Network Management

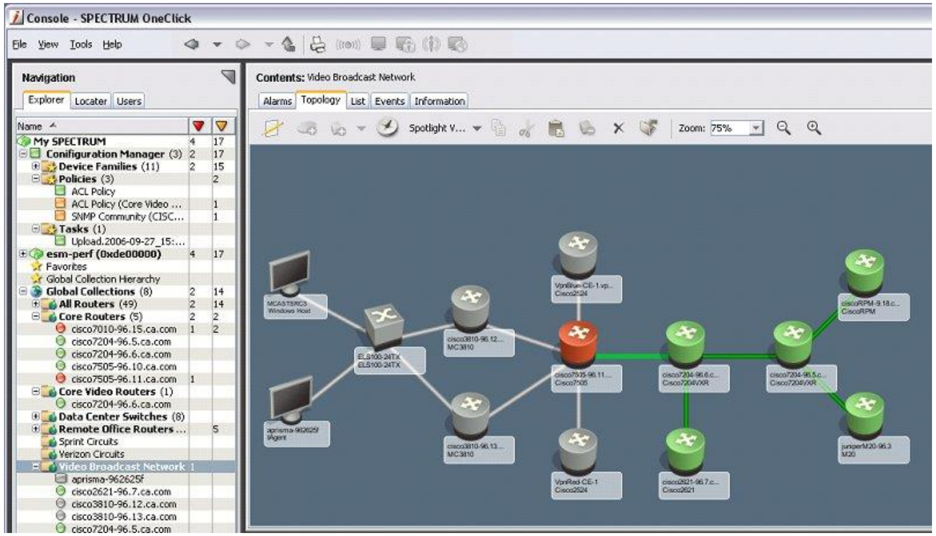
Silvia Batraneanu (UCI)
on behalf of
ALICE, ATLAS, CMS and LHCb

Outline

- Monitoring :
  - Health
  - Performance
  - Flows
- Device Configuration
- MAC/IP/Hostname management
- Device registration
- Network documentation
- Topology discovery
- Physical design
- Conclusions

All-in-one tool – CA Spectrum

- Full network management solution used by IT for many years
- Does:
  - ✓ **Health monitoring(events, alarms),**
  - ✓ Statistics gathering (5 minutes),
  - ✓ Topology discovery,
  - ✓ Configuration management,
  - ✓ Event correlation,
  - ✓ and many more



- Suited for general purpose networks but not necessarily for our needs
- Pretty expensive commercial tool
- Requires a lot of expertise
- Support..hmm..contributes heavily to the Spectrum admin's autonomy

Health monitoring (1)

- What do we need to monitor?
  - ✓ Device and port status
  - ✓ HW and SW failures

- How can we monitor?
  - *Synchronously*(SNMP polls)
  - *Asynchronously* (SNMP traps and switch logs)

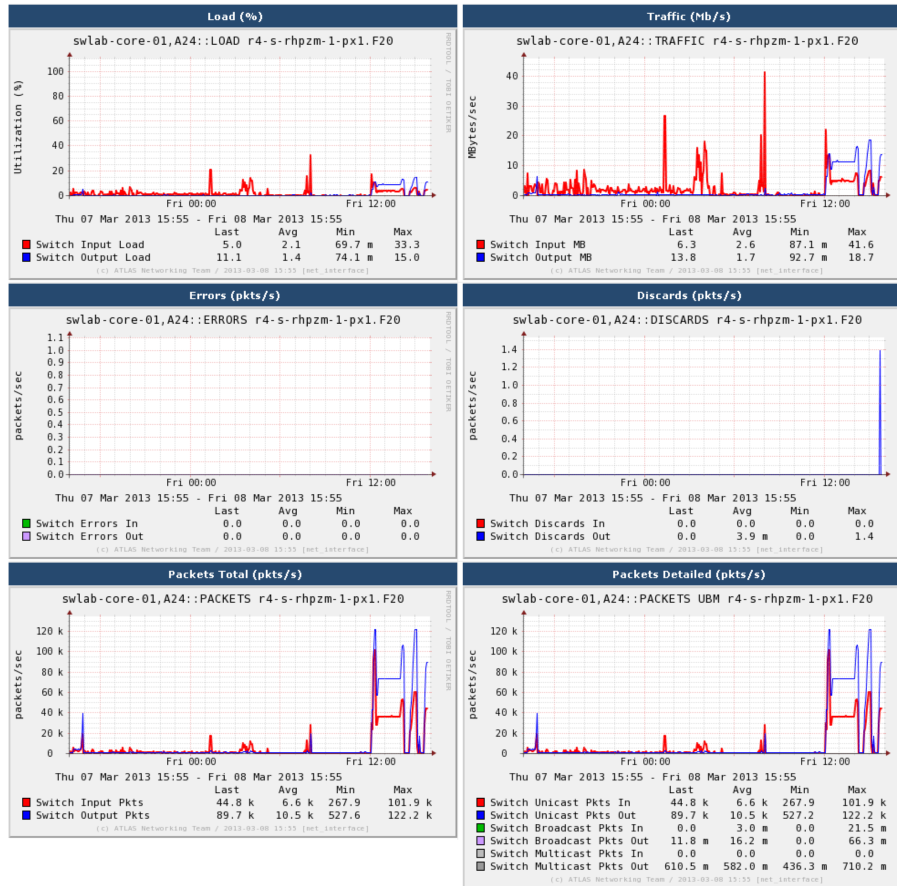
Health monitoring (2)

