How to Avoid Drowning in Logs

Streaming 180 Billion Events/Day and
Batching 150 TB/Hour

Joshua Robinson
Founding Engineer, FlashBlade

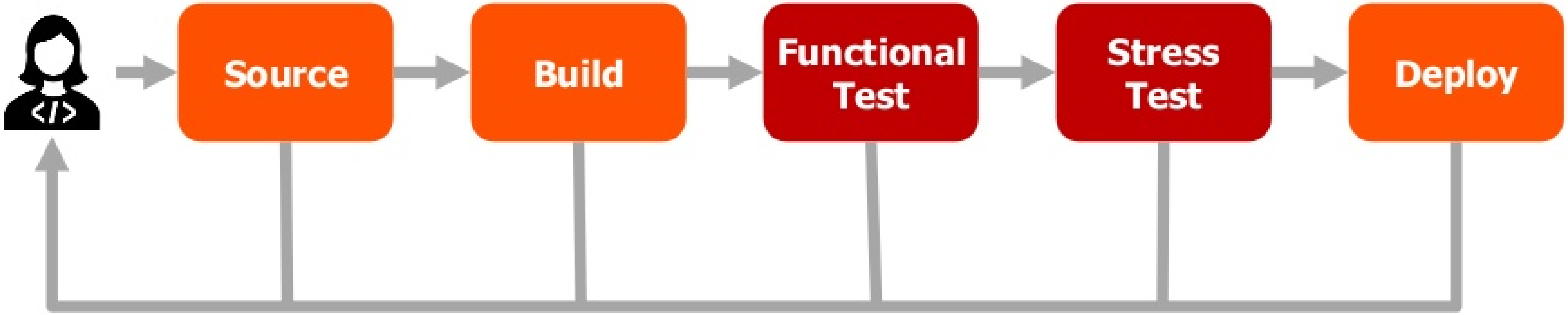
Log Analytics Pipeline in Numbers

✓ **2M** events / second

✓ **5** seconds SLA

