

Log Management

From `grep` to Full-Text Search & Back

Philipp Krenn

@xeraa

Technology

Move & Counter-Move

Mainframe & Terminal

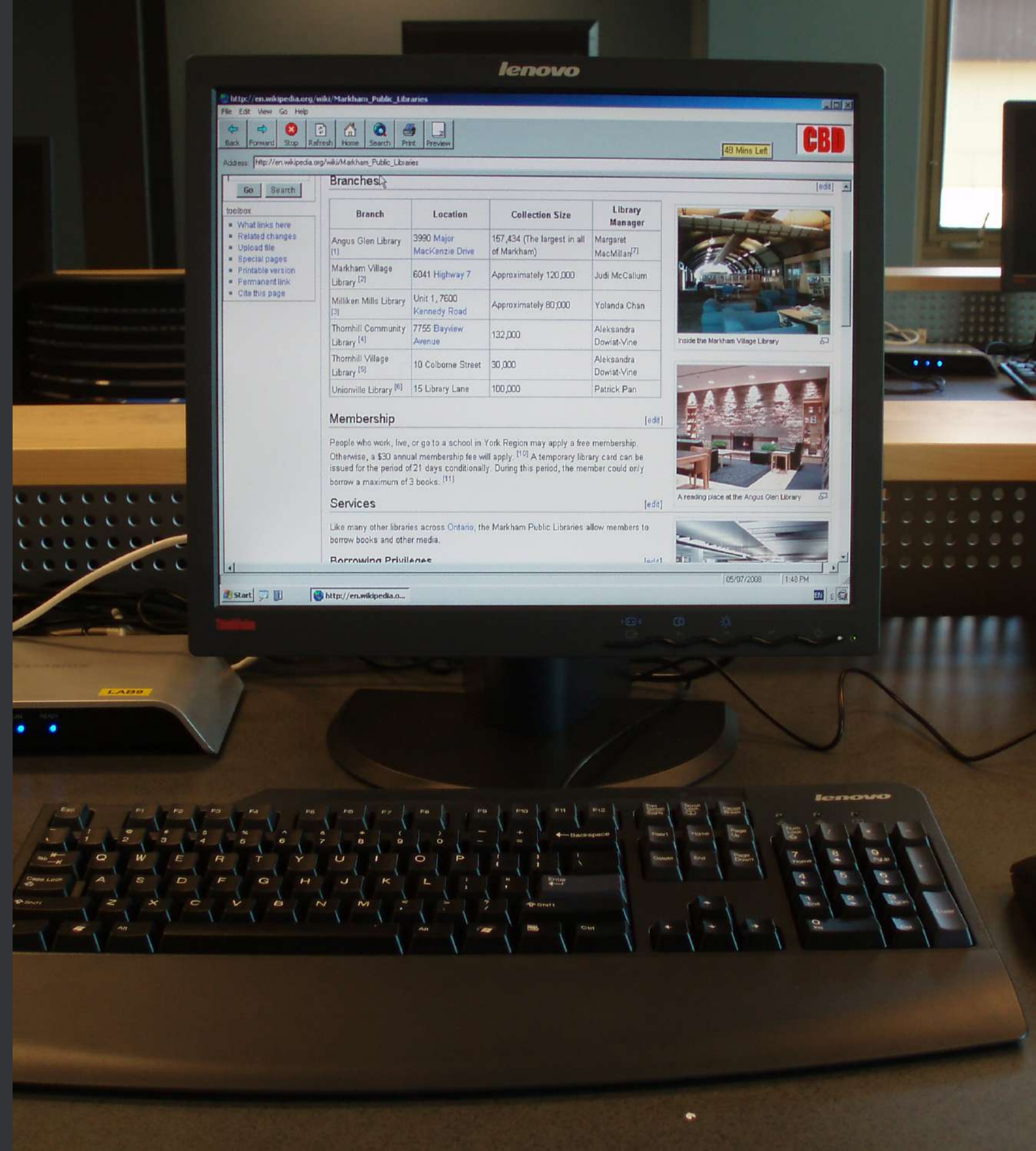




Personal Computer

**Also Fat, Heavy, Rich,
or Thick Client**

Thin Client





Laptop

Cloud



Change

Requirements & Tradeoffs




Developer 🥑

Log Management

tail

```
less +F /var/log/error.log
```

The background image shows a large, rusted metal structure, likely the hull of a ship, with a bright light source in the background. The structure is heavily corroded and has a reddish-brown patina. A bright light source, possibly a searchlight or a flare, is visible in the upper right, casting a beam of light across the scene. The overall tone is dark and industrial.

