RDMA in Data Centers: Looking Back and Looking Forward

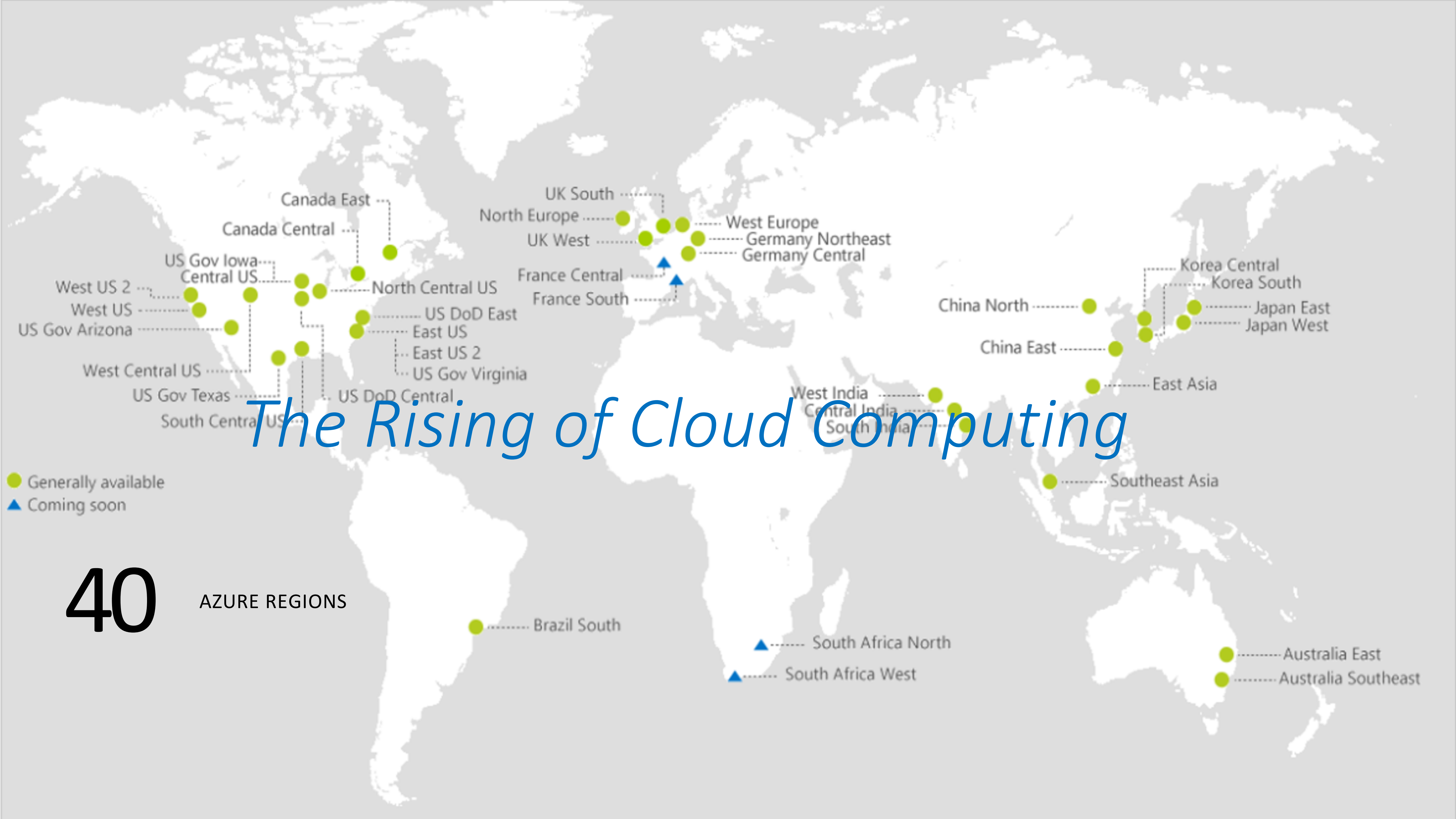
Chuanxiong Guo

Microsoft Research

ACM SIGCOMM APNet 2017

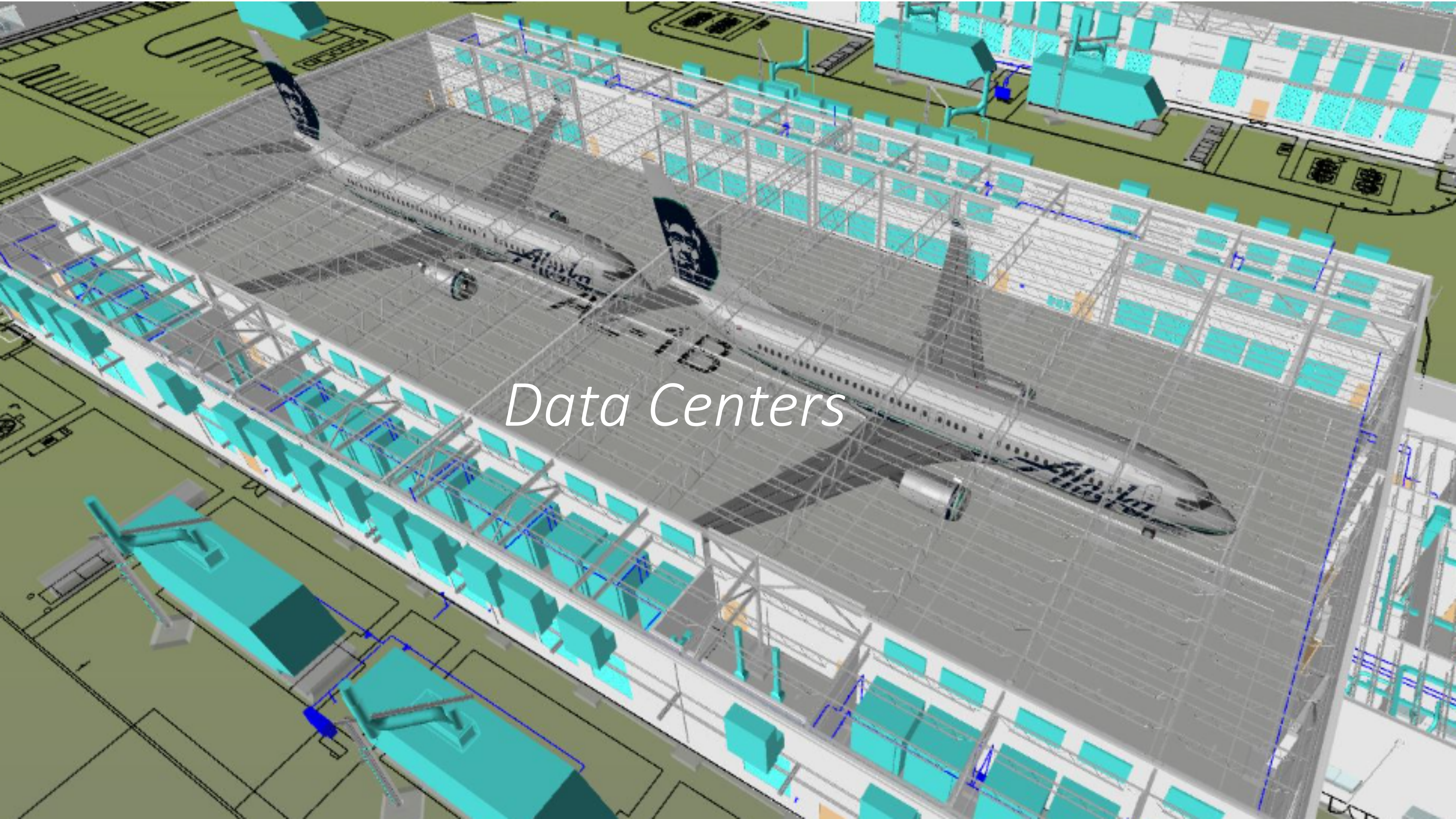
August 3 2017

The Rising of Cloud Computing



40 AZURE REGIONS

Data Centers

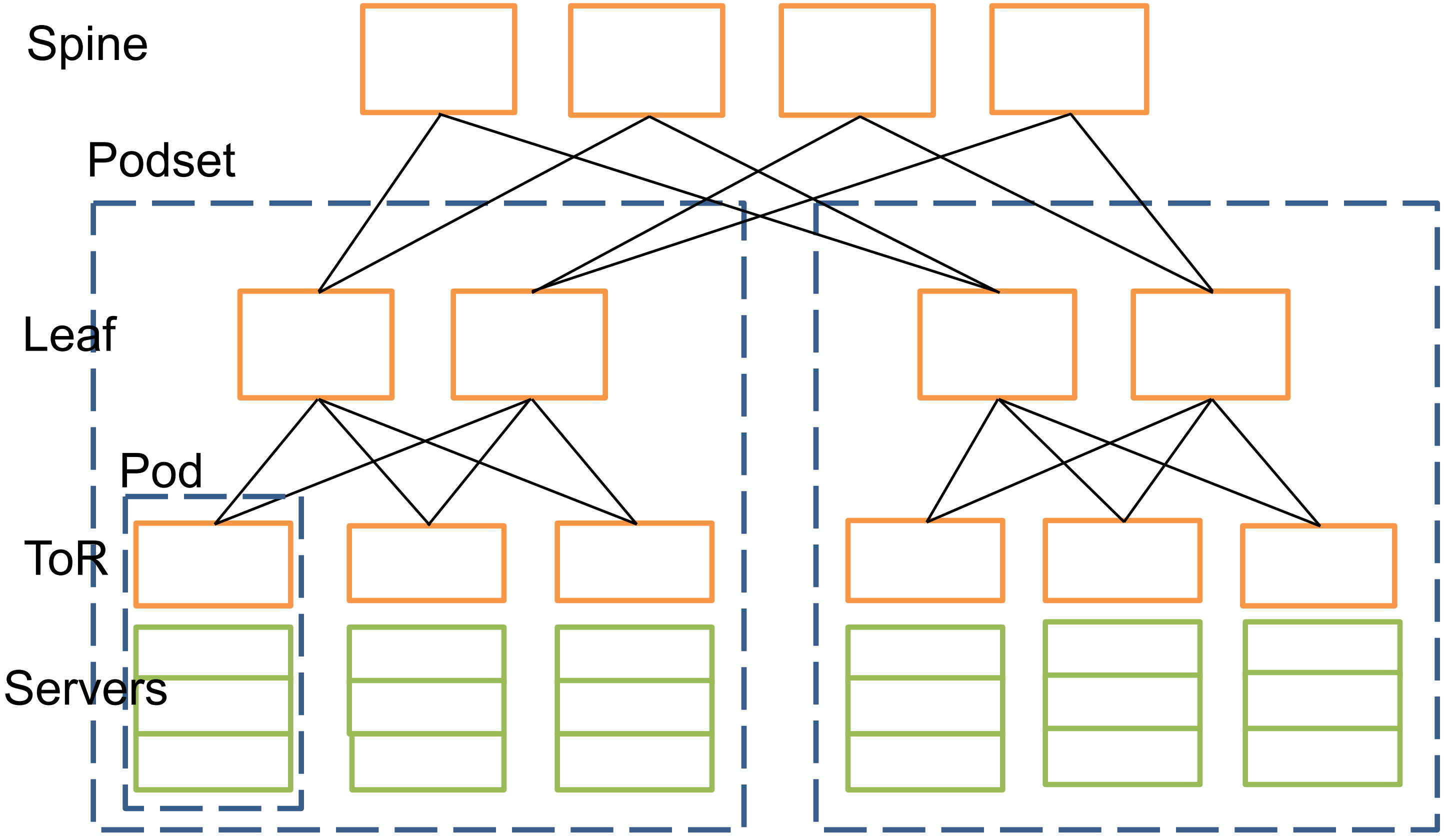


Data Centers

Data center networks (DCN)

- Cloud scale services: IaaS, PaaS, Search, BigData, Storage, Machine Learning, Deep Learning
- Services are latency sensitive or bandwidth hungry or both
- Cloud scale services need cloud scale computing and communication infrastructure

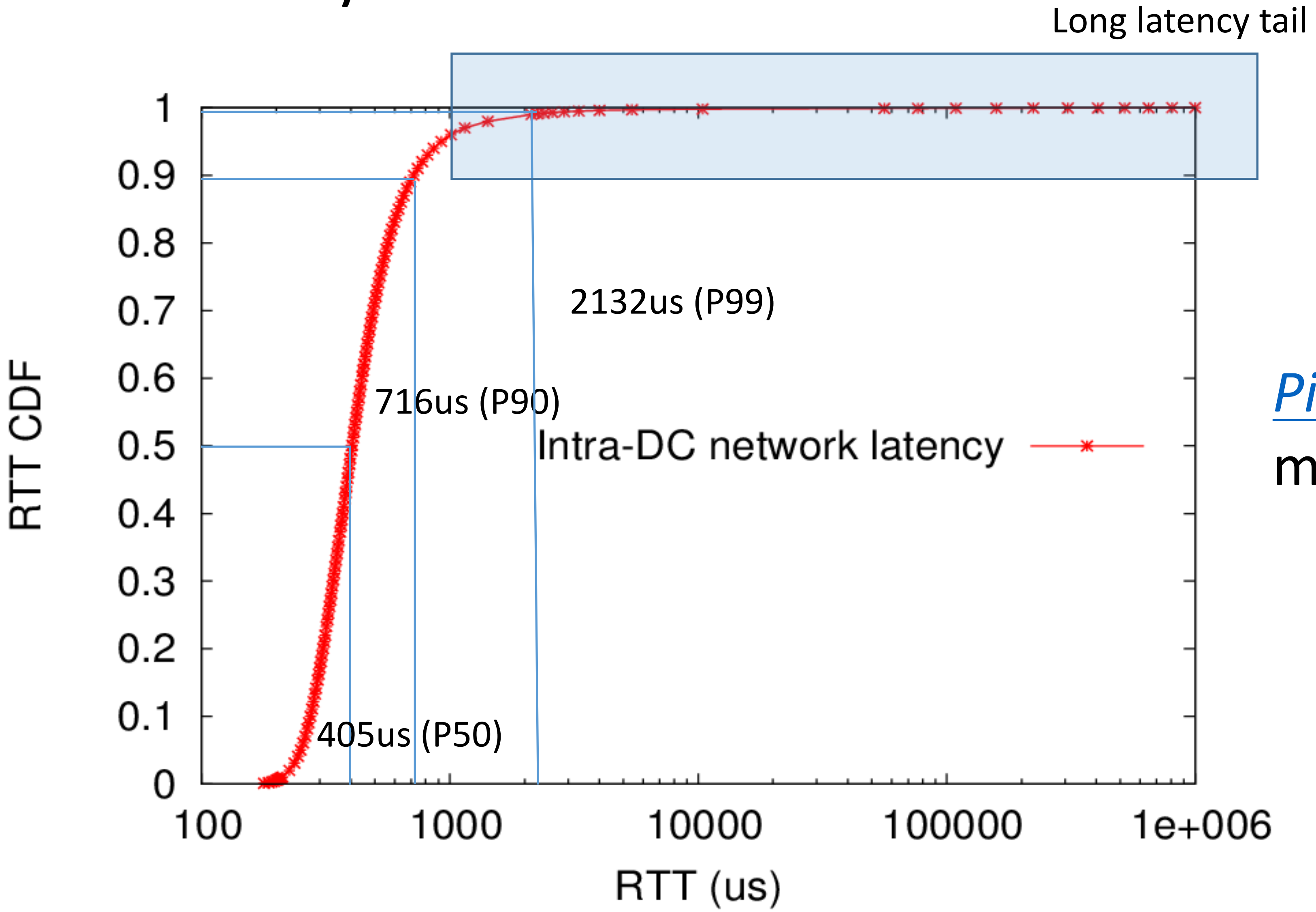
Data center networks (DCN)



