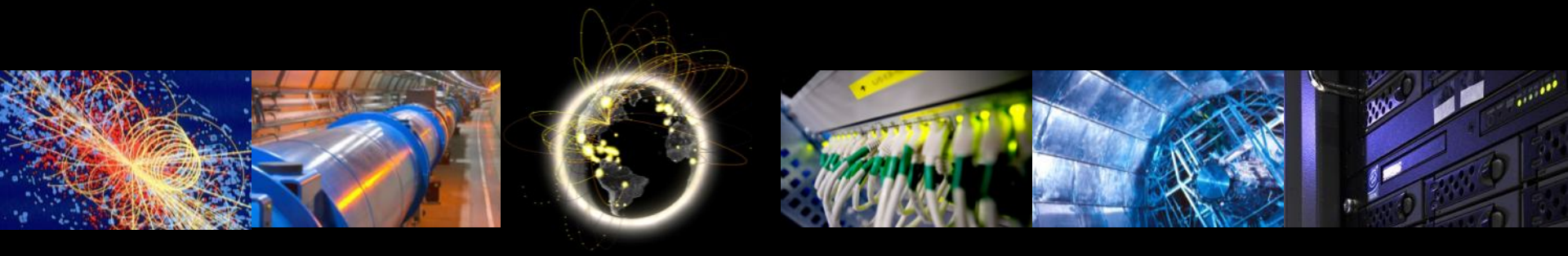
OSG Networking Analytics: Evolution and Status

Ilija Vukotic, Shawn McKee

OSG All-hands Meeting, Salt Lake City, Utah
20 March 2018



WLCG
Worldwide LHC Computing Grid

Open Science Grid

Overview

Ilija and I will cover OSG Networking Analytics

I will cover the **first part**, introducing OSG networking and the evolution of our data pipeline

Ilija will then take over and cover the details of the analytics platform and some of the work exploring how best to use the metrics we are collecting...

OSG Networking Components

- Network Monitoring via **perfSONAR**
  - Having perfSONAR fully deployed with a global dashboard is giving us powerful options for better management and use of our network
- A network **datastore** host all network metrics
- Tools to manage and maintain our infrastructure
  - A Modular dashboard (**MaDDash**); critical for quick "visibility" into networks. We can't manage/fix/respond-to problems if we can't "see" them.
  - **OMD/Check_mk** (used to monitor and verify the state of many globally distributed perfSONAR services); required to maintain the overall proper functioning of the monitoring infrastructure.
  - The development of the "**mesh-configuration**" and corresponding GUI interface; critical to creating a scalable, manageable deployment for WLCG/OSG
- Documentation --- Installation, debugging, How-tos
- Outreach and Support
  - With the network R&E community, VOs, software developers
  - OSG Support provides network ticket triage and routing