- What SNMP-based health monitoring tools are currently used?
  - ALICE: SNMP traps collection integrated with Lemon
  - ATLAS : Dedicated Spectrum instance -> moving to Icinga
    - topics under investigation : traps support, congestion notifications, alarm filtering and display -> could profit from the other experiments prior experience
  - CMS :
    - control network managed by IT and monitored by IT Spectrum instance
    - data network equipment status reported in Icinga, connectivity tests in the online software
  - LHCb: Icinga

- Which switch log collection, display and alerting tools are currently used?
  - ATLAS:
    - ✓ Syslog-ng for collection
    - ✓ In-house tool(NetLog) for display
    - ✓ Custom Spectrum alarms for chassis HW failures and errors
  - LHCb:
    - ✓ Rsyslog for collection
    - ✓ Ossec for analysis, real-time alerting and active response
  - We can knit a complete switch log handling solution!!

Performance monitoring (1)

- What can we gather?
  - Traffic counters
    - ✓Line occupancy and overall packet rates
    - ✓Specialized packet rates: unicast, broadcast, multicast
    - ✓Erroneous, discarded packet rates
    - ✓Interface speed
  - Networking devices statistics
    - ✓CPU and Memory Occupancy
    - ✓Temperature
    - ✓Ping Time, etc



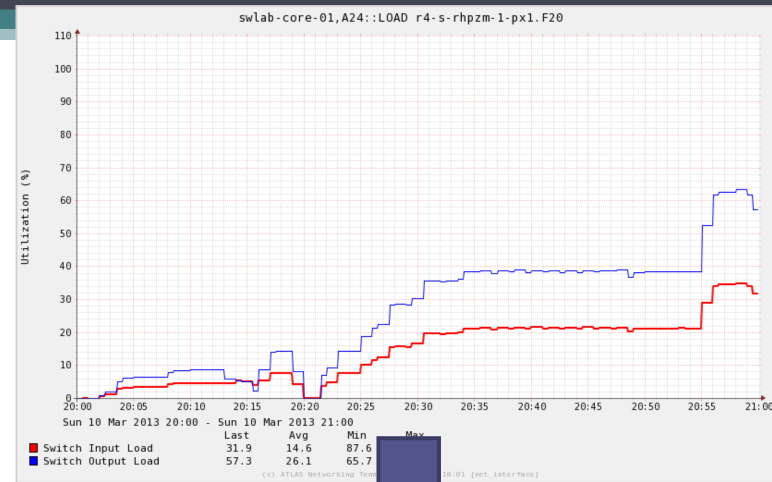
Performance monitoring (2)

- Are these statistics useful? Definitely YES!!!
- Why?
  - Traffic counters
    - ✓ Line occupancy and overall packet rates -> congestion, abnormal traffic shapes, correlation with data taking parameters
    - ✓ Specialized packet rates: unicast, broadcast, multicast -> sporadic ARP broadcast storms
    - ✓ Erroneous, discarded packet rates -> congestion, physical layer (and sometimes hidden) problems
    - ✓ Interface speed -> detecting 100Mb/s links, auto-negotiation doesn't always work correctly
  - Networking devices statistics
    - ✓ CPU and Memory Occupancy -> low priority processes (SNMP agent, logging) not responsive enough
    - ✓ Temperature -> rack or switch fan cooling failures
    - ✓ Ping Time -> abnormal delays in the network
- Initial reflex: blame the network -> Current behavior: look at the stats and, if something is wrong, blame the network

