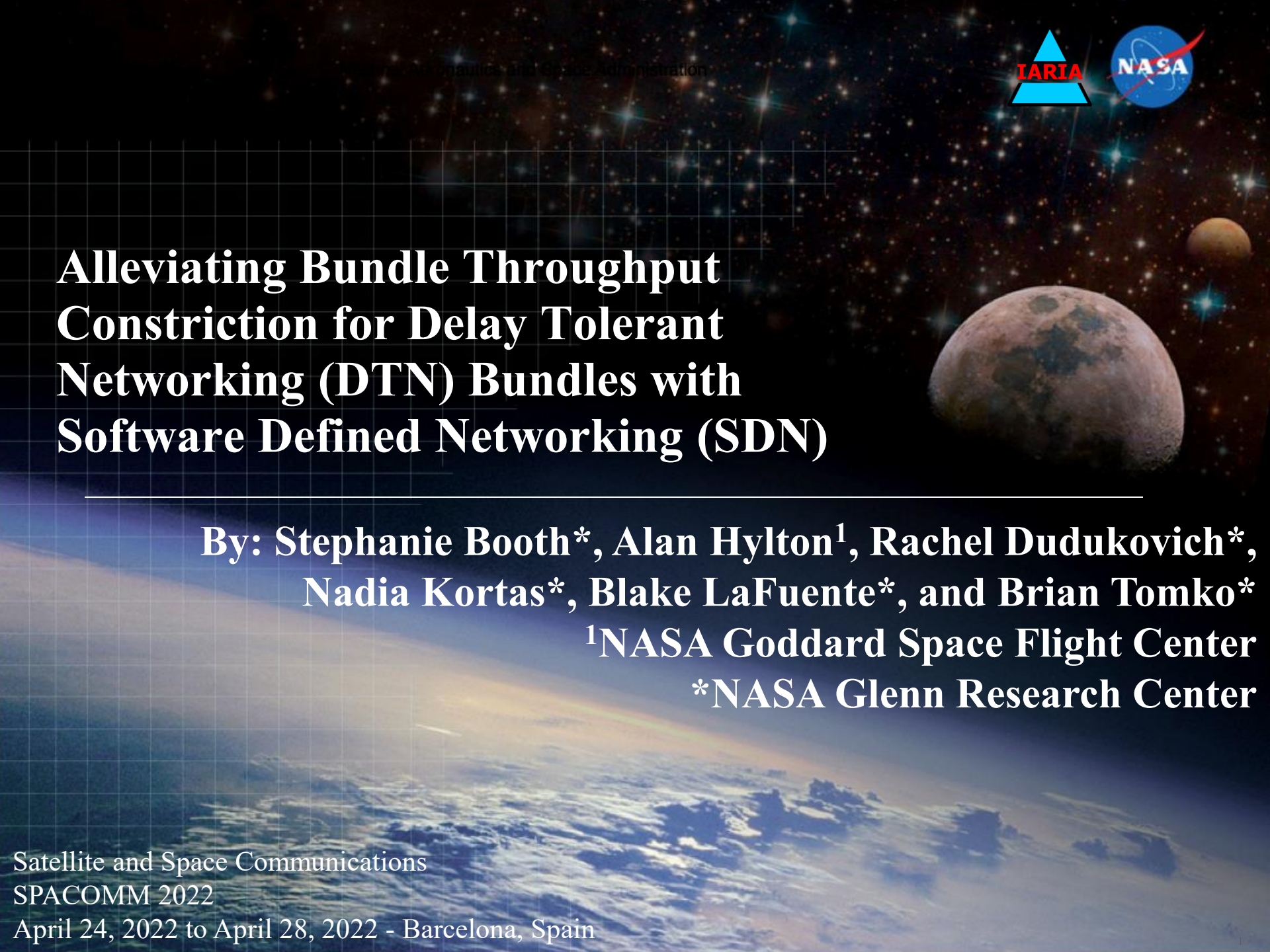


Alleviating Bundle Throughput Constriction for Delay Tolerant Networking (DTN) Bundles with Software Defined Networking (SDN)

By: Stephanie Booth*, Alan Hylton[1], Rachel Dudukovich*, Nadia Kortas*, Blake LaFuente*, and Brian Tomko*

[1]NASA Goddard Space Flight Center

*NASA Glenn Research Center

Satellite and Space Communications
SPACOMM 2022
April 24, 2022 to April 28, 2022 - Barcelona, Spain