OSG Networking Evolution

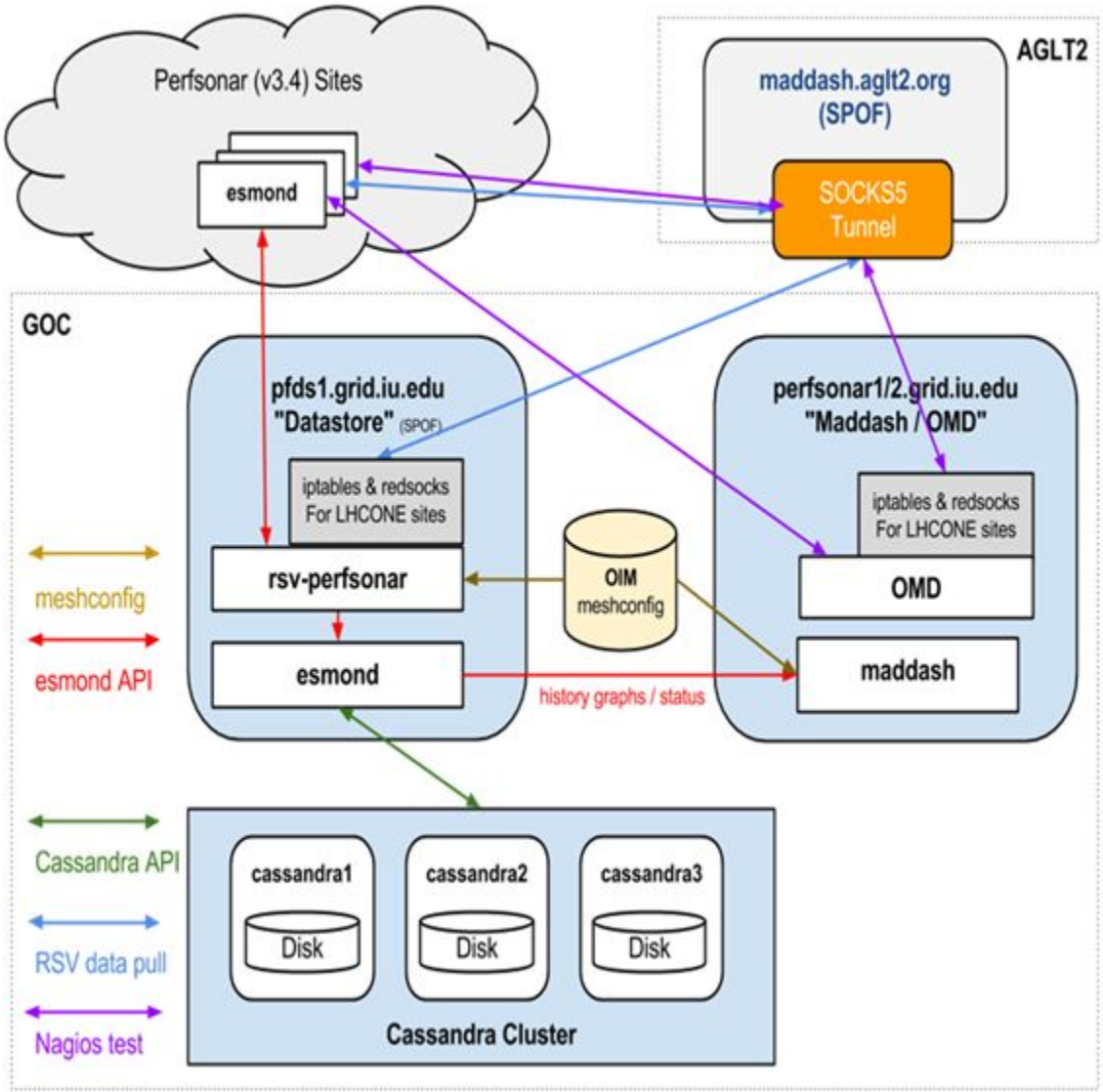
The OSG network data pipeline has significantly evolved since its inception in 2012

- Initially we queried all **perfSONAR** toolkits centrally and stored the gathered metrics in ESmond
  - Users had a significant learning curve to be able to access and analyze the data; it wasn't easy to do analytics
- The system evolved to include parallel publication to an **ActiveMQ** bus at the end of 2015
  - Users could choose to subscribe to data of interest
  - Data could now easily be sent to Elasticsearch
- In 2017 we added a new destination to **RabbitMQ** (run be Nebraska and hosted in **AWS**)
  - New destination to Nebraska Elasticsearch
  - New 'long-term' repository for data
- In 2018 we are focusing on streamlining the system

Original OSG Data Pipeline

In 2014 we had a data pipeline that is shown on the diagram on the right.