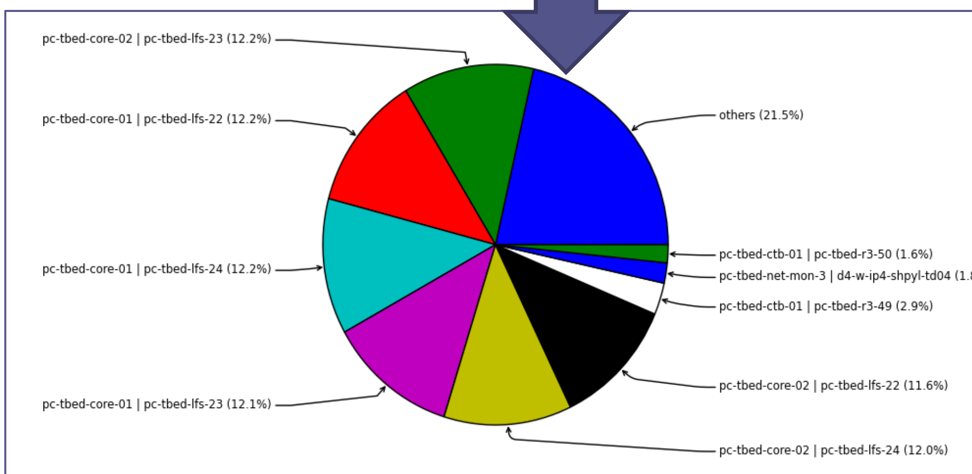
Performance monitoring (3)

- What tools are currently used?
  - ATLAS: Tried it with Spectrum, currently using in-house tools
    - Fast SNMP Poller (APoll)
      - 30 seconds resolution
      - uses C++ and SNMP bulk get requests
    - Custom WebApp (Net-IS) for statistics display
    - Spectrum for congestion alerts -> Icinga-APoll integration under investigation
  - LHCb: Cacti
    - Several useful plugins: boost for caching, Thold for alerting
    - 1 minute resolution
    - Traffic congestion alarms
  - CMS: Real-time traffic plots from Spectrum, no history
    - Started using Cacti for a few links,
    - Started looking into Icinga
- At least a few open-source solutions that are worth investigating: Icinga-Ganglia, Cacti

Flow monitoring



- ❑ Based on statistical packet sampling
- ❑ Provides information about network conversations at different layers
- ❑ Implies support from the switch vendors:
  - ❑ HP, Dell (Force 10), Brocade : sFlow
  - ❑ Cisco: NetFlow
- ❑ Currently only used by ATLAS
  - ❑ In-house tool(Net-SFlow)
    - sFlow collector
    - sFlow processor
    - NetIS for display
  - ❑ Testing open-source tools
    - NFDump collector
    - NFSen display tool
  - ❑ Very useful for troubleshooting
    - Congestion
    - Access/Security
- ❑ Not used on a daily basis and not crucial
  - ❑ Worth using (and adapting) existing open-source tools but not building an in-house solution



5 10	Conversation	Source Address	Destination Address	Bytes TX	Frames TX	Bytes RX	Frames RX	Bytes	Frames
☑	pc-tbed-core-02 | pc-tbed-lfs-23	10.193.2.41	10.193.6.41	140632576	316928	0	0	140632576	316928
☑	pc-tbed-core-01 | pc-tbed-lfs-22	10.193.2.17	10.193.6.40	139708928	309248	215040	1024	139923968	310272
☑	pc-tbed-core-01 | pc-tbed-lfs-24	10.193.2.17	10.193.6.42	136977920	309760	2807808	6656	139785728	316416
☑	pc-tbed-core-01 | pc-tbed-lfs-23	10.193.2.17	10.193.6.41	139087872	309760	71680	1024	139159552	310784
☑	pc-tbed-core-02 | pc-tbed-lfs-24	10.193.2.41	10.193.6.42	134996480	302592	2469888	6144	137466368	308736
☑	pc-tbed-core-02 | pc-tbed-lfs-22	10.193.2.41	10.193.6.40	132777984	297984	0	0	132777984	297984
☑	pc-tbed-ctb-01 | pc-tbed-r3-49	10.193.20.59	10.193.6.34	24685056	33792	8078848	12800	32763904	46592
☑	pc-tbed-net-mon-3 | d4-w-ip4-shpyl-td04	10.193.2.66	10.193.6.3	4537856	10752	16337408	16896	20875264	27648
☑	pc-tbed-ctb-01 | pc-tbed-r3-50	10.193.20.59	10.193.6.35	10089984	14336	8432640	15360	18522624	29696
☑	d4-w-ip4-shpyl-td04 | pc-tbed-net-mon-2	10.193.6.3	10.193.2.65	8857600	13312	6586368	14848	15443968	28160

Device configuration

- How are the experiments doing it?
  - ALICE : Manually (few network devices)
  - ATLAS : Mainly manually for the cores, automated for the edge switches using a in-house Python toolkit(Sw-Script) or Expect scripts
  - CMS: Mainly manually , some automation scripts for the Myrinet setup
  - LHCb: Mainly manually, some automation scripts based on pexpect
  - **Everybody seems to be happy with their current solution**

- Automation : does it help?
  - Depending on the size of the network and on the underlying technology (Ethernet, IB)
  - Very useful for edge switches (when more than a few and having similar functions)
  - Useful in the case of core devices which have similar configuration on multiple interfaces (VLANs, trunks)

