You Don't Know ’Jack’: CXL Fabric Orchestration and Management

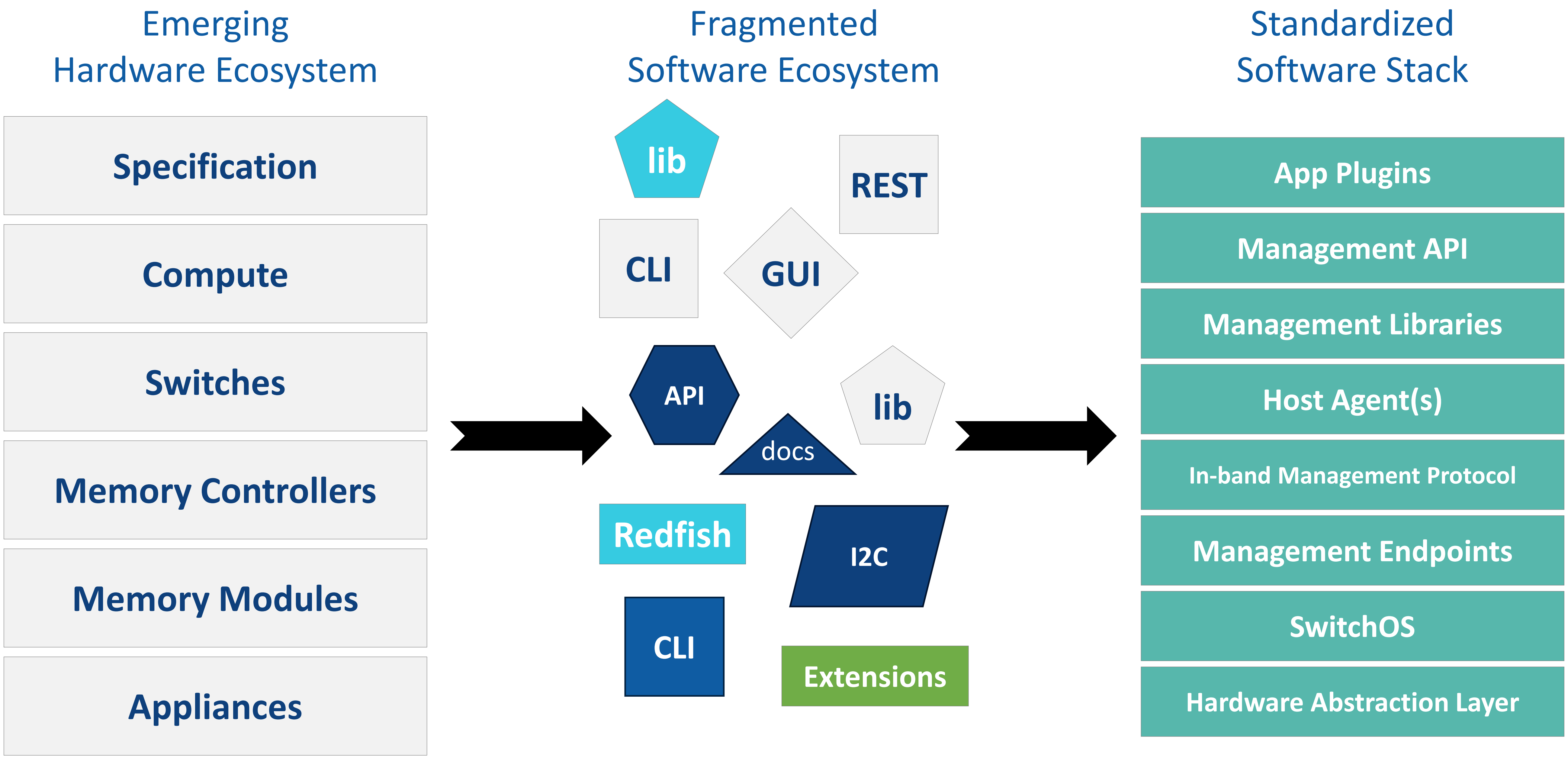
- Grant Mackey – Jackrabbit Labs

Who is Jackrabbit Labs?

The open-source software services company for shared memory management

Why is Open-Source Software Needed for CXL?

Fragmentation slows adoption



Emerging Hardware Ecosystem

- Specification
- Compute
- Switches
- Memory Controllers
- Memory Modules
- Appliances