- Single ownership
- Large scale
- High bisection bandwidth
- Commodity Ethernet switches
- TCP/IP protocol suite

But TCP/IP is not doing well

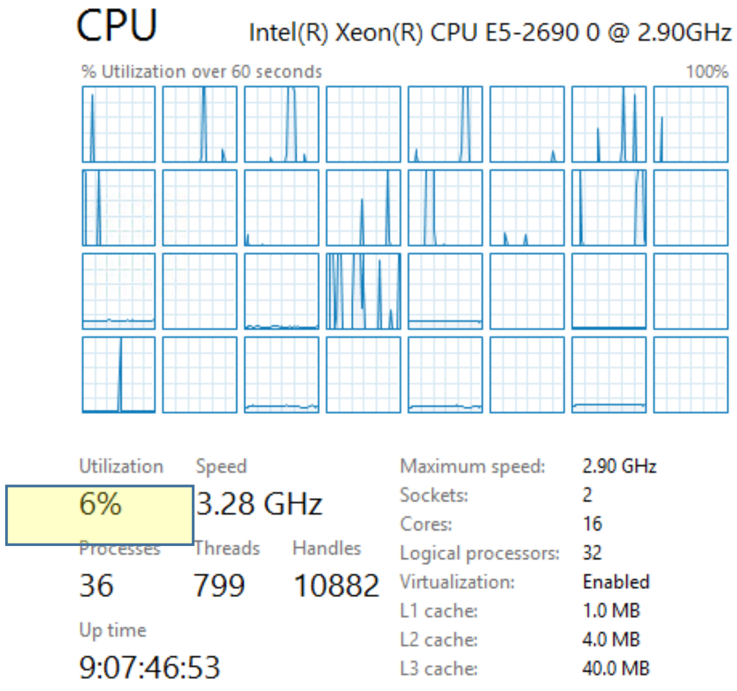
TCP latency



Pingmesh
measurement results

TCP processing overhead (40G)

Sender



CPU Intel(R) Xeon(R) CPU E5-2690 0 @ 2.90GHz

Utilization 6%

Speed 3.28 GHz

Processes 36

Threads 799

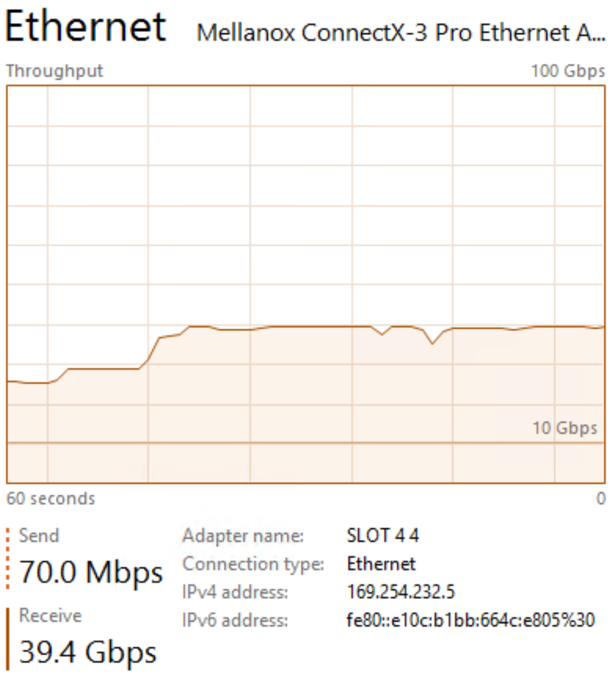
Handles 10882

Up time 9:07:46:53

Maximum speed: 2.90 GHz
Sockets: 2
Cores: 16
Logical processors: 32
Virtualization: Enabled
L1 cache: 1.0 MB
L2 cache: 4.0 MB
L3 cache: 40.0 MB

8 tcp connections

40G NIC



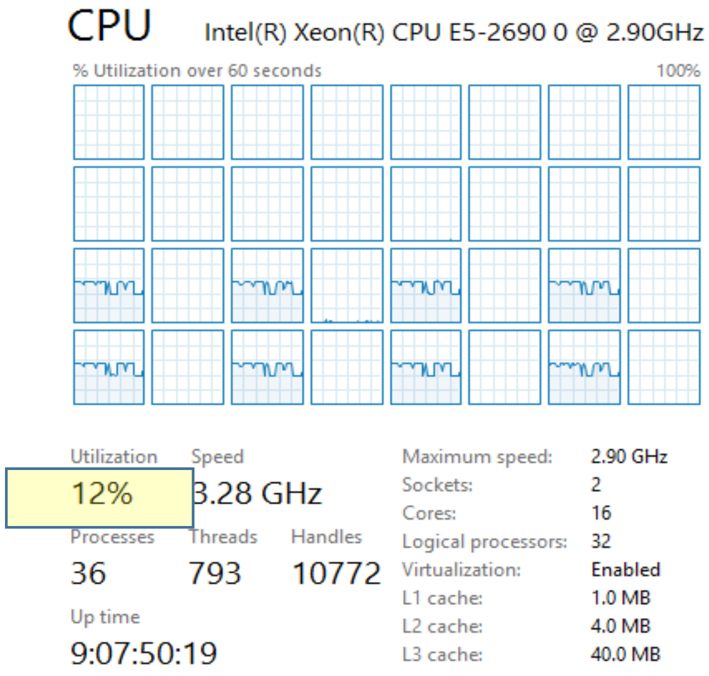
Ethernet Mellanox ConnectX-3 Pro Ethernet A...

Send 70.0 Mbps

Receive 39.4 Gbps

Adapter name: SLOT 4 4
Connection type: Ethernet
IPv4 address: 169.254.232.5
IPv6 address: fe80::e10c:b1bb:664c:e805%30

Receiver



CPU Intel(R) Xeon(R) CPU E5-2690 0 @ 2.90GHz

Utilization 12%

Speed 3.28 GHz

Processes 36

Threads 793

Handles 10772

Up time 9:07:50:19

Maximum speed: 2.90 GHz
Sockets: 2
Cores: 16
Logical processors: 32
Virtualization: Enabled
L1 cache: 1.0 MB
L2 cache: 4.0 MB
L3 cache: 40.0 MB

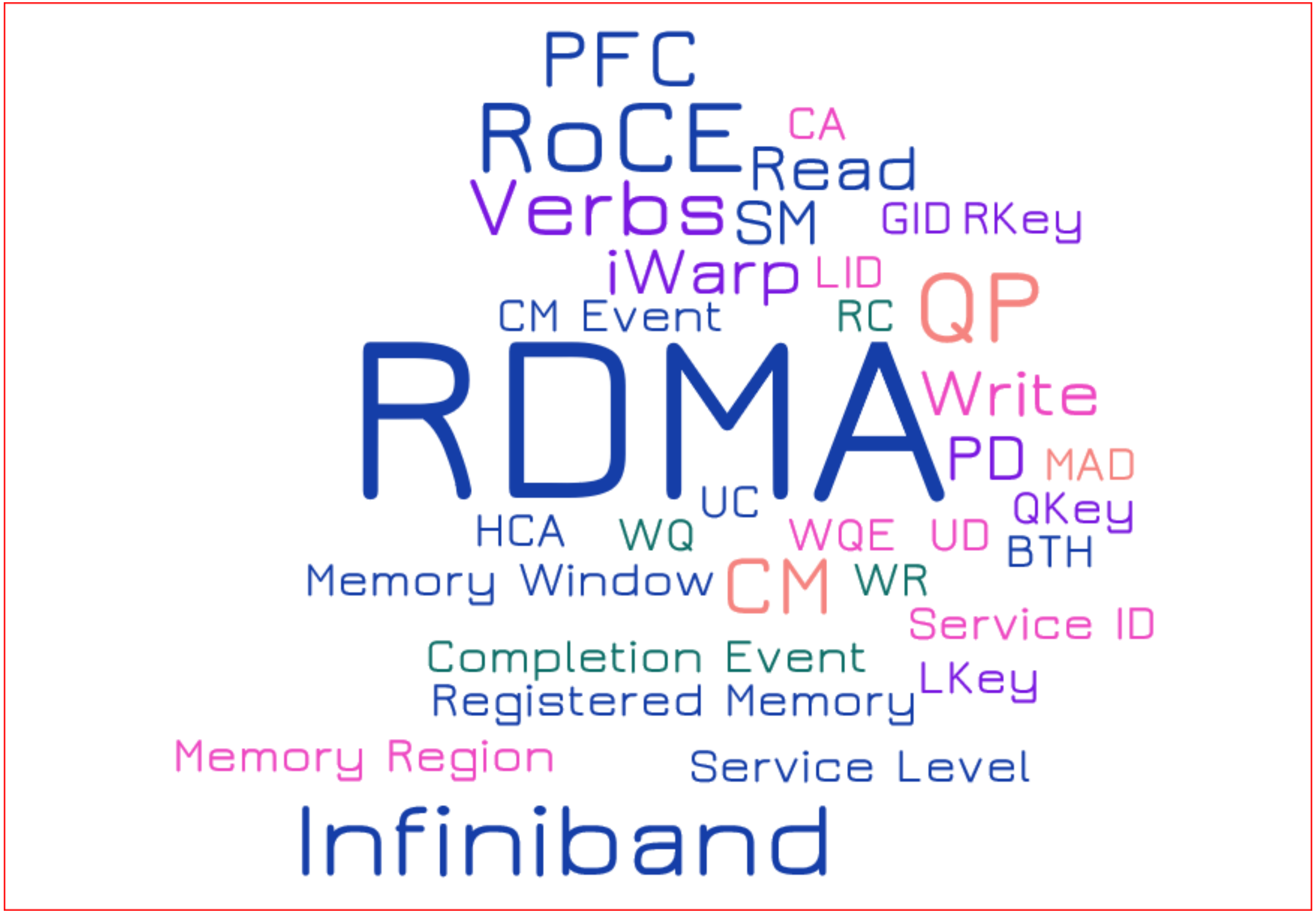
An RDMA renaissance story

Virtual Interface Architecture Spec

1.0	1997

Infiniband Architecture Spec

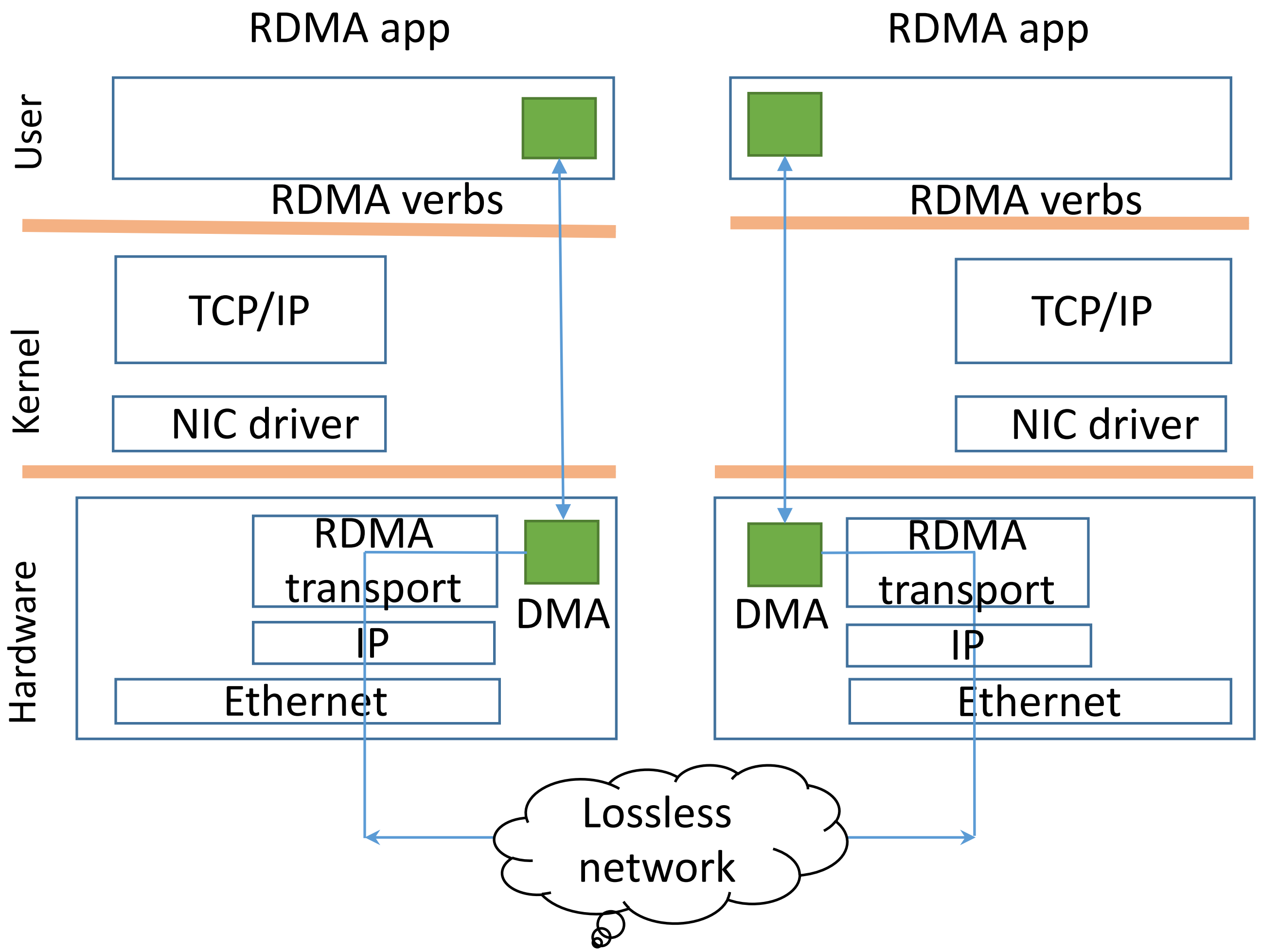
1.0	2000
1.1	2002
1.2	2004
1.3	2015
RoCE	2010
RoCEv2	2014