Stephanie Booth

stephanie.l.booth@nasa.gov

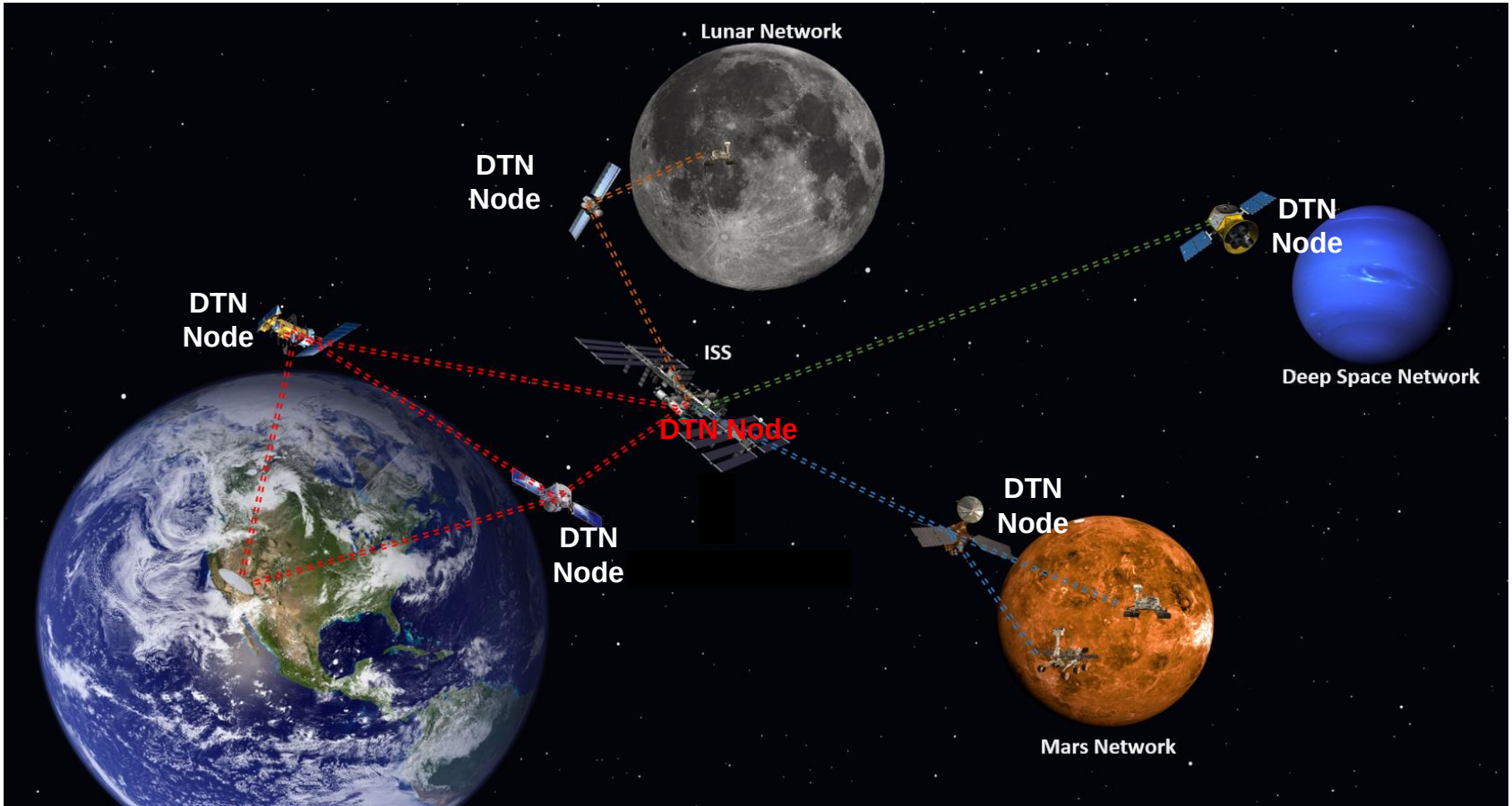
Stephanie Booth received her Bachelor's and Master's degree in Electrical Engineering from the University of Toledo located in Toledo, Ohio USA. She is currently a systems integrator at NASA Glenn Research Center in Cleveland, Ohio USA in the Secure Networks, System Integration and Test (LCN) branch

Her work and interest lies in software defined networking, delay tolerant networking, and testing high temperature SiC electronics. In addition, she has had previous work involving link analyses and free-space optics.

Table Of Contents

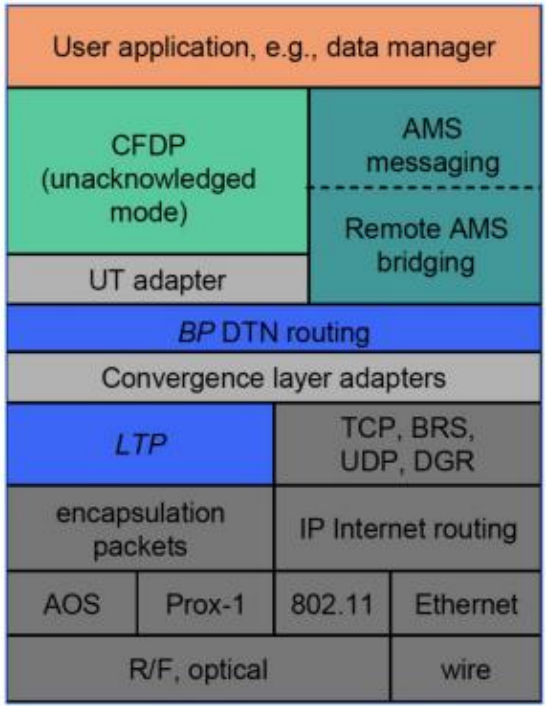
Problem Statement

- **Networks are expanding and becoming more complex. Therefore, they are also increasing bundle packet throughput within the system.**
- ***IF*** **DTN nodes are constricted in packet throughput, how would the system act? Would it just drop the packet without a flag? What happens** ***IF*** **all DTN node's send at their limits to the ISS DTN node?**



DTN Implementation: ION

- **Delay Tolerant Networking is the answer to space's long latencies and intermittent connections.**
  - A bundle can be fragmented into smaller bundles
  - Bundles are made up of blocks: Primary, Extension, Payload
- **Interplanetary Overlay Network (ION) will be the Delay Tolerant Networking (DTN) implementation used and compared against in this presentation.**
- **ION use Bundle Protocol (BP) version 6 (BPv6) but BPv7 is underway.**



