Log Anomaly Detection

Ladislav Ondris

Sep 2022

FTS feeds

Information we might find useful in the process of detection:

Endpoints (source, destination)
Timestamp
Message
State
Facility
Queue time - can be computed from **submit_time**, **staging_start** and **staging_finished** in complete
Transfer time - not available in raw
Latency - not available in raw
Transfer size

TransferComplete Feed (monit_prod_fts_enr_complete)

The feed contains additional interesting information as opposed to the raw feed**:**

- country,
- experiment_site, src_site, dst_site,
- t_error_message (**similar** to data.reason),
- failure_phase,
- tr_timestamp_start, tr_timestamp_complete

Did the previous pipeline use raw or completed logs?

Apparently, it uses latency, which is not available in the raw feed.

Code might tell us more**.**

Optimizer regarding FTS feeds

OptimizerMessage feed: a message is sent each time the Optimizer changes the parameters for a given src/dst pair.

It is probably the component responsible for adaptive optimization of transfers, which dynamically adjusts the number of concurrent file transfers.

FTS feeds and transfer’s lifecycle

The log message is sent to specific feeds depending on when the failure happened in the transfer’s lifecycle

The following feeds receive a log message:

ACTIVE -> FAILED

- TransferState
- TransferComplete

STAGING -> FAILED

- TransferState

Processing and enrichment of logs (*raw -> completed)*

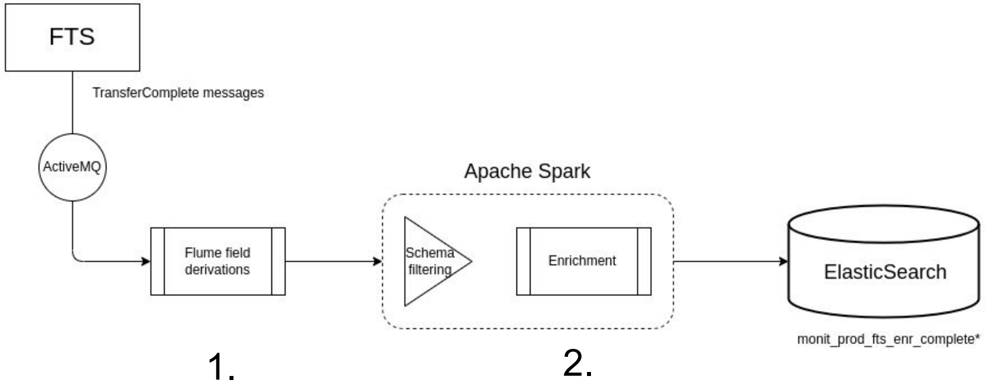
- monit_prod_fts_**enr_complete** corresponds to the **enriched TransferComplete** feed (there is no monit_prod_fts_**raw_complete**)
- monit_prod_fts_**raw_state** corresponds to the **TransferState** feed.

Two places where enrichment happens:

1. Attribute derivations from the same message
2. Enrichment by merging with information from CRIC

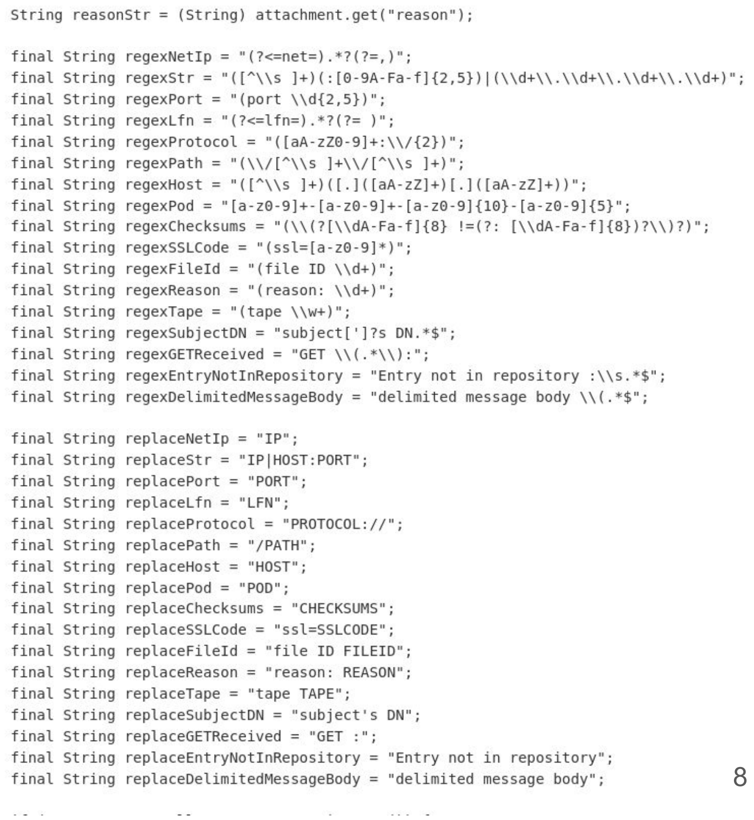
Can be some of it useful?

