Network tuning - a practical guide

Ramiro Voicu

(Caltech/USLHCNet)

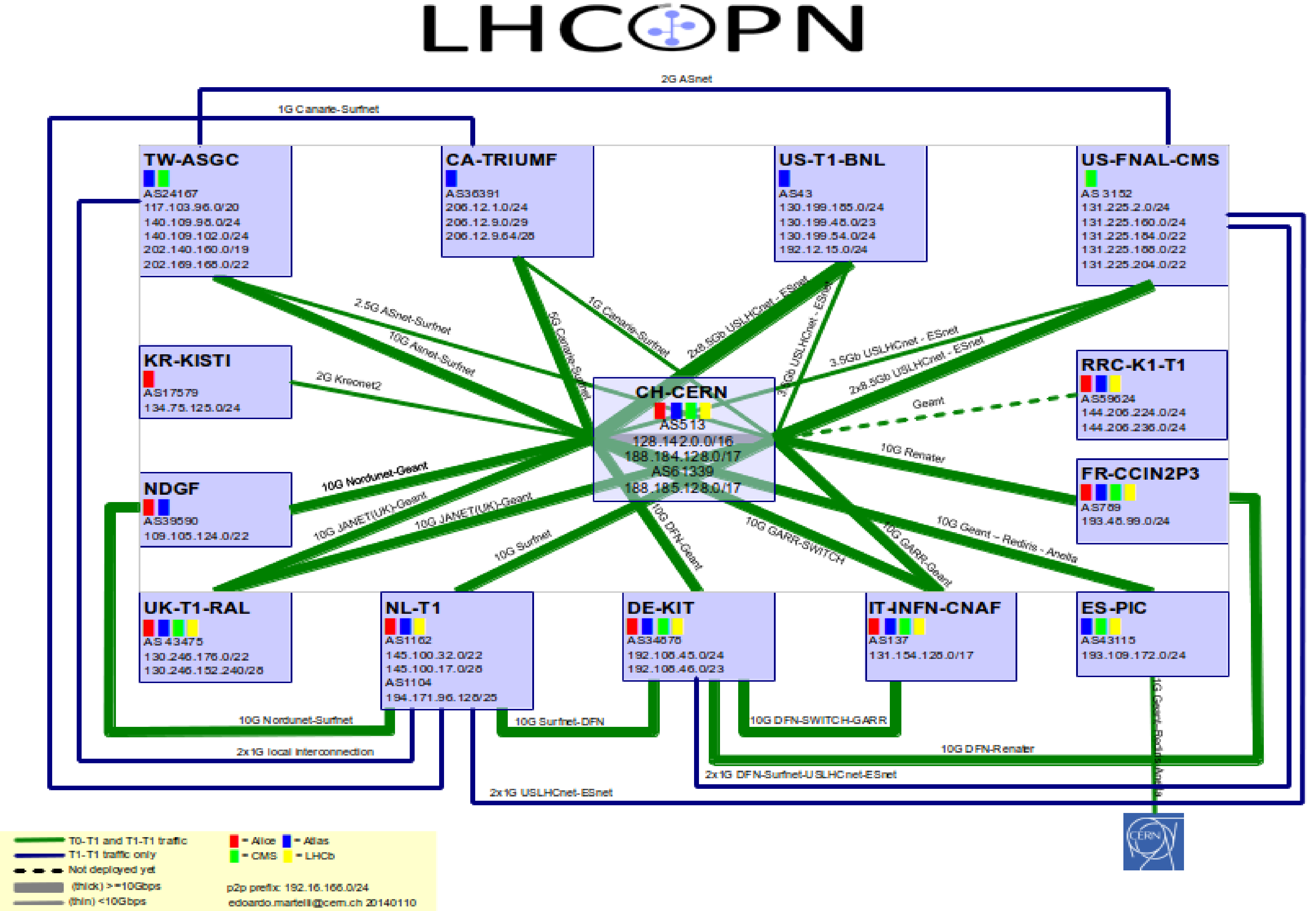
ALICE T1/T2 workshop

Tsukuba, March 7th, 2014

Outline

- US LHCNet
- TCP background
- TCP performance tuning
- NIC/Ethernet tuning
- Network tuning
- Debugging

