Lustre Monitoring

DataDirect Networks Japan, Inc.

Shilong Wang, Shuichi Ihara

2016/10/8

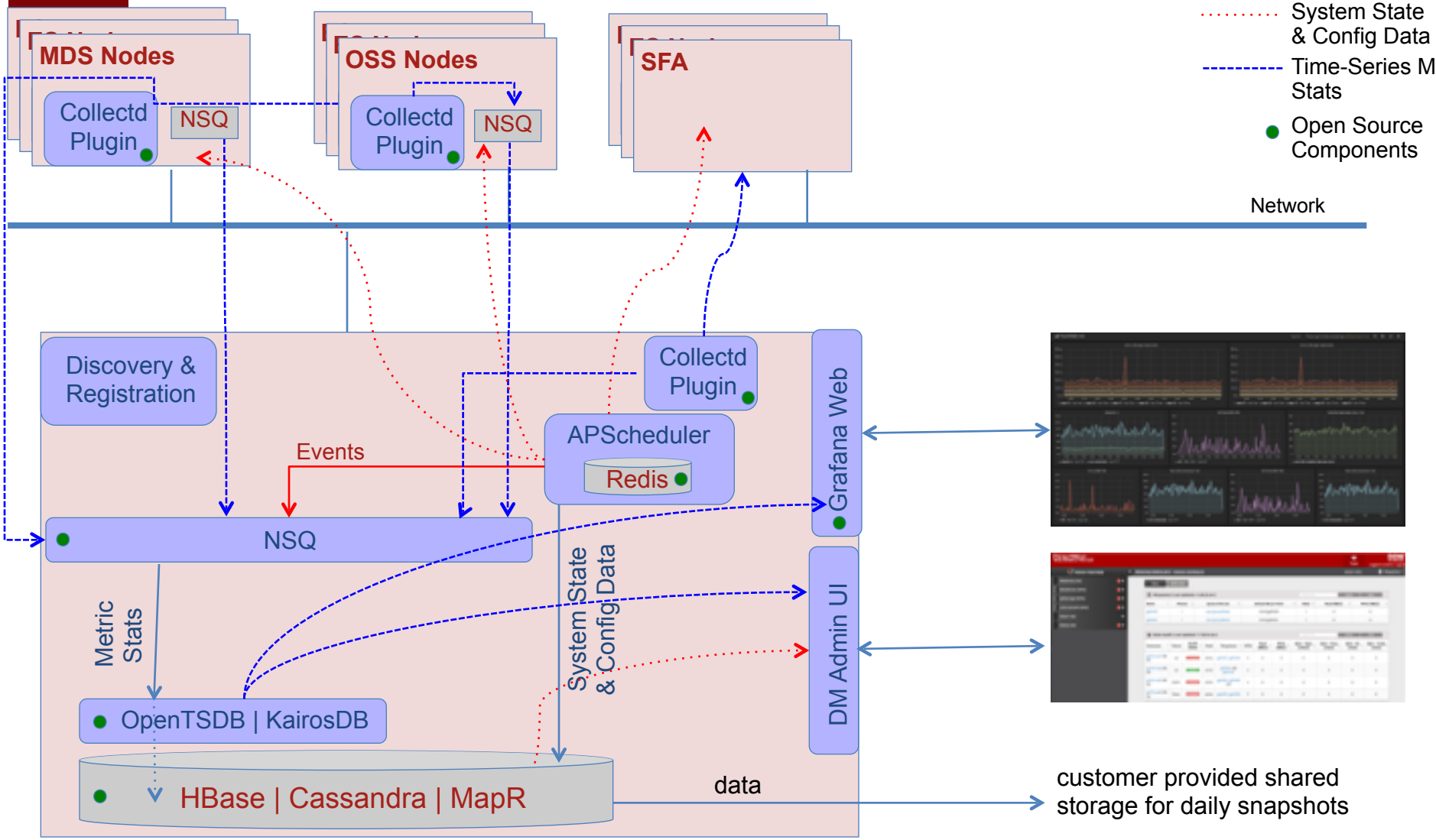
Background of Lustre monitoring

- **Activities on the Lustre are black box**
  - Users and Administrators want to know “what’s going on?”
  - Find “Crazy Jobs” in advance to prevent slow down.
- **Lustre statistics are valuable big data**
  - Not only monitoring and visualization, but also analysis
  - Predictable operations could be possible.
  - It helps optimize applications and data relocation.
- **Open Source based monitoring tool**
  - In general, open source is common in the HPC system and it’s straightforward.
  - Various combination is possible and make new use cases.