Depends on whether we use raw or enriched FTS feed.



Processing of raw log message

- Known entities in the string are replaced using regular expressions, creating purged_reason attribute
- The complete feed does not contain the attribute. **What happened to it?**
- data.t_error_message is probably almost the same as data.reason



```java
String reasonStr = (String) attachment.get("reason");

final String regexNetIp = "(?<=net=).*?(?=,)";
final String regexStr = "([^\\s ]+)(:[0-9A-Fa-f]{2,5})|(\\d+\\.\\d+\\.\\d+\\.\\d+)";
final String regexPort = "(port \\d{2,5})";
final String regexLfn = "(?<=lfn=).*?(?= )";
final String regexProtocol = "([aA-zZ0-9]+:\\/{2})";
final String regexPath = "(\\/[^\\s ]+\\/[^\\s ]+)";
final String regexHost = "([^\\s ]+)([.]([aA-zZ]+)[.]([aA-zZ]+))";
final String regexPod = "[a-z0-9]+-[a-z0-9]+-[a-z0-9]{10}-[a-z0-9]{5}";
final String regexChecksums = "(\\(?[\\dA-Fa-f]{8} !=(?: [\\dA-Fa-f]{8})?\\)?)";
final String regexSSLCode = "(ssl=[a-z0-9]*)";
final String regexFileId = "(file ID \\d+)";
final String regexReason = "(reason: \\d+)";
final String regexTape = "(tape \\w+)";
final String regexSubjectDN = "subject[']?s DN.*$";
final String regexGETReceived = "GET \\(.*\\):";
final String regexEntryNotInRepository = "Entry not in repository :\\s.*$";
final String regexDelimitedMessageBody = "delimited message body \\(.*$";

final String replaceNetIp = "IP";
final String replaceStr = "IP|HOST:PORT";
final String replacePort = "PORT";
final String replaceLfn = "LFN";
final String replaceProtocol = "PROTOCOL://";
final String replacePath = "/PATH";
final String replaceHost = "HOST";
final String replacePod = "POD";
final String replaceChecksums = "CHECKSUMS";
final String replaceSSLCode = "ssl=SSLCODE";
final String replaceFileId = "file ID FILEID";
final String replaceReason = "reason: REASON";
final String replaceTape = "tape TAPE";
final String replaceSubjectDN = "subject's DN";
final String replaceGETReceived = "GET :";
final String replaceEntryNotInRepository = "Entry not in repository";
final String replaceDelimitedMessageBody = "delimited message body";
```

ClusterLogs - FTS log analysis

ClusterLogs was used to cluster FTS logs.

Validated results using GGUS tickets.

In the end, **it did not help** the operators to fix issues.

Lesson learned:

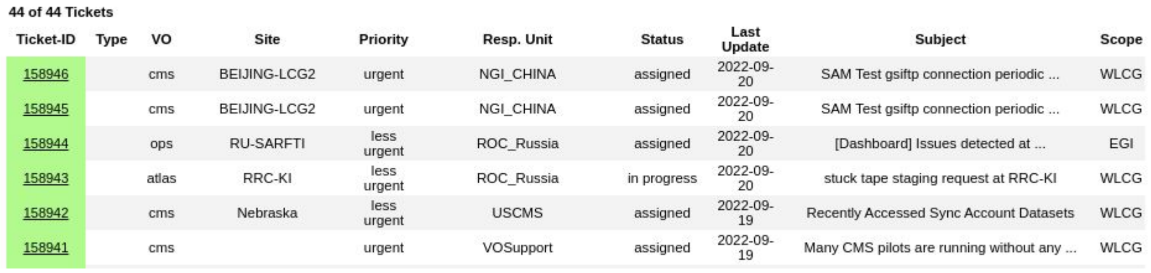
The clusters on their own are useless. They do not produce any meaningful and actionable information.