RDMA

- Remote Direct Memory Access (RDMA): Method of accessing memory on a remote system ***without*** interrupting the processing of the CPU(s) on that system
- RDMA offloads packet processing protocols to the NIC
- RDMA in Ethernet based data centers

RoCEv2: RDMA over Commodity Ethernet

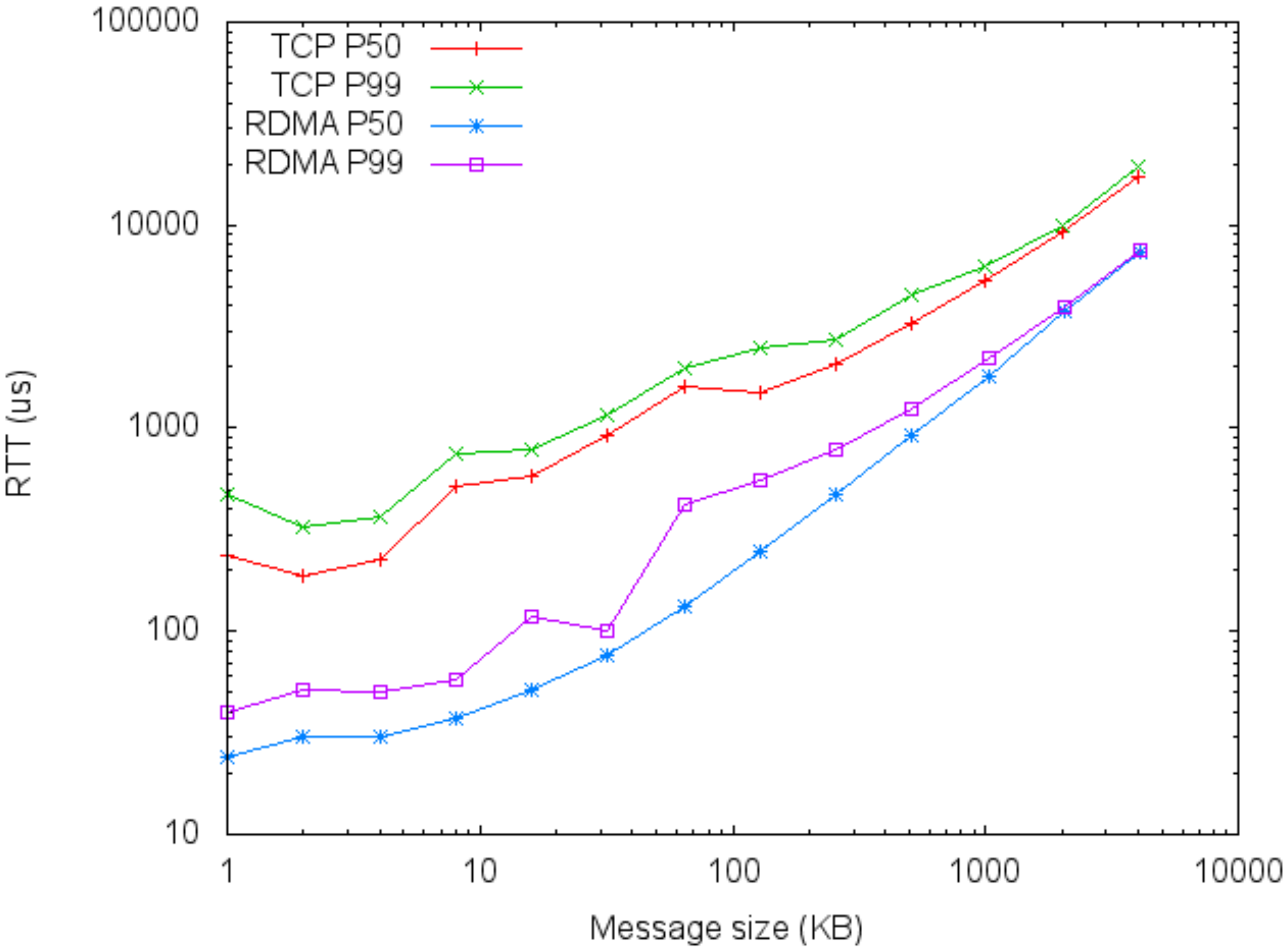


- RoCEv2 for Ethernet based data centers
- RoCEv2 encapsulates packets in UDP
- OS kernel is not in data path
- NIC for network protocol processing and message DMA

RDMA benefit: latency reduction

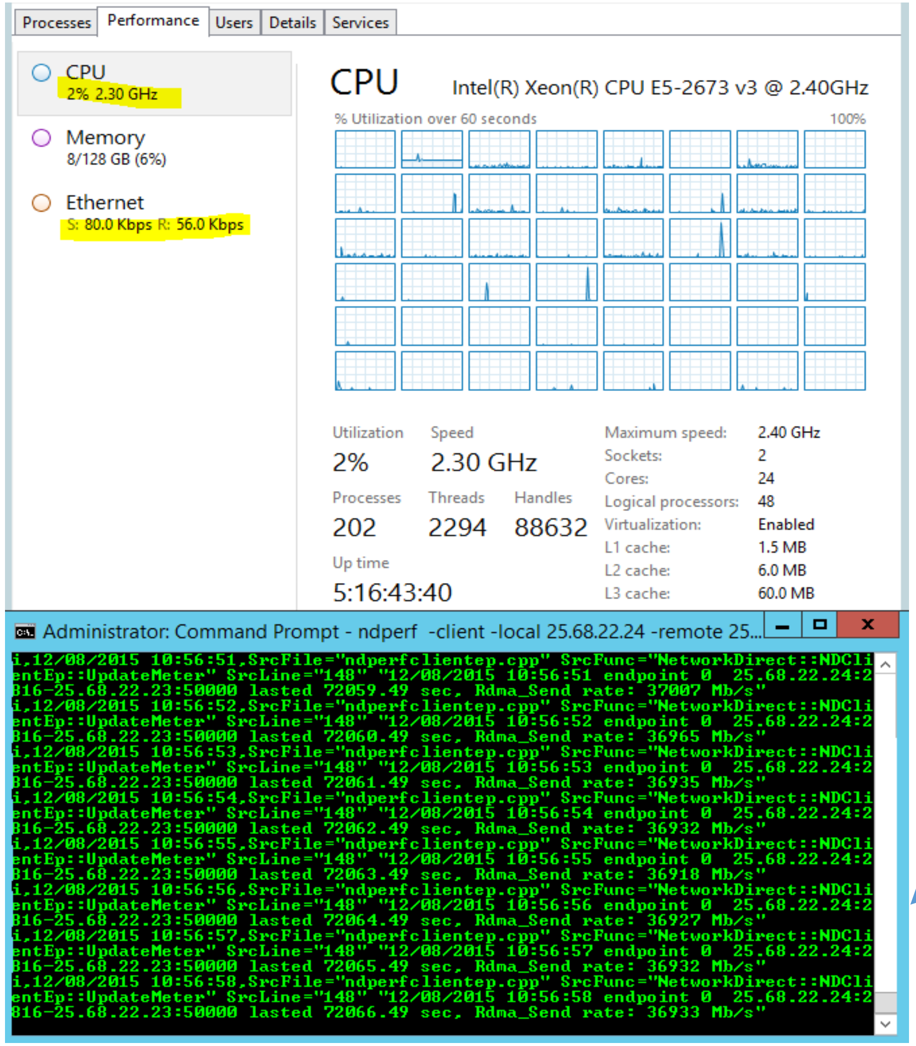
Msg size	TCP P50 (us)	TCP P99 (us)	RDMA P50 (us)	RDMA P99 (us)
1KB	236	467	24	40
16KB	580	788	51	117
128KB	1483	2491	247	551
1MB	5290	6195	1783	2214

- For small msgs (<32KB), OS processing latency matters
- For large msgs (100KB+), speed matters



RDMA benefit: CPU overhead reduction

Sender



CPU
2% 2.30 GHz

Memory
8/128 GB (6%)

Ethernet
S: 80.0 Kbps R: 56.0 Kbps

CPU Intel(R) Xeon(R) CPU E5-2673 v3 @ 2.40GHz

Utilization 2%
Speed 2.30 GHz
Processes 202
Threads 2294
Handles 88632
Up time 5:16:43:40

Maximum speed: 2.40 GHz
Sockets: 2
Cores: 24
Logical processors: 48
Virtualization: Enabled
L1 cache: 1.5 MB
L2 cache: 6.0 MB
L3 cache: 60.0 MB

Administrator: Command Prompt - ndperf -client -local 25.68.22.24 -remote 25...

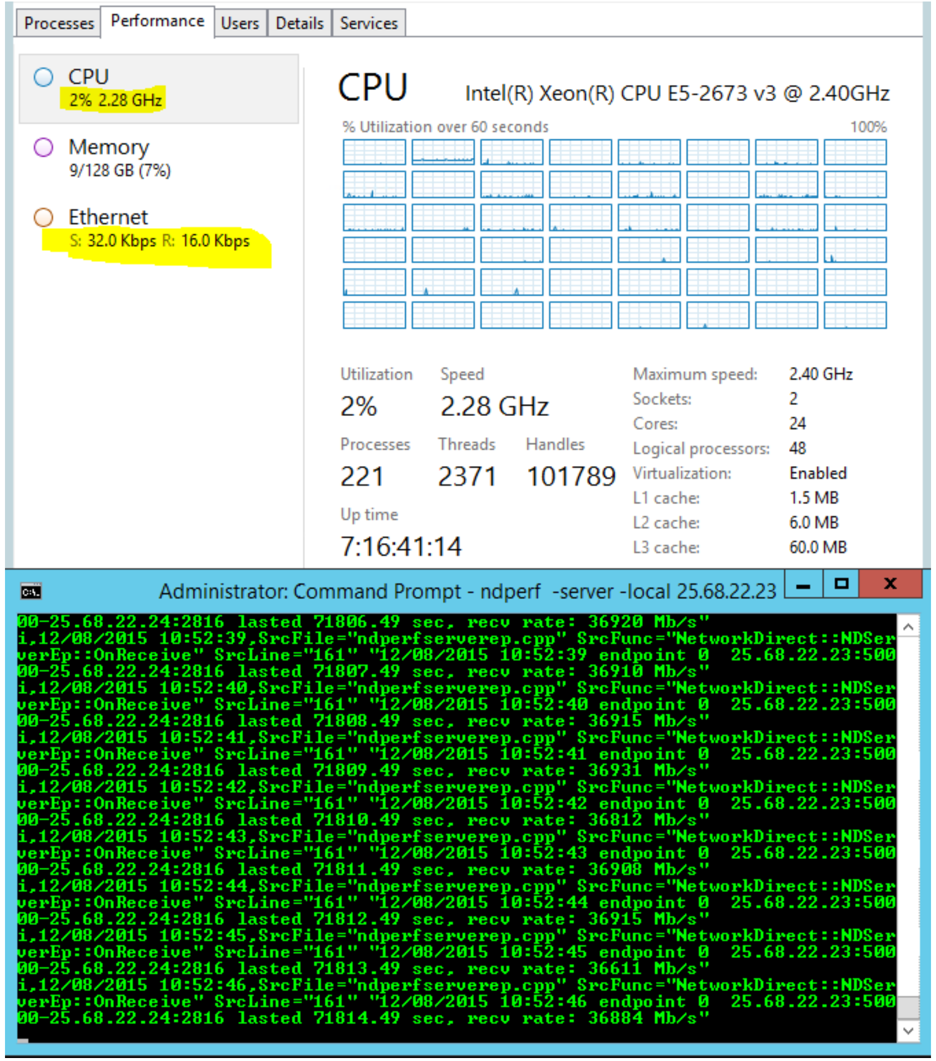
```
i,12/08/2015 10:56:51,SrcFile="ndperfclientep.cpp" SrcFunc="NetworkDirect::NDCli
entEp::UpdateMeter" SrcLine="148" "12/08/2015 10:56:51 endpoint 0  25.68.22.24:2
816-25.68.22.23:50000 lasted 72059.49 sec, Rdma_Send rate: 37007 Mb/s"
i,12/08/2015 10:56:52,SrcFile="ndperfclientep.cpp" SrcFunc="NetworkDirect::NDCli
entEp::UpdateMeter" SrcLine="148" "12/08/2015 10:56:52 endpoint 0  25.68.22.24:2
816-25.68.22.23:50000 lasted 72060.49 sec, Rdma_Send rate: 36965 Mb/s"
i,12/08/2015 10:56:53,SrcFile="ndperfclientep.cpp" SrcFunc="NetworkDirect::NDCli
entEp::UpdateMeter" SrcLine="148" "12/08/2015 10:56:53 endpoint 0  25.68.22.24:2
816-25.68.22.23:50000 lasted 72061.49 sec, Rdma_Send rate: 36935 Mb/s"
i,12/08/2015 10:56:54,SrcFile="ndperfclientep.cpp" SrcFunc="NetworkDirect::NDCli
entEp::UpdateMeter" SrcLine="148" "12/08/2015 10:56:54 endpoint 0  25.68.22.24:2
816-25.68.22.23:50000 lasted 72062.49 sec, Rdma_Send rate: 36932 Mb/s"
i,12/08/2015 10:56:55,SrcFile="ndperfclientep.cpp" SrcFunc="NetworkDirect::NDCli
entEp::UpdateMeter" SrcLine="148" "12/08/2015 10:56:55 endpoint 0  25.68.22.24:2
816-25.68.22.23:50000 lasted 72063.49 sec, Rdma_Send rate: 36918 Mb/s"
i,12/08/2015 10:56:56,SrcFile="ndperfclientep.cpp" SrcFunc="NetworkDirect::NDCli
entEp::UpdateMeter" SrcLine="148" "12/08/2015 10:56:56 endpoint 0  25.68.22.24:2
816-25.68.22.23:50000 lasted 72064.49 sec, Rdma_Send rate: 36927 Mb/s"
i,12/08/2015 10:56:57,SrcFile="ndperfclientep.cpp" SrcFunc="NetworkDirect::NDCli
entEp::UpdateMeter" SrcLine="148" "12/08/2015 10:56:57 endpoint 0  25.68.22.24:2
816-25.68.22.23:50000 lasted 72065.49 sec, Rdma_Send rate: 36932 Mb/s"
i,12/08/2015 10:56:58,SrcFile="ndperfclientep.cpp" SrcFunc="NetworkDirect::NDCli
entEp::UpdateMeter" SrcLine="148" "12/08/2015 10:56:58 endpoint 0  25.68.22.24:2
816-25.68.22.23:50000 lasted 72066.49 sec, Rdma_Send rate: 36933 Mb/s"
```