LHCOPN



TW-ASGC
🟦🟩
AS24167
117.103.96.0/20
140.109.98.0/24
140.109.102.0/24
202.140.160.0/19
202.169.168.0/22

CA-TRIUMF
🟦
AS36391
206.12.1.0/24
206.12.9.0/29
206.12.9.64/28

US-T1-BNL
🟦
AS43
130.199.185.0/24
130.199.48.0/23
130.199.54.0/24
192.12.15.0/24

US-FNAL-CMS
🟩
AS 3152
131.225.2.0/24
131.225.160.0/24
131.225.184.0/22
131.225.188.0/22
131.225.204.0/22

KR-KISTI
🟥
AS17579
134.75.125.0/24

CH-CERN
🟥🟦🟩🟨
AS513
128.142.0.0/16
188.184.128.0/17
AS61339
188.185.128.0/17

RRC-K1-T1
🟥🟦🟨
AS59624
144.206.224.0/24
144.206.236.0/24

NDGF
🟥🟦
AS39590
109.105.124.0/22

FR-CCIN2P3
🟥🟦🟩🟨
AS789
193.48.99.0/24

UK-T1-RAL
🟥🟦🟩🟨
AS 43475
130.246.176.0/22
130.246.152.240/28

NL-T1
🟥🟦🟨
AS1162
145.100.32.0/22
145.100.17.0/28
AS1104
194.171.96.128/25

DE-KIT
🟥🟦🟩🟨
AS34878
192.108.45.0/24
192.108.46.0/23

IT-INFN-CNAF
🟥🟦🟩🟨
AS137
131.154.128.0/17

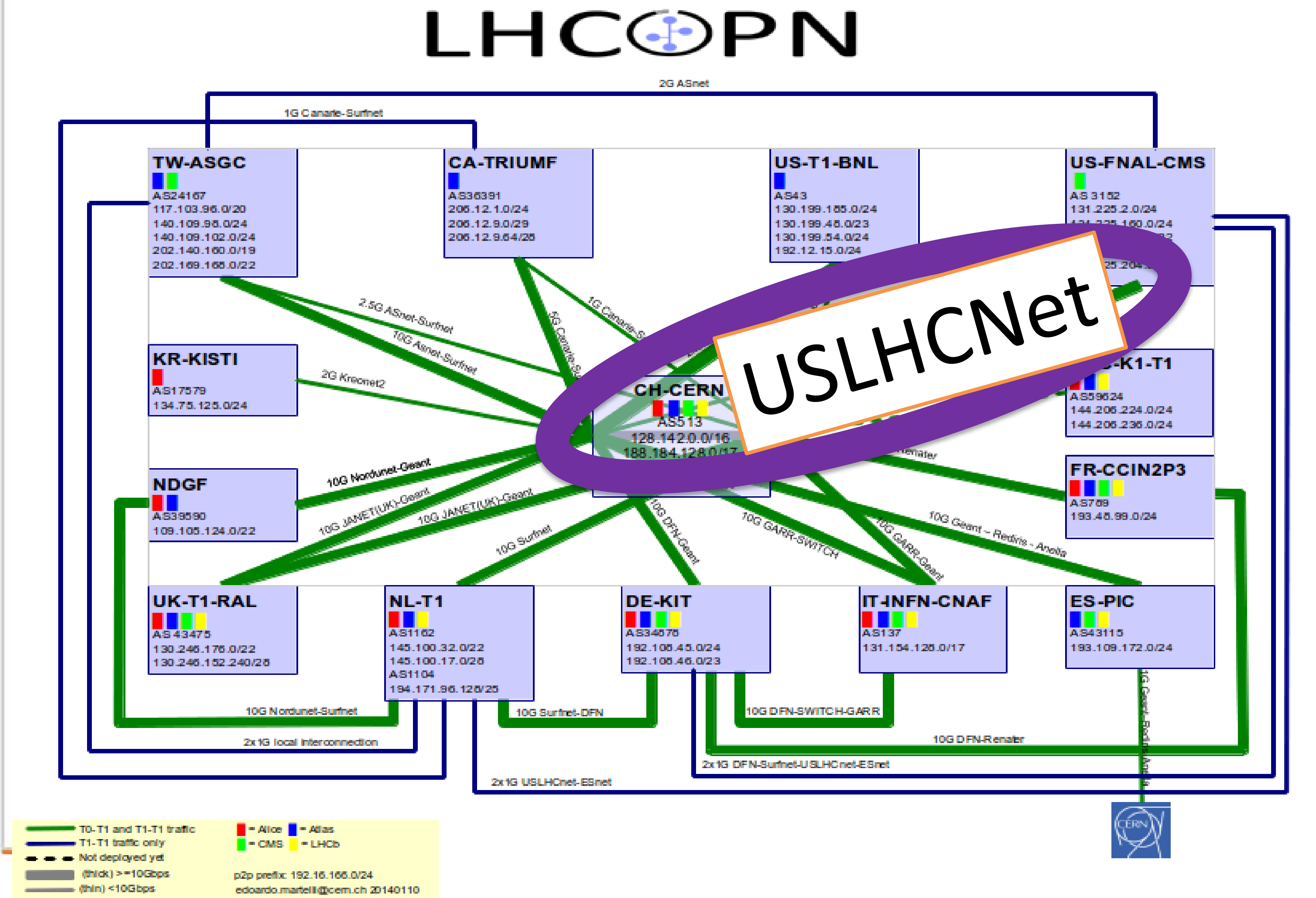
ES-PIC
🟦🟩🟨
AS43115
193.109.172.0/24

T0-T1 and T1-T1 traffic
T1-T1 traffic only
Not deployed yet
(thick) >=10Gbps
(thin) <10Gbps

🟥 = Alice 🟦 = Atlas
🟩 = CMS 🟨 = LHCb

p2p prefix: 192.16.166.0/24
edoardo.martelli@cern.ch 20140110

US LHCNet


LHCOPN
2G ASnet
1G Canarie-Surfnet
TW-ASGC
AS24167
117.103.96.0/20
140.109.98.0/24
140.109.102.0/24
202.140.160.0/19