Fragmented Software Ecosystem

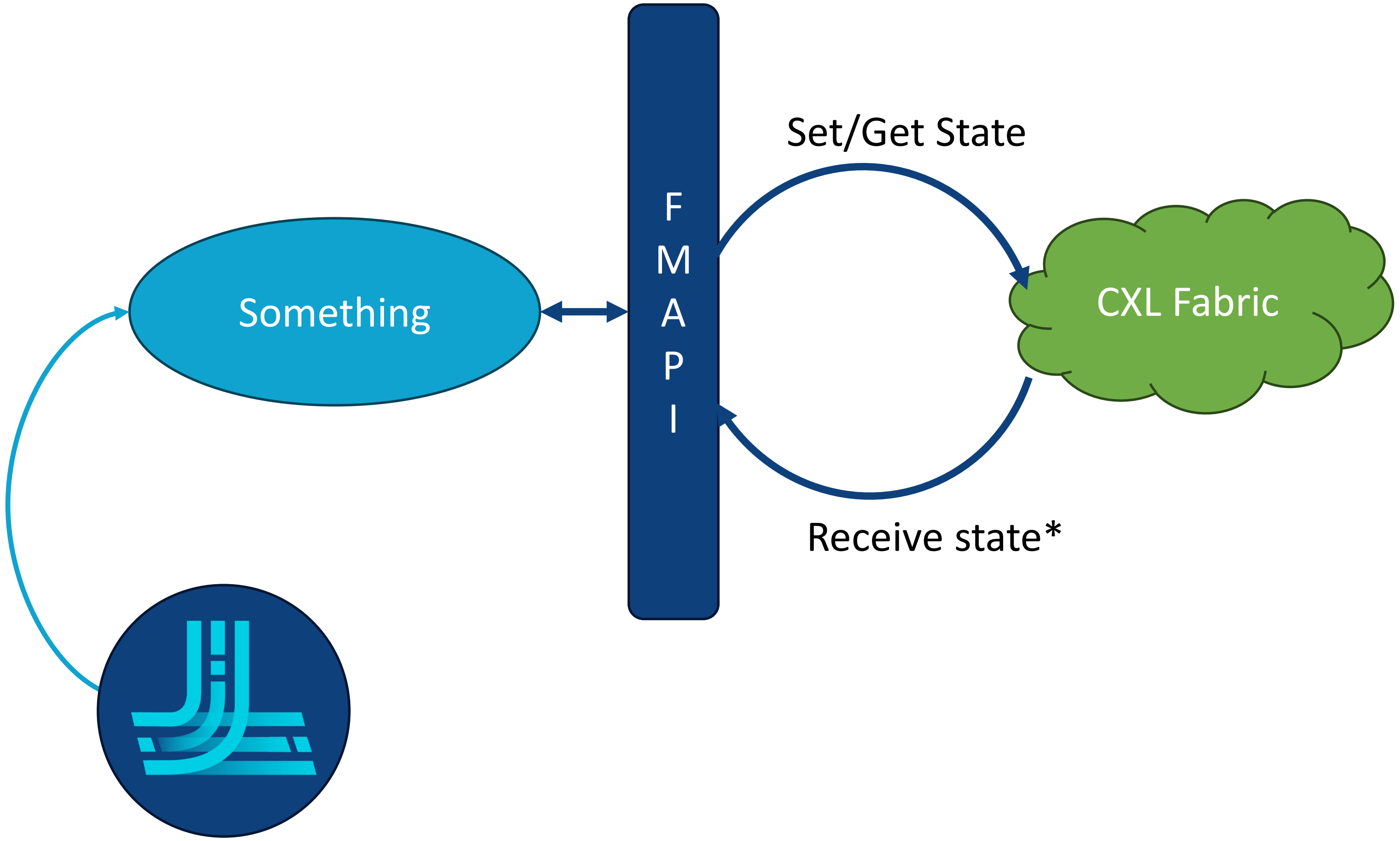
Standardized Software Stack

- App Plugins
- Management API
- Management Libraries
- Host Agent(s)
- In-band Management Protocol
- Management Endpoints
- SwitchOS
- Hardware Abstraction Layer

This is the CXL FMAPI slide

An API is not a platform

- The CXL spec contains a Fabric Management API, but FMAPI is not orchestration!
  - FMAPI is just an API to complete actions on the fabric, not a tool to manage state
  - The number of command sets grows quickly with major version updates