One ND connection

40G NIC

37Gb/s goodput

Receiver



CPU
2% 2.28 GHz

Memory
9/128 GB (7%)

Ethernet
S: 32.0 Kbps R: 16.0 Kbps

CPU Intel(R) Xeon(R) CPU E5-2673 v3 @ 2.40GHz

Utilization 2%
Speed 2.28 GHz
Processes 221
Threads 2371
Handles 101789
Up time 7:16:41:14

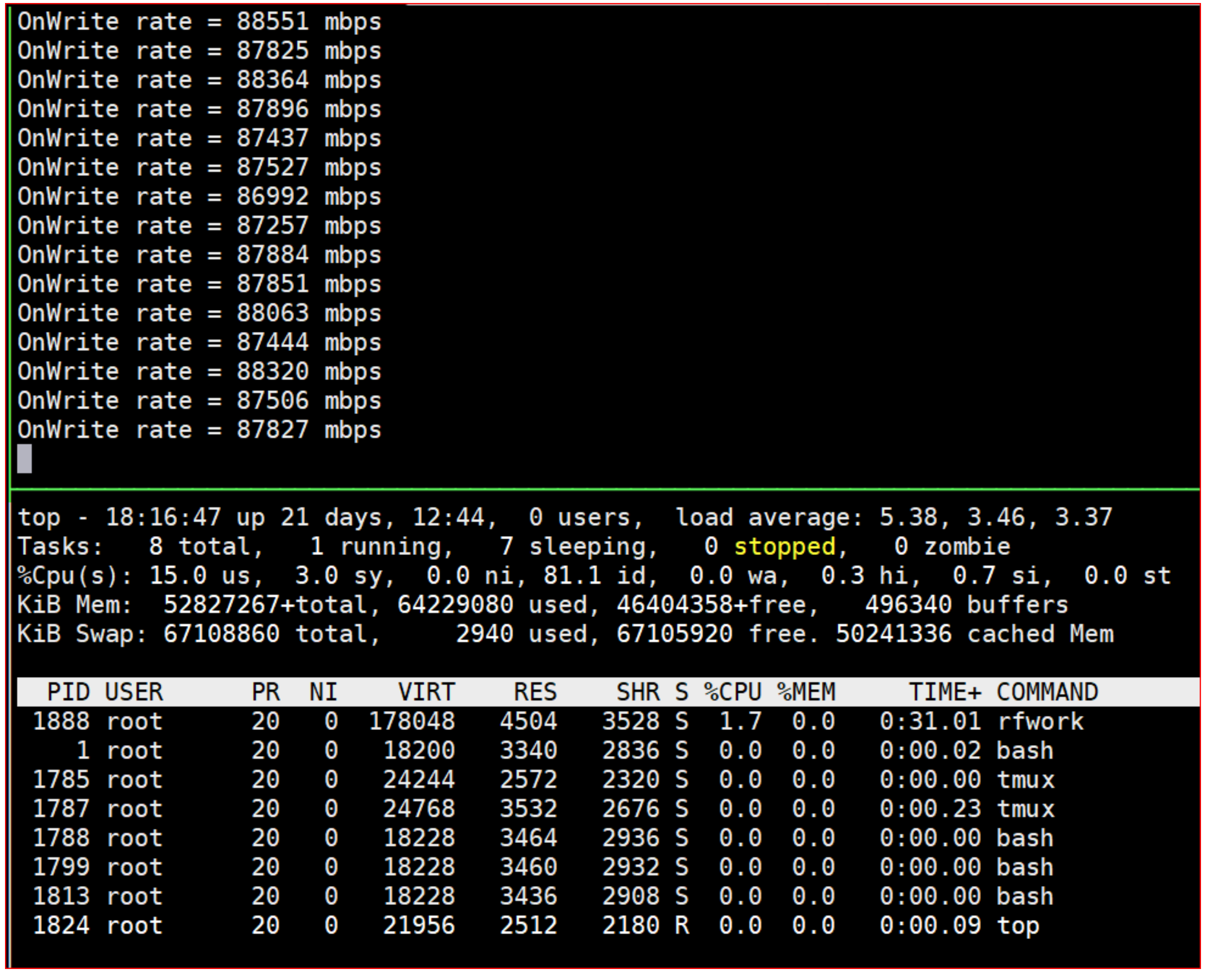
Maximum speed: 2.40 GHz
Sockets: 2
Cores: 24
Logical processors: 48
Virtualization: Enabled
L1 cache: 1.5 MB
L2 cache: 6.0 MB
L3 cache: 60.0 MB

Administrator: Command Prompt - ndperf -server -local 25.68.22.23

```
00-25.68.22.24:2816 lasted 71806.49 sec, recv rate: 36920 Mb/s"
i,12/08/2015 10:52:39,SrcFile="ndperfserverep.cpp" SrcFunc="NetworkDirect::NDSer
verEp::OnReceive" SrcLine="161" "12/08/2015 10:52:39 endpoint 0  25.68.22.23:500
00-25.68.22.24:2816 lasted 71807.49 sec, recv rate: 36910 Mb/s"
i,12/08/2015 10:52:40,SrcFile="ndperfserverep.cpp" SrcFunc="NetworkDirect::NDSer
verEp::OnReceive" SrcLine="161" "12/08/2015 10:52:40 endpoint 0  25.68.22.23:500
00-25.68.22.24:2816 lasted 71808.49 sec, recv rate: 36915 Mb/s"
i,12/08/2015 10:52:41,SrcFile="ndperfserverep.cpp" SrcFunc="NetworkDirect::NDSer
verEp::OnReceive" SrcLine="161" "12/08/2015 10:52:41 endpoint 0  25.68.22.23:500
00-25.68.22.24:2816 lasted 71809.49 sec, recv rate: 36931 Mb/s"
i,12/08/2015 10:52:42,SrcFile="ndperfserverep.cpp" SrcFunc="NetworkDirect::NDSer
verEp::OnReceive" SrcLine="161" "12/08/2015 10:52:42 endpoint 0  25.68.22.23:500
00-25.68.22.24:2816 lasted 71810.49 sec, recv rate: 36812 Mb/s"
i,12/08/2015 10:52:43,SrcFile="ndperfserverep.cpp" SrcFunc="NetworkDirect::NDSer
verEp::OnReceive" SrcLine="161" "12/08/2015 10:52:43 endpoint 0  25.68.22.23:500
00-25.68.22.24:2816 lasted 71811.49 sec, recv rate: 36908 Mb/s"
i,12/08/2015 10:52:44,SrcFile="ndperfserverep.cpp" SrcFunc="NetworkDirect::NDSer
verEp::OnReceive" SrcLine="161" "12/08/2015 10:52:44 endpoint 0  25.68.22.23:500
00-25.68.22.24:2816 lasted 71812.49 sec, recv rate: 36915 Mb/s"
i,12/08/2015 10:52:45,SrcFile="ndperfserverep.cpp" SrcFunc="NetworkDirect::NDSer
verEp::OnReceive" SrcLine="161" "12/08/2015 10:52:45 endpoint 0  25.68.22.23:500
00-25.68.22.24:2816 lasted 71813.49 sec, recv rate: 36611 Mb/s"
i,12/08/2015 10:52:46,SrcFile="ndperfserverep.cpp" SrcFunc="NetworkDirect::NDSer
verEp::OnReceive" SrcLine="161" "12/08/2015 10:52:46 endpoint 0  25.68.22.23:500
00-25.68.22.24:2816 lasted 71814.49 sec, recv rate: 36884 Mb/s"
```

RDMA benefit: CPU overhead reduction

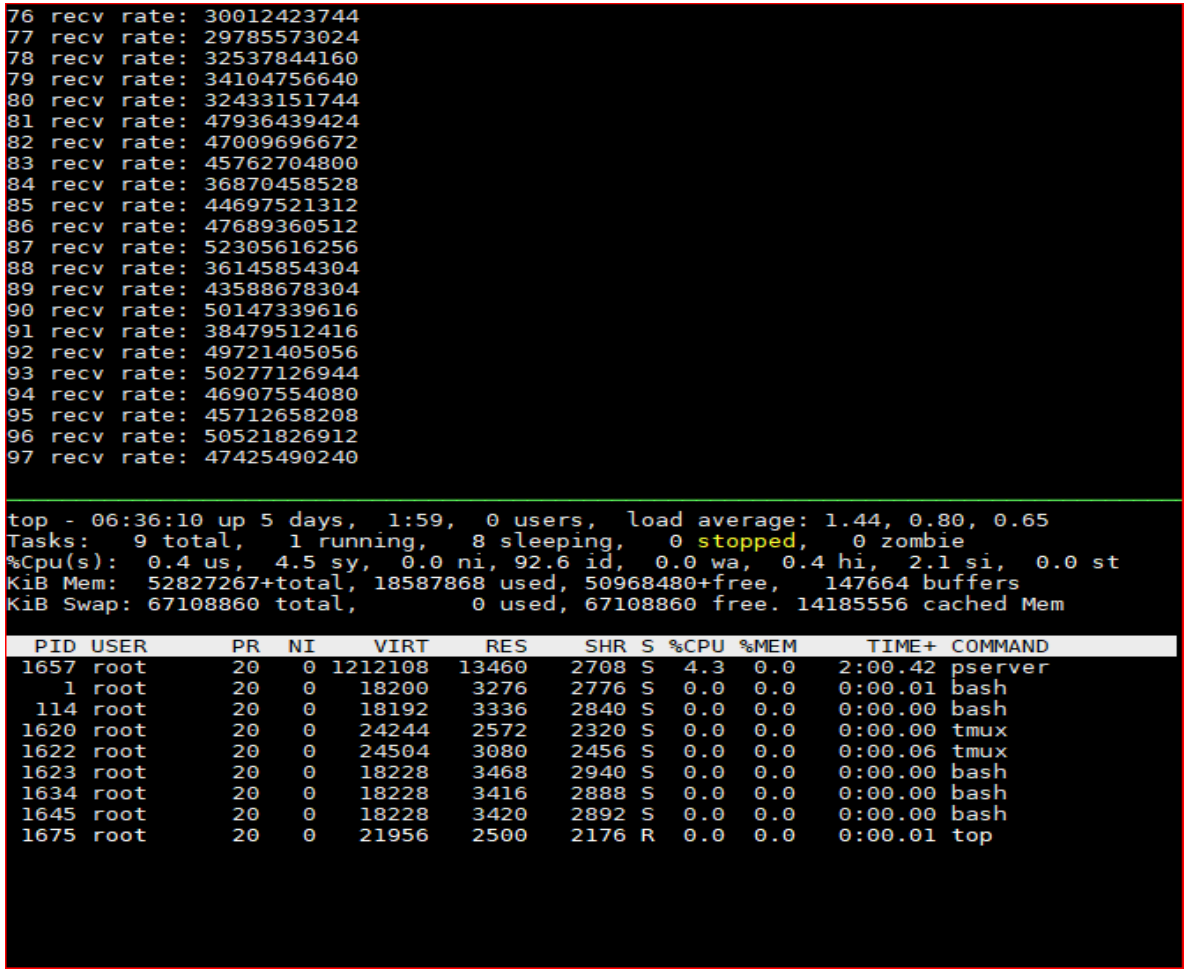
Intel(R) Xeon(R) CPU E5-2690 v4 @ 2.60GHz, two sockets 28 cores



```
OnWrite rate = 88551 mbps
OnWrite rate = 87825 mbps
OnWrite rate = 88364 mbps
OnWrite rate = 87896 mbps
OnWrite rate = 87437 mbps
OnWrite rate = 87527 mbps
OnWrite rate = 86992 mbps
OnWrite rate = 87257 mbps
OnWrite rate = 87884 mbps
OnWrite rate = 87851 mbps
OnWrite rate = 88063 mbps
OnWrite rate = 87444 mbps
OnWrite rate = 88320 mbps
OnWrite rate = 87506 mbps
OnWrite rate = 87827 mbps

top - 18:16:47 up 21 days, 12:44,  0 users,  load average: 5.38, 3.46, 3.37
Tasks:   8 total,   1 running,   7 sleeping,   0 stopped,   0 zombie
%Cpu(s): 15.0 us,  3.0 sy,  0.0 ni, 81.1 id,  0.0 wa,  0.3 hi,  0.7 si,  0.0 st
KiB Mem:  52827267+total, 64229080 used, 46404358+free,   496340 buffers
KiB Swap: 67108860 total,     2940 used, 67105920 free. 50241336 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
 1888 root      20   0  178048   4504   3528 S  1.7  0.0   0:31.01 rfwork
    1 root      20   0   18200   3340   2836 S  0.0  0.0   0:00.02 bash
 1785 root      20   0   24244   2572   2320 S  0.0  0.0   0:00.00 tmux
 1787 root      20   0   24768   3532   2676 S  0.0  0.0   0:00.23 tmux
 1788 root      20   0   18228   3464   2936 S  0.0  0.0   0:00.00 bash
 1799 root      20   0   18228   3460   2932 S  0.0  0.0   0:00.00 bash
 1813 root      20   0   18228   3436   2908 S  0.0  0.0   0:00.00 bash
 1824 root      20   0   21956   2512   2180 R  0.0  0.0   0:00.09 top
```



```
76 recv rate: 30012423744
77 recv rate: 29785573024
78 recv rate: 32537844160
79 recv rate: 34104756640
80 recv rate: 32433151744
81 recv rate: 47936439424
82 recv rate: 47009696672
83 recv rate: 45762704800
84 recv rate: 36870458528
85 recv rate: 44697521312
86 recv rate: 47689360512
87 recv rate: 52305616256
88 recv rate: 36145854304
89 recv rate: 43588678304
90 recv rate: 50147339616
91 recv rate: 38479512416
92 recv rate: 49721405056
93 recv rate: 50277126944
94 recv rate: 46907554080
95 recv rate: 45712658208
96 recv rate: 50521826912
97 recv rate: 47425490240

top - 06:36:10 up 5 days,  1:59,  0 users,  load average: 1.44, 0.80, 0.65
Tasks:   9 total,   1 running,   8 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.4 us,  4.5 sy,  0.0 ni, 92.6 id,  0.0 wa,  0.4 hi,  2.1 si,  0.0 st
KiB Mem:  52827267+total, 18587868 used, 50968480+free,   147664 buffers
KiB Swap: 67108860 total,        0 used, 67108860 free. 14185556 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
 1657 root      20   0 1212108  13460   2708 S  4.3  0.0   2:00.42 pserver
    1 root      20   0   18200   3276   2776 S  0.0  0.0   0:00.01 bash
  114 root      20   0   18192   3336   2840 S  0.0  0.0   0:00.00 bash
 1620 root      20   0   24244   2572   2320 S  0.0  0.0   0:00.00 tmux
 1622 root      20   0   24504   3080   2456 S  0.0  0.0   0:00.06 tmux
 1623 root      20   0   18228   3468   2940 S  0.0  0.0   0:00.00 bash
 1634 root      20   0   18228   3416   2888 S  0.0  0.0   0:00.00 bash
 1645 root      20   0   18228   3420   2892 S  0.0  0.0   0:00.00 bash
 1675 root      20   0   21956   2500   2176 R  0.0  0.0   0:00.01 top
```