- Backup:
  - regular jobs retrieving configuration files from the switches via TFTP
  - version control on the retrieved files
  - ATLAS: started using Rancid

```
console# configure
console (config)# interface vlan 1
console (config-if)# ip address <ip address x.x.x.x> <subnet mask x.x.x.x or /x>
console (config-if)# exit
console (config)# ip default-gateway <ip address x.x.x.x>
console (config)# exit
console# copy running-config startup-config
```

MAC/IP/Hostname management

- ATLAS, ALICE and CMS rely on CERN Network DB
    - Advantages
        - ✓ Centralized IP space management,
        - ✓ Powerful SOAP interface offering many bindings: Python, Perl, Java, etc
        - ✓ Managed by IT
    - Disadvantages
        - ✓ No VLAN support,
        - ✓ No support for multiple subdomains,
        - ✓ Core operations not allowed to users,
        - ✓ Extracting information takes some time,
        - ✓ What others?
- LHCb has a private DB
- All experiments use dedicated DHCP servers
- DNS:
    - ALICE and ATLAS: slave servers from the IT ones
    - CMS : custom server, multiple domain names
    - LHCb: custom server

Device registration and network documentation

Device registration

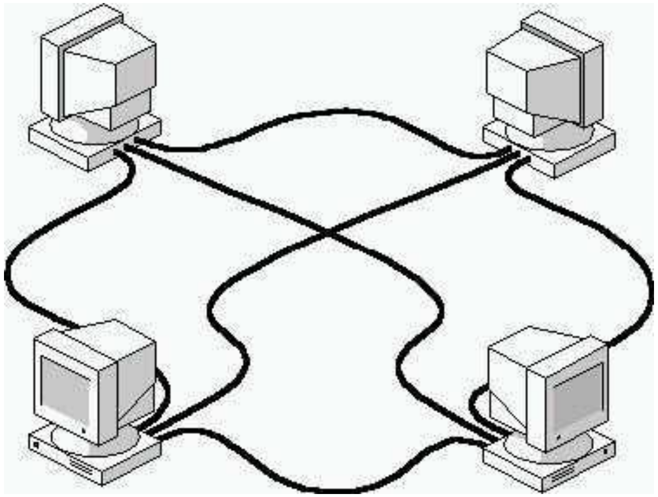
- ALICE, ATLAS, CMS: SOAP interface to CERN Network DB, mainly Perl and Python bindings
- LHCb : Device entries created using Quattor templates

Network description

- In general stored in dedicated DBs
- In general augmenting the data present in CERN Network DB
  - Network topology + link type, trunk info
  - Network configuration(eg. VLANs, SNMP and logging)
  - Logical grouping
  - Physical layout
  - ..and others

How different are our dedicated DBs?

Topology discovery



- Is it useful? ..The most reliable way to track changes especially when multiple teams are involved
- Strongly dependant on the underlying technology
  - Ethernet: MAC address tables and LLDP
  - IB: dedicated inter-switch protocol
- Strongly dependant on the link types and device manufacturers
- What tools are used in the experiments?
  - ATLAS : In-house tool (NetDiscovery)
    - No LLDP support for Brocade
    - No VM support
  - LHCb : NEDI open source tool

Physical design

- Implies cable layout and labeling
- Currently done in a more or less manual way
- What are the experiments using?
  - ALICE, CMS, LHCb : Local DBs, Spreadsheets
  - ATLAS:
    - ✓ Visio drawings made with a dedicated (but deprecated) plugin
      -> cable spreadsheets
    - ✓ Started building a Web-based application mainly for manual editing and integration with the discovered topology
- Capabilities included into high end data center management solutions
  BUT at prohibitive prices
- A domain which calls for unified effort and hopefully a common solution
  What is your wish-list?
  What are the common points?
  Would it be worth investing in a good layout tool?

Conclusions

- All experiments do health monitoring
  - converging to Icinga
  - seem to have all the answers for log handling
- Performance monitoring proved very useful
  - Open source solutions available(Icinga, Ganglia, Cacti)
- Flow monitoring is useful in some situations
- Depending on the network particularities, automatic configuration may or may not be needed
- Topology discovery is useful but strongly dependent on the underlying technology and workflow
- IP allocation and device registration mainly dependant on CERN IT Network DB
  - How can we overcome the disadvantages?
- Physical design :
  - We should do better than that
  - Perfect topic for a common discussion and possible initiative!

Thank you!

- ALICE – Ulrich Fuchs
- ATLAS – Giovanna Lehmann, Eigil Obrestad, Sergio Ballestrero, Eukeni Pozo
- CMS – Marc Dobson, Andrea Petrucci
- LHCb – Guoming Liu