In order for the information to be useful, we must provide specific issue description. Giving the operators a bunch of logs is not enough.

We must ask the question, how should the system help the operators? Is duplicating tickets created by users going to help?

GGUS [1]

- Ticketing system
- Contains **unstructured** user-made description of the problem and attributes specifying VO, Scope, etc.



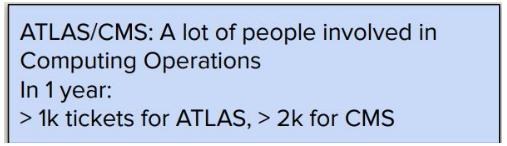
44 of 44 Tickets

Ticket-ID	Type	VO	Site	Priority	Resp. Unit	Status	Last Update	Subject	Scope
158946		cms	BEIJING-LCG2	urgent	NGI_CHINA	assigned	2022-09-20	SAM Test gsiftp connection periodic ...	WLCG
158945		cms	BEIJING-LCG2	urgent	NGI_CHINA	assigned	2022-09-20	SAM Test gsiftp connection periodic ...	WLCG
158944		ops	RU-SARFTI	less urgent	ROC_Russia	assigned	2022-09-20	[Dashboard] Issues detected at ...	EGI
158943		atlas	RRC-KI	less urgent	ROC_Russia	in progress	2022-09-20	stuck tape staging request at RRC-KI	WLCG
158942		cms	Nebraska	less urgent	USCMS	assigned	2022-09-19	Recently Accessed Sync Account Datasets	WLCG
158941		cms		urgent	VOSupport	assigned	2022-09-19	Many CMS pilots are running without any ...	WLCG

Number of tickets

Cca 4k tickets for the past year of which 900 are CMS.

But an older presentation shows much higher number per year. Why?



ATLAS/CMS: A lot of people involved in Computing Operations
In 1 year:
> 1k tickets for ATLAS, > 2k for CMS

https://ggus.eu/

GGUS vs Service Portal

What is the difference between Service Portal (or Service Now) and GGUS?

Are there any other ticketing websites?

Existing solution: Log Anomaly Detector

Can it use other attributes besides a log message?

- The current solution does not support this, but it can be extended.

To what extent is it useful?

- We can reuse some of the code, but our solution will be mostly tailored to our use case.
- Also, it could save some of the easy programming, which might not cover the time spent trying to integrate this solution.

Can it be integrated into the CERN's infrastructure?

- ?

Is the solution used by others?

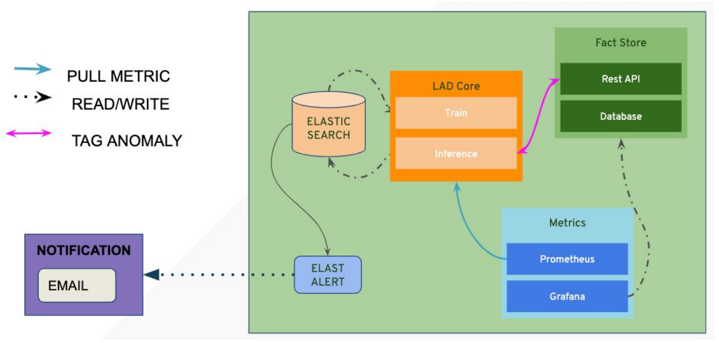
- 253 stars on GitHub -> Yes.

What is the license?

- GNU General Public Licence -> Commercial use allowed.

Scalability?