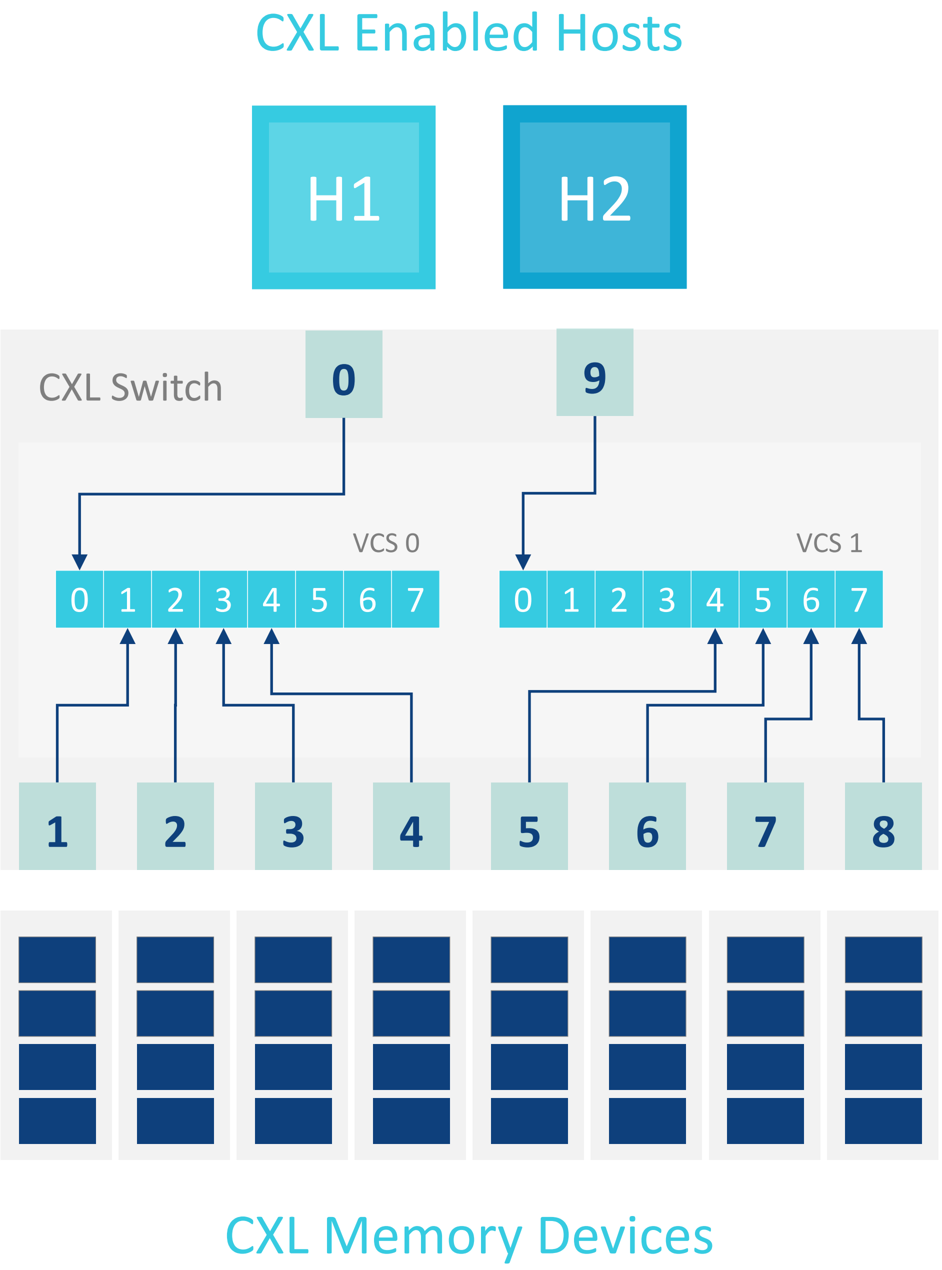


CMD set	CXL 3.1	CXL 2.0
Physical Switch	✓+	✓
Virtual Switch	✓	✓
MLD Port	✓	✓
MLD Components	✓	✓
Multi-Headed Devices	✓	x
DCD mgmt.	✓	x
PBR switch	✓	x
Global memory access EP	✓	x
GMA EP mgmt.	✓	x

*There is presently no mechanism to acknowledge a set state cmd, orchestrator has to explicitly verify

Jack – CXL Fabric Management CLI Tool

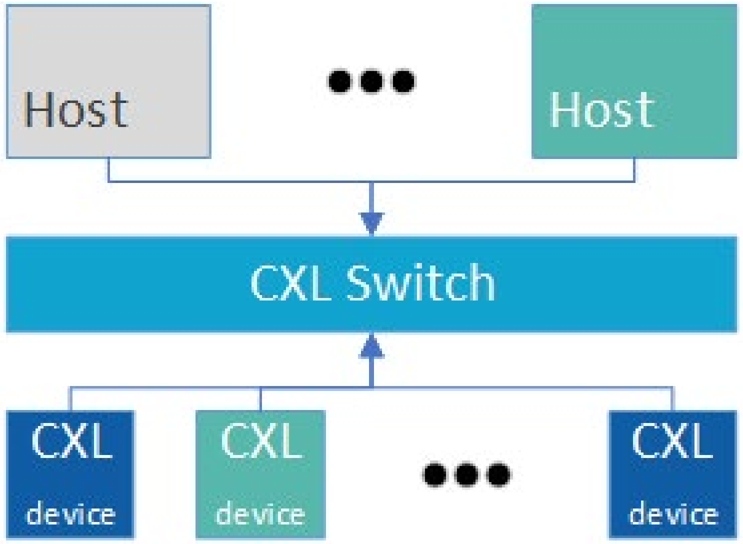
Implements the CXL Fabric Management API



```
# jack show port

#    @  Port State  Type    LD  Ver  CXL Ver   MLW  NLW  MLS  CLS  Speeds    LTSSM     LN  Flags
---  -  ----------  ------  --  ---  --------  ---  ---  ---  ---  --------  --------  --  ------------
0    +  Upstream    T1      -   2.0  AB        16   16   5.0  5.0      45    L0        0     P
1    +  Downstream  T3-MLD  16  2.0  AB        16   16   5.0  5.0      45    L0        0     P
2    +  Downstream  T3-MLD  16  2.0  AB        16   16   5.0  5.0      45    L0        0     P
3    +  Downstream  T3-MLD  16  2.0  AB        16   16   5.0  5.0      45    L0        0     P
4    +  Downstream  T3-MLD  16  2.0  AB        16   16   5.0  5.0      45    L0        0     P
5    +  Downstream  T3-MLD  16  2.0  AB        16   16   5.0  5.0      45    L0        0     P
6    +  Downstream  T3-MLD  16  2.0  AB        16   16   5.0  5.0      45    L0        0     P
7    +  Downstream  T3-MLD  16  2.0  AB        16   16   5.0  5.0      45    L0        0     P
8    +  Downstream  T3-MLD  16  2.0  AB        16   16   5.0  5.0      45    L0        0     P
9    +  Upstream    T1      -   2.0  AB        16   16   5.0  5.0      45    L0        0     P

# jack show vcs 0

Show VCS:
VCS ID  : 0
State   : Enabled
USP ID  : 0
vPPBs   : 8

vPPB  PPID LDID Status
----  ---- ---- -----------
   0:    0    - Bound Physical Port
   1:    1    0 Bound LD
   2:    2    0 Bound LD
   3:    3    0 Bound LD
   4:    4    0 Bound LD
   5:    -    - Unbound
   6:    -    - Unbound
   7:    -    - Unbound
```