202.169.168.0/22
CA-TRIUMF
AS36391
206.12.1.0/24
206.12.9.0/29
206.12.9.64/26
US-T1-BNL
AS43
130.199.185.0/24
130.199.48.0/23
130.199.54.0/24
192.12.15.0/24
US-FNAL-CMS
AS 3152
131.225.2.0/24
USLHCNet
KR-KISTI
AS17579
134.75.125.0/24
2.5G ASnet-Surfnet
10G Asnet-Surfnet
5G Canarie-Surfnet
1G Canarie-Surfnet
2G Kreonet2
CH-CERN
AS513
128.142.0.0/16
188.184.128.0/17
-K1-T1
AS59624
144.206.224.0/24
144.206.236.0/24
Renater
NDGF
AS39590
109.105.124.0/22
10G Nordunet-Geant
10G JANET(UK)-Geant
10G JANET(UK)-Geant
10G Surfnet
10G DFN-Geant
10G GARR-SWITCH
10G GARR-Geant
10G Geant - Rediris - Anella
FR-CCIN2P3
AS789
193.48.99.0/24
UK-T1-RAL
AS 43475
130.246.176.0/22
130.246.152.240/28
NL-T1
AS1162
145.100.32.0/22
145.100.17.0/28
AS1104
194.171.96.128/25
DE-KIT
AS34878
192.108.45.0/24
192.108.46.0/23
IT-INFN-CNAF
AS137
131.154.128.0/17
ES-PIC
AS43115
193.109.172.0/24
1G Geant-Rediris-Anella
10G Nordunet-Surfnet
10G Surfnet-DFN
10G DFN-SWITCH-GARR
10G DFN-Renater
2x1G local interconnection
2x1G DFN-Surfnet-USLHCnet-ESnet
2x1G USLHCnet-ESnet
CERN
T0-T1 and T1-T1 trafic
T1-T1 traffic only
Not deployed yet
(thick) >=10Gbps
(thin) <10Gbps
= Alice
= Atlas
= CMS
= LHCb
p2p prefix: 192.16.166.0/24
edoardo.martelli@cern.ch 20140110