DTN Protocol Stack

ION's Design Overview

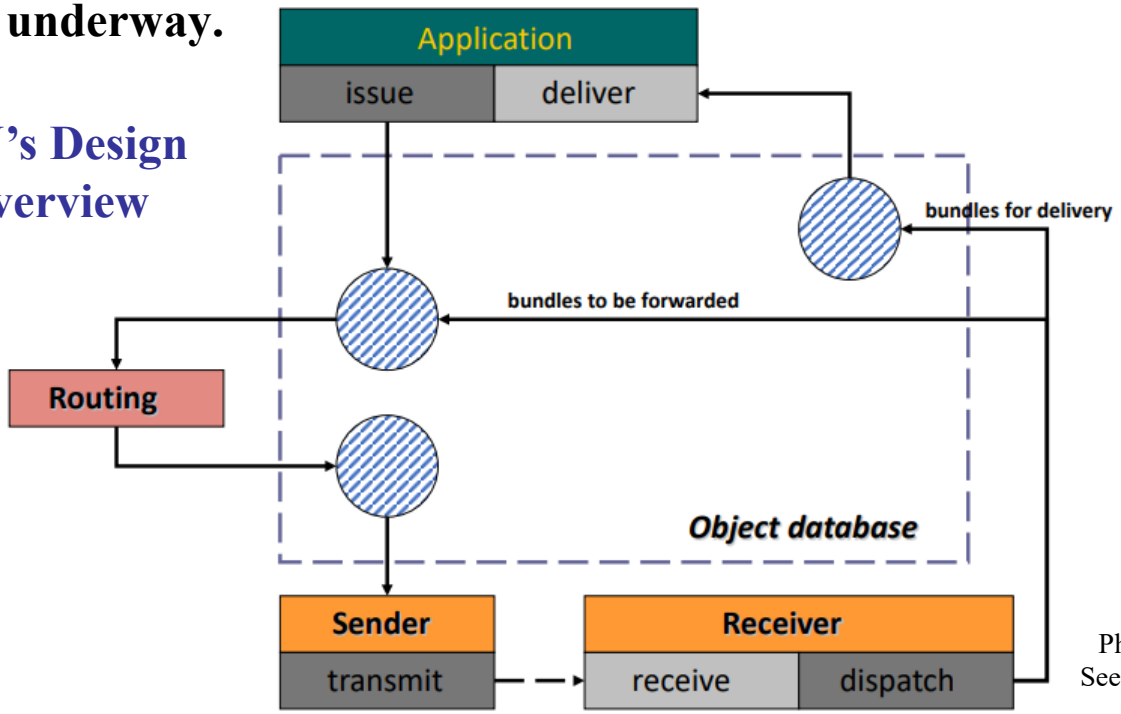


Photo Credit:
See Reference 1

Software Defined Networking (SDN) Switch

- **An Aurora 710 networking switch by Netberg was used.**
- **This switch contains**
  - 32 x 100 GbE four-lane Quad Small Form-factor Pluggable (QSFP) interfaces
  - Intel Tofino switching Integrated Circuit (IC)
- **This switch is Programming Protocol-Independent Packet Processors (P4) capable.**

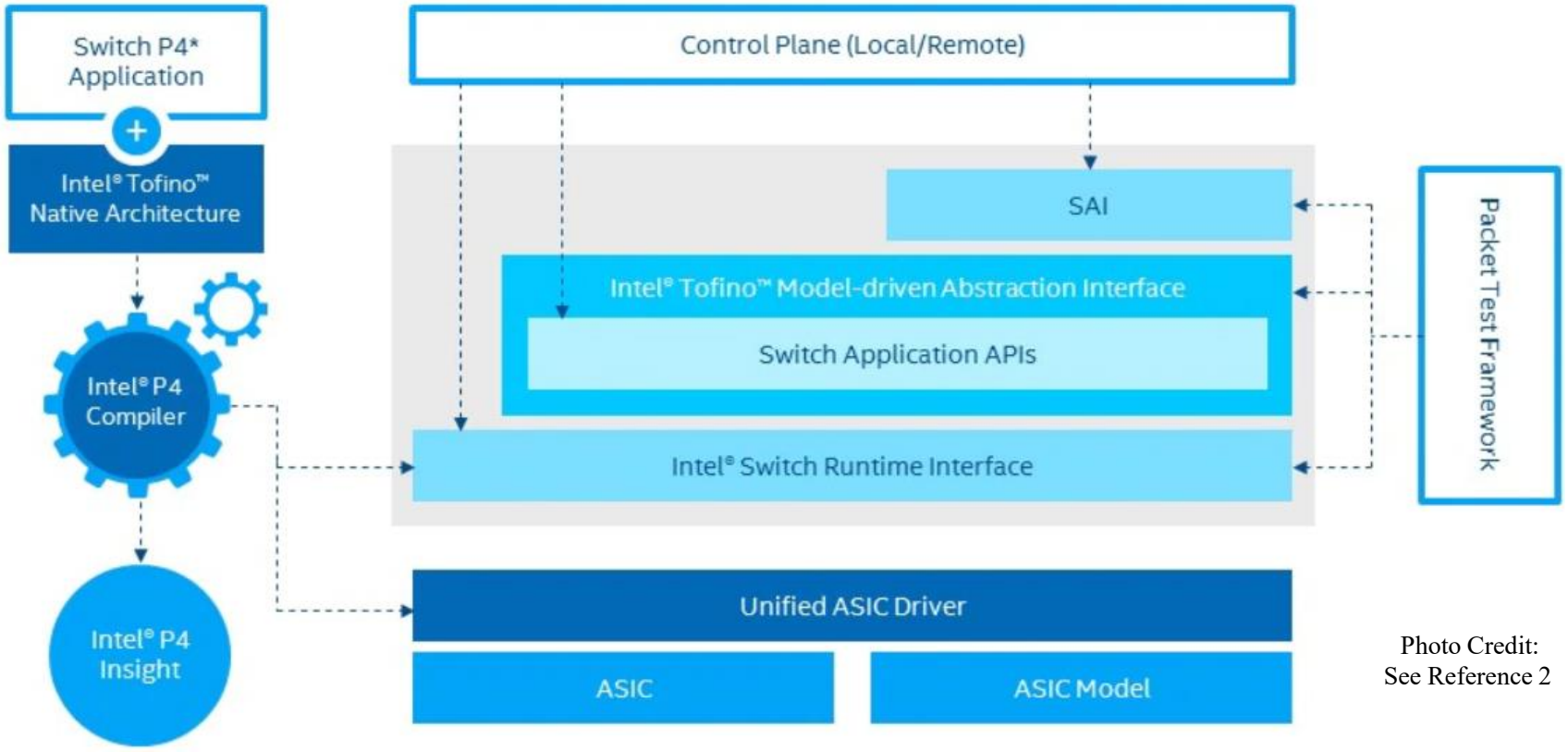
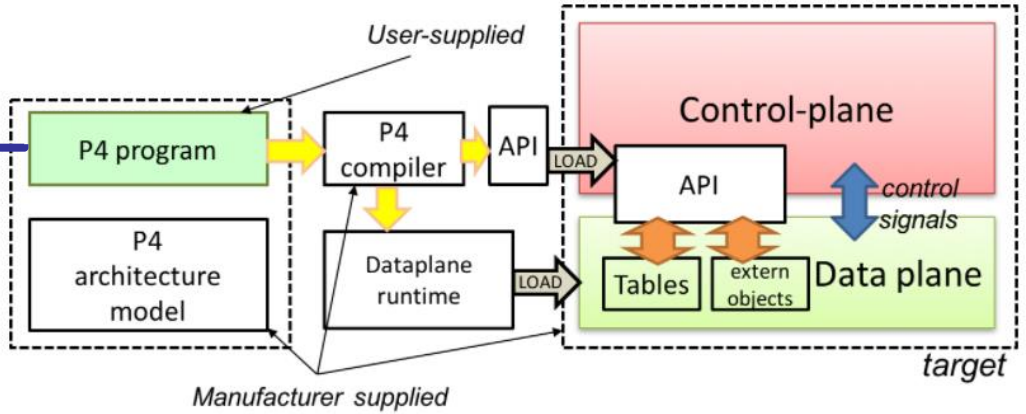