✓ **0.5 - 1 PB** of data / day

Continuous Integration & Continuous Deployment



Source

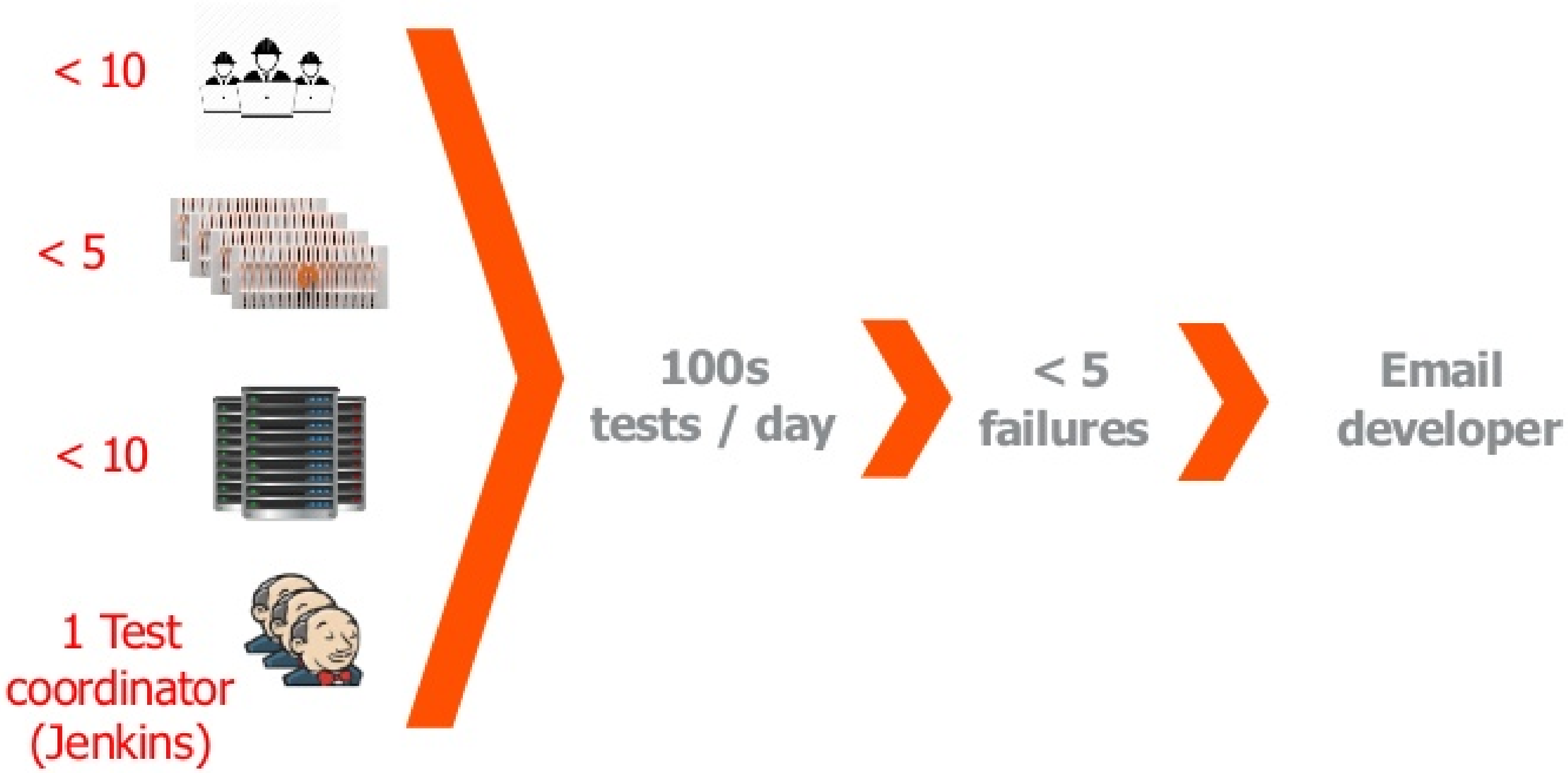
Build

Functional Test

Stress Test

Deploy

CI/CD works!



< 10

< 5

< 10

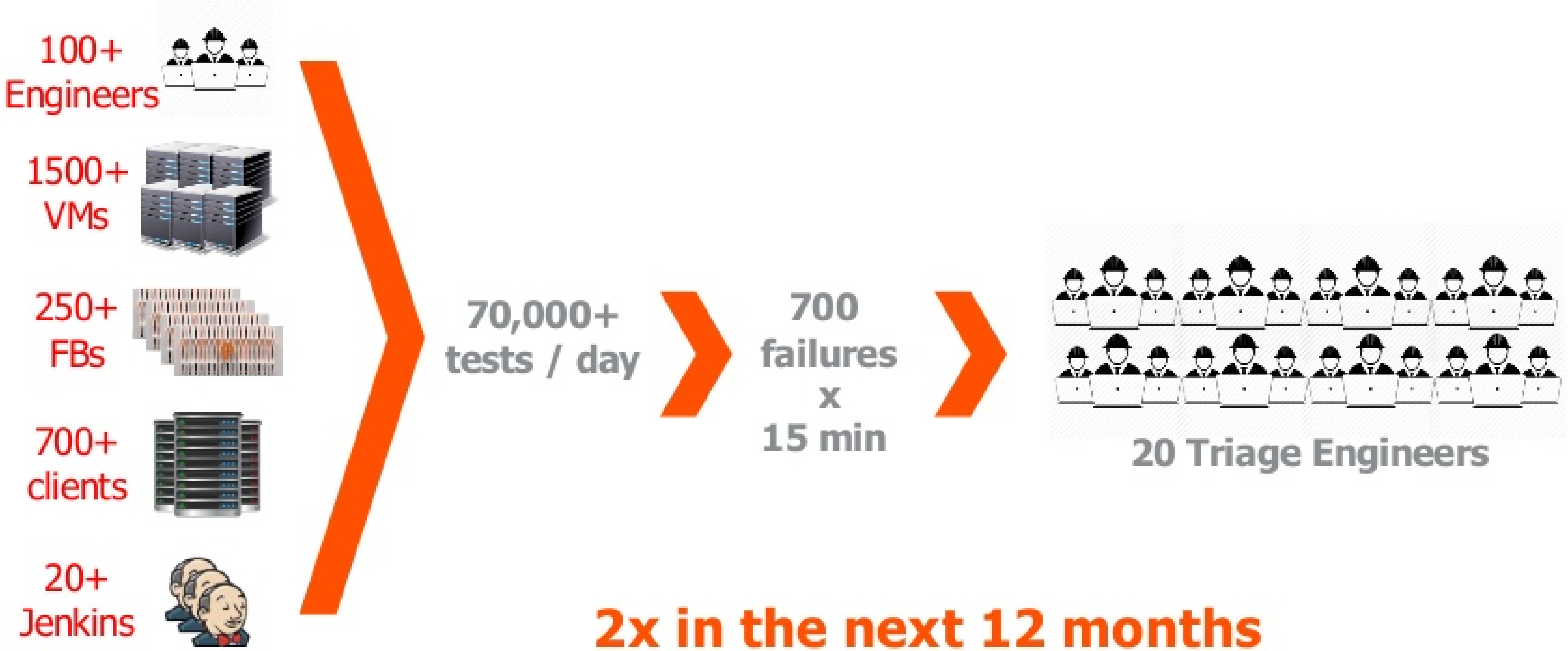
1 Test coordinator (Jenkins)