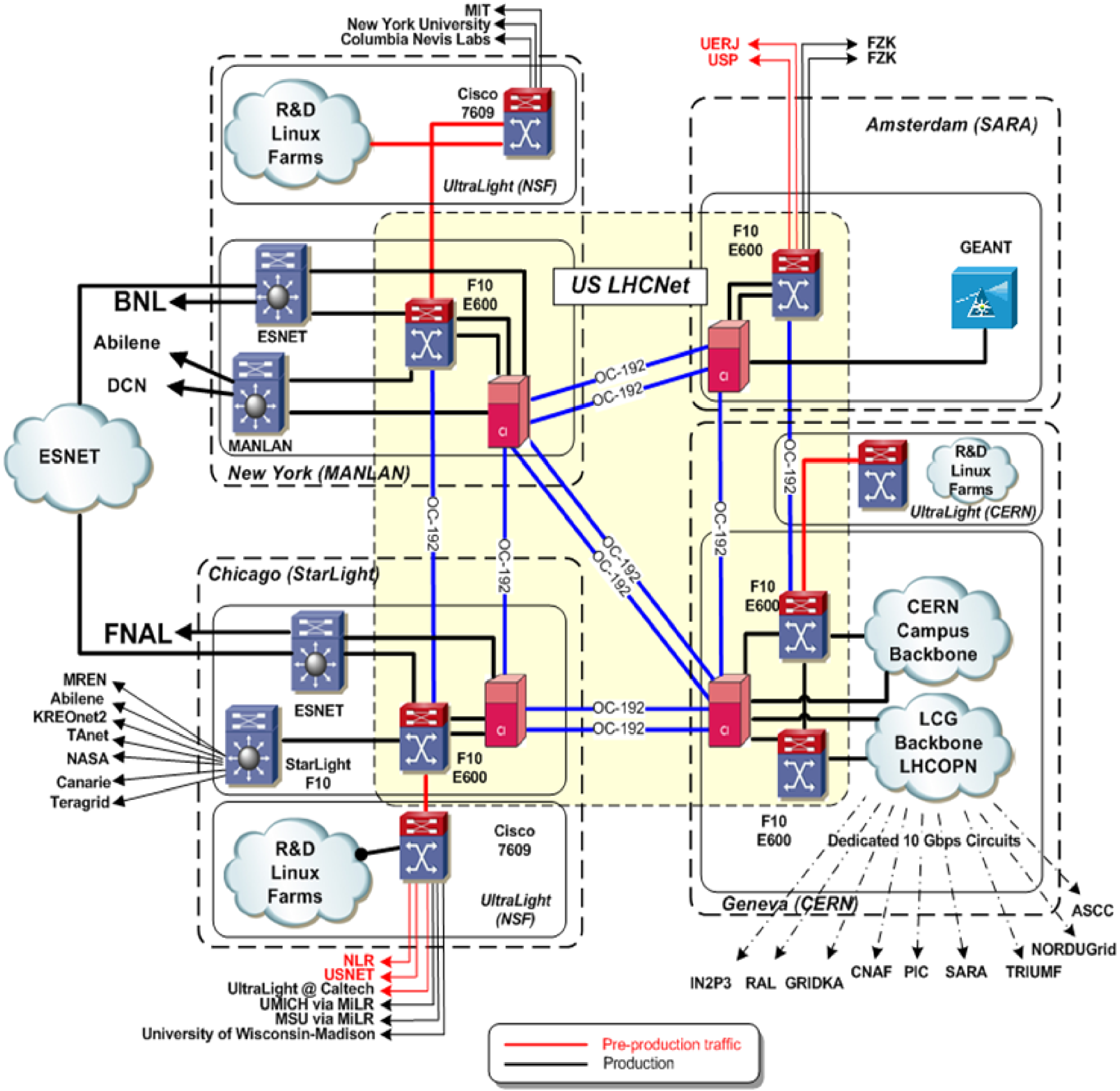
US LHCNet

- LHCOPN from CERN to US
  - FNAL
  - BNL

- 6 x 10G TA circuits

- 4 PoPs
  - Geneva
  - Amsterdam
  - Chicago
  - New York

- The core is based on Ciena CD/CI (Layer 1.5)



TCP BACKGROUND

Internet Protocol (IP) Suite

- Designed in the 1973 by Vinton Cerf and Robert E. Kahn
  - Declared in 1982 by DoD as the standard for military computer networking
- Four layer model (vs 7 layer ISO/OSI)
  - Application (xrootd, ftp, smtp, http, etc)
  - Transport
    - TCP - Transmission Control Protocol (reliable, byte stream connection-oriented)
    - UDP - User Datagram Protocol (connection-less)