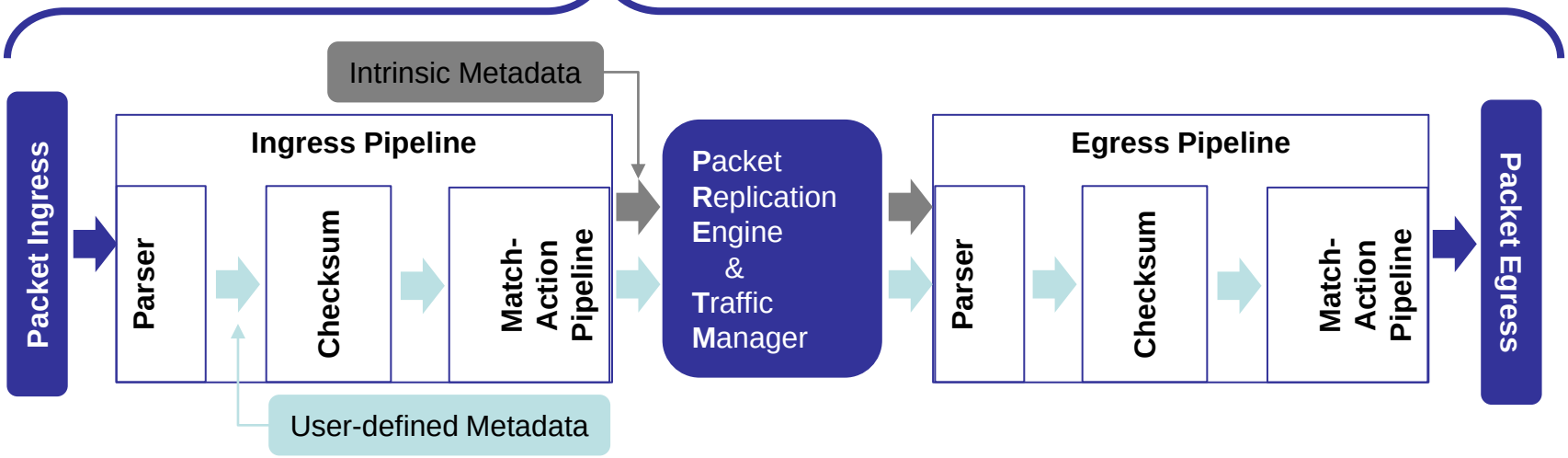
Photo Credit:
See Reference 2

P4 – Programming Protocol-independent Packet Processors

- **P4 allows implementing specific behavior in a network within minutes.**
- **P4 enables control over the programmable data plane.**
- **P4 can be used for a network to handle one or many, current and future protocols.**
- **Two major language reworks ie $P4_{14}$ and $P4_{16}$**

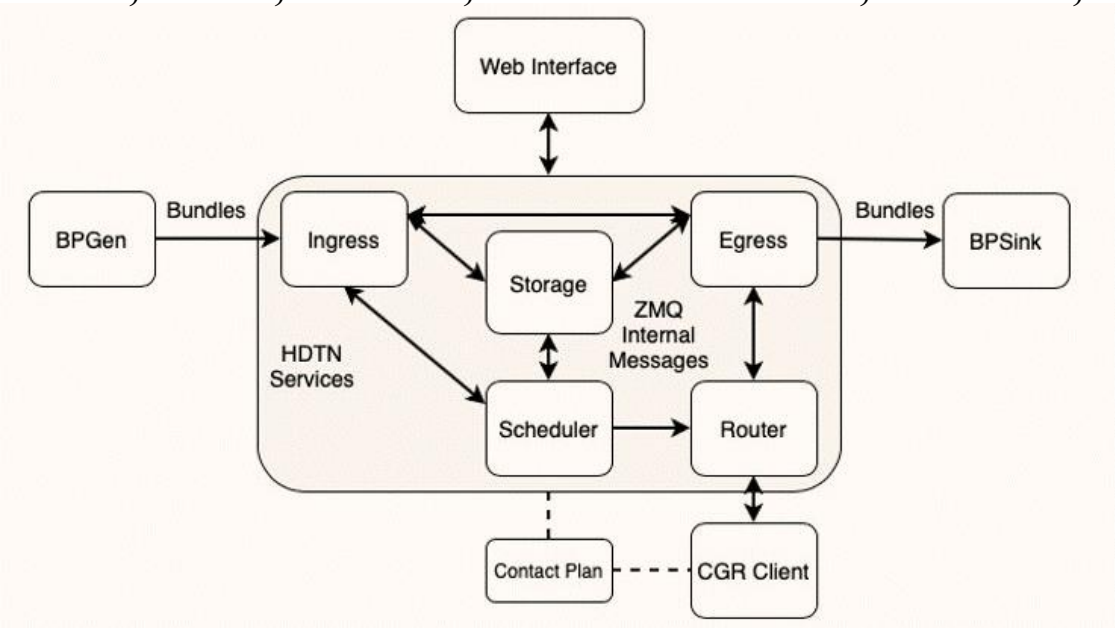


↑ Photo Credit: See Reference 3



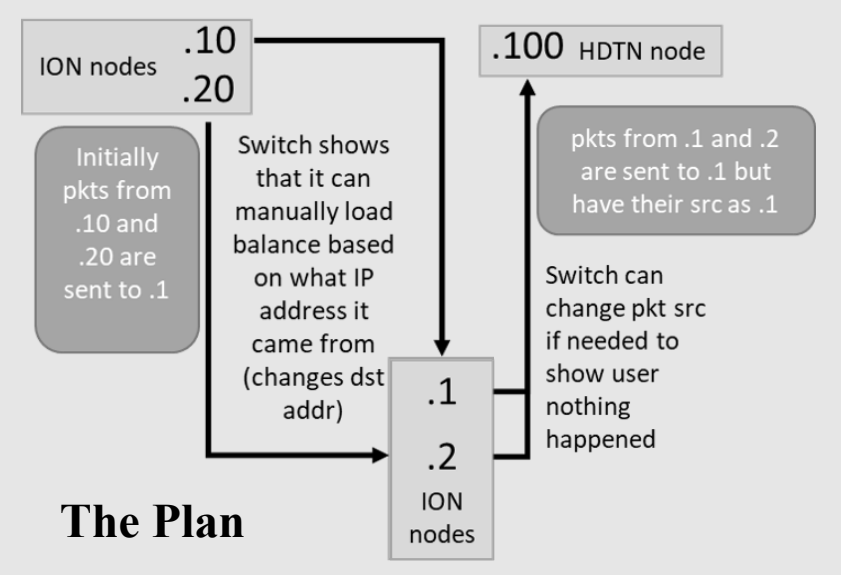
HDTN – High-rate DTN

- **HDTN was initially developed to support optical communication data rates exceeding 1 Gbps.**
- **Software bottlenecks caused by shared memory and locking mechanisms have been removed to allow for high speed, asynchronous message processing.**
- **Based on message bus architecture using ZeroMQ.**
- **Released under NASA Open Source Agreement:**
  - https://github.com/nasa/HDTN
- **Supports: BP v6, BP v7, UDPCL, TCPCL v3 and v4, STCPCL, and LTPCL.**