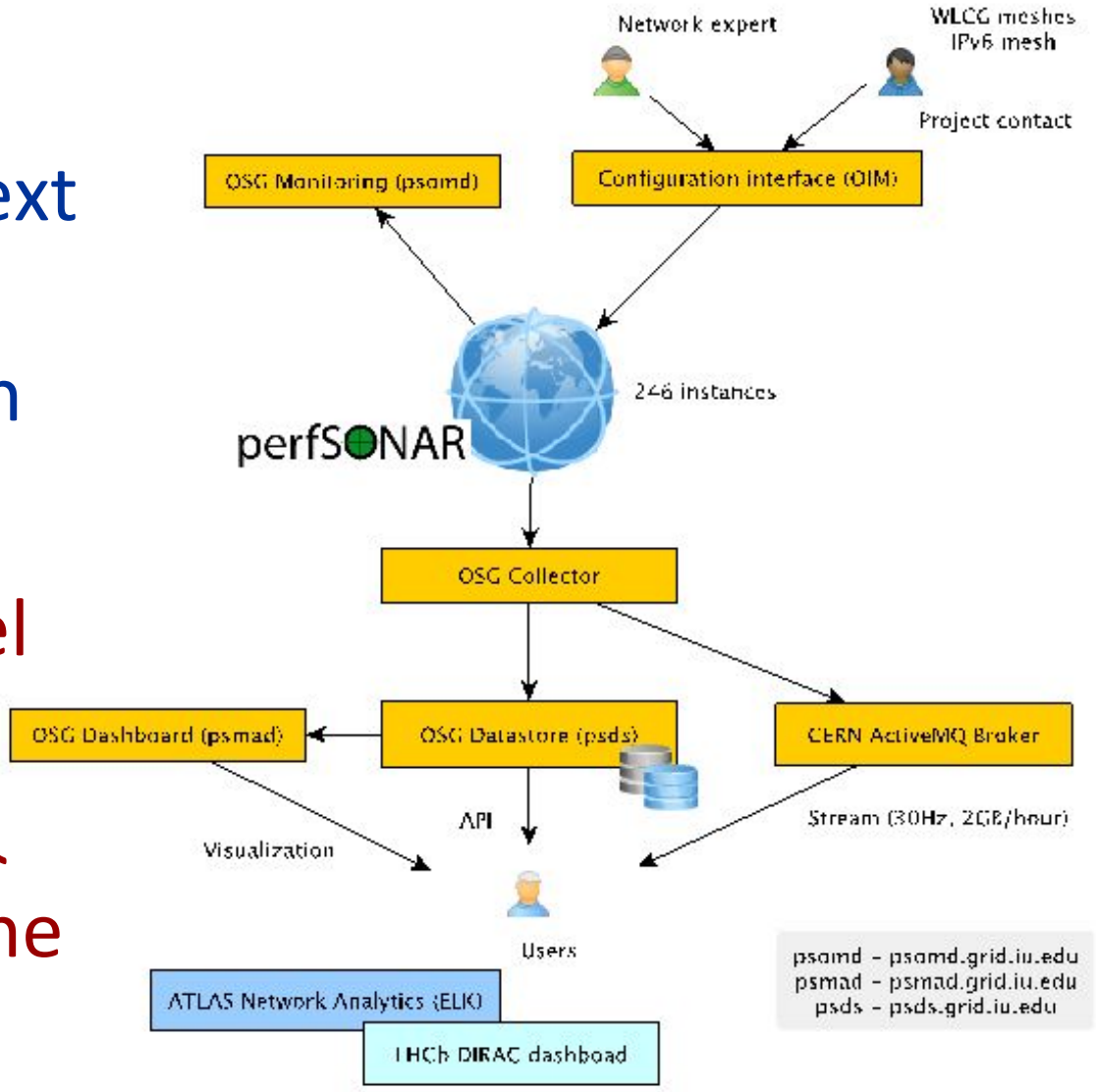
A conglomeration of various technologies to meet our initial goals



