# Table of Contents

# WLCG Data Analytics platform

**Scalable data store and analytics platform for monitoring WLCG, a distributed data-intensive scientific infrastructure**

# Project Introduction

The current WLCG monitoring system has proven to be a solid and reliable solution to support WLCG functions and operations during LHC data-taking years. A variety of data coming from different services and experiment-specific frameworks is gathered, processed and archived and a generic web-based dashboard provides a uniform and customisable monitoring interface for scientists and sites. In the next future, the WLCG monitoring infrastructure have to cope with an extension of the volume (e.g. higher LHC luminosity) and the variety (e.g. new data-transfer protocols and new resource-types, as cloud-computing) of the monitoring data. Nevertheless, traditional architecture where relational database systems are used to store, to process and to serve monitoring events has **clear limitations**. Scalability is difficult to achieve, technique like sharding push complexity at application level, leading to higher maintenance and operational costs and to human-faults. Moreover, effective monitoring requires low-latency read access to real-time data, while built-in database procedures impose constraints on the format, granularity and timeliness of monitoring information.

The Monitoring section of the Support for Distributed Computing ( SDC) group, at the CERN IT department, is working on the research, the design and the development of the new data store and analytics platform for the evolution of the WLCG monitoring, able to cope with the scalability, flexibility and fault-tolerance requirements foreseen in the long-term WLCG scenario. The task requires a sound knowledge of distributed systems theory and concepts together with a deep understanding of the new technology stack for large-scale data analysis. The project will be done relying on facilities and experience from the Agile Monitoring initiative of the CERN IT department. In recent years, the challenge of handling big volume of data has been taken by many companies, particularly on the internet domain, leading to a full **paradigm shift** on **data archiving**, **processing** and **visualization**. A stack of new technologies appeared, each one targeting specific aspects on big-scale distributed data-processing. However, all these technologies, such as batch computation system (e.g. Hadoop) and non-structured databases, can handle very large data-volumes with little cost but with serious **trade-offs**. The goal is to architect the new platform in a tool-chain approach leveraging on the most appropriate technologies and computing techniques for the WLCG use case.

The project can be decomposed in three main objectives and areas of work.

- The first objective is the **batch layer**, to store a constantly growing dataset providing the ability to compute arbitrary functions on it.
- The second objective is the **serving layer**, to store the batch-processed views, using indexing techniques to make them efficiently query-able.
- The third objective is the **real-time processing** layer able to perform analytics on fresh data with incremental algorithms to compensate for batch-processing latency. Moreover, the real-time analytics layer can be used as input for active-reaction, adopting classical pattern matching approach to promptly detect errors and failures on the stream of monitoring events.

Depending on the result of the technology evaluation in respect to WLCG use cases, the project architecture can be considered a case for the lambda approach, where different tools are used to address each functionality, or a more suitable scenario for a single-tool solution providing all required capabilities.

# Resources

- Git Batch-processing on Hadoop: https://git.cern.ch/web/?p=wlcgmon/analytics.git
- Git Real-time investigation: https://git.cern.ch/web/?p=wlcgmon/analytics/rt.git
- Git Spark: https://git.cern.ch/reps/wdt-spark
- Git Batch & Stream-processing on Spark for XRootD: https://git.cern.ch/web/wlcgmon/analytics/xrootd-spark.git
- Git shared library: https://git.cern.ch/web/wlcgmon/analytics/xrootd-lib.git
- DirQ 2 HDFS: https://git.cern.ch/web/?p=wlcgmon/analytics/dirq2hdfs.git]]

- Mailing List/ e-groups: wlcg-data-analytics@cernNOSPAMPLEASE.ch
- Jira: https://its.cern.ch/jira/browse/WLCGANALYT

# Contributors

- Julia Andreeva/CERN: IT-SDC/MI section leader
- Luca Magnoni/CERN: project architect
- Professor Akram Khan/Dr David Smith (Brunel University): PhD supervisors
- Uthayanath Suthakar/Brunel University: PhD student - researcher & developer
- Cristovao Jose Domingues Cordeiro: IT-SDC/MI fellow - researcher & developer
- Georgiou Maria Varvara: IT-SDC/MI summer student - researcher & developer

# Project Status

The project is currently at the initial stage where the focus is on **technology investigation** and understanding of the best combination for the WLCG use cases. The preferred approach for the investigation is to quickly prototype a well defined set of WLCG use cases using the different candidate technologies, to get insight on advantages and limitations in a concrete scenario. CERN will provide the testing infrastructure using the IT Agile Cloud/Openstack facility. The result of this initial stage will be a **design document with the proposed architecture** for the project. Later stages will be the implementation of the different layers and validation with real WLCG data flow and volume.

A fully working prototype of Hadoop-based computing for WLCG transfer monitoring has been developed and tested with excellent results (30 June 2014).

A study on the possibility to use in-memory computation with Esper for transfers monitoring has been presented and a proof of concepts is available on git repo. (27 August 2014)

# Spark Evaluation

https://twiki.cern.ch/twiki/bin/view/LCG/SparkEvaluation

# Investigation on Real-Time processing

The real-time processing layer seems the most dynamic area of research, where several technologies exists offering a similar set functionalities. The investigation should understand the key features of each technologies, with focus on what they are meant for and overall architecture, together with language supported, API and connection points.

## Candidate technologies

- Storm (Twitter, now Apache)
- Kinesis (Amazon)
- Samza (Linkedin)
- S4 (Yahoo)
- evolution of Hadoop ecosystem, e.g. Apache Spark
- Esper

### Log collection and ingestion technologies

- Chukwa
- Flume

- Fluentd
- Logstash
- Kafka

# project with Brunel University

A collaboration between CERN IT-SDC/MI section and the Brunel University - Uxbridge has been established to work on a doctoral research project on the design and development of the WLCG data analytics platform.