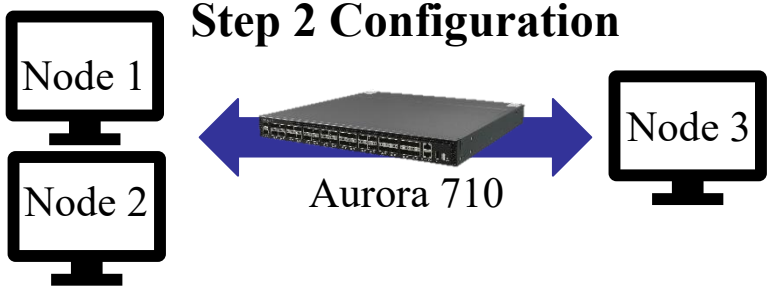
Initial Setup

- **The overarching goal is to follow The Plan figured below. First stage will be Step 1 configuration between the two ION nodes of the plan. Second stage follows Step 2 configuration for two ION nodes to one ION node or to one HDTN node**
- **The Aurora 710 networking switch was provided blank, meaning, without understanding any networking protocol. The steps taken were:**
  - Code in P4 to understand packet switching on layer 3/IP logical addressing
  - Then take layer 4/protocol into account, ie, UDP
  - Since packet altering happens if UDP packet is on a certain port, conditioning code was written



Step 1 Configuration

Step 2 Configuration



Initial Setup (continued)

- **Aurora platform contained compiled code, hardware ports were enabled and up, and tables were loaded**
- **Once operational, preliminary tests of the P4 software was conducted using a packet creation software called Scapy before adding in the DTN implementation. Following the NAT rules Table, the results showed that:**
  - Code changes packet if UDP bundle on port 4556
  - Source IP address changes follows Table I properly
  - Destination IP address changes follows Table I properly
  - TCP/IP and all other UDP packets are switched as normal

Table 1

Original		Modification
IP Source	IP Destination	
10.10.10.[10-19, 100]	10.10.10.1	No change
10.10.10.[20-29, 100]	10.10.10.1	Destination → 10.10.10.2
10.10.10.[30-39, 100]	10.10.10.1	Destination → 10.10.10.3
10.10.10.1	10.10.10.[10-19, 100]	No change
10.10.10.2	10.10.10.[20-29, 100]	Source → 10.10.10.1
10.10.10.3	10.10.10.[30-39, 100]	Source → 10.10.10.1

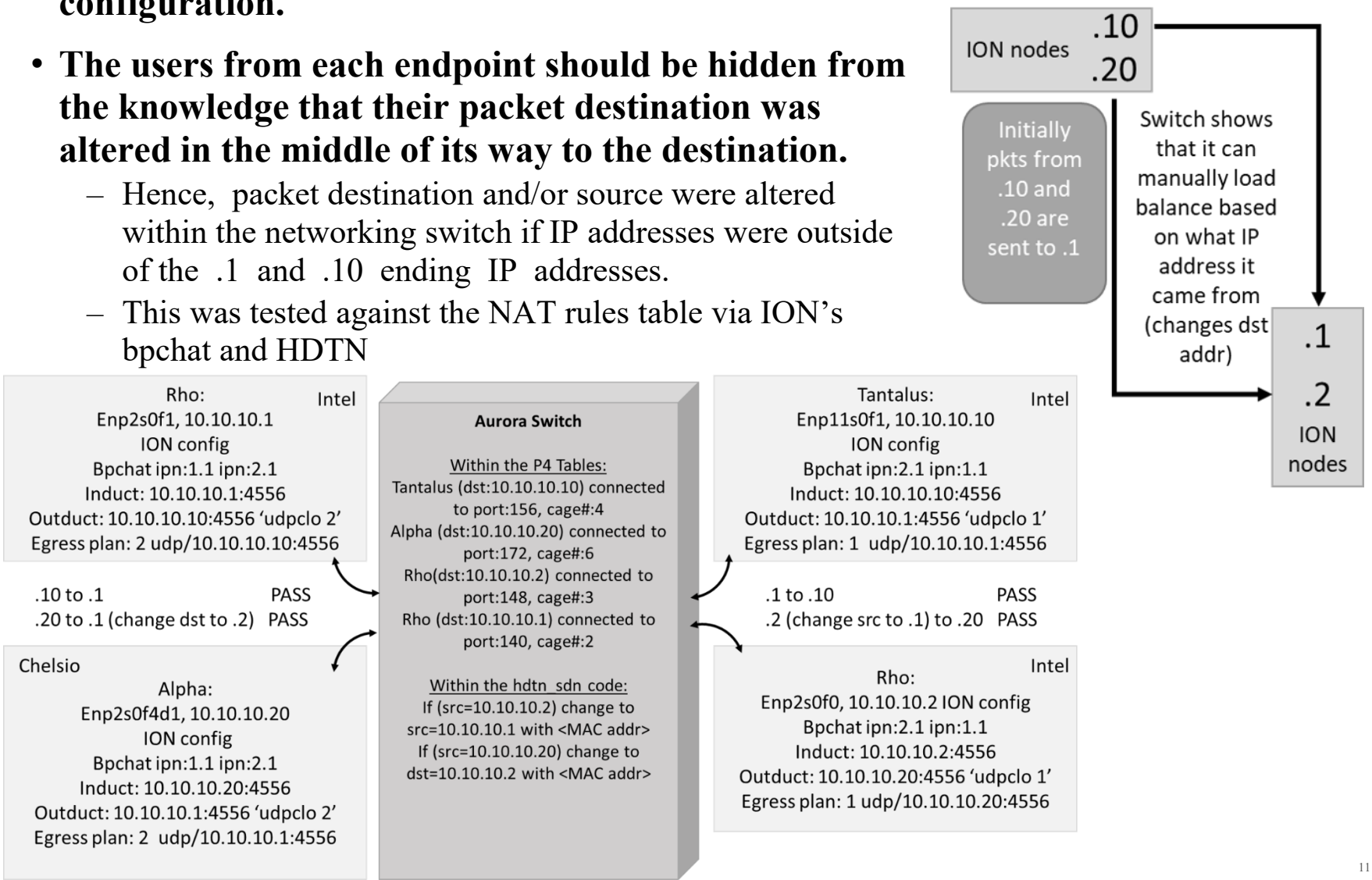
```
bf-sde> pm show
-----+----+---+----+-------+----+---+---+---+--------+----------------+----------------+-
PORT |MAC |D_P|P/PT|SPEED  |FEC |RDY|ADM|OPR|LPBK    |FRAMES RX       |FRAMES TX       |E
-----+----+---+----+-------+----+---+---+---+--------+----------------+----------------+-
1/0  |23/0|132|2/ 4|40G    |NONE|YES|ENB|UP |  NONE  |             213|               0|
2/0  |22/0|140|2/12|40G    |NONE|YES|ENB|UP |  NONE  |             219|               0|
3/0  |21/0|148|2/20|40G    |NONE|YES|ENB|UP |  NONE  |             220|               0|
4/0  |20/0|156|2/28|40G    |NONE|YES|ENB|UP |  NONE  |             216|               0|
5/0  |19/0|164|2/36|40G    |NONE|YES|ENB|UP |  NONE  |               3|               0|
6/0  |18/0|172|2/44|40G    |NONE|YES|ENB|UP |  NONE  |             218|               0|
```

```
Table ipv4_host:
----- ipv4_host Dump Start -----
Default Entry:
Entry data (action : NoAction):

pipe.Ingress.ipv4_host entries fo
hdr.ipv4.dst_addr    port
-------------------  ------
0x0A0A0A0A           0x9C
0x0A0A0A14           0xAC
0x0A0A0A02           0xA4
0x0A0A0A64           0x84
0x0A0A0A01           0x8C
```