C/S monitoring

- Collecting Data from monitor target, usually it could be MDS/MGS, OSS, client.
- Sending collected data to persistent Storage.
- Collected data could be reviewed by Users friendly.(Time series, Rates etc.)

Standalone Configuration



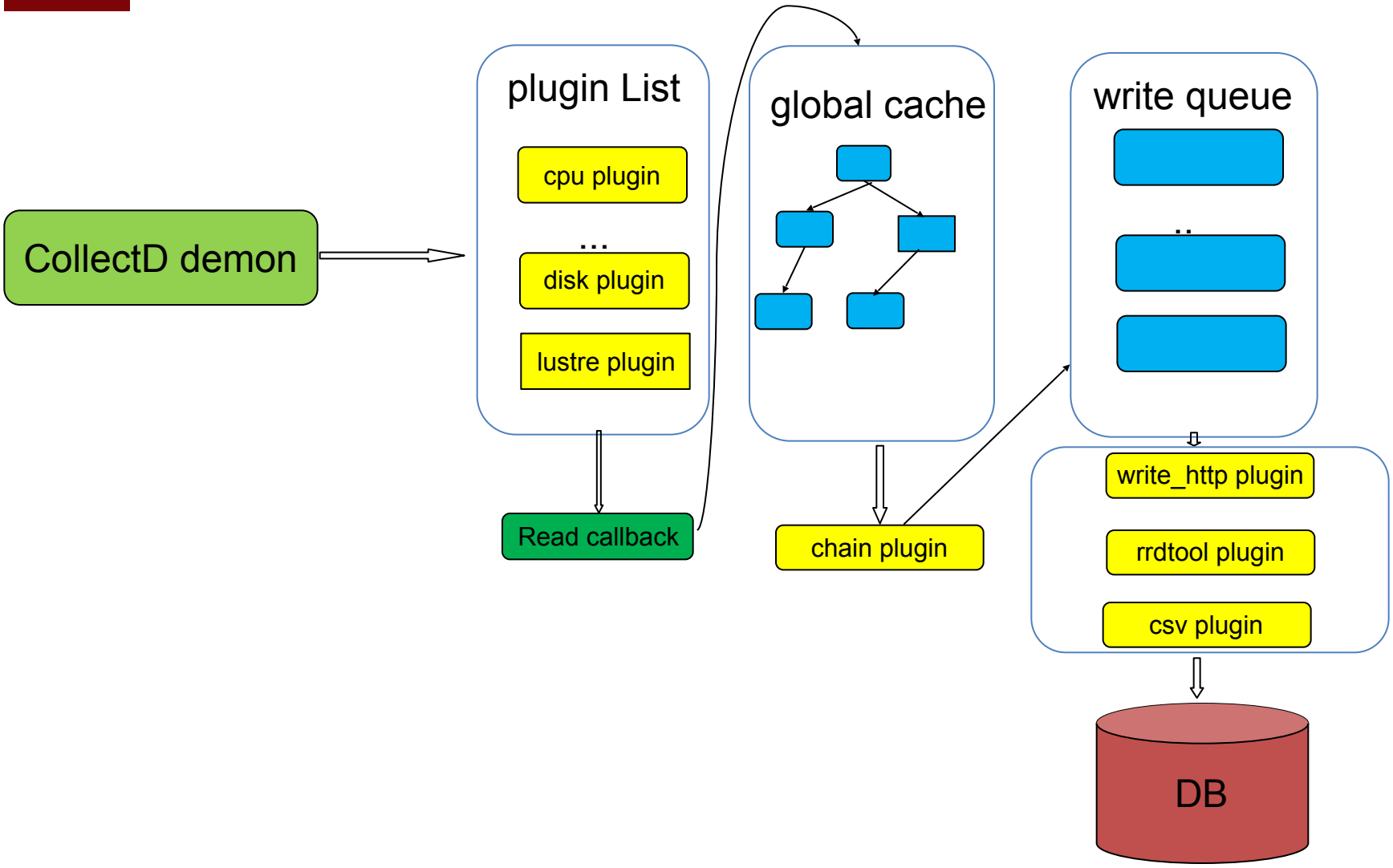
Components of Lustre monitoring

- **Data collecter**
  - Collects statisticsfrom Lustre /proc and sends them to monitoring server over the network.
  - Runs on servers as well as client and routers.
- **Backend Storage**
  - Receive stats from agents and store them into database.
  - It can be historical and query-able data
- **Frontend**
  - Collected data is not only visualized, but also analytics.
  - Application I/O analytics and