- WLCGMon_DataAnalytics_PhD.pdf: Short PhD project description

### Meeting & Executive summary

- Kick-off - 21 March 2014
- Update - 11 April 2014
    - ♦ Uthay presented a preliminary investigation on the existing technology for batch processing/serving layer/ real-time analytics [slides ppt| pdf ]
    - ♦ Outcome/Actions:
        - ◊ CERN will host the testing infrastructure (IT Cloud/Openstack seems the ideal solution)
        - ◊ CERN will provide a first use case (data set and aggregation examples), to be presented next meeting
        - ◊ Uthay will circulate a summary doc with the results of the preliminary investigation by the next meeting
        - ◊ Uthay is invited to present the result of his investigation at the next SDC-MI section meeting (28 April 2014 10:00 CEST)
- CANCELED - Next meeting scheduled: 17 April 2014 10:00 (CEST)
- Update: 25 April 2014 10:00 (CEST)
    - ♦ Main topic: review of Uthay's presentation for section meeting
    - ♦ Outcome/Actions:
        - ◊ Uthay will start working on the collection of WLCG use cases
        - ◊ Luca will formalize the structure of a use case
        - ◊ First use case: data transfer
- Update: 09 May 2014 10:00 CEST
    - ♦
        - ◊ The EOS use case seems a perfect match for MapReduce approach
        - ◊ Uthay will work a first prototype of MapReduce based EOS statistics generation
        - ◊ Technology is HDFS for storing data and the Hadoop framework for computation
        - ◊ The prototype have to cope with EOS data load, so particular attention on the structuring of data for efficient parallelization over time bin
        - ◊ Prototype will include map and reduce Java code and data structure description
        - ◊ CERN will provide git repo and Jira project (as soon as Uthay get an account...)
        - ◊ Tentative deadline: 30th May 2014
- Update: 23 May 2014
    - ♦
        - ◊ Meeting with DBA group, the analytic use case is interesting for their evaluation of Hadoop. A test cluster (4 node, old DB machine) will be provided by DBA by the beginning of June.
        - ◊ They interest is on the evaluation of exporter from db to hadoop, data format (parquet), and abstraction (Impala)
- Meeting with Dirk Duellmann (IT-DSS) 11 June 2014:
    - ♦
        - ◊ Meeting was to share WLCG Analytics requirements and ideas in the light of a common IT analytics effort
        - ◊ The IT-DSS/Hadoop cluster will host the batch processing part (at least) of the WLCG Analytics project when in production

- ◊ There are periodic (but rare) meeting to share experimentation on Hadoop (report from exps, etc.), it may be interesting for us too
- ◊ Investigation on Avro where done for last CHEP, to be investigated.

- Uthay Visit@CERN from 25 June to 4 July 2014:
  - ♦ ◊ First version of the Hadoop-based prototype for batch-processing of transfer logs completed. ( WLCGANALYT-1)
  - ◊ Prototype tested on DBA test Hadoop cluster with very good results (1 day ~ 1 minutes, 1 month ~ 15 minutes)
  - ◊ Next ( WLCGANALYT-13) :
    - · Consolidate code , test AVRO
    - · Setup data pipeline ORACLE/HDFS
    - · Schedule regular batch-processing of the data
    - · Deploy a working dashboard UI on the generated data
- Mary's project definition - 3 June 2014: * Investigate on how to use Esper as real-time processing engine to generated statistics on fresh monitoring data
- Next Uthay meeting: 12 September 2014
- Update performance result: 24 September 2014
  - ♦ ◊ Avro optimization + compression cut data size by 4 and processing time by 2, as by the new plots
- Uthay visit: Feb 2015
- Maria join back the team: March 2015

### Presentations

- IT-SDC/MI section meeting - 30 June 2014 - Project's status update and first results of Hadoop prototype
- IT-SDC/MI section meeting - 28 April 2014 - Report on project's objectives and candidate technologies
- IT-SDC Group meeting / Summer students report - 27 August 2014 - Maria's report on integrating Esper for in-memory statistics computation
- IT-SDC/MI section meeting - 13 October 2014 - Quick update on latest performance optimization
- IT DATA ANALYTICS WG - 22 October 2014 - Update on Experiment Dashboard evolution on Hadoop
- White Area leacture - 1 April 2015 - Monitoring WLCG with lambda architecture - CHEP rehearsal

## Reports:

- ReportOnTech_V1.pdf: Technology investigation
- WLCG_Analytics_Platform-XRootD_Monitoring_Use_Case_Results.pdf: WLCG Analytics Platform: XRootD Monitoring Use Case Results using a Hadoop and MapReduce Framework
- mgeorgiouReport.pdf: Report on Esper integration for in-memory analytics

# WLCG Consolidation project

This project profits from the work done in the context of the WLCG Consolidation project, which fostered a loosely-couple architecture for WLCG monitoring applications. The data analytics project focus on data archiving and processing solutions.

- From the WLCG consolidation project, this is an interesting high-level summary of the storage/aggregation use cases WLCG Consolidation analysis
- Hadoop technology overview : https://twiki.cern.ch/twiki/bin/view/LCG/NoSQLStorageResearch

# IT-Agile collaboration

- Discussion with IT-Mon on how to use Hadoop cluster for EOS/XRootd use case - 12th May 2014

# Use cases

## Data transfers - FTS

- Raw data and aggregated statistics for the FTS dashboard: http://dashb-fts-transfers.cern.ch/ui/#date.from=201404010000&date.interval=0&date.to=201404010100
- The zip includes a readme.txt file with some explanation and a link to the Oracle PL/SQL that does the statistics generation
- some_fts_data.zip: some_fts_data.zip