me looking
for the bug

7.2 GB
of log file

grep

```
grep "ERROR" /var/log/error.log  
| sort  
| uniq -c  
| sort -r  
| head -n5
```

Problem

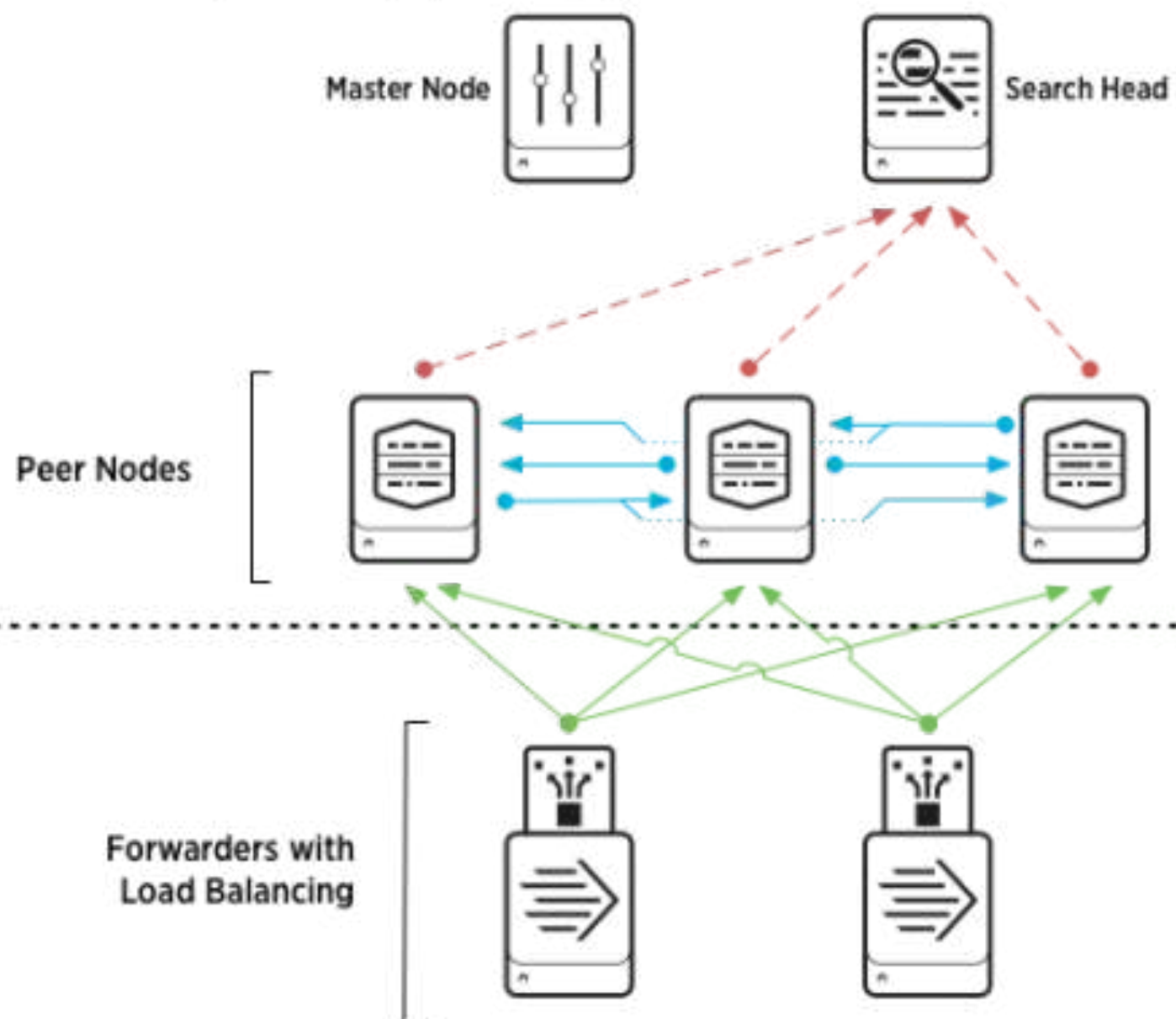
Horizontal Scaling

Distributed Applications

Ephemeral Logs

splunk[®] >

Cluster: master node, search head; 3 peer nodes; replication factor = 3



LEGEND



Search data



Peer node replicated data