Node to Node Configuration

- **Each link was tested individually following Step 1 configuration.**
- **The users from each endpoint should be hidden from the knowledge that their packet destination was altered in the middle of its way to the destination.**
  - Hence, packet destination and/or source were altered within the networking switch if IP addresses were outside of the .1 and .10 ending IP addresses.
  - This was tested against the NAT rules table via ION's bpchat and HDTN



ION to ION bpchat Results

SDN Manually Load Balancing ION to ION

Tantalus [.10] → Rho [.1]

Wireshark capture on Tantalus (shows how packet left):

```
Frame 1: 99 bytes on wire (792 bits), 99 bytes captured (792 bits) on interface enp11s0f1, id 0
Ethernet II, Src: [redacted], Dst: [redacted]
Internet Protocol Version 4, Src: 10.10.10.10, Dst: 10.10.10.1
User Datagram Protocol, Src Port: 47258, Dst Port: 4556
Bundle Protocol
```

Wireshark capture on Rho (shows no change):

```
Frame 1: 99 bytes on wire (792 bits), 99 bytes captured (792 bits) on interface 0
Ethernet II, Src: [redacted], Dst: [redacted]
Internet Protocol Version 4, Src: 10.10.10.10, Dst: 10.10.10.1
User Datagram Protocol, Src Port: 47258, Dst Port: 4556
Bundle Protocol
```

Rho [.1] → Tantalus [.10]

Wireshark capture on Rho (shows how packet left):

```
Frame 2: 99 bytes on wire (792 bits), 99 bytes captured (792 bit
Ethernet II, Src: [redacted], Dst: [redacted]
Internet Protocol Version 4, Src: 10.10.10.1, Dst: 10.10.10.10
User Datagram Protocol, Src Port: 54574, Dst Port: 4556
Bundle Protocol
```

Wireshark capture on Tantalus (shows no change):

```
Frame 2: 99 bytes on wire (792 bits), 99 bytes captured (792 bit
Ethernet II, Src: [redacted], Dst: [redacted]
Internet Protocol Version 4, Src: 10.10.10.1, Dst: 10.10.10.10
User Datagram Protocol, Src Port: 54574, Dst Port: 4556
Bundle Protocol
```

ION bpchat working!

ION bpchat on Rho

```
slbooth@rho:~/ion-3.7.0/tests/bpchat$ sudo bpchat
from rho to tant
from tant to rho
```

ION bpchat on Tantalus

```
slbooth@tantalus:~/ion-3.7.0/tests/bpchat$ sudo bp
from rho to tant
from tant to rho
```

Alpha [.20] → Rho [.1]

Will see it on Rho [.2] (IP dst change!) only if UDP dst port 4556

Wireshark capture on Alpha (shows how packet left):

```
Frame 2: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface 0
Ethernet II, Src: [redacted], Dst: [redacted]
Internet Protocol Version 4, Src: 10.10.10.20, Dst: 10.10.10.1
User Datagram Protocol, Src Port: 57932, Dst Port: 4556
Bundle Protocol
```

Wireshark capture on Rho (shows the change):

```
Frame 2: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface 0
Ethernet II, Src: [redacted] Dst: [redacted]
Internet Protocol Version 4, Src: 10.10.10.20, Dst: 10.10.10.2
User Datagram Protocol, Src Port: 57932, Dst Port: 4556
Bundle Protocol
```

Rho [.2] → Alpha [.20]

Will see if on Alpha [.2] as source Rho [.1] only if UDP dst port 4556

Wireshark capture on Rho (shows how packet left):

```
Frame 1: 91 bytes on wire (728 bits), 91 bytes captured (728 bits) on interface 0
Ethernet II, Src: [redacted] Dst: [redacted]
Internet Protocol Version 4, Src: 10.10.10.2, Dst: 10.10.10.20
User Datagram Protocol, Src Port: 44734, Dst Port: 4556
Bundle Protocol
```

Wireshark capture on Alpha (shows the change):

```
Frame 2: 91 bytes on wire (728 bits), 91 bytes captured (728 bits) on interface 0
Ethernet II, Src: [redacted] Dst: [redacted]
Internet Protocol Version 4, Src: 10.10.10.1, Dst: 10.10.10.20
User Datagram Protocol, Src Port: 44734, Dst Port: 4556
Bundle Protocol
```