## EOS/XRootD transfers monitoring

### General information

- EOS generate logs for each file access, propagated at the end of the file operation via messaging (stomp?)
  - `filename, source, destination, posix_operation, users, ...`
- Dedicated collectors write logs in Oracle in tables for raw data
- Periodic PL/SQL procedures compute statistics per predefined time bins (10 min and 1 day resolution). Support for more time resolution would be a plus.
  - `source, destination, time_bin, bytes`
  - in the future, more parameter may appear on statistics: (e.g. `file, user`, source, destination, time_bin, size...)
- New computation approach: a transfer is accounted for each bin it spawns upon (uniform distribution (size/#bins))
  - Old approach: transfer accounted only on the arrival bin (still used for the EOS use case cause the new one cannot be implemented for performance issue on Oracle)
- Each access logs may "touch" several bins, requiring re-computation
  - not very frequent in ATLAS, many small transfer on the current bin
  - CMS has bigger file -> longer transfer
  - timeout of 24 hours
- The current architecture is composed by:
  - DB backend (Oracle)
  - web-server (dashboard python code, connecting to DB via custom dao and serving structured JSON via HTTP)
  - JS-UI (dashboard framework) building plots on reading structured JSON via ajax request
- In the current system, filtering and grouping is done on the web-server layer:
  - request like the "transfer plot for inbound connection at BNL for the last 24hrs"
  - translate into `select source=*, dest=*, time=now-24hr` to Oracle
  - the web-server does the filtering (or the UI?)
- Some numbers:
  - Input rate: 20 Hz for CMS, 150Hz with ~ 1KHz spikes (1 day long) for ATLAS
  - Max latency for new data on the UI = 10 minutes
  - #Daily single users= 10/15
  - #Daily requests on the web-server? and on the DB?
  - How long is the history available from UI? (e.g. 3 months, 1 year?)
  - Parameter dimension: source/destination is O(100), O(1000) if considering laptop as sources. user is O(1000). file is potentially millions, but only few are used on the same interval.

- As a example reference, this it FAX dashboard: http://dashb-ai-520.cern.ch/ui/#p.grouping=dst&src.site=(BNL)

### Prototype for /Hadoop based batch-processing

The aim of this prototype is to compute the statistics for EOS/XRootd transfer data with MapReduce jobs, and save it back to a dedicated Oracle table, to be served for standard Dashboard UI visualization. The prototype supports both CSV and AVRO data format. Below technical details and test results.

#### Avro Data Model:

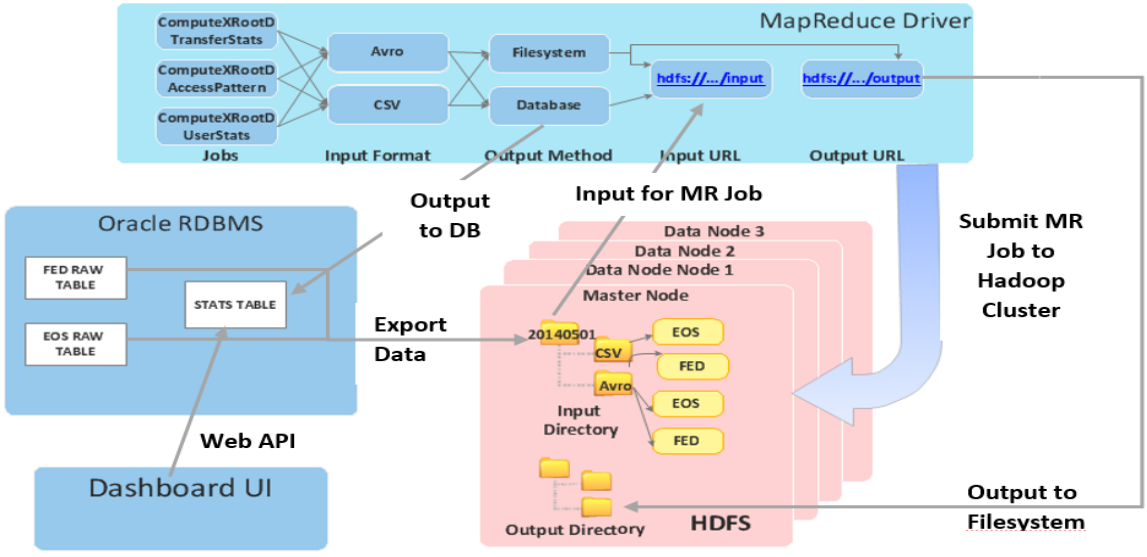
- EOS/FAX - Schema: https://twiki.cern.ch/twiki/pub/LCG/WLCGMonDataAnalytics/eos.avsc
- EOS/FAX - Sample data: https://twiki.cern.ch/twiki/pub/LCG/WLCGMonDataAnalytics/eos-samp.json
- Converted sample data and schema to Avro format : https://twiki.cern.ch/twiki/pub/LCG/WLCGMonDataAnalytics/eos.avro
  - Command used to convert: `java -jar avro-tools-1.7.6.jar fromjson fax_samp.json --schema-file eos.avsc > eos.avro`
- Upload avro data file to HDFS (in order to use with Avro MapReduce job):
  - `hdfs fs -put eos.avro hdfs://us1.brunel.ac.uk/wlcg/eos/20140603/eos.avro`

#### More info

- XRootD monitoring with Hadoop: https://twiki.cern.ch/twiki/bin/view/LCG/XrootdMonitoring#Federation_level_monitoring_AN1
- WLCG Data Transfer monitoring: https://twiki.cern.ch/twiki/bin/view/LCG/WLCGDataTransferMonitoring

### : Overview of Hadoop Architecture

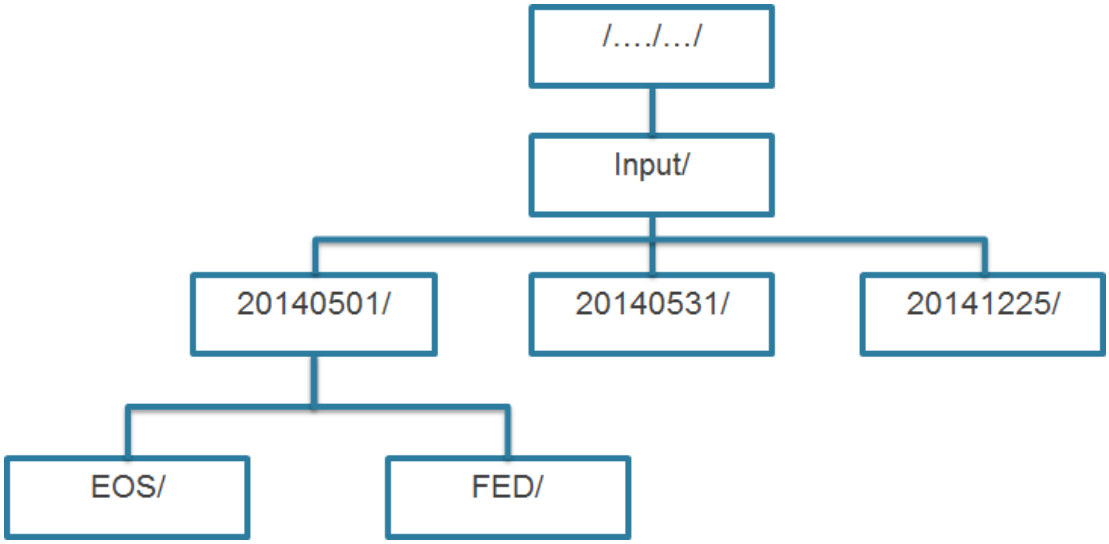
- architecture_v1.png:




**XRootD: Overview of Hadoop Architecture**

### Data structure

- Decided to partition the data by dates.
- Structure:



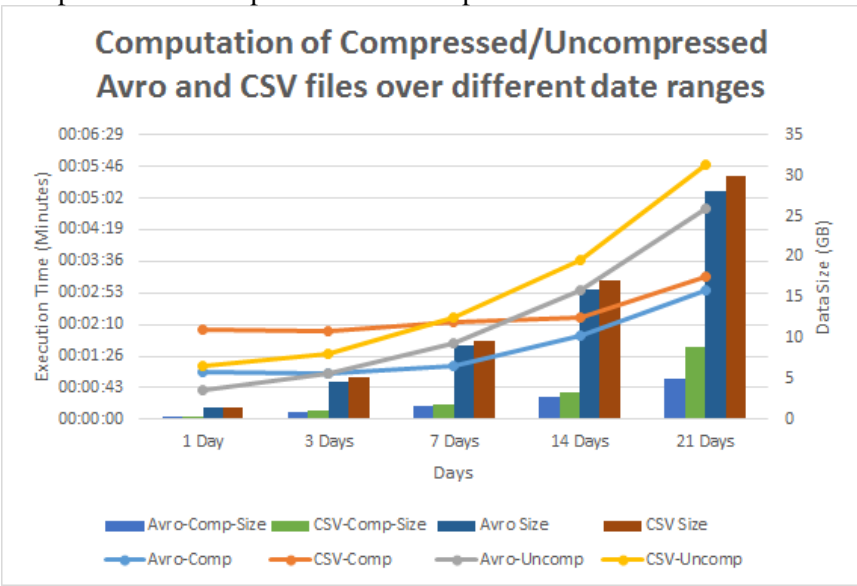

## Data set

- 1 day FAX Data (all data accesses on the ATLAS XRootD federation but EOS):
  - http://dashb-ai-555.cern.ch/ui/FAX-03-MAY.dsv
- 1 day EOS Data accesses
  - http://dashb-ai-555.cern.ch/ui/EOS-ATLAS-03-MAY.dsv

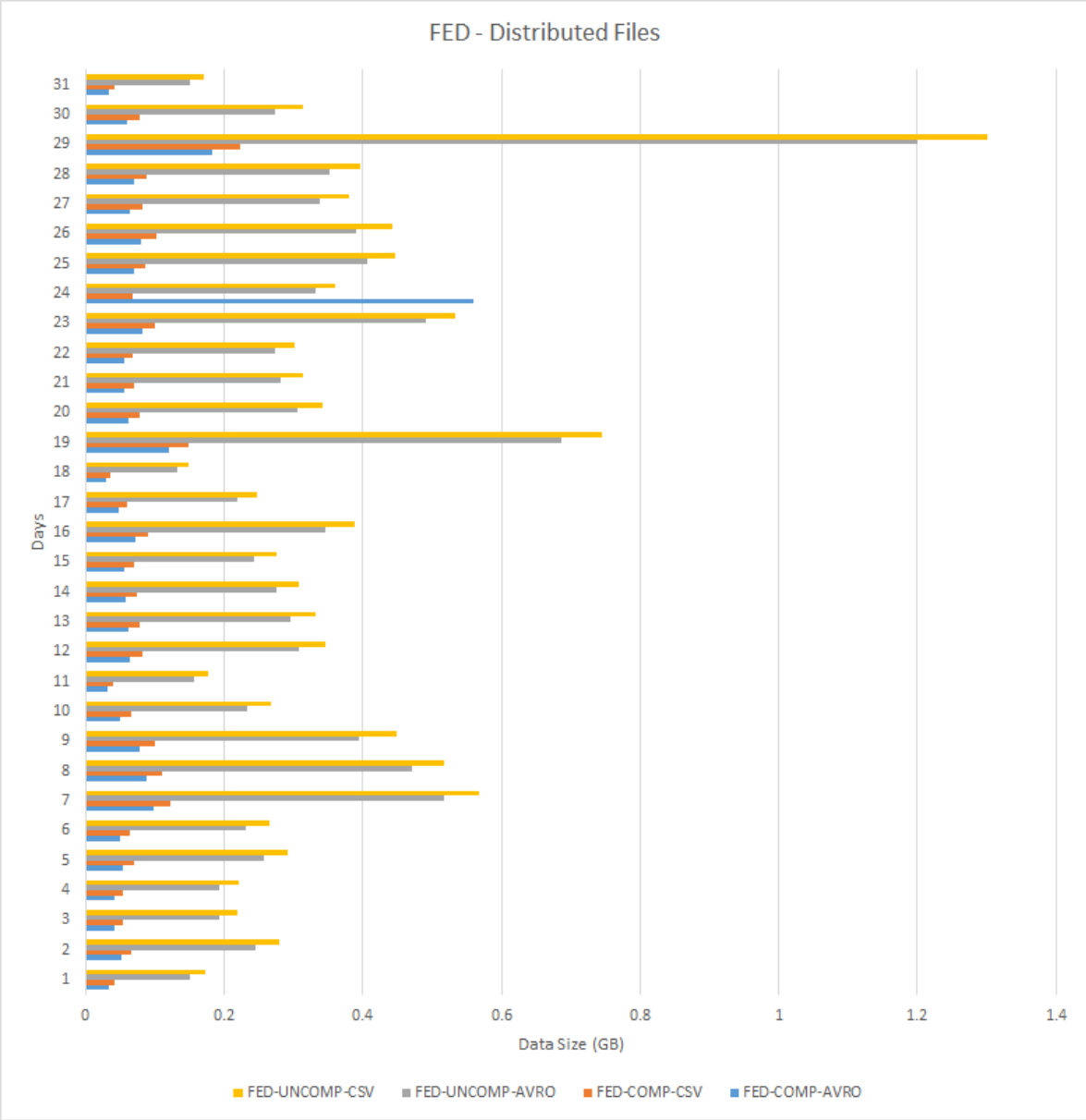
#### Tests and Results:

- Tested datasets:
  - 1 month of historical dataset from both EOS and FED.
  - FED dataset is worth ~11.3GB and EOS is worth ~58.5GB.