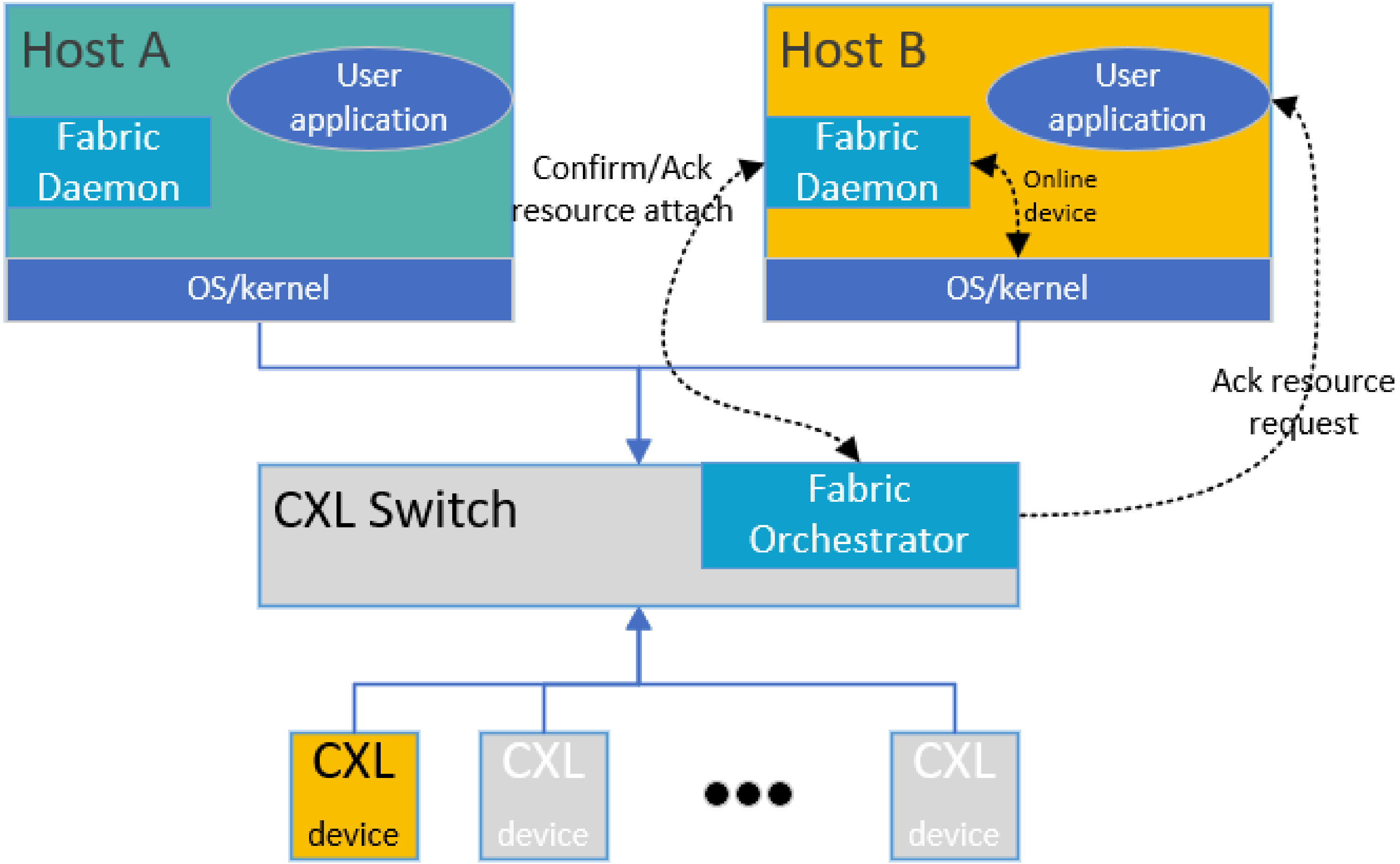
CXL Fabrics Need a Platform to Shine

Specs and state management won't get you there



Once a host is assigned ownership of a cxl device the fabric cannot take it back via any CXL specification mechanisms

Cc
resc

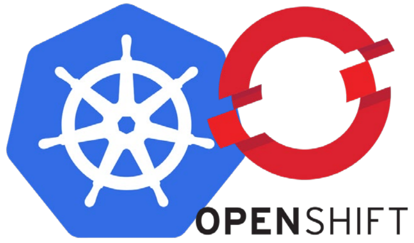


Orchestration outside of the CXL spec is needed to enable a composable memory system rather than a statically allocated at boom memory topology

Interacting with cluster schedulers and resource managers

I don’t care what you are, give me ‘X’

- Resource schedulers don’t want to know how memory fabrics like CXL work
  - They don’t care about Ultra/Ethernet/Infiniband/NV or UALink either.
  - They want the OS or a module to handle it so they can schedule resources



Container ‘x’ interfaces
- Resource, CRI
- Storage, CSI
- Network, CNI

AND device plugin support

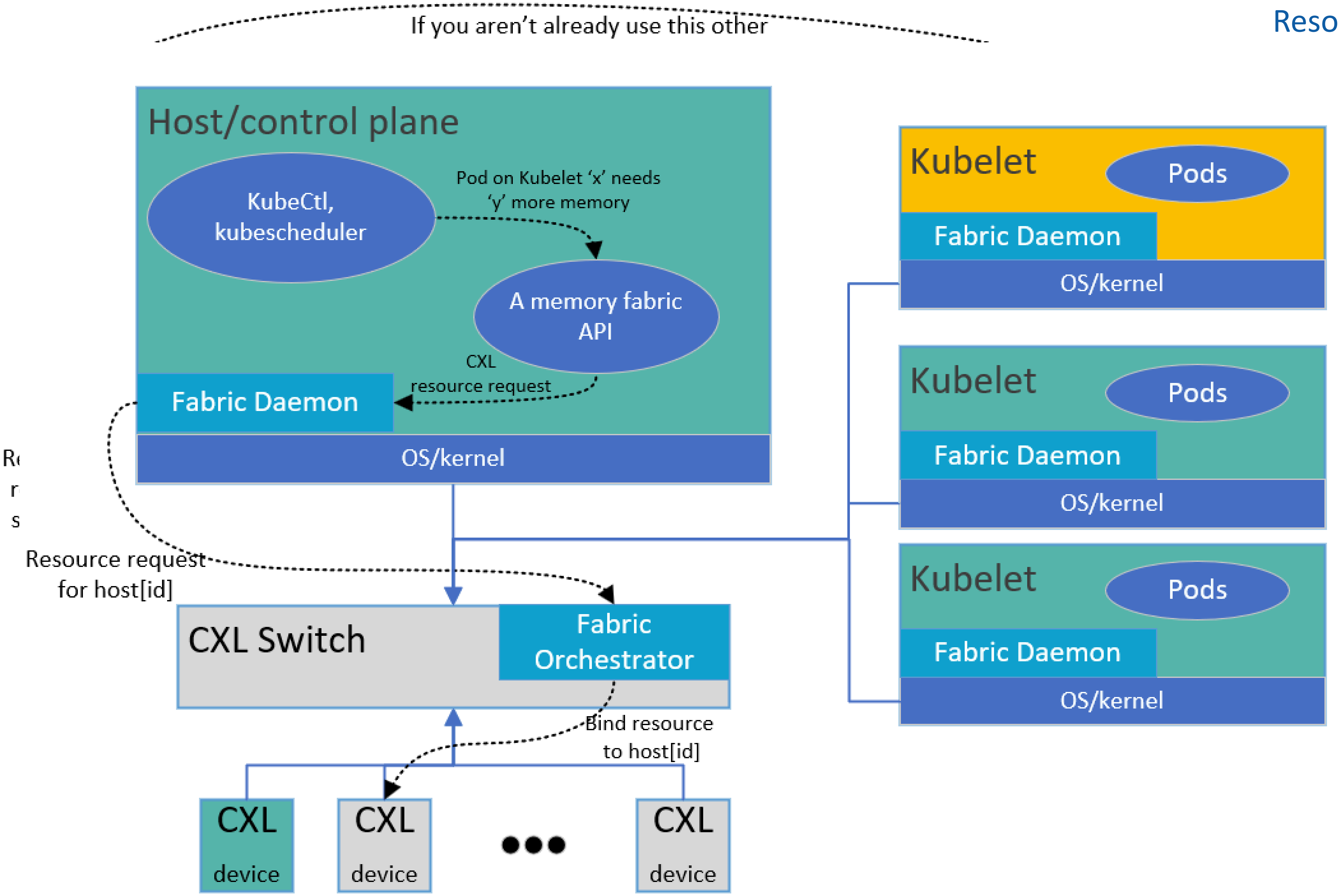
completely punt on caring about hardware