    - SCTP – Stream Control Transmission Protocol – reliable, message stream connection oriented, connection multiplexing)
  - Internet (routing and addressing IPv4, IPv6)
  - Link (Network Access) Layer (Ethernet, MAC)

TCP

- TCP is the workhorse of data communication between applications, especially for (high-performance) data transfers

- Designed when the losses were equivalent to congestion in the network

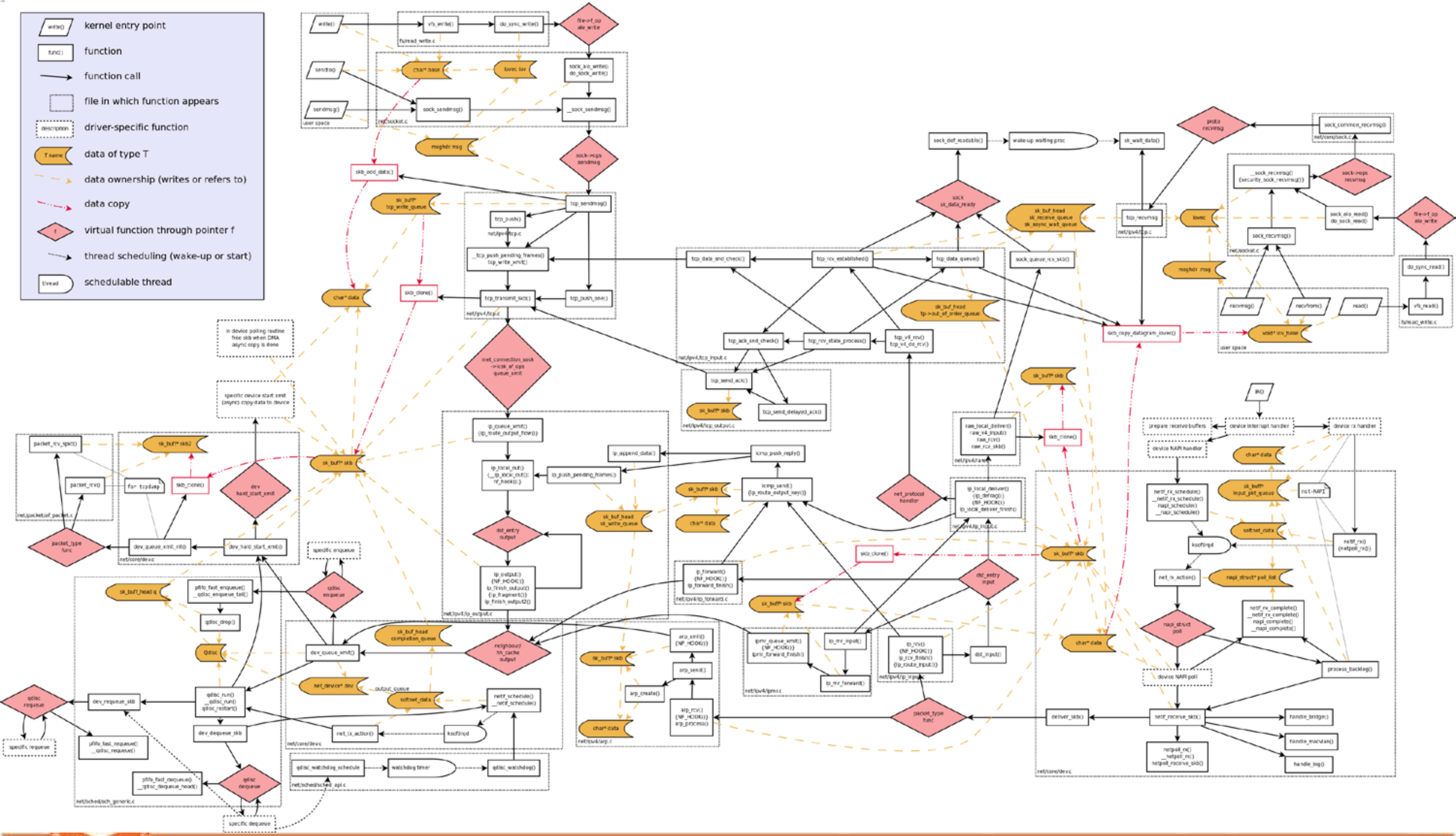
- Not suitable for LFN (Long Fat Networks)
  - LFN are networks with large BDP
  - BDP – Bandwidth Delay Product – represents the maximum “in flight” data