RDMA: Single QP, 88 Gb/s, 1.7% CPU

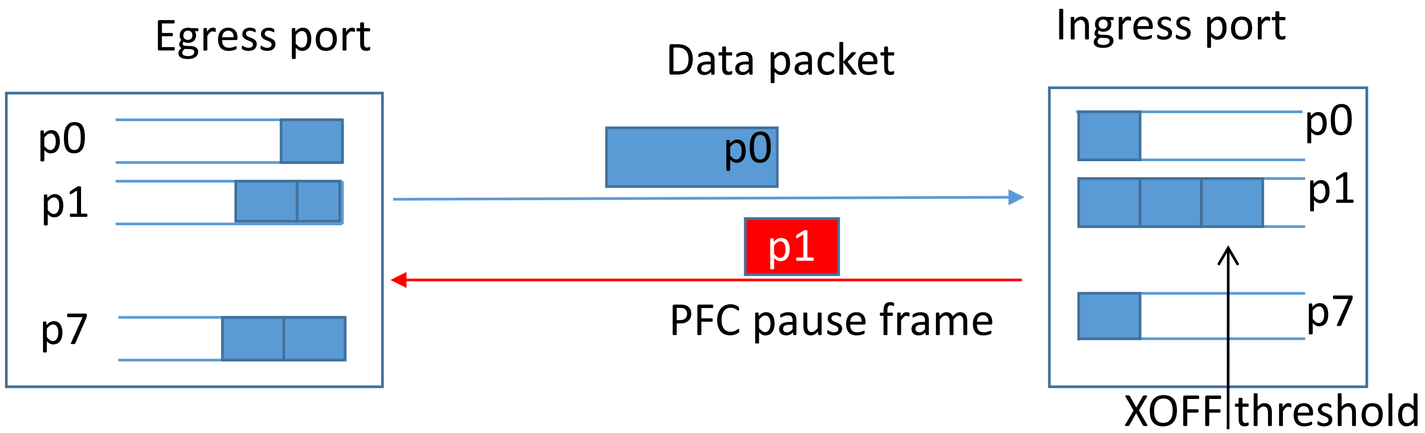
TCP: Eight connections, 30-50Gb/s,
Client: 2.6%, Server: 4.3% CPU

RoCEv2 needs a lossless Ethernet network

- PFC for hop-by-hop flow control
- DCQCN for connection-level congestion control

Priority-based flow control (PFC)

- Hop-by-hop flow control, with eight priorities for HOL blocking mitigation
- The priority in data packets is carried in the VLAN tag or DSCP
- PFC pause frame to inform the upstream to stop
- PFC causes HOL and colleterial damage



Data packet

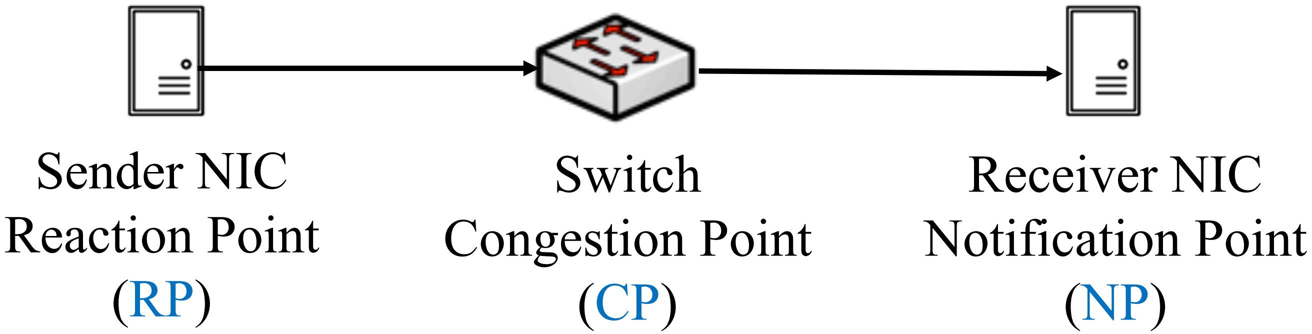
DMAC	SMAC	VLAN TAG	EtherType	Payload	FCS

16bits	3bits	1bits	12bits
TPID 0x8100	TCI		
	PCP	DEI	VID

PFC pause frame

DMAC 01-80-C2-00-00-01	SMAC	EtherType 88-08	OpType 01-01	Priority-enable-vector (2B)	Time-vector (16B)	Padding	FCS

DCQCN



Sender NIC
Reaction Point
(RP)

Switch
Congestion Point
(CP)

Receiver NIC
Notification Point
(NP)

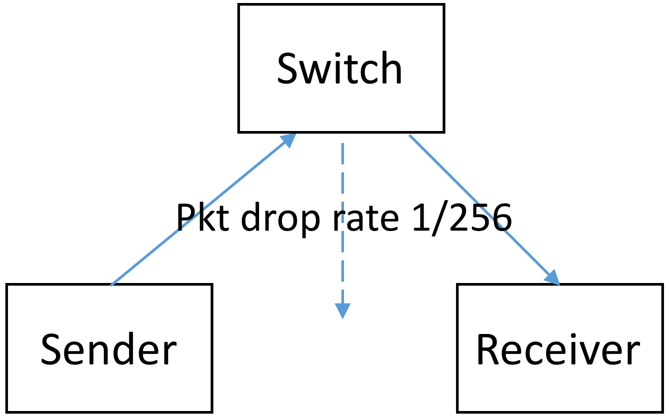
DCQCN = Keep PFC + Use ECN +
hardware rate-based congestion control

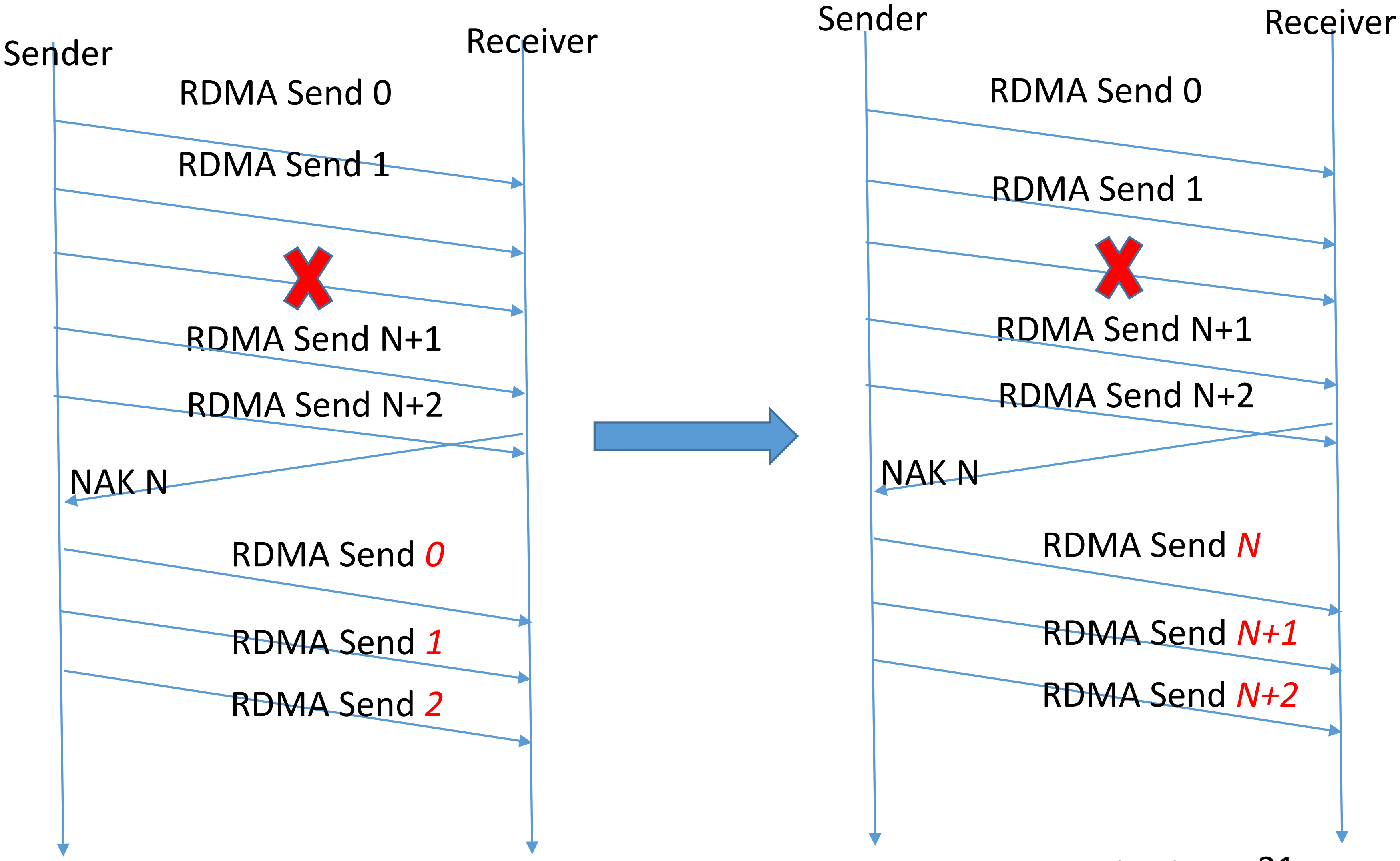
- CP: Switches use ECN for packet marking
- NP: periodically check if ECN-marked packets arrived, if so, notify the sender
- RP: adjust sending rate based on NP feedbacks

The lossless requirement causes safety and performance challenges

- RDMA transport livelock
- PFC deadlock
- PFC pause frame storm
- Slow-receiver symptom