Flexible data collector

- **A lot of agents existed to collect Lustre performance statistics**
- **Collectd is one of reasonable options**
  - **Actively developed, supported and documented**
  - **Running at many Enterprise/HPC system**
  - **Written in C and over 100 plugins are available.**
  - **Supports many backend database for data store.**
  - **Unfortunately, Lustre plugin is not available, but we made it!**
  - **Publish the lustre-plugin codes? Yes, we want, but there were several discussions at LUG15. (e.g. stats in /proc or /sys in the future?)**

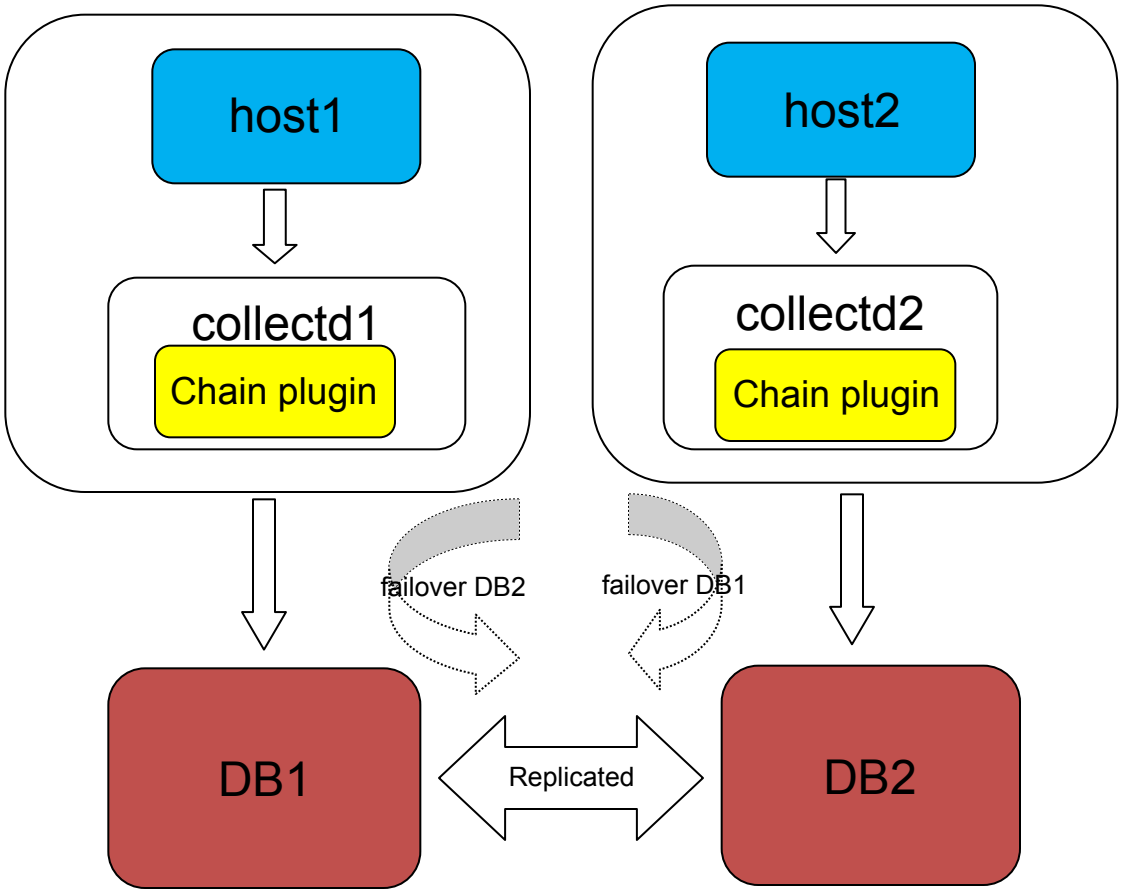
Architecture of Collectd



Scalable backend data store

- **RDD and SQL based data store dose not scale**
  - RDD works well on small system, writing 10M statics into files are very challenging (few million IOPS!)
  - SQL is faster than RDD, but still hit next level scalability. And it's complex to make database deign.
- **NoSQL based key-value store shines**
  - OpenTSDB/Hbase. KairosDB/Cassandra
  - key, value and tags are easy adaption for Lustre statics data store. No need complex database schema.
  - Need to be aware of managing for statics data archiving. (retention)