E.g. BDP for 1Gbps network, RTT 5ms

BDP = 1000 * 0.005 = 5 Mb = 640 KBytes

TCP Linux implementation

We will analyze each component in the following slides



TCP PERFORMANCE

TCP performance

- What influences the TCP performance?
  - **Packet Loss**
  - **Out of order delivery**
  - **Round-trip**
  - **Congestion avoidance algorithm**

- Matt Mathis formula

$$\textbf{BW} <= \left(\frac{\boldsymbol{MSS}}{\boldsymbol{RTT}}\right) * \frac{\mathbf{1}}{\sqrt{\boldsymbol{p}}}$$

where :

MSS – Maximum Segment Size – maximum amount of data which can be received in a single TCP segment

RTT – Round Trip Time

p – probability of packet loss

TCP buffer size

- Default TCP send and receive buffer size were initially **64KBytes**

- Increased via the setsockopt(SO_SNDBUF, SO_RCVBUF) sys call in the application

- Auto-tuning of the buffer size were introduced recently:
  - in Linux (kernel 2.4 and refined in 2.6)
    - Linux 2.6 started with 256KB max (now at 4MB) SL(C)5?
  - Windows Vista/7 16MB
  - Mac OS 10.5

- The autotuning works far better in the recent kernels, no need to call setsockopt(SO_*)

TCP settings Linux

/etc/sysctl.conf

```
net.core.rmem_max = 33554432
net.core.wmem_max = 33554432

net.ipv4.tcp_rmem = 4096 87380 33554432
net.ipv4.tcp_wmem = 4096 65536 33554432
```

DO NOT SET net.ipv4.tcp_mem as suggested in some tuning recipes from some Ethernet vendors. This is computed at boot time.

In 2006 when we started with **FDT** we used to recommend 8MB, but 16MB or 32MB should be more suitable for recent hardware.