First Evolution: OSG Data Pipeline

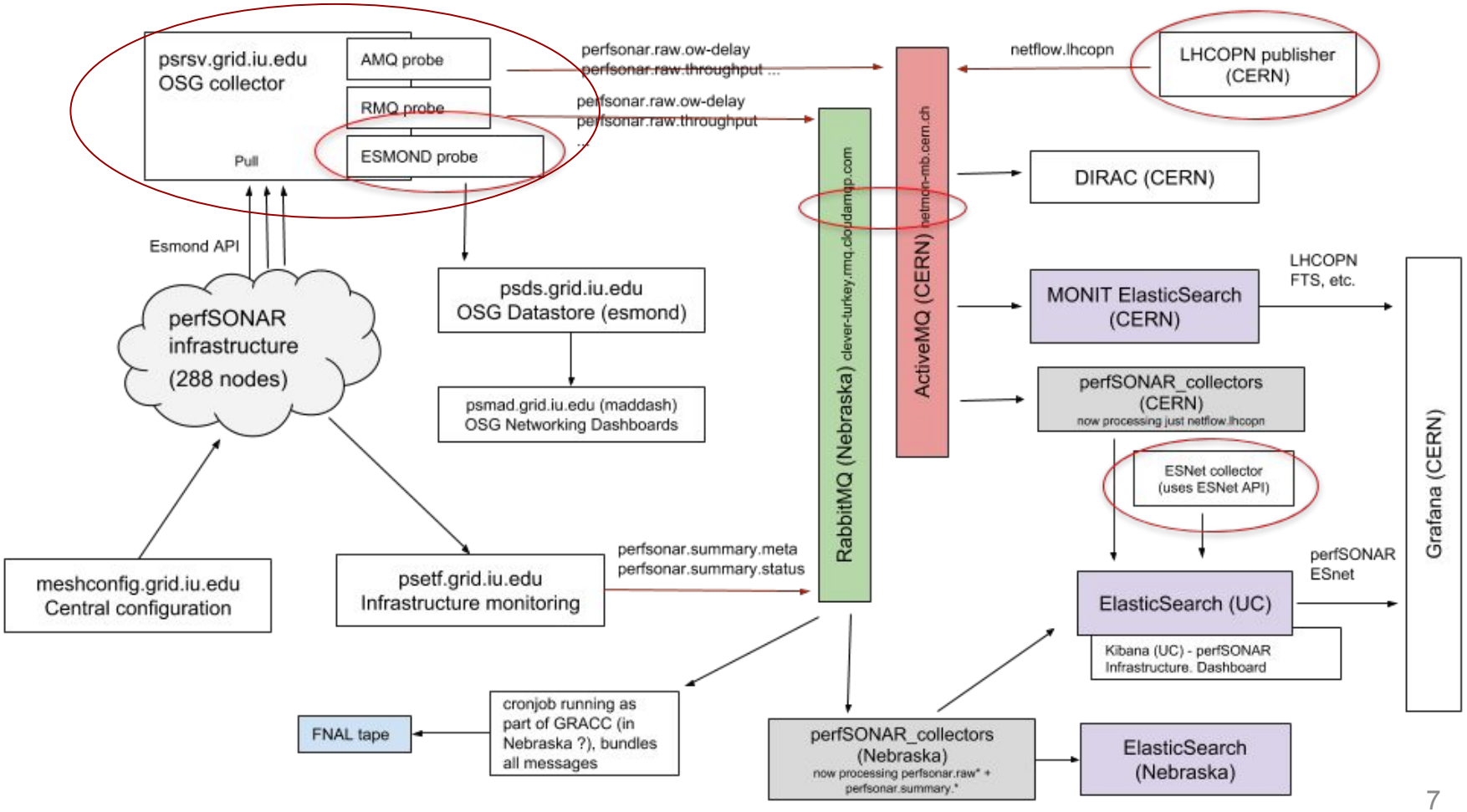
A simplified diagram of the next step in the data pipeline evolution