Ceph – storage
corosync – state sync

The chimera!
Has 41 types of resource services with varying levels of hardware abstraction

Where Jack and the Orchestrator are going

Resource manager API



Review

Challenges

- Potential fragmentation of the shared memory software ecosystem will delay adoption
- Lack of application development in the open
- Lack of platforms emulated or real to do said development on

Call to Action

- Experiment with QEMU today! QEMU supports more CXL features (i.e. CXL 3.0+) than CPU HW today
- Software application development doesn't have to wait until hardware (i.e. switches) are available
- Evaluate where open-source tools / libraries / APIs can be used in your projects

Intel	libcxl	https://github.com/pmem/ndctl
Jackrabbit Labs	libmem	https://github.com/JackrabbitLabs/libmem
Jackrabbit Labs	Jack - CXL FM API CLI Tool	https://github.com/JackrabbitLabs/jack
Jackrabbit Labs	CXL Switch Emulator	https://github.com/JackrabbitLabs/cse
Samsung	Scalable Memory Development Kit (SMDK)	https://github.com/OpenMPDK/SMDK
Micron	CXL Memory Resource Kit (CMRK)	https://github.com/cxl-micron-reskit/cxl-reskit
SK Hynix	Heterogeneous Memory Software Development Kit (HMSDK)	https://github.com/skhynix/hmsdk
Micron	CXL Library CLI	https://github.com/cxl-micron-reskit/mxcli
Micron	FAMFS	https://github.com/cxl-micron-reskit/famfs
Intel	Unified Memory Framework	https://github.com/oneapi-src/unified-memory-framework
QEMU	QEMU	https://github.com/qemu/qemu
Samsung	libcxlmi	https://github.com/computexpresslink/libcxlmi

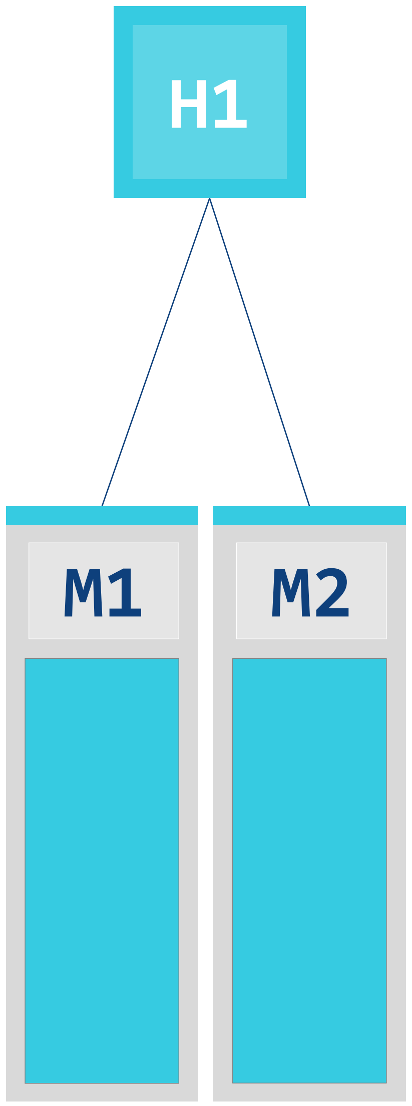
JACKRABBIT LABS

Driving CXL Adoption with Open Source

CXL Memory Topologies

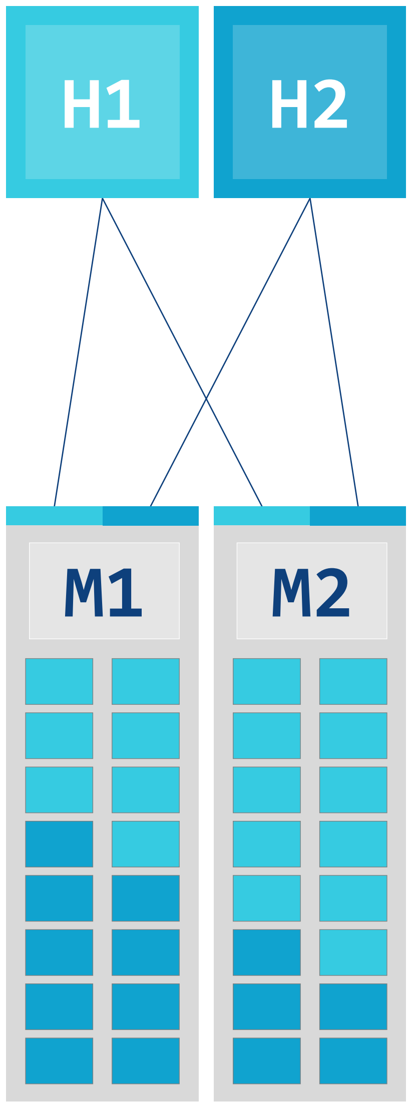
Hosts, Switches, and Devices can be connected in a Direct or Switched Topology