More settings:
http://monalisa.cern.ch/FDT/documentation_syssettings.html
http://fasterdata.es.net/host-tuning/

TCP settings Linux

- A few more settings for Linux (same **/etc/sysctl.conf**)

```
net.core.netdev_max_backlog = 25000
net.ipv4.tcp_congestion_control = cubic
```

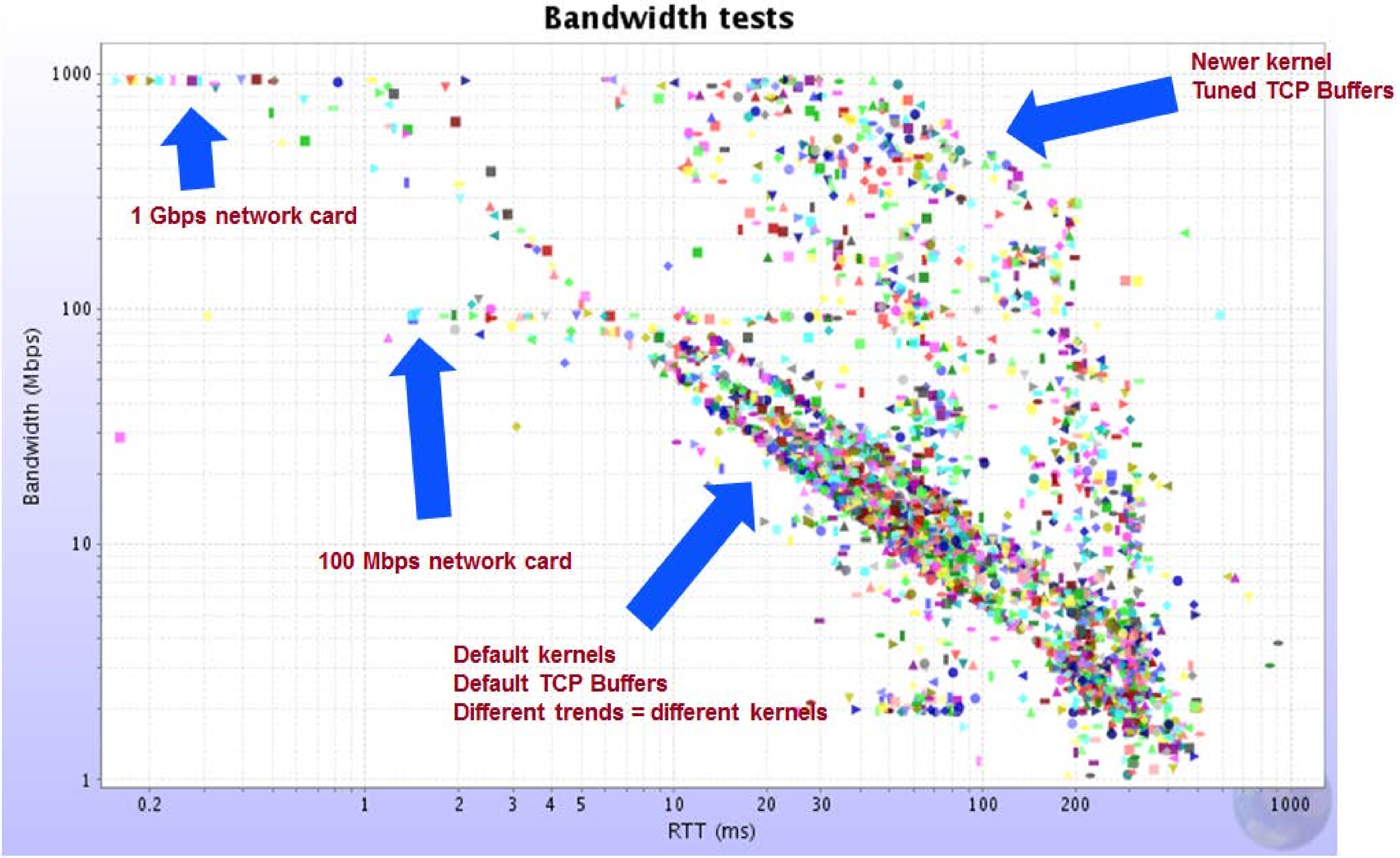
(a few series of kernels 2.6.15-18 had issues with cubic)

```
net.ipv4.tcp_no_metrics_save = 1
```