RDMA transport livelock



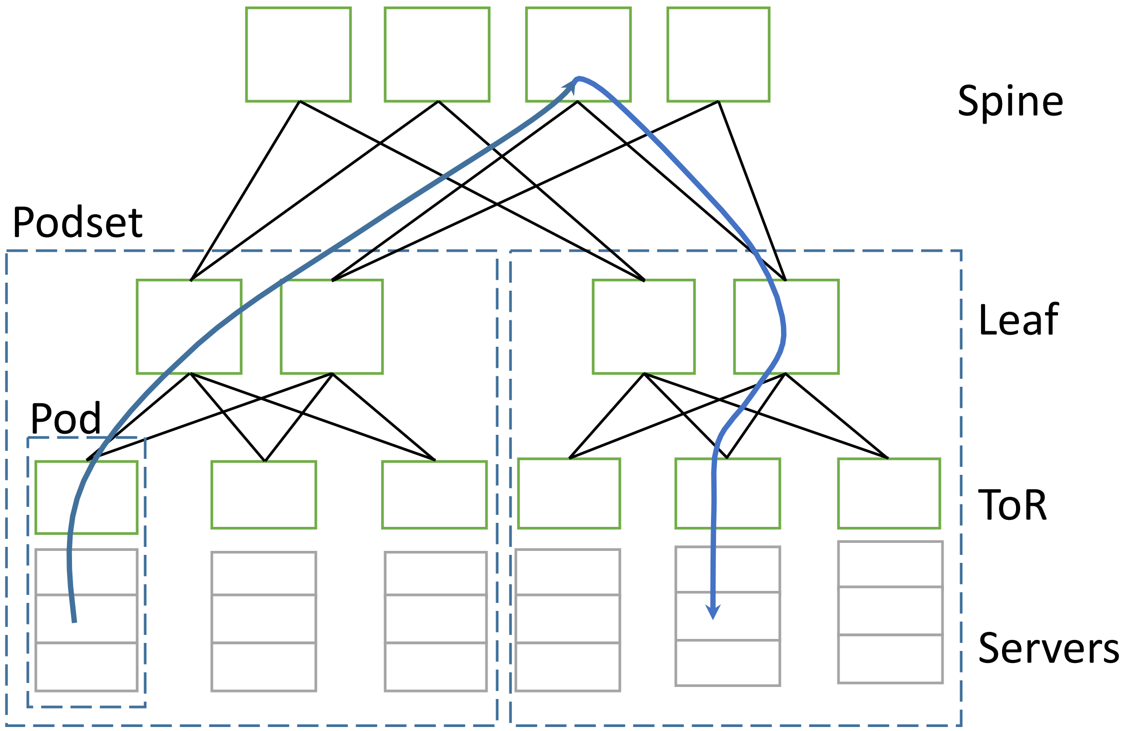


Go-back-0

Go-back-N

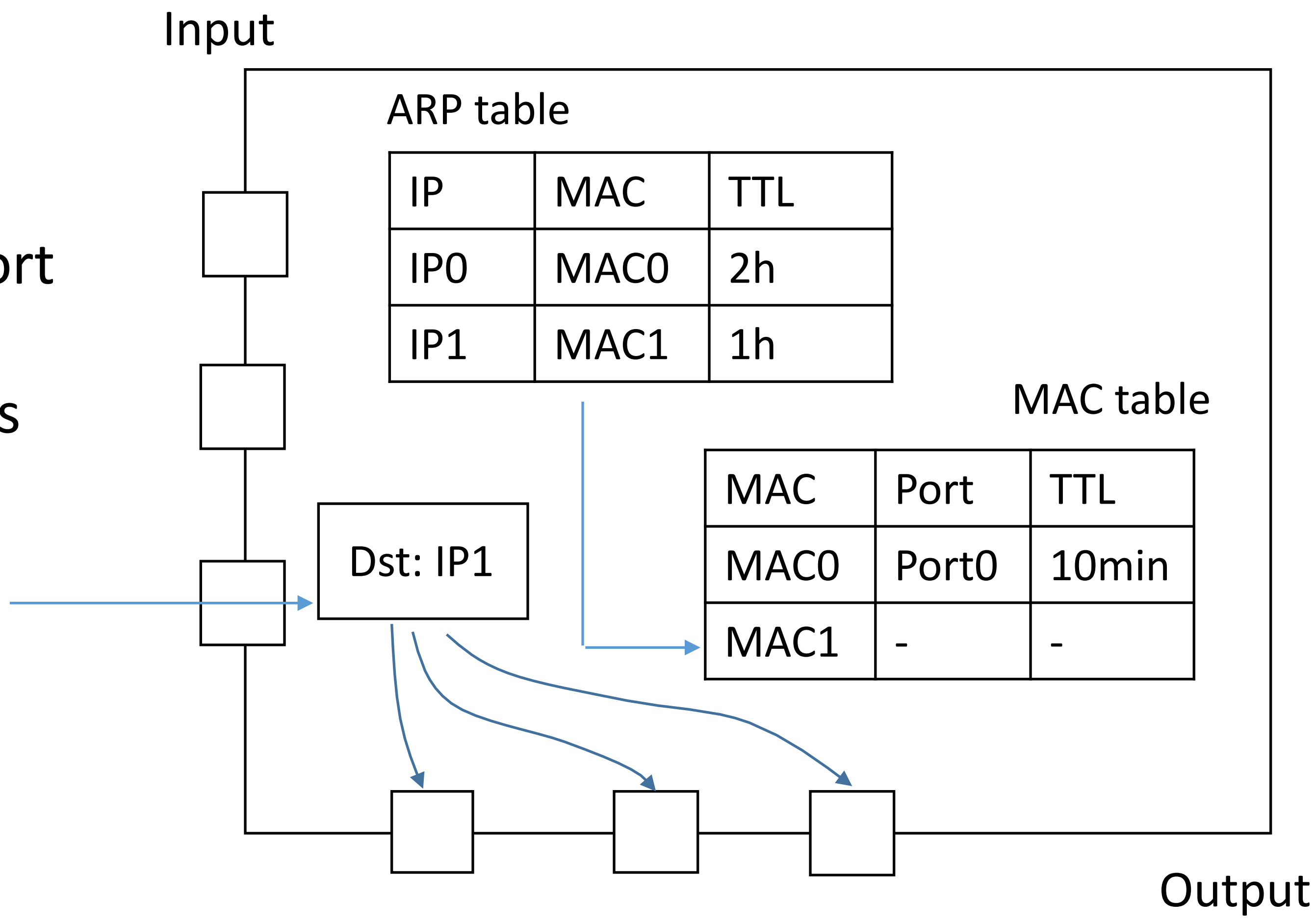
PFC deadlock

- Our data centers use Clos network
- Packets first travel up then go down
- No cyclic buffer dependency for up-down routing -> no deadlock
- But we did experience deadlock!



PFC deadlock

- Preliminaries
  - ARP table: IP address to MAC address mapping
  - MAC table: MAC address to port mapping
  - If MAC entry is missing, packets are flooded to all ports



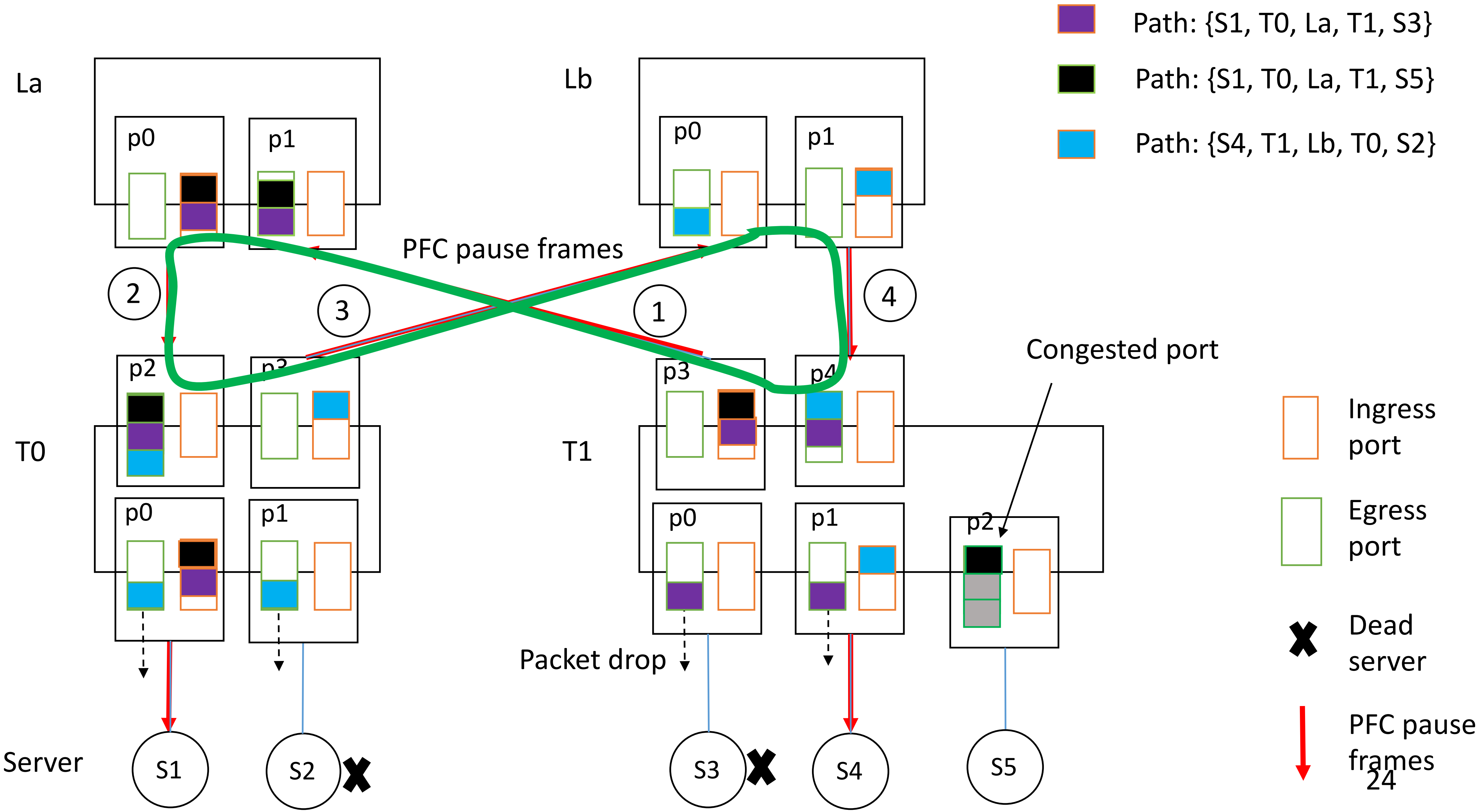
ARP table

IP	MAC	TTL
IP0	MAC0	2h
IP1	MAC1	1h

MAC table

MAC	Port	TTL
MAC0	Port0	10min
MAC1	-	-

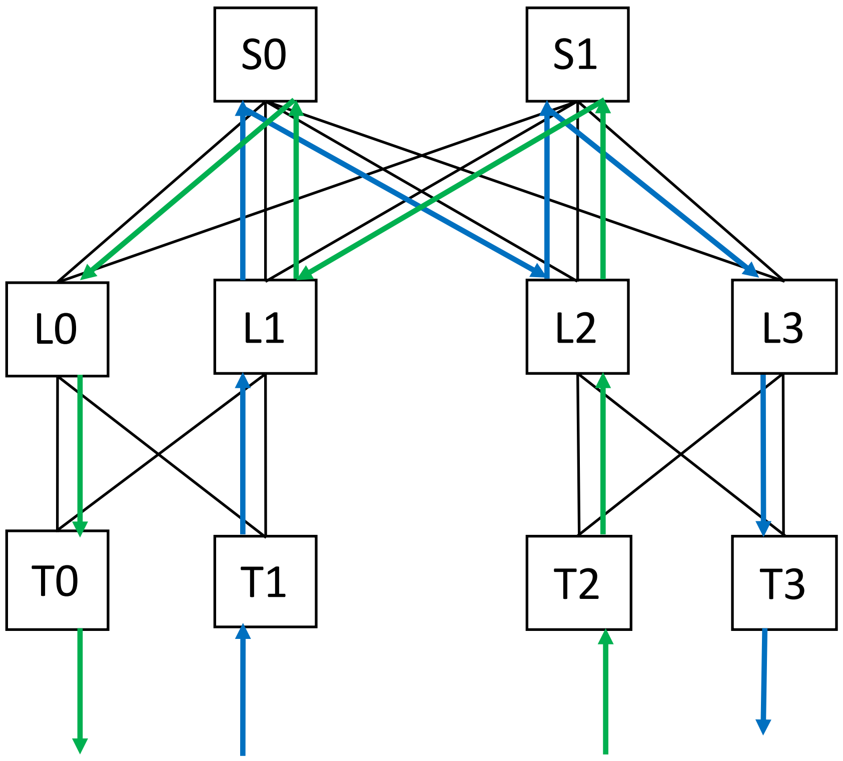
PFC deadlock



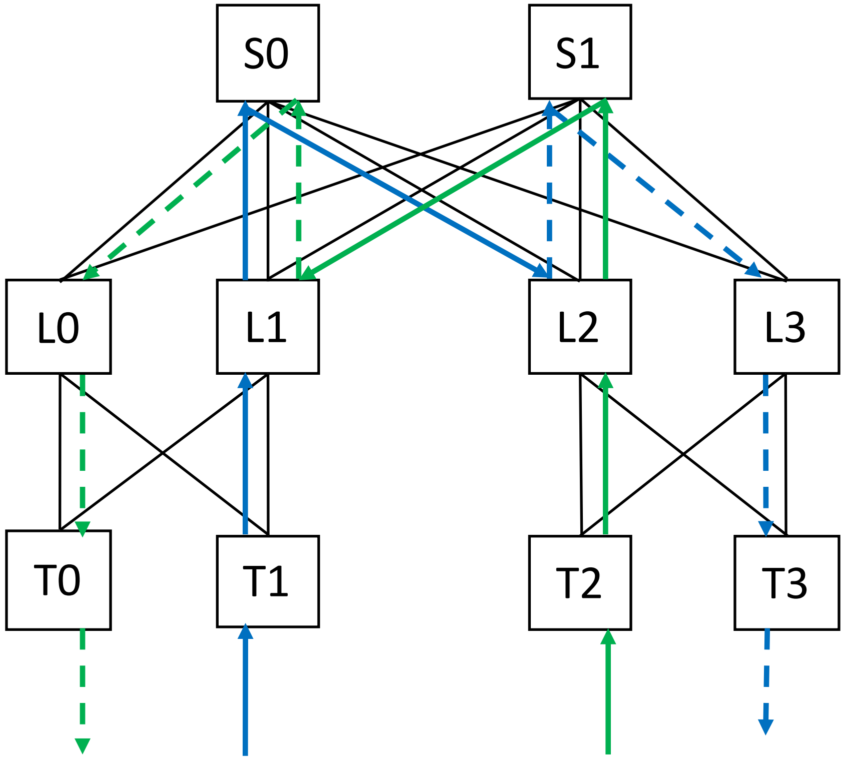
PFC deadlock

- The PFC deadlock root cause: the interaction between the PFC flow control and the Ethernet packet flooding
- Solution: drop the lossless packets if the ARP entry is incomplete
- Recommendation: do not flood or multicast for lossless traffic

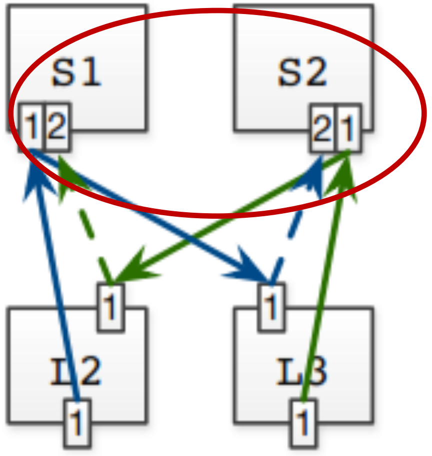
Tagger: practical PFC deadlock prevention



- Concept: Expected Lossless Path (ELP) to decouple Tagger from routing

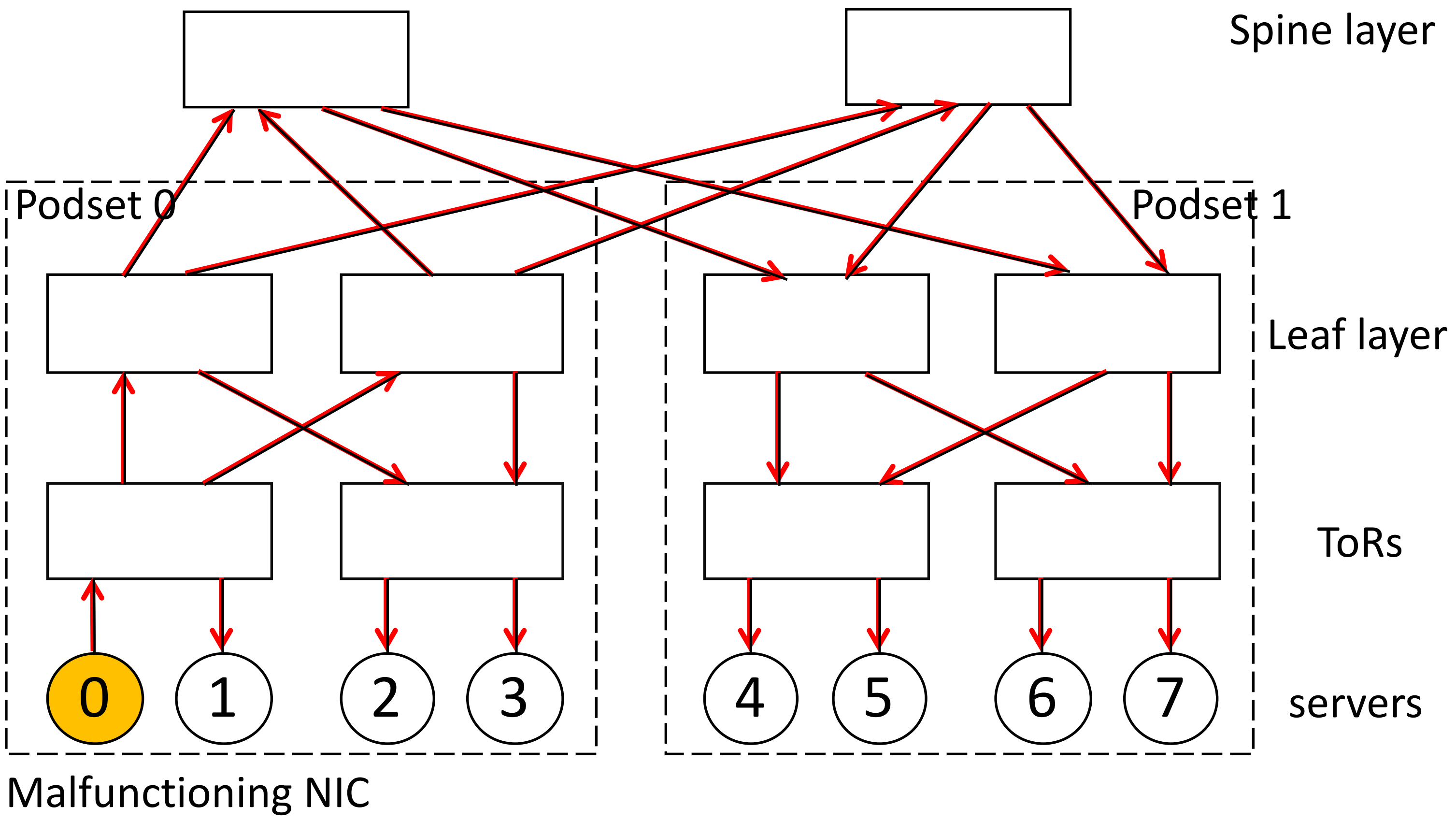


- Strategy: move packets to different lossless queue before CBD forming



- Tagger Algorithm works for general network topology
- Deployable in existing switching ASICs