Direct Attach
Memory Expansion



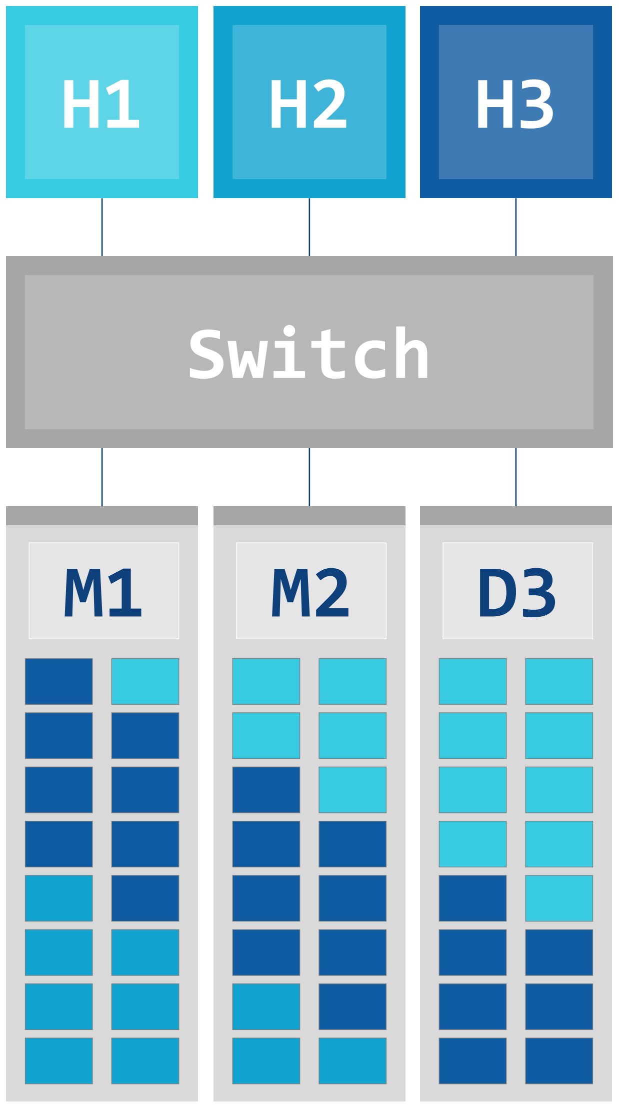
SH-SLD – DRAM Drives
Single-Headed Single Logical Devices

Direct Attach
Device Pooling / Sharing



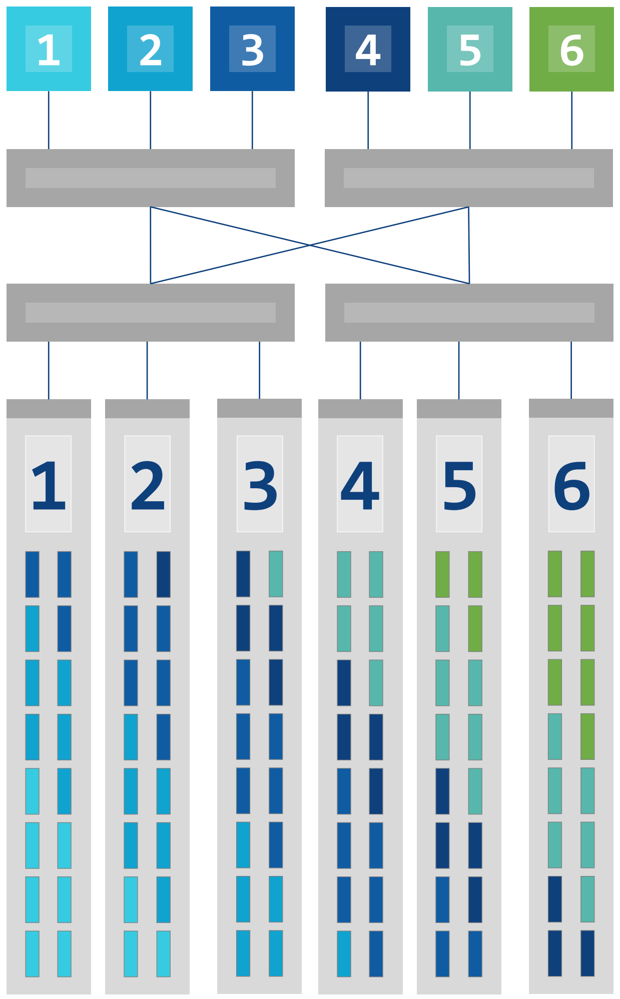
MH-MLD – DRAM Drives
Multi-Headed Multi-Logical Devices