- It is highly recommended to keep the kernel updated
  - Last “goodies” – automatic TSO sizing and FQ scheduler for TCP pacing came in 3.12.4 end of 2013 … http://lwn.net/Articles/564978/

FDT bandwidth tests in Alice

Single TCP stream performance



Newer kernel
Tuned TCP Buffers

1 Gbps network card

100 Mbps network card

Default kernels
Default TCP Buffers
Different trends = different kernels

ETHERNET / NETWORK INTERFACE CARD

Ethernet/NIC

- MTU – Maximum Transfer Unit
  - Jumbo frames MTU 9000 (on the end-host), 9192 in the WAN
  - caution when using it; 1500 still the norm

    ```
    Test before: ping –s 8000 –Mdo endHost
              (actually ping –s 8972 –Mdo endHost)
    ```

  - Can cause “black-holes” (one direction fine, not the other one)

    ```
    use net.ipv4.tcp_mtu_probing=1 in /etc/sysctl.conf
    ```

  - MSS = MTU – (ip header + tcp headers)
- Ethernet FLOW control
  - mechanism for temporarily stopping the transmission of data via PAUSE frame
  - Whenever the queue at the receiving port get full a PAUSE frame is sent back to the sender to stop the transmission
  - pacing at lower level (instead of TCP)
  - can be disabled via ethtool, or kernel module parameter
  - required at least for 40Ge cards
  - as well as RoCE transfers (RDMA over Converged Ethernet) - Lossless Ethernet
  - Seen increases from 4-5Gbps to 22Gbps per TCP stream

Ethernet/NIC

- IRQ pinning (also know as IRQ affinity)
  - Manually assign the interrupts per CPU (core, HT)
  - Usually is needed whenever a new generation of network cards appears (E.g. migration from 100Mbps to 1Gbps, then 10Gbps, 40Gbps, ...)
  - Most of the time needed at the receiver due to CPU saturation (use mpstat, htop to monitor per core utilization)
  - Hint: **irqbalance daemon** will be disabled if manual config

```
E.g. FOR Linux
cat /proc/interrupts | grep ethX
echo bitMask > /proc/irq/<irqNo>/smp_affinity
```

where `bitMask` represents the core#

```
  (core 2 = 04, core 4 = 16, core 5 = 32, etc)
```

NETWORK TUNING

Network tuning

- **TCP congestion control mechanism produces bursty traffic as seen by the network devices**

- **Site Firewalls**
  - Usually the firewalls are able to handle less traffic than their network interfaces
  - Needs big input buffers to accommodate the TCP burstiness
  - It is recommended to isolate data-intensive traffic (HEP) and by-pass the firewall

- **Router/Switch Buffer Size Issues**
  - Packet drops, losses (if any) usually appear at the output queue in the ToR (Top of the Rack) switches and/or cluster aggregation switch
  - Recommended network devices with larger buffers, even though more expensive
  - “Buffer bloat” may arise in congested networks == bigger buffers along congested paths

DEBUGGING

Debugging

- What to measure?

  - SNMP monitoring of network devices
    - Interfaces utilization, errors on the interfaces, congested paths, packet drops

  - Same for the hosts (/proc FS, ifconfig, ethtool –S):
    - Interface utilization, errors on the interfaces
    - Look for CPU saturation (per core)

  - Packet loss, route (traceroute, tracepath),
    - UDP at different rates and look for losses along the path (recommended nuttcp and/or iperf3)
      - Initially smaller rates (if errors) check the connectors
      - Increase the rate

```
nuttcp -i 5 -T 60 -R 8G -u 10.100.100.4
```

Debugging

- Continuous monitoring via active probes

- Alarms whenever performance drops below a certain threshold

- Tools: nuttpc, bwctl, iperf, FDT mem2mem

- Orchestrated: PerfSONAR PS (WLCG service), MonALISA

あなたのためにこれ以上の刺身！

有り難う御座います