Loadbalance on DB Cluster



Frontend

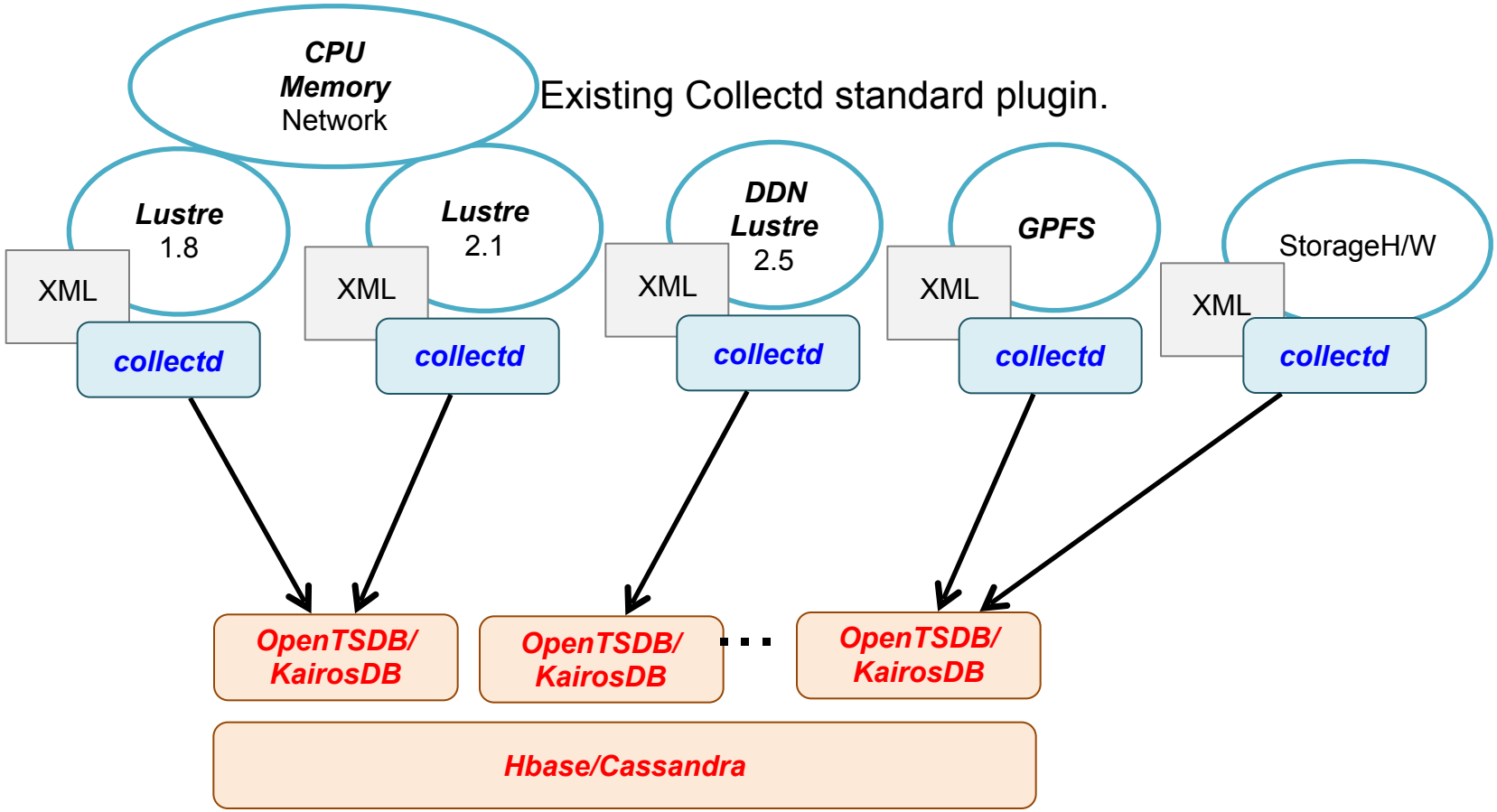
➢ Grafana

- **Grafana is an open source, feature rich metrics dashboard and graph editor for Graphite, Elasticsearch, OpenTSDB, Prometheus and InfluxDB.**