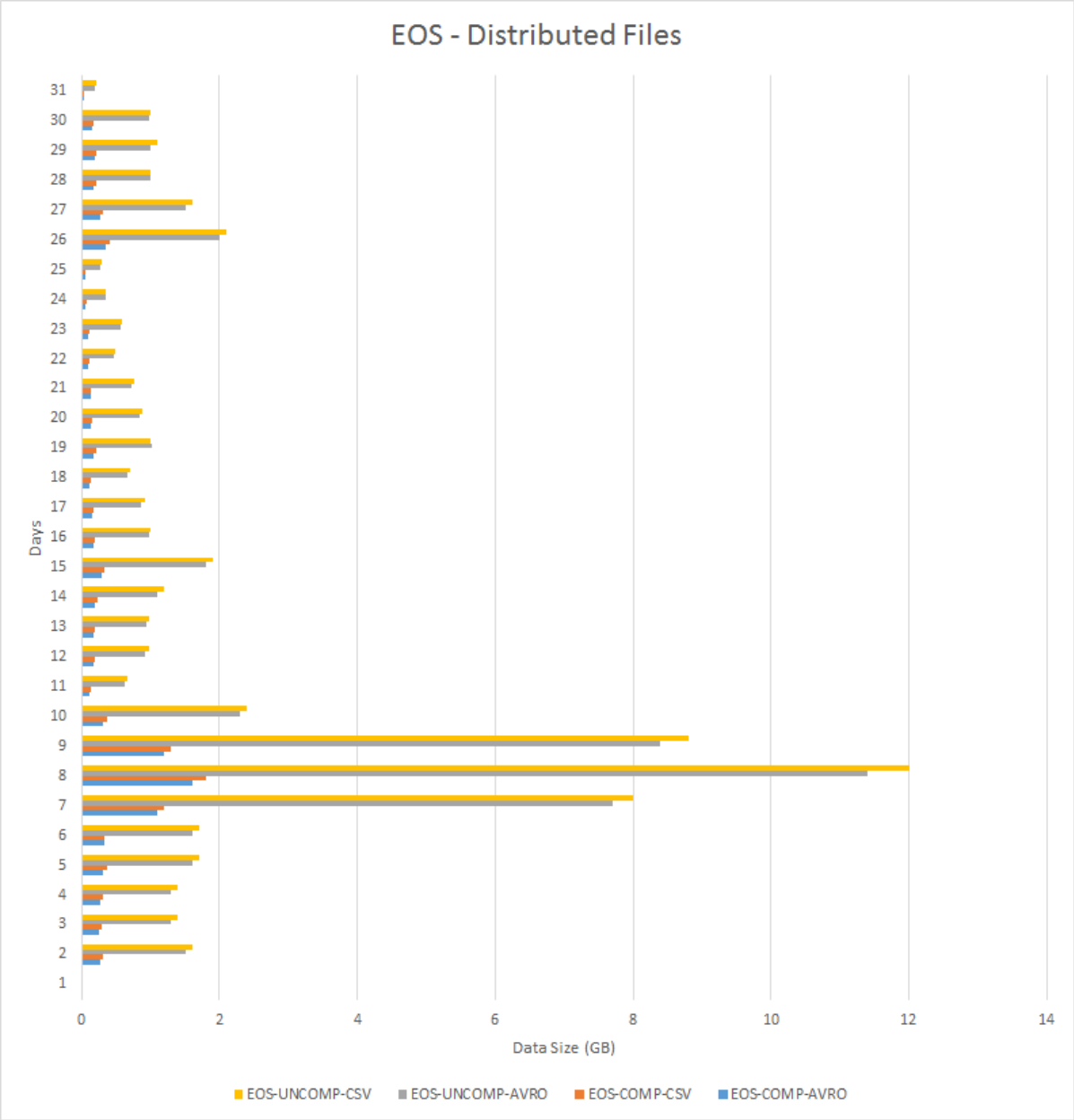
- Computation of Compressed & Uncompressed Avro and CSV files:




- FAX File Distribution:




- EOS File Distribution:




- Results table:

| Day(s) | Avro-Comp (Mins) | CSV-Comp (Mins) | Avro-Uncomp (Mins) | CSV-Uncomp (Mins) | Avro-Comp-Size (GB) | CSV-Comp-Size (GB) | Avro Size (GB) | CSV Size (GB) |
|---|---|---|---|---|---|---|---|---|
| 1 Day | 00:01:04 | 00:02:03 | 00:00:39 | 00:01:14 | 0.26 | 0.3 | 1.4 | 1.50 |
| 3 Days | 00:01:03 | 00:02:01 | 00:01:02 | 00:01:29 | 0.9 | 1.08 | 4.7 | 5.10 |
| 7 Days | 00:01:12 | 00:02:12 | 00:01:45 | 00:02:20 | 1.62 | 1.91 | 8.99 | 9.65 |
| 14 Days | 00:01:55 | 00:02:19 | 00:02:57 | 00:03:38 | 2.83 | 3.35 | 15.89 | 17.00 |
| 21 Days | 00:02:56 | 00:03:15 | 00:04:47 | 00:05:48 | 4.95 | 8.82 | 28.07 | 29.92 |