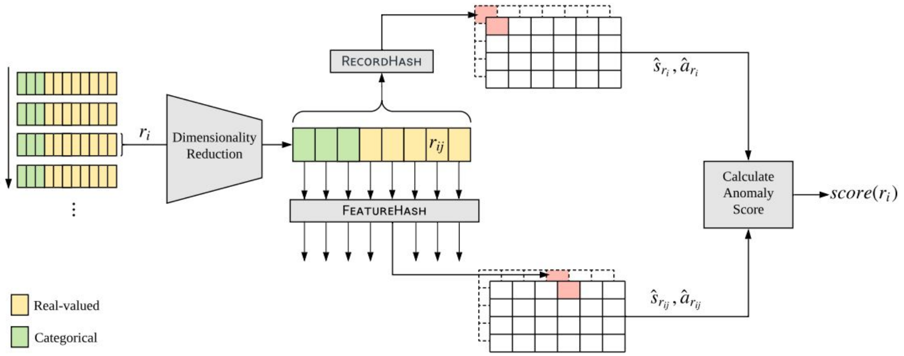
- ?

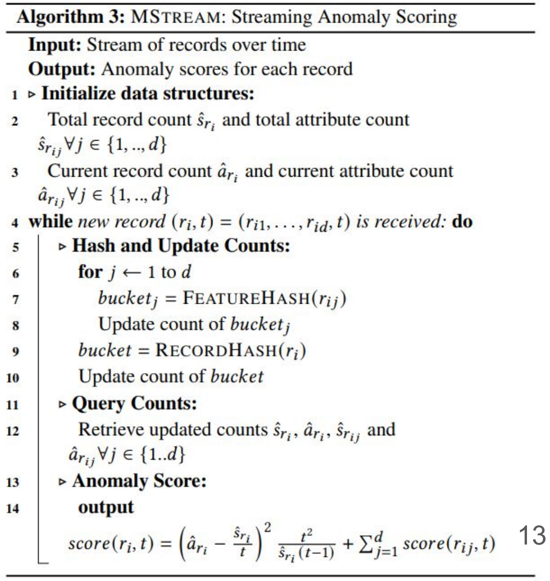


Let's not focus on this right now. We can get back to it once some of the code may seem useful.

MStream

- Extended MIDAS. Defines an anomalousness score for each record instead of a source-destination edge.
- Detects anomalies in data including categorical and numerical attributes
- Computes locality-sensitives hashes, which places the features into buckets
- Determines anomaly scores as a sum of Chi-Squared statistics of past counts and current counts




Algorithm 3: MSTREAM: Streaming Anomaly Scoring

```
Input: Stream of records over time
Output: Anomaly scores for each record
1  ▹ Initialize data structures:
2    Total record count $\hat{s}_{r_i}$ and total attribute count
     $\hat{s}_{r_{ij}} \forall j \in \{1, .., d\}$
3    Current record count $\hat{a}_{r_i}$ and current attribute count
     $\hat{a}_{r_{ij}} \forall j \in \{1, .., d\}$
4  while new record $(r_i, t) = (r_{i1}, \ldots, r_{id}, t)$ is received: do
5      ▹ Hash and Update Counts:
6        for $j \leftarrow 1$ to $d$
7          $bucket_j$ = FEATUREHASH($r_{ij}$)
8          Update count of $bucket_j$
9        $bucket$ = RECORDHASH($r_i$)
10       Update count of $bucket$
11     ▹ Query Counts:
12       Retrieve updated counts $\hat{s}_{r_i}, \hat{a}_{r_i}, \hat{s}_{r_{ij}}$ and
         $\hat{a}_{r_{ij}} \forall j \in \{1..d\}$
13     ▹ Anomaly Score:
14       output
         $score(r_i, t) = \left(\hat{a}_{r_i} - \frac{\hat{s}_{r_i}}{t}\right)^2 \frac{t^2}{\hat{s}_{r_i}(t-1)} + \sum_{j=1}^{d} score(r_{ij}, t)$
```

Experimenting with ***Kubeflow***

- Figuring out how to share data between components
    - Took a little while to figure out how to pass numpy arrays
- Initially problems with versions and imports
- Versioning - integrated support for git and terminal!

Next steps

- Find a way to write and test code comfortably (e.g., the web environment does not check whether the variable has been initialized before)
- How to make a streaming pipeline?

Next steps

- Define the system’s goal comprehensively
  - Talk to FTS members, who resolve issues and tickets
- Examine the existing pipeline’s code to see which FTS feed is used
- Decide which FTS feed to use