Single Layer Switch
Device Pooling / Sharing



SH-MLD – DRAM Drives
Single-Headed Multi-Logical Devices

Multi-Layer Switch
Device Pooling / Sharing

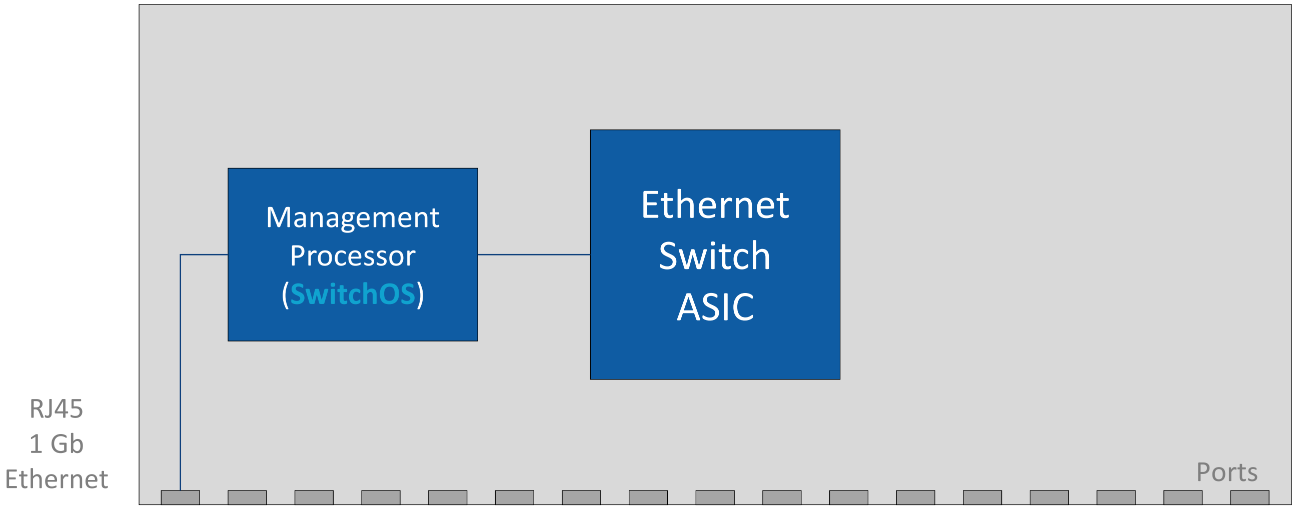


SH-MLD – DRAM Drives
Single-Headed Multi-Logical Devices

SwitchOS

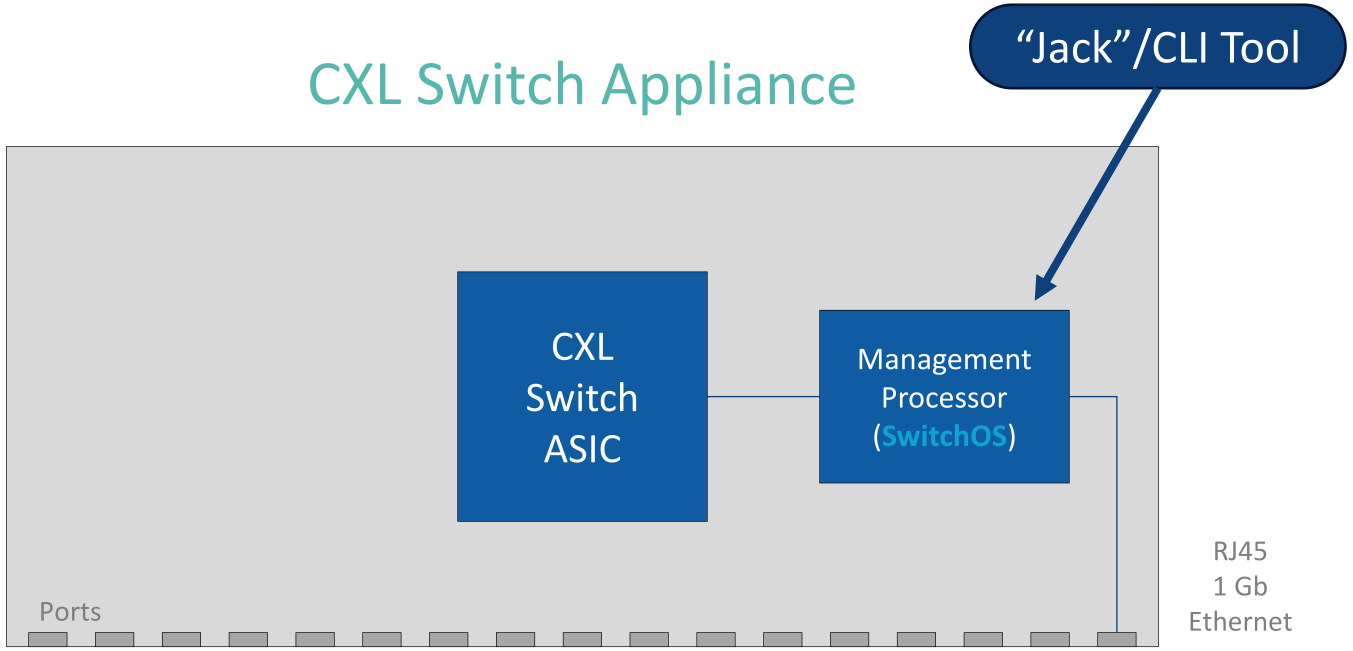
The Management Abstraction Layer to the Silicon

Ethernet Switch Appliance



- Managed Ethernet switches run a SwitchOS
- e.g. SONiC, Cumulus, FBOSS, EOS, NX-OS
- Managed through in-band / out-of-band Ethernet links
- Hardware Abstraction Layer (HAL)
- Can be run on a low-end BMC or larger x86 processor
- SONiC = Debian + Ethernet Management Containers
- Typically has a CLI shell + Web API / GUI