# EOS/FAX computation on Hadoop

## IT-DSS Hadoop cluster

Login on: p01001532975913.cern.ch or lxfsrk49a05.cern.ch

### HDFS file structure

```
-bash-4.1$  kinit wdtmon
-bash-4.1$  hdfs  dfs -ls /user/wdtmon/atlas_xrd_mon
Found 2 items
drwxr-xr-x   - wdtmon supergroup          0 2014-10-30 14:02 /user/wdtmon/atlas_xrd_mon/input
drwxr-xr-x   - wdtmon supergroup          0 2014-11-06 15:40 /user/wdtmon/atlas_xrd_mon/output
```

### Monitoring

- Job monitoring: http://p01001532965510.cern.ch:8088/cluster
- HDFS status: http://p01001532965510.cern.ch:50070/
- Ganglia http://p01001532965510.cern.ch/ganglia/?m=load_one&r=day&s=descending&c=p01001532965510.cern.ch&
- Meter: https://meter.cern.ch/public/_plugin/kibana/#/dashboard/elasticsearch/Metrics:%20Host

### Access

**wdtmon** service account, used to control data access and MR jobs

## dashb-ai-617

### Statistics Job

#### Cron

```
[root@dashb-ai-617 dirq2hdfs_consumer]# crontab -l | grep mr
#*/15 * * * * /root/dirq2hdfs_consumer/mr_trans_stats_10mins.sh
#*/10 * * * * /root/dirq2hdfs_consumer/mr_access_pattern_10mins.sh
#*/10 * * * * /root/dirq2hdfs_consumer/mr_user_stats_10mins.sh
```

#### Logs

```
[root@dashb-ai-617 ~] cd /root/dirq2hdfs_consumer/
[root@dashb-ai-617 dirq2hdfs_consumer] ./hadoop-2.3.0/bin/yarn logs -applicationId  > mr.log
```

### Map Reduce Jobs options:

```
Usage: [main class] [options]

  Options:
  * -db
       Enter the db config path
  * -input
       Enter an input path.
  * -inputformat
       Enter an input Format
    -output
       Enter an output path.
  * -outputformat
       Enter an output Format
  * -processdays
       Process data from ...
       Default: 3
  * -topology
       Enter the topology file path
  * -vo
       Enter the virtual organisation
```

e.g.

```
yarn jar /root/dirq2hdfs_consumer/xrootd-06112014.jar com.cern.dashboard.xrootd.TransferStats -to
```

### writer

The Dirq2HdfsDaemon has been configured to run every 10 minutes.

The daemon script is located under:

/root/dirq2hdfs_consumer/dirq2hdfsdae

To start|stop EOS, FAX_US and FAX_EU daemons:

```
./dirq2hdfsdae start|stop eos
./dirq2hdfsdae start|stop fax_us
./dirq2hdfsdae start|stop fax_eu
```

Configuration files are located in same Dir:

```
[root@dashb-ai-617 ]# ls -ltr *config*
-rwxrwxrwx. 1 root root 347 Oct 29 17:30 eos.config.properties
-rwxr-xr-x. 1 root root 353 Oct 29 17:31 fax_us.config.properties
-rwxr-xr-x. 1 root root 353 Oct 29 17:32 fax_eu.config.properties
```

# Hadoop Test Cluster:

### Nodes:

- itrac911.cern.ch
- itrac912.cern.ch
- itrac925.cern.ch
- itrac926.cern.ch

### The current setup is:

- itrac925 - Mater node with majority of roles: datanode + httpfs + namenode + nfs gateway + hive + impala + sqoop + yarn (MR).
- itrac911 - Datanode has impala and oozie
- itrac912 - same as itrac911
- itrac926 is similar to itrac911 & itrac912 but with secondary namenode too.

All members of wlcg-data-analytics e-group will have access to above cluster.

Use NICE accounts to log in.

# PL/SQL Procedure jobs for transfer analysis

Show... Hide

## computeXrootdTransferStatsAvg

### Steps:

This procedure calculates a group of statistics every 10 minutes for a given stream of data. After it calculates those statistics it groups them by some properties which will be described below. The steps of the procedure are as follows:

1. Selects from the log messages:
    - write_bytes_at_close
    - read_bytes_at_close
2. Check:
    - If **writes_bytes_at_close>0** then we have a **client_domain** else we have a **serve_domain** and set it as **src_domain**
    - If **read_bytes_at_close>0** then we have a **server_domain** else we have a **client_domain** and set it as **dst_domain**
        - ◊ if **client_domain=server_domain** set **remote_access** 0 else set is as 1.
    - if **writes_bytes_at_close+read_bytes_at_close=file_size** set **is_transfer=1** else **is_transfer=0**.
    - if **read_bytes_at_close>0** then set **activity='r'**
    - if **write_bytes_at_close>0** then set **activity='w'**
    - if **write_bytes_at_close<=0** and **read_bytes_at_close=<0** then set **activity='u'**
3. Set:
    - **write_bytes_at_close+read_bytes_at_close** as **bytes**
    - the time that a transaction started as **start_time**
    - the time that a transaction ended as **end_time**
    - set **total_time = end_time-start_time**
4. Compute:
    - if **total_time<>0** then **active_time*bytes/total_time** (bytes that are transferred in each bin)
    - count of filelfn and set it as **active**
    - sum the finished transactions and set it as **finished**
    - sum the total amount of bytes and set it as **bytes**
    - sum the active_time of a transaction and set it as **active_time**
5. Group the above by :
    - src_domain
    - dst_domain
    - user_protocol
    - is_remote_access
    - is_transfer
6. Merge the following into **t_stats_avg** table
    - src_domain
    - dst_domain
    - user_protocol
    - is_remote_access
    - is_transfer
    - activity
    - period_end_time
    - active
    - bytes
    - active_time

## computeEOSStatsAvg

### Steps:

This procedure follows the same steps as the previous one (computeXrootdTransferStatsAvg) with the only difference that the log messages come from the **t_raw_eos** table which concerns CERN data storage.

