if transport asks for it
VPP Host Stack: SVM

- Fifo segments:
  - Shared memory segments allocated by the app-interface sub-layer and mapped by applications
  - Preferred without file backing (memfd). Support for segments with file backing (shm) will eventually be deprecated
VPP Host Stack: SVM

- Message queue
  - Allocated in first shared memory segment
  - Has two rings for control and io events from vpp to app
  - Supports condvar or eventfd signaling

Diagram:

- App Interface
  - Session
  - TCP
  - IP, DPDK
  - mq
  - rx
  - tx
VPP Host Stack: SVM

- **fifos**
  - Fixed header position and linked list of memory chunks for actual data
  - Can grow/shrink by adding/removing chunks
  - Lock free enqueue/dequeue but some atomic operations needed
  - 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
- Timestamps, SACKs
- High scale timers implementation
- NewReno and Cubic congestion control
- Fast recovery, timer based retransmissions
- Tx pacing
- Checksum offloading
- Protocol correctness tested with Defensics Codenomicon
VPP Host Stack: Comms Library (VCL)

- Apps can directly use the raw session layer APIs but then need to:
  - Manage binary api and message queue interaction with vpp
  - Maintain session state, potentially deal with thread safety
  - Implement async communication mechanisms
VPP Host Stack: Comms Library (VCL)

VPP Comms library (VCL)
- Manages interaction with session layer
- Abstracts sessions to integer session handles
- Exposes epoll/select/poll functions
- Supports multi-worker applications
- Can handle mq notifications with both mutex-condvar pair and eventfd signaling
VPP Host Stack: Comms Library (VLS)

- VCL Locked Sessions (VLS)
  - Exposes northbound a vls handle table shared by all workers
  - Detects app threads and enforces vls table and session locking on rw access
  - Detects app forks and registers new processes as vcl workers
VPP Host Stack: Comms Library (LDP)

- LDP library
  - Uses LD_PRELOAD to intercept and redirect syscalls to VLS
  - Manages fd to vls session handle translation
  - When it works, it requires no changes to applications
  - Do not expect it to always work
  - Functionally works with iperf, nginx, sshd etc.
  - Not optimized for performance
Data Transfer

enqueue to fifo
tx io event

enqueue to fifo
rx io notification

Poll tx events
dequeue to buffer
add tcp header

App Interface

Session

TCP

IP, DPDK

client

mq

rx

tx

server

mq

rx

tx

App Interface

Session

TCP

IP, DPDK

Congestion control
Reliable transport
Data Transfer

Some rough numbers on an E2699 w/XL710: ~36Gbps/core (1.5k MTU) half-duplex!
Redirected Connections (Cut-through)

client

connect

App Interface

Session

Cut-Through

bind

server

IP, DPDK

FD.io Mini-Summit at KubeCon Europe 2019
Redirected Connections (Cut-through)

- Cut-through transport:
  - Tracks the sessions
  - Allocates ssvm segment for fifos
  - Asks apps to map segment
Redirected Connections (Cut-through)

Cut-through transport:
- Tracks the sessions
- Allocates ssvm segment for fifos
- Asks apps to map segment

Throughput is around ~120Gbps half-duplex if receiver does not touch the data!
VPP builtin apps

- Use app-interface C apis
- Applications provide at attachment time callback functions for io/ctrl events
- Shm segment/fifo segment allocated in process memory
VPP builtin apps

- Ctrl/rx io events are delivered to app within vpp worker context
- Tx io events from app to session layer rely on session layer message queue
- E.g. http_static, echo apps
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