ION bpchat working!

```
slbooth@rho:~/ion-3.7.0/tests/bpchat$ sudo bpchat ipn:2.1 ipn:1.1
from alpha to rho
from rho to alpha
```

```
slbooth@alpha:~/ion-3.7.0/tests/bpchat$ sudo bpchat ipn:1.1 ipn:2.1
from alpha to rho
from rho to alpha
```

ION to ION bpchat Results

Tantalus [.10] → Rho [.1

Wireshark capture on

```
Frame 1: 99 bytes on wi
Ethernet II, Src:
Internet Protocol Versi
User Datagram Protocol,
Bundle Protocol
```

Wireshark capture on

```
Frame 1: 99 bytes on wi
Ethernet II, Src:
Internet Protocol Versi
User Datagram Protocol,
Bundle Protocol
```

Rho [.1] → Tantalus [.10

Wireshark capture on

```
Frame 2: 99 bytes on wi
Ethernet II, Src:
Internet Protocol Versi
User Datagram Protocol,
Bundle Protocol
```

Wireshark capture on

```
Frame 2: 99 bytes on wi
Ethernet II, Src:
Internet Protocol Versi
User Datagram Protocol,
Bundle Protocol
```

ION bpchat working!

ION bpchat on Rho

```
slbooth@rho:~/ion-
from rho to tant
from tant to rho
```

ION bpchat on Tantalu

```
slbooth@tantalus:~
from rho to tant
from tant to rho
```

Alpha [.20] → Rho [.1]

Will see it on Rho [.2] (IP dst change!) only if UDP dst port 4556

Wireshark capture on Alpha (shows how packet left):

```
Frame 2: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface 0
Ethernet II, Src:                                    , Dst:
Internet Protocol Version 4, Src: 10.10.10.20, Dst: 10.10.10.1
User Datagram Protocol, Src Port: 57932, Dst Port: 4556
Bundle Protocol
```

Wireshark capture on Rho (shows the change):

```
Frame 2: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface 0
Ethernet II, Src:                                    Dst:
Internet Protocol Version 4, Src: 10.10.10.20, Dst: 10.10.10.2
User Datagram Protocol, Src Port: 57932, Dst Port: 4556
Bundle Protocol
```

Rho [.2] → Alpha [.20]

Will see if on Alpha [.2] as source Rho [.1] only if UDP dst port 4556

Wireshark capture on Rho (shows how packet left):

```
Frame 1: 91 bytes on wire (728 bits), 91 bytes captured (728 bits) on interface 0
Ethernet II, Src:                                    Dst:
Internet Protocol Version 4, Src: 10.10.10.2, Dst: 10.10.10.20
User Datagram Protocol, Src Port: 44734, Dst Port: 4556
Bundle Protocol
```

Wireshark capture on Alpha (shows the change):

```
Frame 2: 91 bytes on wire (728 bits), 91 bytes captured (728 bits) on interface 0
Ethernet II, Src:                                    Dst:
Internet Protocol Version 4, Src: 10.10.10.1, Dst: 10.10.10.20
User Datagram Protocol, Src Port: 44734, Dst Port: 4556
Bundle Protocol
```

ION bpchat working!

```
slbooth@rho:~/ion-3.7.0/tests/bpchat$ sudo bpchat ipn:2.1 ipn:1.1
from alpha to rho
from rho to alpha
```

```
slbooth@alpha:~/ion-3.7.0/tests/bpchat$ sudo bpchat ipn:1.1 ipn:2.1
from alpha to rho
from rho to alpha
```

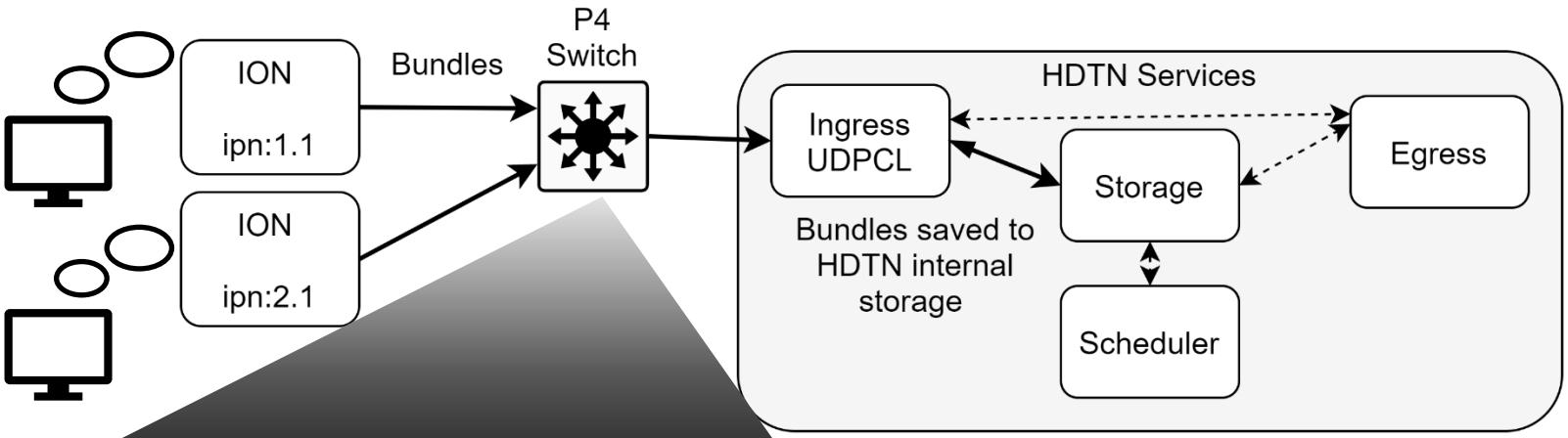
Node to Node Benchmarking Results

- **Once the networking switch was checked out, benchmarking results were collected to see bottleneck limits. ION to ION node results are shown for comparison with a HDTN test.**
- **Results only PASS when all bundles are received.**