A deign of lustre-plugin in collectd

- **A framework consists of two core components**
  - Core logic layer (Lustre plugin)
  - Statistics definition layer(XML file and XML parser)
- **Defined XML for Lustre /proc information**
  - A single XML file for all definitions of Lustre data collection
  - No need to maintain massive error-prone scripts.
  - Extendable without core logic layer change.
  - Easy to support multiple Lustre version and Lustre distributions in the same cluster.

Architecture of lustre-plugin and configuration



An demonstration of 1.7 Trillion data point store in 24 hours

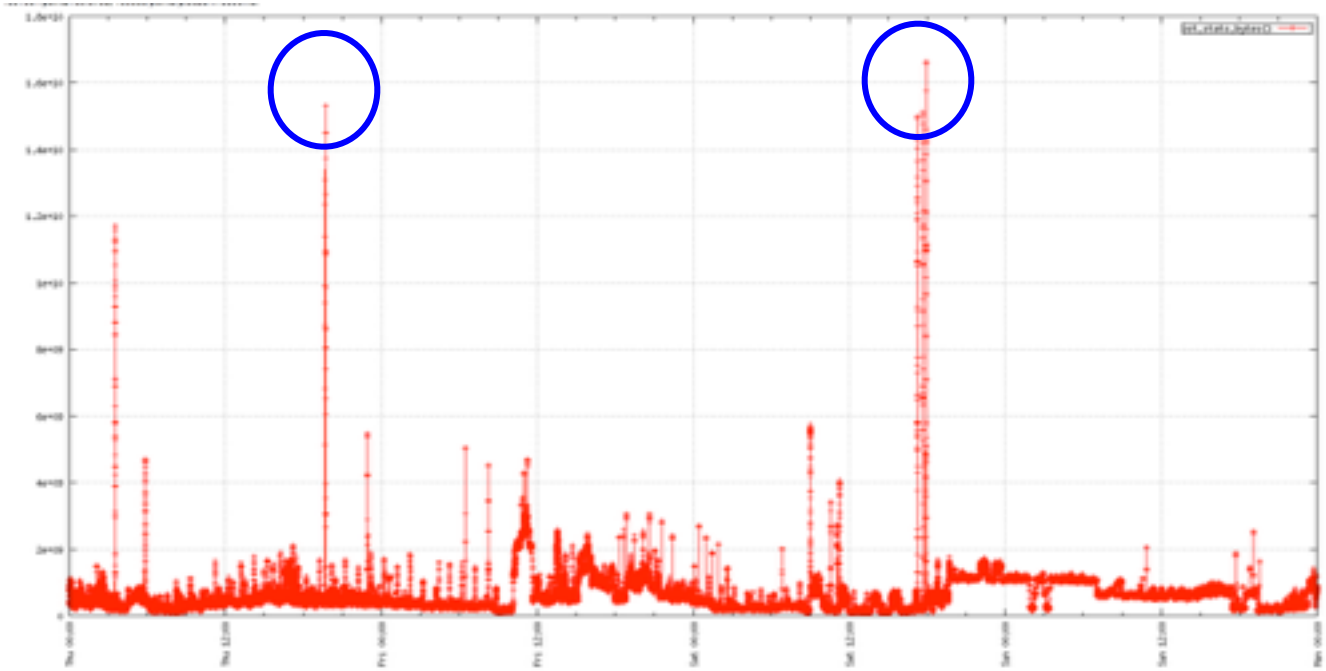
- **Developed "Stress" plugin for collectd**
  - Generating "dummy" stats with collectd
  - For regression tests and benchmark tool of Lustre monitoring.
  - It works conjunctions with other collectd plugins.
- **Stress test on Lustre-plugin and OpenTSDB**
  - Setup two nodes hadoop/hbase cluster and OpenTSDB runs top of it.
  - 16 metrics generators(servers). Total 20M stats generated every second and send two OpenTSDB servers.
  - Passed 24 hours stress test and stored 1.7 Trillion stats without any problems.