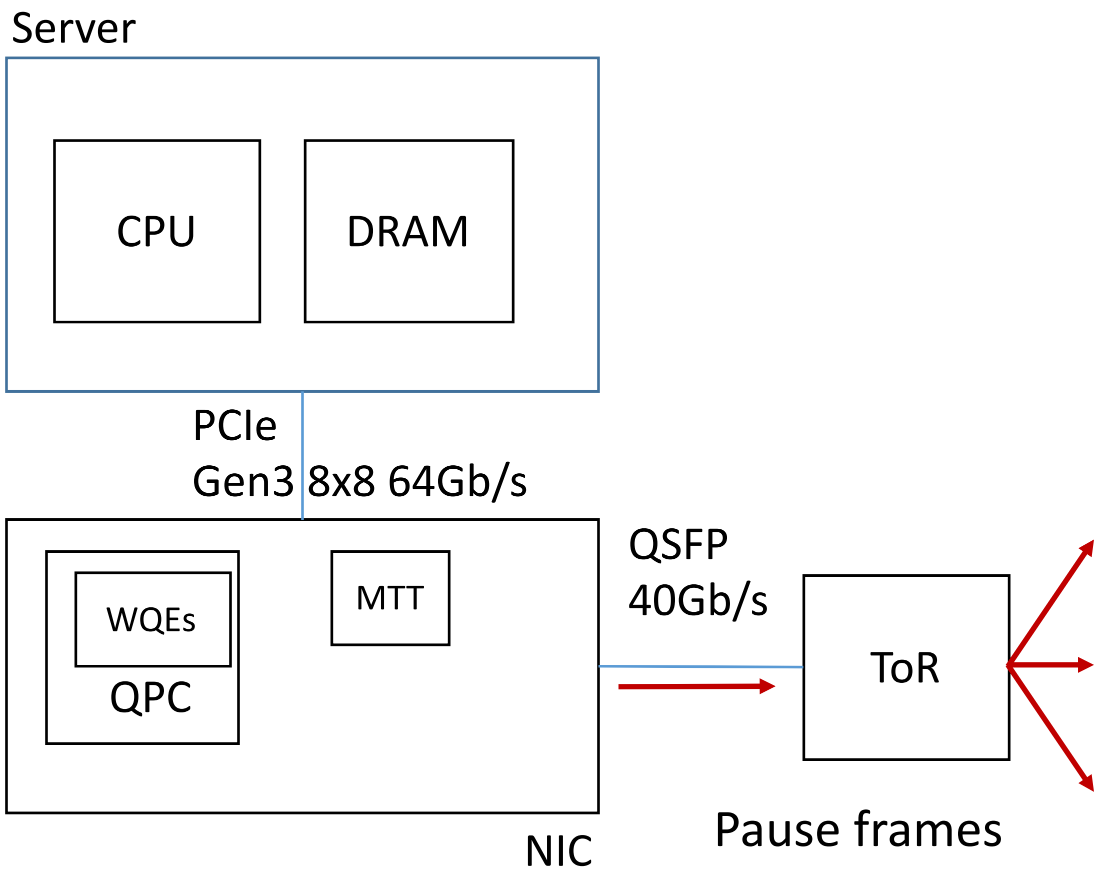
NIC PFC pause frame storm



- A malfunctioning NIC may block the whole network
- PFC pause frame storms caused several incidents
- Solution: watchdogs at both NIC and switch sides to stop the storm

The slow-receiver symptom

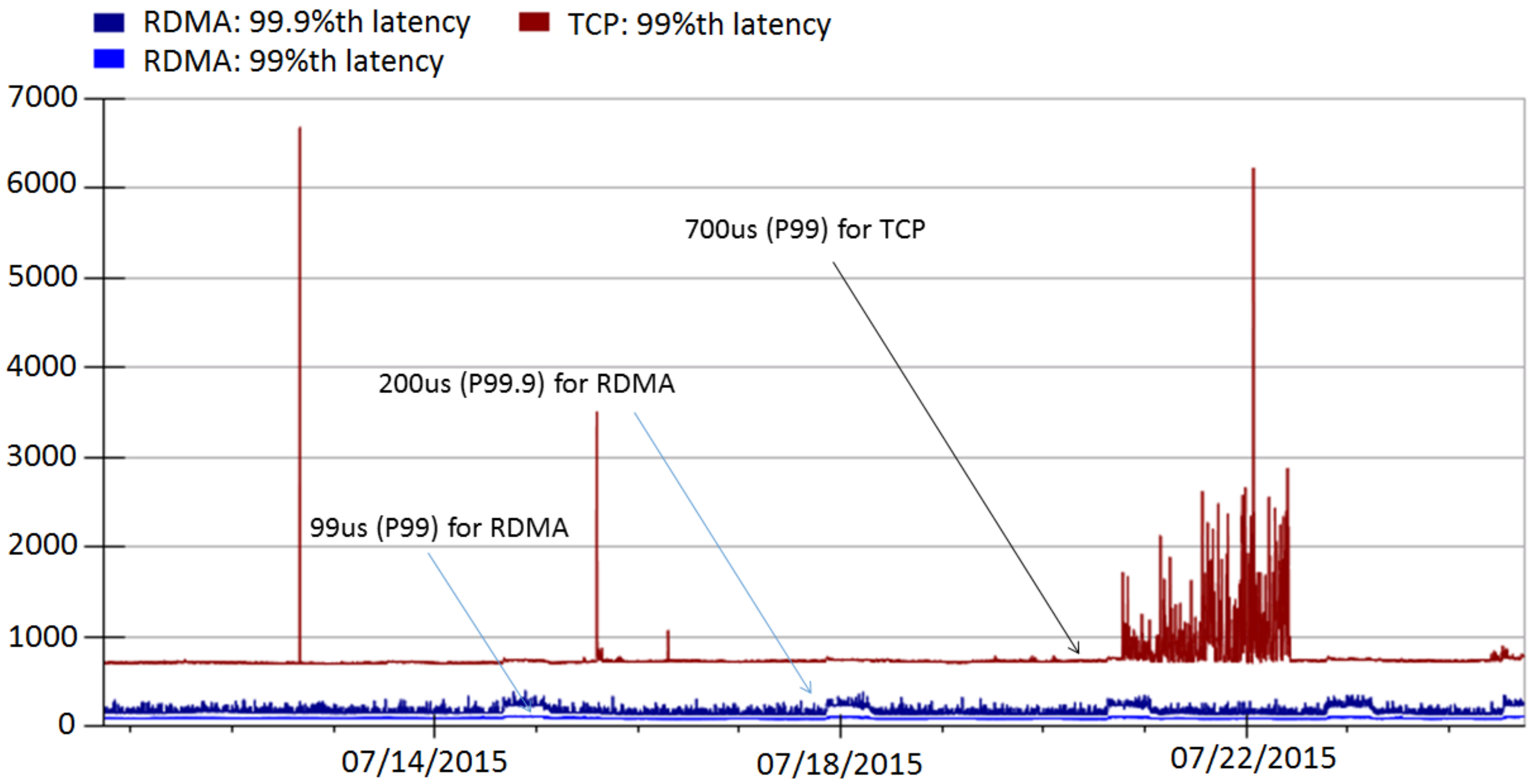
- ToR to NIC is 40Gb/s, NIC to server is 64Gb/s
- But NICs may generate large number of PFC pause frames
- Root cause: NIC is resource constrained
- Mitigation
  - Large page size for the MTT (memory translation table) entry
  - Dynamic buffer sharing at the ToR



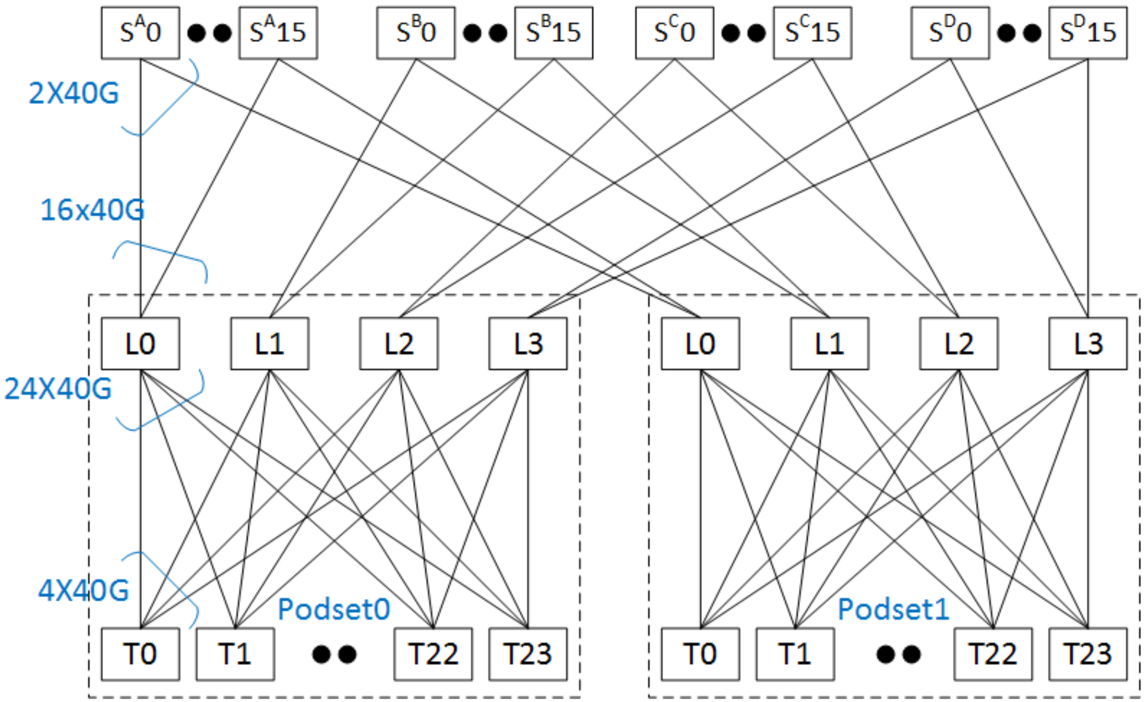
Deployment experiences and lessons learned

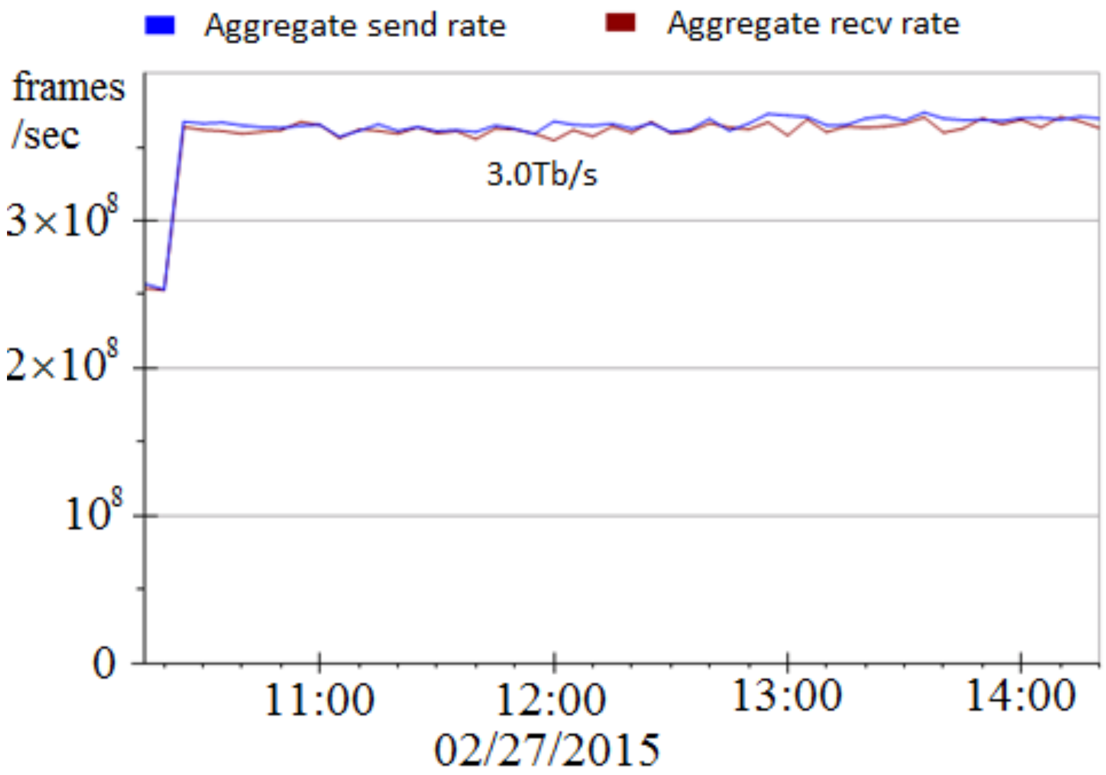
Latency reduction

- RoCEv2 deployed in Bing world-wide for two and half years
- Significant latency reduction
- Incast problem solved as no packet drops