We added parallel publication to CERN's ActiveMQ message bus at the end of 2015



Current Network Data Pipeline

Current pipeline. Red ovals outline areas we are changing



OSG Networking Data

The OSG networking data pipeline is gathering many kinds of data

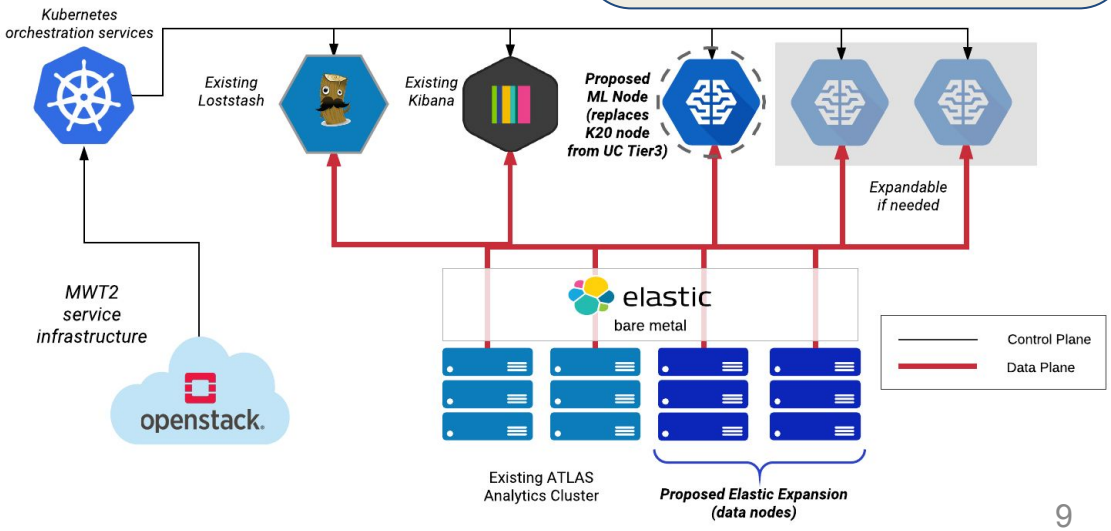
- perfSONAR metrics
  - **Latency** and **packet-loss** (10Hz of test packets per source-destination)
  - **Bandwidth** (one test per 4 or 6 hour window)
  - **Traceroute**: monitoring network path topology
- ESnet and LHCOPN interface and flow data
- FTS (File Transfer Service) metrics
  - Rate per file
  - Total traffic generated per site and link
  - Queue times

Given this data, the interesting work is in using it, Ilija?

Analytics platform

- Monitoring
  - Long retention
  - Fast & reliable
  - Has all data (ESnet, PerfSONAR, FTS,...)
- Debugging issues
  - Has to have raw data
- Analysis
  - Open remote access to all the data
  - ML platform
    - Python, Jupyterl
    - Keras, Scikit-lea
    - GPU support
- Alarm & Alert

Upgrade:
- Adding 5 ES nodes (doubling disk space)
- Adding dedicated ML node (8 GTX 1080 Ti)



First steps

- Simple alarms for now based only on PerfSONAR*
  - To make sure we have all the data, we set up alarms on issues with the data transport chain.
  - An alarm on capital problems at individual sites
    - if packet loss from a site to more than 6 other sites is greater than 2%. While highly specific, this test is not sensitive enough.
    - Coming soon: alarms on large increases in latencies and significant changes in path.

* For now FTS data doesn't tell us much about network performance (limited by the FTS itself).