## computeXrootdTransferStats

### Steps:

This procedure calculates a group of statistics every 10 minutes for a given stream of data. After it calculates those statistics it groups them by some properties which will be described below. The steps of the procedure are as follows:

1. Selects from the table t_raw_fed:
    - src_domain
    - dst_domain
    - user_protocol
    - If **writes_bytes_at_close>0** then we have a **client_domain** else we have a **serve_domain** and set it as **src_domain**
    - If **read_bytes_at_close>0** then we have a **server_domain** else we have a **client_domain** and set it as **dst_domain*** if **client_domain=server_domain** set **remote_access** 0 else set is as 1.
    - if **writes_bytes_at_close+read_bytes_at_close=file_size** set **is_transfer=1** else **is_transfer=0**.
    - if **read_bytes_at_close>0** then set **activity='r'**
    - if **write_bytes_at_close>0** then set **activity='w'**
2. Select:
    - src_domain
    - dst_domain
    - user_protocol
    - is_remote_access
    - is_transfer
    - activity
    - sum(write_bytes_at_close) as bytes
3. Group them by:
    - src_domain
    - dst_domain
    - user_protocol
    - is_remote_access
    - is_transfer
    - activity
4. Merge the above following into **t_Stats_** table

### Steps:

This procedure calculates a group of statistics for each day given a group of statistics calculated every 10 minutes. After it calculates those statistics it groups them by some properties which will be described below. The steps of the procedure are as follows:

1. Selects from the t_stats table:
    - src_domain
    - dst_domain
    - user_protocol
    - is_remote_access
    - is_transfer
    - activity
    - beginning of the day that this transaction begun as period_end_time
    - sum of bytes as bytes
    - sum of finished as finished
2. Group the above by :
    - src_domain

- dst_domain
- user_protocol
- is_remote_access
- is_transfer
- is_remote_access
- activity
- period_end_time

3. Merge them into t_stats_a

### Steps:

This procedure calculates a group of statistics every 10 minutes for a given stream of data. . After it calculates those statistics it groups them by some properties which will be described below. The steps of the procedure are as follows:

1. Selects from the t_raw_fed table:
   - client_domain
   - server_domain
   - user_protocol
   - read_bytes_at_close
   - write_bytes_at_close
   - if (client_domain==server_domain) then is_remote_access=1 else 0
   - if (read_bytes_at_close+write_bytes_at_close=file_size) then is_transfer=1 else 0
2. :
   - if user_dn==null then replace it with server_username. In case that server_username is also null replace it with "n/a"
   - if file_lfn==null then replace it with "n/a"
   - AVG(file_size) as file_size
   - if (read_bytes_at_close>0) then number_of_read=1 else is 0
   - sum(read_bytes_at_close) as bytes
   - if (read_bytes_at_close>0) then sum(end_time-start_time) as read_time else read_time=0
   - if (write_bytes_at_close>0) then number_of_write=1 else is 0
   - sum(write_bytes_at_close) as bytes
   - if (write_bytes_at_close>0) then sum(end_time-start_time) as write_time else read_time=0
3. Group the above by :
   - client_domain
   - server_domain
   - user_protocol
4. Merge them into t_stats_access_pattern_a

### Steps:

This procedure calculates a group of statistics every 10 minutes for a given stream of data. . After it calculates those statistics it groups them by some properties which will be described below. The steps of the procedure are as follows:

1. Selects from the t_raw_fed table:
   - user_protocol
   - if (client_domain==server_domain) then is_remote_access=1 else 0
   - if (read_bytes_at_close+write_bytes_at_close=file_size) then is_transfer=1 else 0
   - if user_dn==null then replace it with server_username. In case that server_username is also null replace it with "n/a"

    - read_single_bytes
    - read_vector_bytes
    - File_size
2. :Select
    - if user_dn
    - user_protocol
    - is_remote_access
    - is_transfer
    - sum(read_single_bytes) as read_single_bytes
    - sum(read_vector_bytes) as read_vector_bytes
    - sum(file_size) as aggregated_filesize
3. Group the above by :
    - user_dn
    - user_protocol
    - is_remote_access
    - is_transfer
4. Merge them into T_USER_ACTIVITY

- mgeorgiouReport.pdf: Report on Esper integration for in-memory analytics

# Log ingestion from Message Queues to HDFS using Apache Flume

Apache Flume is designed for distributed and fault tolerant service for collecting a large amount of data and moving them to HDFS An agent within Apache Flume is responsible for data (event) flow from external to designated destination, which consist of three components; Source, Channel and Sink:

- Source provides interface for consuming events from external entities through various protocols of data transmission. It supports Avro, log4j, syslog, HTTP Post and JSON for data transmission.
- Channel stores those events received by the source into one or more channels (temporally) until it is permanently stored into Sink. Channel could be local Filesystem or memory.
- Sink consume events from the channel(s) and store them into external storage such as HDFS, Hive, etc Or hop to next agent.

In XrootD use case, source could consume logs from message queues and store them into HDFS as big files as HDFS is better at serving a small number of large files, rather than a large number of small files. There are three ways to close current file and create a new one periodically based on the elapsed time or size of data or number of events, which can be achieved at the sink configuration:

- `hdfs.rollCount` close the current file and create a new file based on number of events (0 is used to ignore this option).
- `hdfs.rollSize` close the current file and create a new file based on the data size (0 is used to ignore this option).
- `hdfs.rollInterval` - close the current file and create a new file based on elapsed time (0 is used to ignore this option).