RDMA throughput

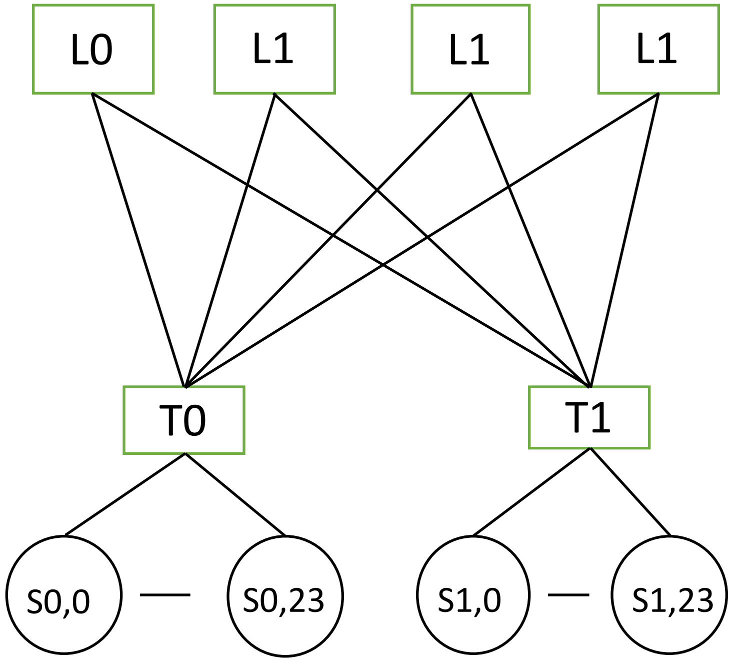


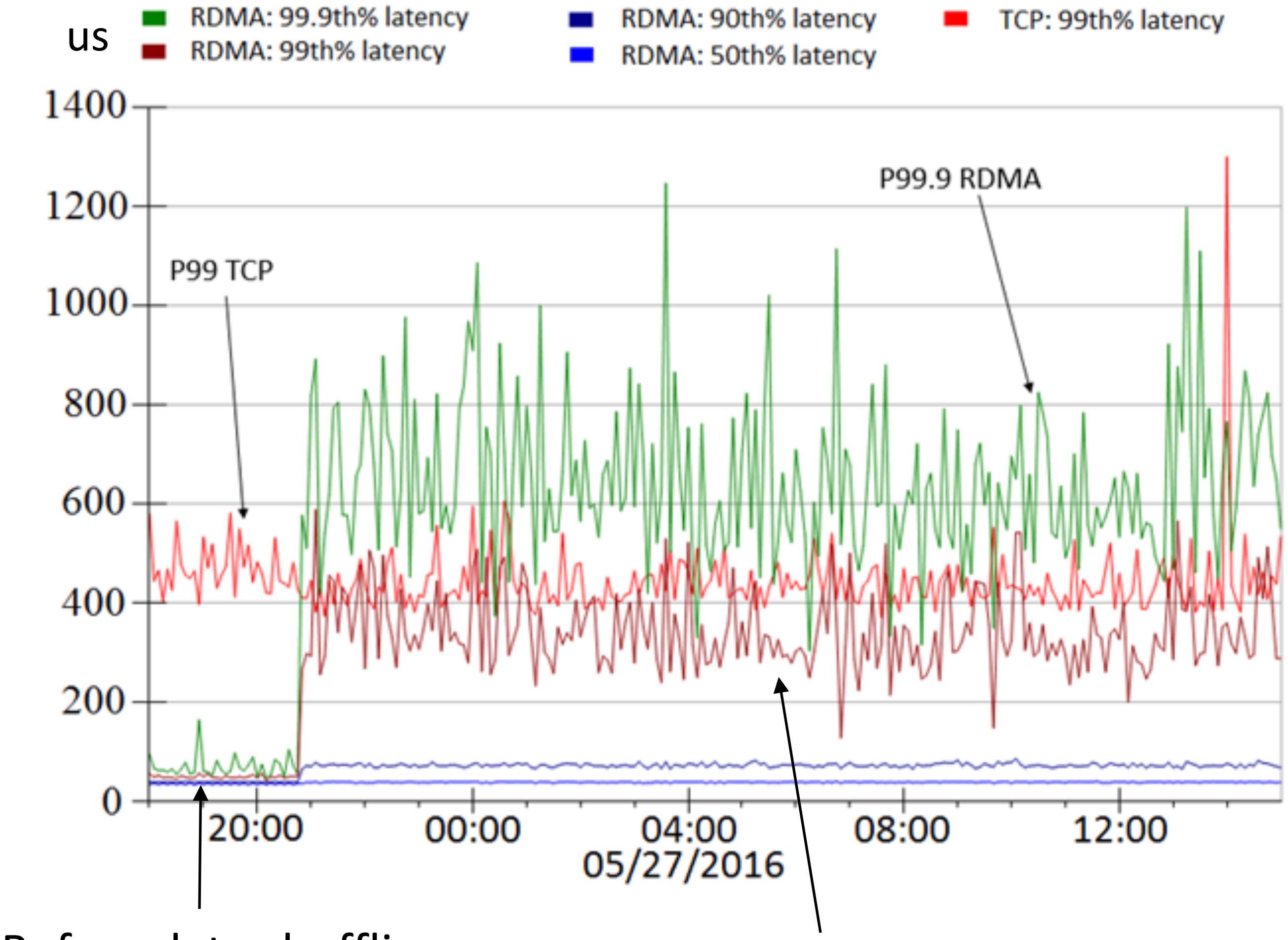


- Using two podsets each with 500+ servers
- 5Tb/s capacity between the two podsets

- Achieved 3Tb/s inter-podset throughput
- Bottlenecked by ECMP routing
- Close to 0 CPU overhead

Latency and throughput tradeoff





Before data shuffling

During data shuffling

- RDMA latencies increase as data shuffling started
- Low latency vs high throughput

Lessons learned

- Providing lossless is hard!
- Deadlock, livelock, PFC pause frames propagation and storm did happen
- Be prepared for the unexpected
  - Configuration management, latency/availability, PFC pause frame, RDMA traffic monitoring
- NICs are the key to make RoCEv2 work

What’s next?

Applications

- RDMA for X (Search, Storage, HFT, DNN, etc.)

Architectures

- Software vs hardware
- Lossy vs lossless network
- RDMA for heterogenous computing systems

Technologies

- RDMA programming
- RDMA virtualization
- RDMA security
- Inter-DC RDMA

Protocols

- Practical, large-scale deadlock free network
- Reducing colleterial damage

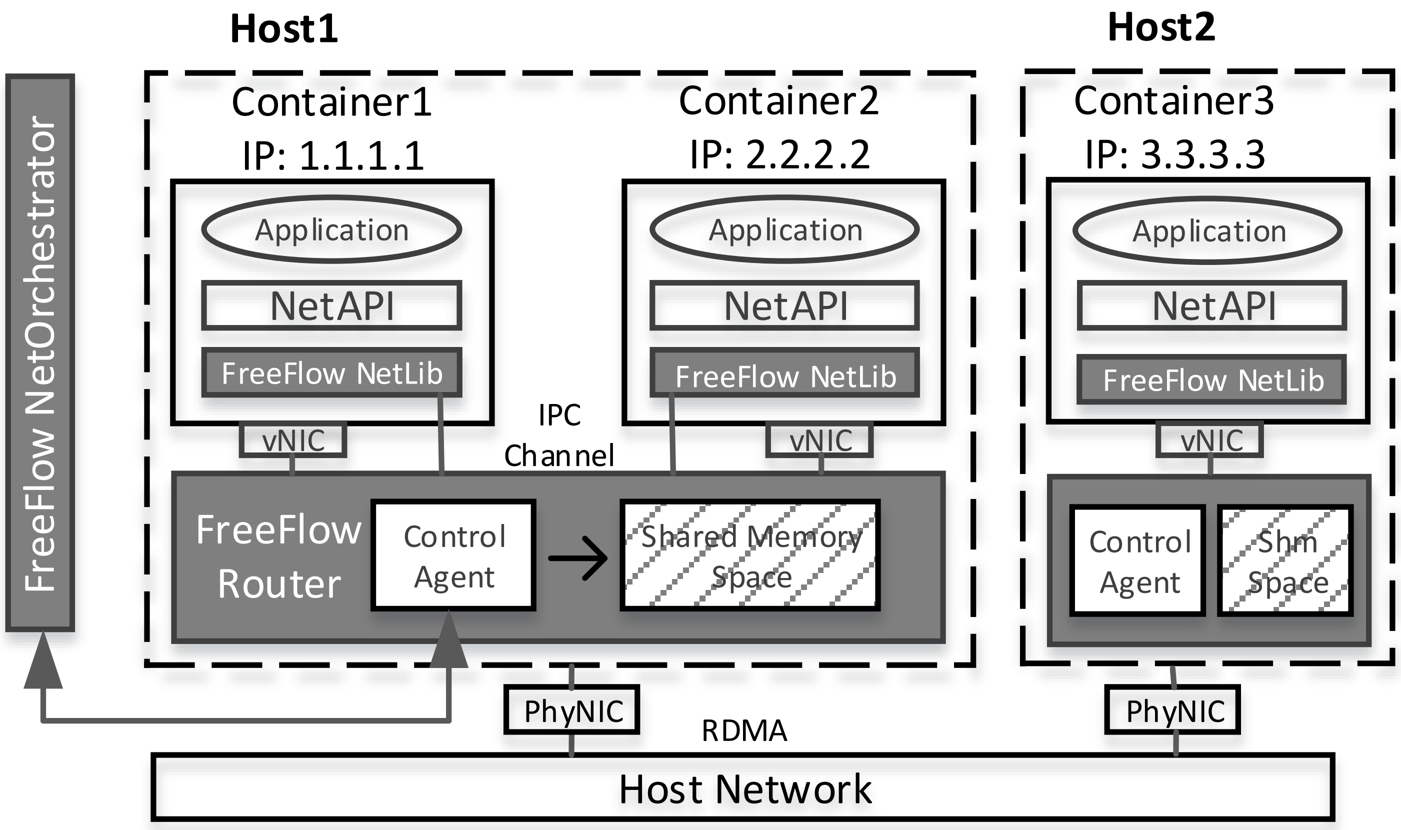
Will software win (again)?

- Historically, software based packet processing won (multiple times)
  - TCP processing overhead analysis by David Clark, et al.
  - Non of the stateful TCP offloading took off (e.g., TCP Chimney)
- The story is different this time
  - Moore's law is ending
  - Accelerators are coming
  - Network speed keep increasing
  - Demands for ultra low latency are real

Is lossless mandatory for RDMA?

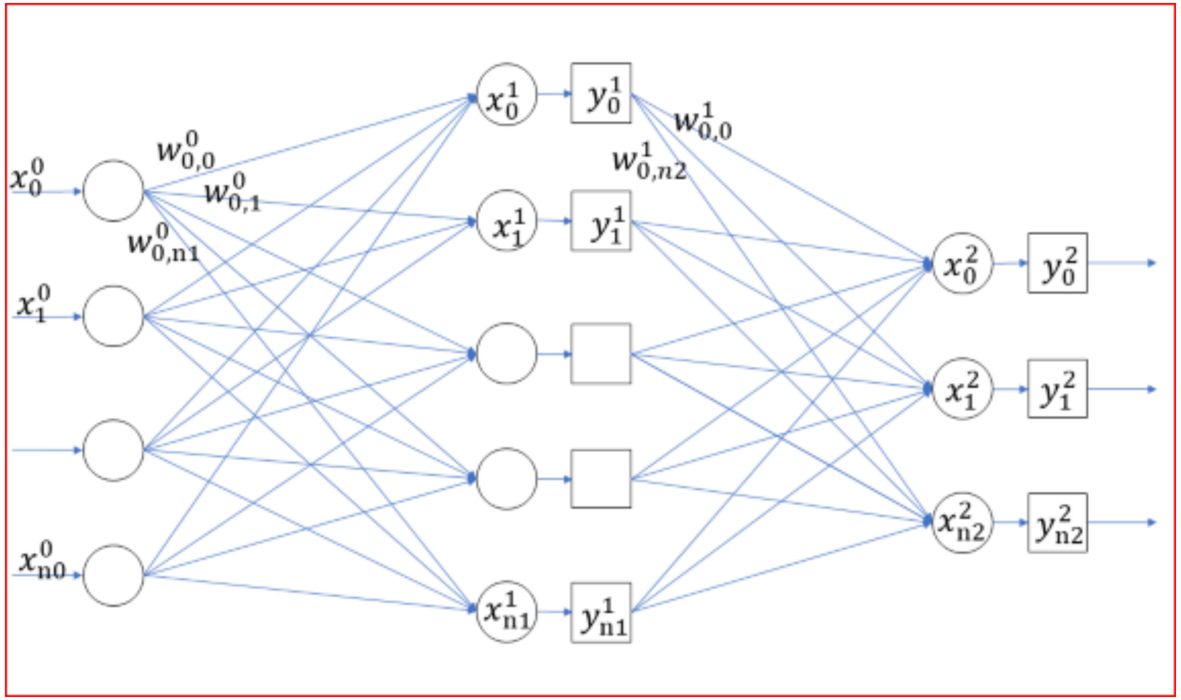
- There is no binding between RDMA and lossless network
- But implementing more sophisticated transport protocol in hardware is a challenge

RDMA virtualization for the container networking



- A router acts as a proxy for the containers
- Shared memory for improved performance
- Zero copy possible

RDMA for DNN



- TCP does not work for distributed DNN training
- For 16-GPU, 2-host speech training with CNTK, TCP communications dominant the training time (72%), RDMA is much faster (44%)

RDMA Programming

- How many LOC for a “hello world” communication using RDMA?
- For TCP, it is 60 LOC for client or server code
- For RDMA, it is complicated …
  - IBVerbs: 600 LOC
  - RCMA CM: 300 LOC
  - Rsocket: 60 LOC

RDMA Programming

- Make RDMA programming more accessible
  - Easy-to-setup RDMA server and switch configurations
  - Can I run and debug my RDMA code on my desktop/laptop?
  - High quality code samples
- Loosely coupled vs tightly coupled (Send/Recv vs Write/Read)

Summary: RDMA for data centers!

- RDMA is experiencing a renaissance in data centers
  - RoCEv2 has been running safely in Microsoft data centers for two and half years
- Many opportunities and interesting problems for high-speed, low-latency RDMA networking
- Many opportunities in making RDMA accessible to more developers

Acknowledgement

- Yan Cai, Gang Cheng, Zhong Deng, Daniel Firestone, Juncheng Gu, Shuihai Hu, Hongqiang Liu, Marina Lipshteyn, Ali Monfared, Jitendra Padhye, Gaurav Soni, Haitao Wu, Jianxi Ye, Yibo Zhu
- Azure, Bing, CNTK, Philly collaborators
- Arista Networks, Cisco, Dell, Mellanox partners

Questions?