Forwarder load-balanced data

Queries

```
source="error.log"
```

```
| regex _raw=".*Fatal.*"
```

```
source="/var/log/nginx/access.log"
```

```
| head 1000
```

```
| top 50 clientip
```

Index Types

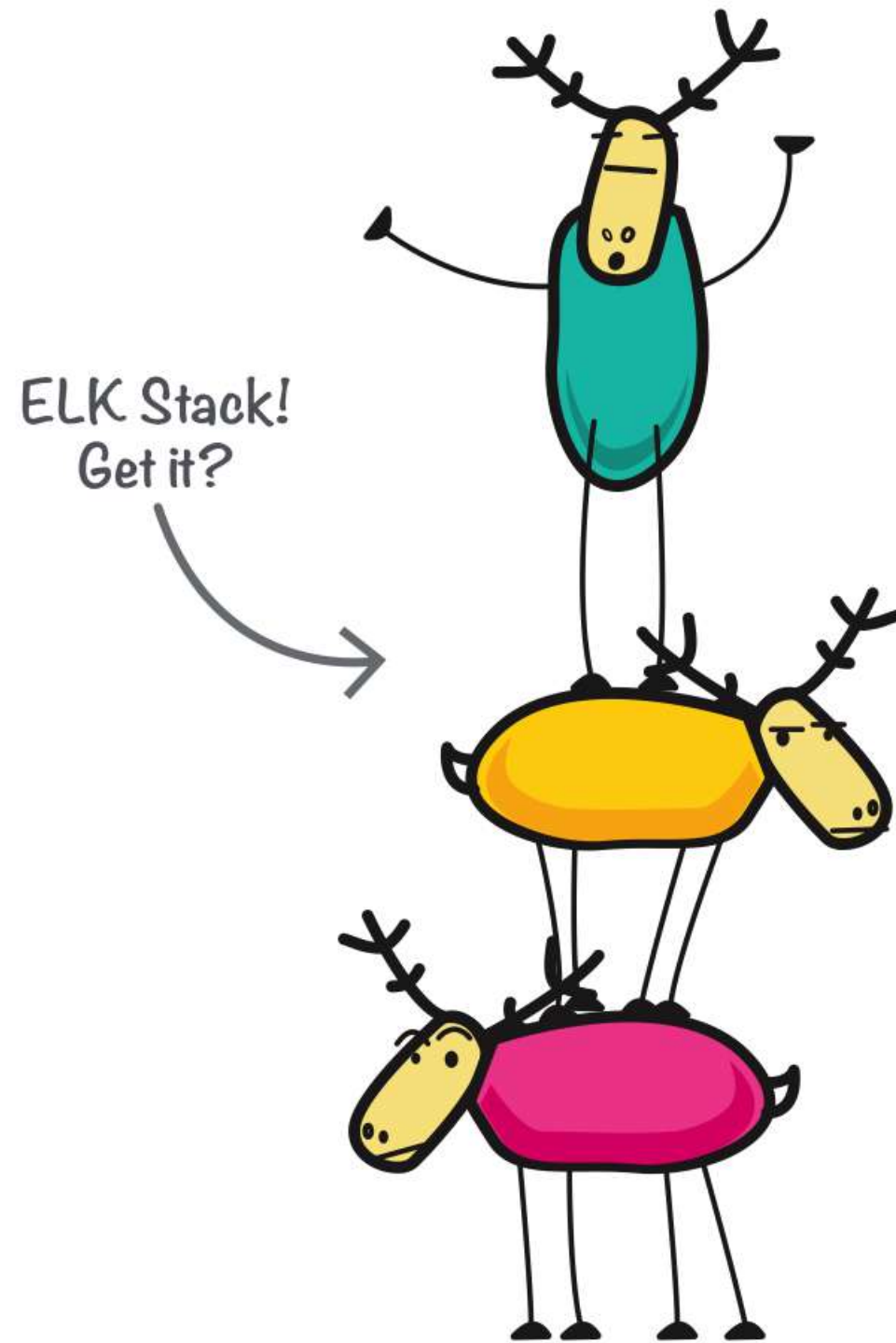
**Event Index: Minimally Structured
(default)**

Metrics Index: Highly Structured

Problem

Speed





E Elasticsearch

L Logstash

K Kibana

Full-Text Search for Logs?