CXL Switch Appliance

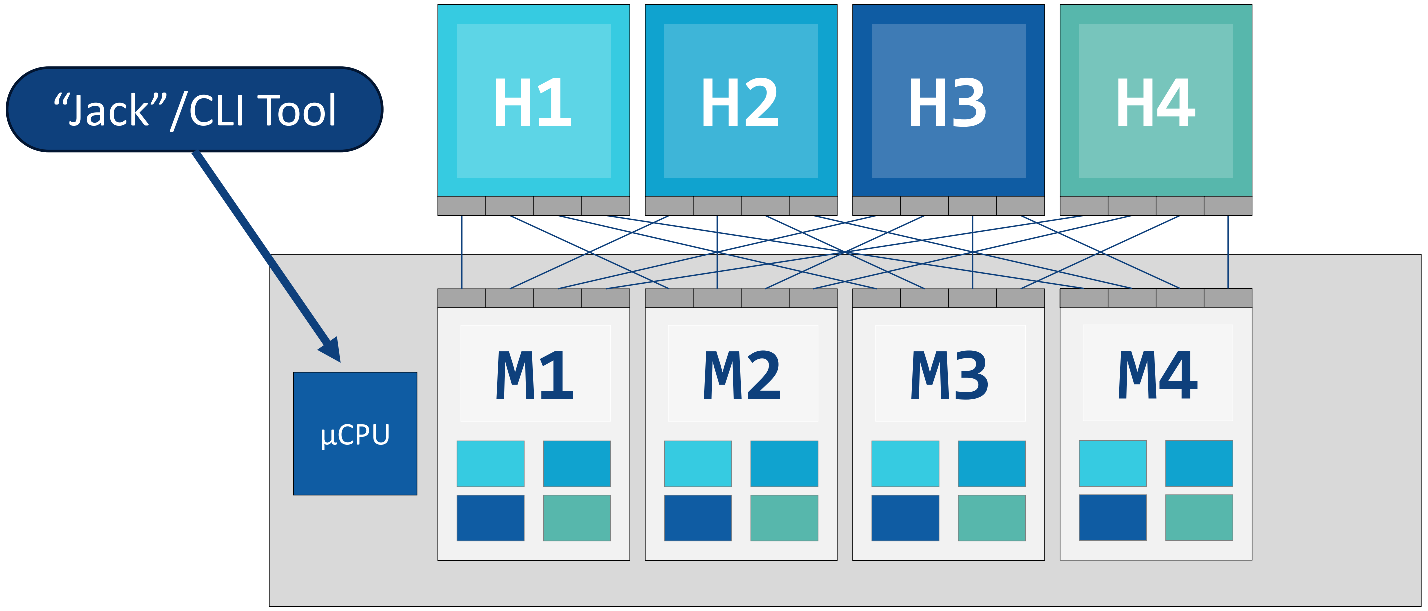


- CXL Switch appliances are equivalent to Ethernet switches
- Will run a SwitchOS to manage CXL switch silicon
- The "Fabric Manager" lives in this SwitchOS
  (Or at least a software agent of a larger orchestration system)
- Has a Hardware Abstraction Layer (HAL) for CXL switch silicon
- External interface can be REST, GUI over Ethernet or an in-band protocol over CXL links

SwitchOS

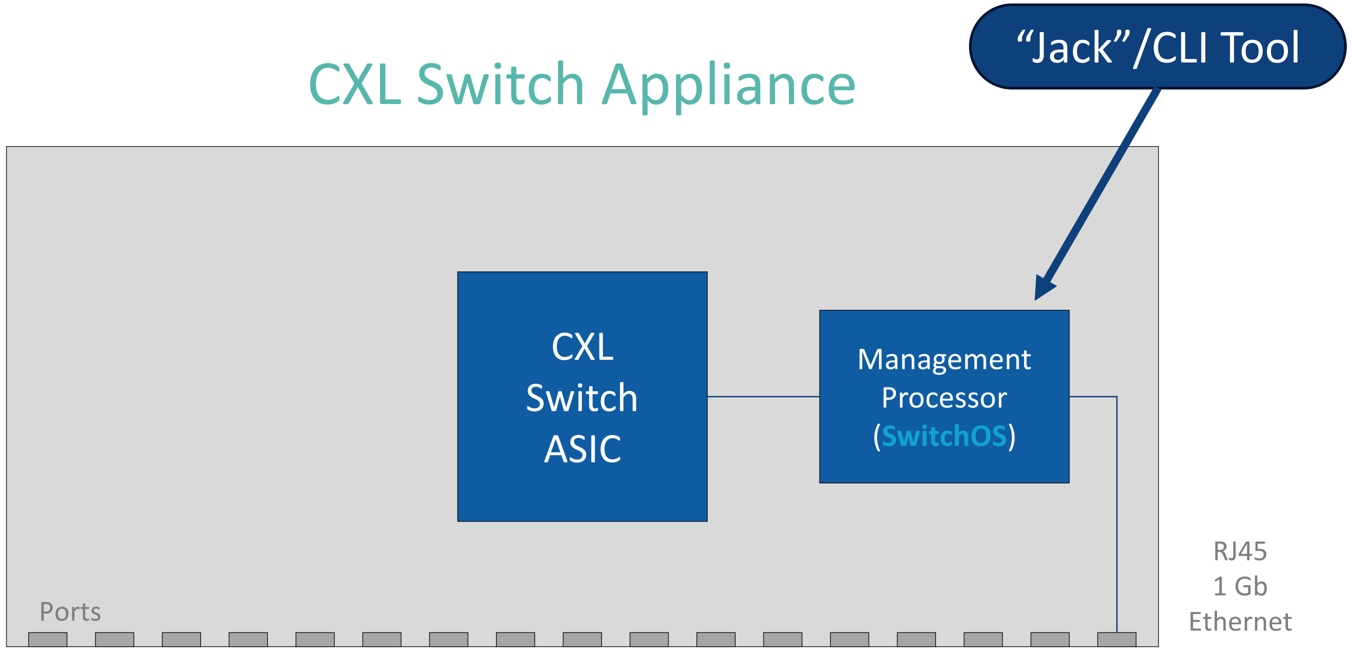
The Management Abstraction Layer to the Silicon

Direct Attach Multi-Port Devices



- Directly connected Multi-Headed (Multi-Port) devices
- No switch architecture
- Memory devices housed in separate / bladed enclosure
- Lower latency – more cables / complex enclosure
- Still requires separate management entity

CXL Switch Appliance



- CXL Switch appliances are equivalent to Ethernet switches
- Will run a SwitchOS to manage CXL switch silicon
- The "Fabric Manager" lives in this SwitchOS
  (Or at least a software agent of a larger orchestration system)
- Has a Hardware Abstraction Layer (HAL) for CXL switch silicon
- External interface can be REST, GUI over Ethernet or an in-band protocol over CXL links