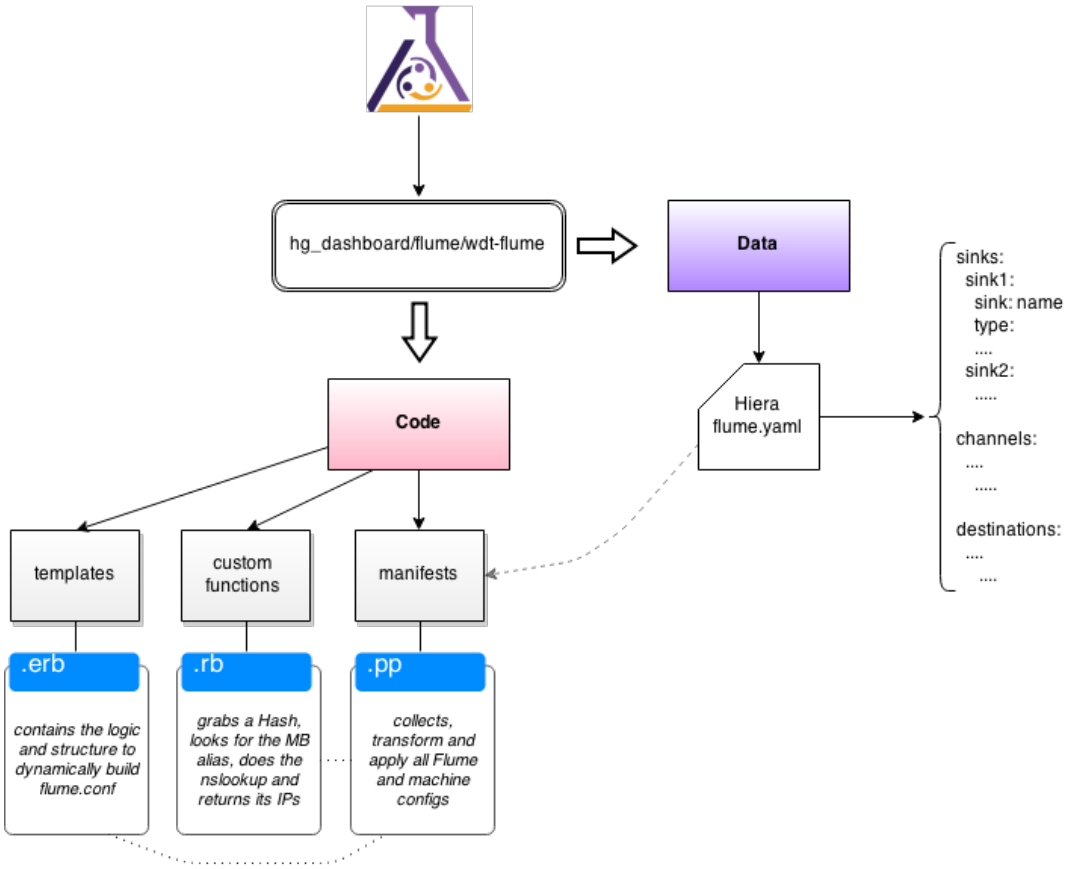
## Using Flume

To facilitate and automatize the deployment and configuration of a Flume machine/instance, some Puppet procedures were written under the **dashboard** hostgroup at https://git.cern.ch/web/it-puppet-hostgroup-dashboard.git.

Steps:

To deploy a new Flume instance one should create a puppetized machine that point to the *dashboard/flume/wdt_flume* hostgroup.

The way the deployment works internally is summarized in the following image:




Basically, in order to fully configure Flume, one must only build their own data structure on Hiera and refer to it on Foreman, which by steps means:

1. With the right accesses, clone https://git.cern.ch/web/it-puppet-hostgroup-dashboard.git and go to *data/hostgroup/dashboard* (under the right environment - master or qa)
2. Open **flume.yaml** (or create another YAML file...it shouldn't matter, as long as the file is on the directory specified in **1.**
    1. Inside one shall find/create a structure like the following

```
channels_key:
  channel1_doesntMatterTheName:
    channel: c1
    type: memory
    capacity: 10000
    transactionCapacity: 10000
    byteCapacity: 200000000
  channelWhatever:
    channel: c2
    type: memory
    capacity: 10000
    transactionCapacity: 10000
    byteCapacity: 200000000
sinks_key:
  sink1:
    sink: k1
    type: hdfs
    channel: c1
```

```
    path: "hdfs://p01001532965510.cern.ch:9000/user/wdtmon/xrootd/atlas/eos/%y-%m-%d
    kerberosPrincipal: wdtmon@CERN.CH
    kerberosKeytab: /etc/wdtmon.keytab
    filePrefix: data.%Y-%m-%d.%H
    rollInterval: 3600
    rollCount: 0
    rollSize: 0
    batchSize: 5000
  sink2:
    sink: k2
    type: hdfs
    channel: c2
    path: "hdfs://p01001532965510.cern.ch:9000/user/wdtmon/xrootd/atlas/fax/%y-%m-%d
    kerberosPrincipal: wdtmon@CERN.CH
    kerberosKeytab: /etc/wdtmon.keytab
    filePrefix: data.%Y-%m-%d.%H
    rollInterval: 3600
    rollCount: 0
    rollSize: 0
    batchSize: 5000
destinations_key:
  Eos:
    destinationName: xrootd.atlas.eos
    channels: c1
    providerURL: dashb-mb.cern.ch
    brokerPort: 61113
    destinationType: TOPIC
    userName: wdtmon
    batchSize: 100
    passwordFile: /opt/flume/amq_pass
  faxUs:
    destinationName: xrootd.atlas.fax.us
    channels: c2
    providerURL: dashb-mb.cern.ch
    brokerPort: 61113
    destinationType: TOPIC
    userName: wdtmon
    batchSize: 100
    passwordFile: /opt/flume/amq_pass
```

When using this structure, one should pay attention to the following:

- ◊ The ***_key** names that define the Hash should be unique in Hiera, otherwise Puppet might have problems while fetching this data. By default these Hash keys are named to **sinks**, **channels** and **destinations**. Please don't overwrite those.
- ◊ The *subkeys* names of the Hash don't really matter as they are just used to differentiate the differentiate the nested Hashes
- ◊ the *providerURL* is the alias of the Messabe brokers, and should always be passed as an hostname. Puppet will later *nslookup* that hostname and replace it with the corresponding IP addresses

2. Save the YAML file and push it to Git
3. Go on Foreman, and submit the following variables: **hiera_get_sinks** pointing to the sinks_key from above; **hiera_get_channels** pointing to the channels_key; **hiera_get_destinations** pointing to the destinations_key.

3. Boot a puppetized VM from the hostgroup specified in 1)

---

This topic: LCG > WLCGMonDataAnalytics
Topic revision: r59 - 2015-05-01 - UthayanathSuthakar