- 1: Winter is coming.
2: Ours is the fury.
3: The choice is yours.



<u>term</u>	<u>freq</u>	<u>documents</u>
choice	1	3
coming	1	1
fury	1	2
is	3	1, 2, 3
ours	1	2
the	2	2, 3
winter	1	1
yours	1	3
Dictionary		Postings

Data Structures

Full-Text Search

Aggregation / Filter / Sort

Queries

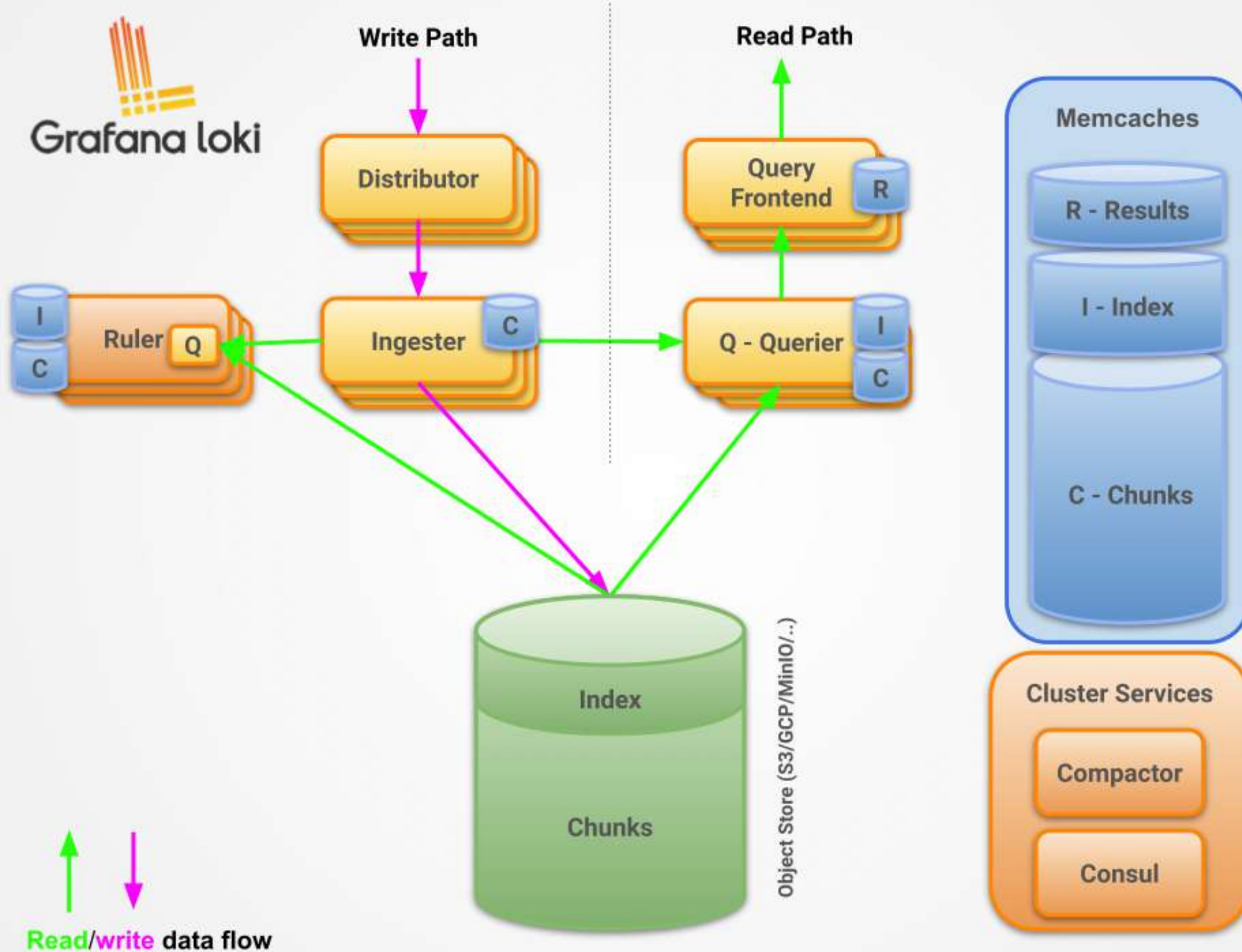
```
GET logs/_search
{
  "query": {
    "bool": {
      "filter": {
        "term": {
          "log.level": "error"
        }
      }
    }
  }
}
```