Anomaly detection

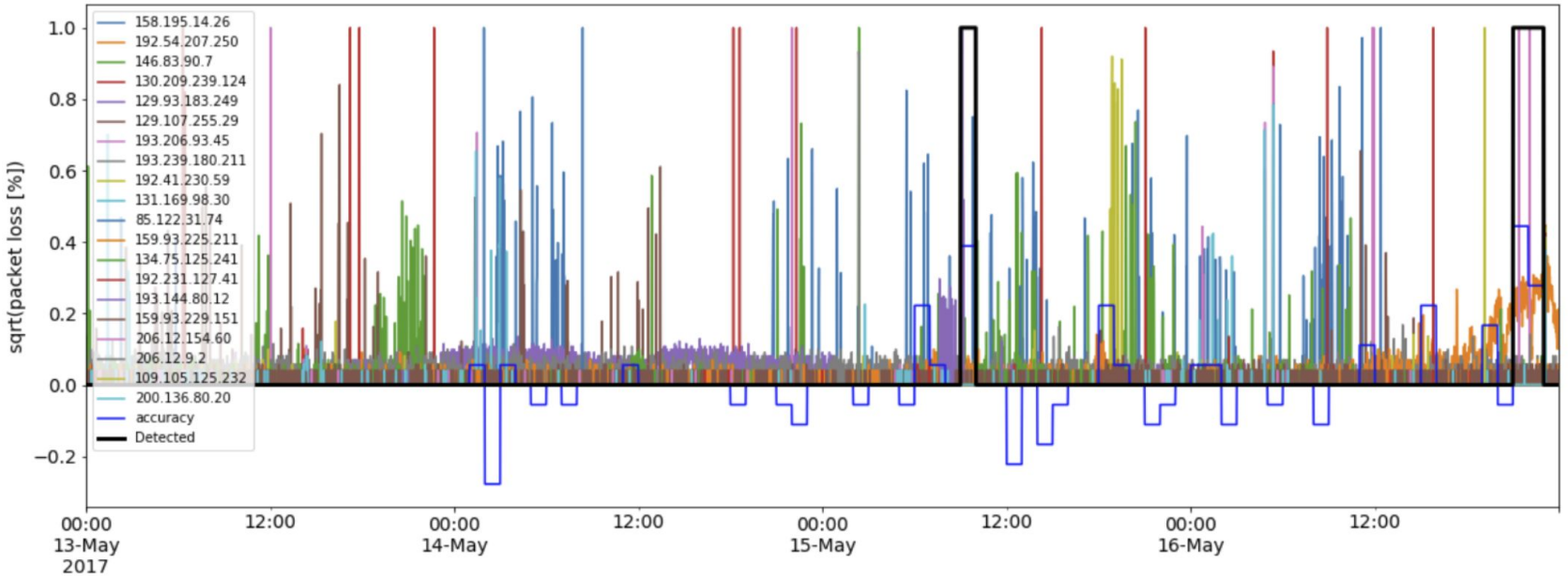
Key system requirements:

- **Fast enough** - alarm to be raised in <3h.
- **Scalable** - with so many sites and links to monitor only minutes are available to evaluate (unless we go for a distributed system)
- **Sensitive** - down to single links.
- **Super Specific** - in a mesh that big, there are always some anomalies. Goal is to figure out which ones are significant enough.
- **Informative** - method should give us a debugging clue on:
  - **What happened.**
  - **Where it happened.**

Anomaly detection - cont.

Tested few approaches:

- Single class SVM - dimensionality problem
- BASIC - Bayesian inference of simultaneous change-points in multiple timeseries - slow.
- BDT and ANN - works well but needs validation



*https://github.com/ATLAS-Analytics/AnomalyDetection/raw/f5851dfddd74fd6d053b7a5de64ac416c89f8f58/PerfSONAR/Paper/AD.pdf

Anomaly detection - cont.

Still some things to test:

- Data Smashing*
- Reinforcement Learning (Bootstrapped Q-learner)
- Create an OpenAI environment, see what others can come up with

In addition to alerts we could provide services predicting

* http://rsif.royalsocietypublishing.org/content/royinterface/11/101/20140826.full.pdf

Containerized analytics

A lot of people interested in trying out new methods, or debugging their cloud/site network.

To make it easy to start we created a container with all the tools (data access, neo4j for topology, jupyterlab, ML packages), codes, examples, documentation.

opensciencegrid/network_analytics

We will run one instance in cloud.

Summary

- We have a working infrastructure in place to monitor and measure our networks
- Network dashboards can be used to get quick, specific details for subset of the network
- We have opportunities to improve our network and storage resource use, lowering the latencies involved in moving data
  - Focusing on the details here could be very beneficial
- We need (mostly) automated ticketing/alarming that might include predictive components (increasing packet loss may cause transfer issues in the next few hours…)

Questions or Comments?

References / Links

- OSG networking documentation https://opensciencegrid.github.io/networking/
- OSG networking dashboards that rely on the analytics platform:
  - http://atlas-kibana.mwt2.org:5601/goto/7a9e388f4685965bcec6ced7705f7d9a
  - http://monit-grafana-open.cern.ch/dashboard/db/home?orgId=16
  - http://atlas-kibana.mwt2.org/app/kibana#/dashboard/Perfsonar-Alarms
- Service monitoring using check_mk/Nagios: https://psetf.grid.iu.edu/etf/check_mk (requires x.509 certificate in your browser)