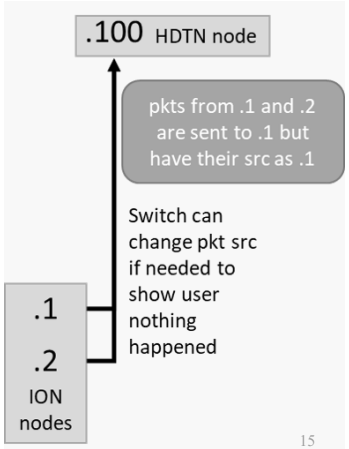
Tx Computer	Rx Computer	Status
Rho	**Eta**	
Stopping bpdriver. Total bundles: 100 Time (seconds): 0.535 Total bytes: 100000 Throughput (Mbps): 1.495	Stopping bpcounter; bundles received: 100 Time (seconds): 9.486 Total bytes: 100000 Throughput (Mbps): 0.084	PASS
Stopping bpdriver. Total bundles: 1000 Time (seconds): 5.338 Total bytes: 1000000 Throughput (Mbps): 1.499	Stopping bpcounter; bundles received: 438 Time (seconds): 20.130 Total bytes: 438000 Throughput (Mbps): 0.174	FAIL
Omicron	**Eta**	
Stopping bpdriver. Total bundles: 100 Time (seconds): 0.558 Total bytes: 100000 Throughput (Mbps): 1.433	Stopping bpcounter; bundles received: 100 Time (seconds): 12.765 Total bytes: 100000 Throughput (Mbps): 0.063	PASS
Stopping bpdriver. Total bundles: 1000 Time (seconds): 2.656 Total bytes: 1000000 Throughput (Mbps): 3.012	Stopping bpcounter; bundles received: 810 Time (seconds): 23.921 Total bytes: 810000 Throughput (Mbps): 0.271	FAIL

HDTN System Setup

- **The HDTN node was configured first to non-volatile storage.**
- **The HDTN node only listened to an IP destination of 10.10.10.100.**
- **Each packet received is to come from and IP address ending in .1.**



Original		Modification
IP Source	IP Destination	
10.10.10.[10-19, 100]	10.10.10.1	No change
10.10.10.[20-29, 100]	10.10.10.1	Destination → 10.10.10.2
10.10.10.[30-39, 100]	10.10.10.1	Destination → 10.10.10.3
10.10.10.1	10.10.10.[10-19, 100]	No change
10.10.10.2	10.10.10.[20-29, 100]	Source → 10.10.10.1
10.10.10.3	10.10.10.[30-39, 100]	Source → 10.10.10.1



2 Nodes to 1 HDTN Node

- **The results brought forth all successes for both 2 nodes to 1 node**

Tx Computer	Rx Computer	Status
Rho -hdtn_host1.rc **Omicron -hdtn_benchmark1.rc**	**Eta ./runscript.sh** **Only to Storage**	
@rho: bpdriver 100 ipn:1.1 ipn:2.1 -1000 t30 Stopping bpdriver. Total bundles: 100 Time (seconds): 0.580 Total bytes: 100000 Throughput (Mbps): 1.378 **@omicron:** bpdriver 200 ipn:1.1 ipn:2.1 -1000 t30 Stopping bpdriver. Total bundles: 200 Time (seconds): 0.782 Total bytes: 200000 Throughput (Mbps): 2.046	**@eta:** hdtn m_bundleCountStorage: 302 m_bundleCountEgress: 0 m_bundleCount: 302 m_bundleData: 312083	PASS
@rho: bpdriver 1000 ipn:1.1 ipn:2.1 -1000 t30 Stopping bpdriver. Total bundles: 1000 Time (seconds): 2.565 Total bytes: 1000000 Throughput (Mbps): 3.119 **@omicron:**	**@eta:** hdtn m_bundleCountStorage: 2002 m_bundleCountEgress: 0 m_bundleCount: 2002	PASS

2 Nodes to 1 HDTN Node

- The

Tx Computer	Rx Computer	Status
Rho -hdtn_host1.rc **Omicron -hdtn_benchmark1.rc**	**Eta ./runscript.sh** **Only to Storage**	
@rho: bpdriver 100 ipn:1.1 ipn:2.1 -1000 t30 Stopping bpdriver. Total bundles: 100 Time (seconds): 0.580 Total bytes: 100000 Throughput (Mbps): 1.378 **@omicron:** bpdriver 200 ipn:1.1 ipn:2.1 -1000 t30 Stopping bpdriver. Total bundles: 200 Time (seconds): 0.782 Total bytes: 200000 Throughput (Mbps): 2.046	**@eta:** hdtn m_bundleCountStorage: 302 m_bundleCountEgress: 0 m_bundleCount: 302 m_bundleData: 312083	PASS
@rho: bpdriver 1000 ipn:1.1 ipn:2.1 -1000 t30 Stopping bpdriver. Total bundles: 1000 Time (seconds): 2.565 Total bytes: 1000000 Throughput (Mbps): 3.119 **@omicron:** bpdriver 1000 ipn:1.1 ipn:2.1 -1000 t30 Stopping bpdriver. Total bundles: 1000 Time (seconds): 3.109 Total bytes: 1000000 Throughput (Mbps): 2.573	**@eta:** hdtn m_bundleCountStorage: 2002 m_bundleCountEgress: 0 m_bundleCount: 2002 m_bundleData: 2083906	PASS

On-Going and Future Work

Conclusion

- **With increasing network throughput and capability needs, bottle-necks are important to avoid. Some options have been found to circumvent performance restriction of the network for DTN implementation nodes.**
  - Manually load balancing packets with a SDN switch and/or
  - Using an HDTN receiver node
- **Current work involves**
  - SDN development to provide solutions to 100Gbps data rates for HDTN.
  - Teaching the Aurora network switch platform how to automatically load balance traffic between ports without the need of the IP changes hard-coded into the P4 code.
- **Future work will incorporate the bundle egress to neighboring nodes and finding limitations to HDTN's capabilities.**

Any Questions?

References

1. https://www.nasa.gov/sites/default/files/atoms/files/1.2_lecture_-_intro_to_ion_implentation_of_the_dtn_architecture.pdf
2. https://www.intel.com/content/www/us/en/products/network-io/programmable-ethernet-switch/p4-suite/p4-studio.html
3. https://p4.org/p4-spec/docs/P4-16-v1.2.0.html

Extra

Helpful Websites

- ION
  - https://www.nasa.gov/directorates/heo/scan/engineering/technology/disruption_tolerant_networking_software_options_ion
- P4
  - www.P4.org
- HDTN
  - https://github.com/nasa/HDTN
- Intel Tofino / SDN
  - https://opennetworking.org/wp-content/uploads/2021/05/2021-P4-WS-Vladimir-Gurevich-Slides.pdf