Application aware I/O monitoring

- **Scalable backend data store**
  - Now, we have scalable backend data store OpenTSDB.
  - Store any type of mercies whatever we want to collect.
- **Lustre Job stats is awesome, but need to be integration.**
  - Lustre JOB stats feature is useful, but administrator is not interested in I/O stats just only based on JOBID. (Array jobs. Job associates with another jobs, e.g. Genmic pipeline)
  - Lustre performance stats should be associated with all JOBID/GID/UID/NID or custom any IDs.

What Jobs Caused burst I/O?

- What happened here?
- Who(Users or Group) or What jobs caused burst I/O?
- If it's not single job and single user. What is top10 user/ group and job of active Lustre I/O?



TopN Query with OpenTSDB

- **Implemented TopN Query based on OpenTSDB**

```
# topn -m mdt_jobstats_samples -r 1434973270 -s -t slurm_job_uid
Time(ms): 2015-06-22 20:41:25.000000        Interval: 5(s)
    rate |  fs_name | mdt_index | slurm_job_uid |  fqdn | slurm_job_id | slurm_job_gid |
15264.00 | scratch1 |   MDT0000 |          1044 | mds06 |        13290 |          1044 |
15076.80 | scratch1 |   MDT0000 |          1045 | mds06 |        13286 |          1045 |
13812.40 | scratch1 |   MDT0000 |          1049 | mds06 |            - |          1049 |
13456.80 | scratch1 |   MDT0000 |          1048 | mds06 |            - |          1048 |
 9180.80 | scratch1 |   MDT0000 |          1050 | mds06 |        13285 |          1050 |
 8909.40 | scratch1 |   MDT0000 |          1047 | mds06 |        13289 |          1047 |
 8779.60 | scratch1 |   MDT0000 |           502 | mds06 |            - |           503 |
 5049.00 | scratch1 |   MDT0000 |           501 | mds06 |        13291 |           502 |
```

- **Not only monitoring, but also it's diagnostic tool!**
  - You can issue query on live and specific time period. (Rewind feature)
  - Lustre job stats associated with not only single ID, but all UID/GID/ JOBID/NID (or custom ID) into OpenTSDB.

Iflyteck(KairosDB)

- 1PB Lustre Filesystem(2 OSS, 1MDS, 1MGS)
- They are using Lustre very heavy, and need monitor IO throughout, CPU, Memory usage for Servers
- Inodes total, free?
- OST space usaed and how fast it grow?
- What is MDS IOPS?
- What is Filesystem throughout? per OSS/OST?

Examples
DDN monitoring in Iflytech(1)

Examples
DDN monitoring in Iflytech(2)

Tokyo Institute of Technology

- **More than 1.2M stats into single monitoring server every 30 sec**
  - **16 Lustre servers, 120 OSTs for 3 lustre filesystems**
  - **1700 clients mount all 3 filesystems**
  - **Lustre-2.1 is running. No jobstats! But collecting client based stats from “export” directory in /proc on Lusre servers.**
    (Total stats = #OST * #Client * #metrics)
  - **Demonstrated more than half Trillion stats stored into opentsdb for 6 months.**

- **3PB Lustre filesystem (12 x OSS, 400 x client)**
- **Lustre jobstats integrated with SLRUM and running on the production system**
  - **Unique Lustre Jobstats configuration with Collectd Lustre plugin and runs on existing on Jobstats framework.**
  - **Collect jobs stats associated with all UID/GID/JOBID and store them into OpenTSDB.**
  - **It helps to find a root cause so quickly when unexpected burst I/O happen. Who and what jobs caused problem.**

Conclusions

- Developed new lustre plugin of collected. It's flexible, extendable and easy maintainable.
- Deigned a Lustre monitoring framework based on lustre-plugin and OpenTSDB. The framework is running on several production systems to solve today's lustre monitoring limitation.
- Demonstrated 1.7 Trillion data store into OpenTSDB in 24 hours. We will continue scalable testing for multi-Trillion data store in few hours.
- Started data analysis

Thank you!