Problem

Parsed Data

Overhead: Ingestion, Disk, Memory







`{component="printer",location="f2c16",level="error"} "Printing is not supported by this printer"`

Label key/values hashed to form **Stream ID**: 3b2cea09797978fc

The log entry is added to a "chunk"

Additional log messages with the same labels are added to the same "chunk":

`{component="printer",location="f2c16",level="error"} "Out of paper"`

`{component="printer",location="f2c16",level="error"} "Too much paper"`

Chunks are filled then compressed and stored:

Printing is not supported by this printer
Out of paper
Too much paper



Printing is not supported by this printer
Out of paper
Too much paper

A separate and small index is kept to lookup chunks

Different label keys or values will hash to a different stream and different chunk:

`{component="printer",location="f2c16",level="info"} "Consider the environment before printing this log message"`

fd9a709ddf43a93a

Queries

```
{job="mysql"} |="error"
```

```
{name="kafka"} |~ "tsdb-ops.*io:2003"
```

```
{name="cassandra"} |~ `error=\w+`
```

Problem

Optimized for Specific Use

Convergence

lucene-grep

<https://github.com/dainiusjocas/lucene-grep>

**"Grep-like utility based on Lucene
Monitor compiled with GraalVM
native-image"**

Structure

Splunk: More Structured Fields

Elasticsearch: Runtime Fields

Features

Loki: Broader Feature Set

Multi-Line Logs

Elasticsearch: Log-Specific Features

wildcard Field Type

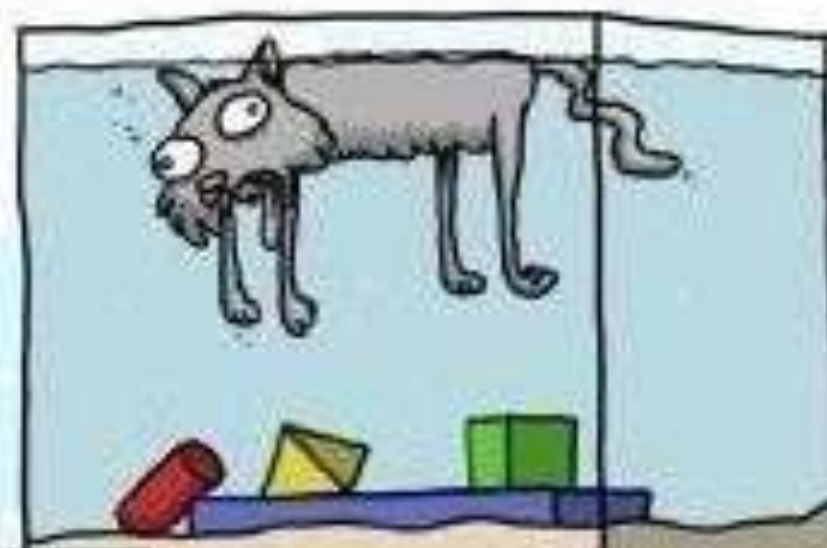
Searchable Snapshots

match_only_text Field Type

Conclusion

**It is not necessary to change.
Survival is not mandatory.**

— W. Edwards Deming



Professor Zapinsky proved that the squid is more intelligent than the housecat when posed with puzzles under similar conditions

Discussion

Features vs Speed vs Cost

Log Management

From `grep` to Full-Text Search & Back

Philipp Krenn

@xeraa