100s tests / day

< 5 failures

Email developer

Scale Problems



100+ Engineers

1500+ VMs

250+ FBs

700+ clients

20+ Jenkins

70,000+ tests / day

700 failures x 15 min

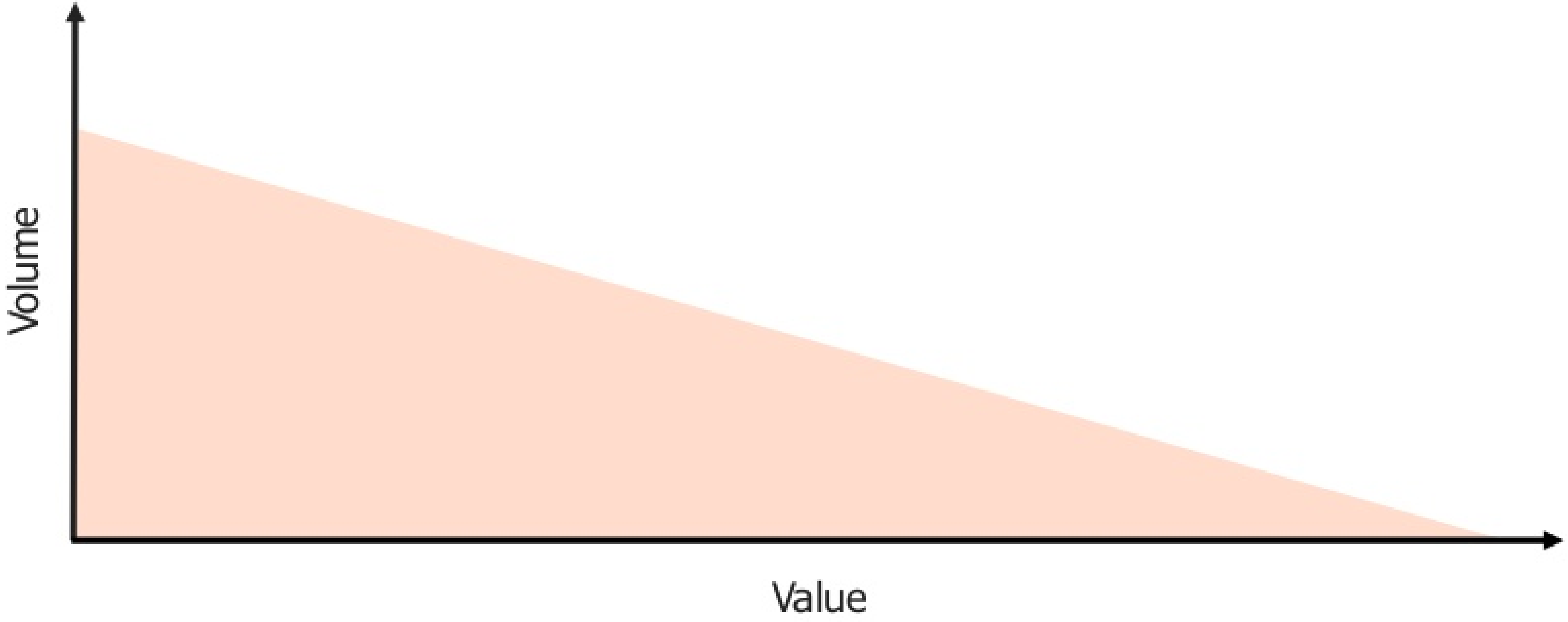
20 Triage Engineers

2x in the next 12 months

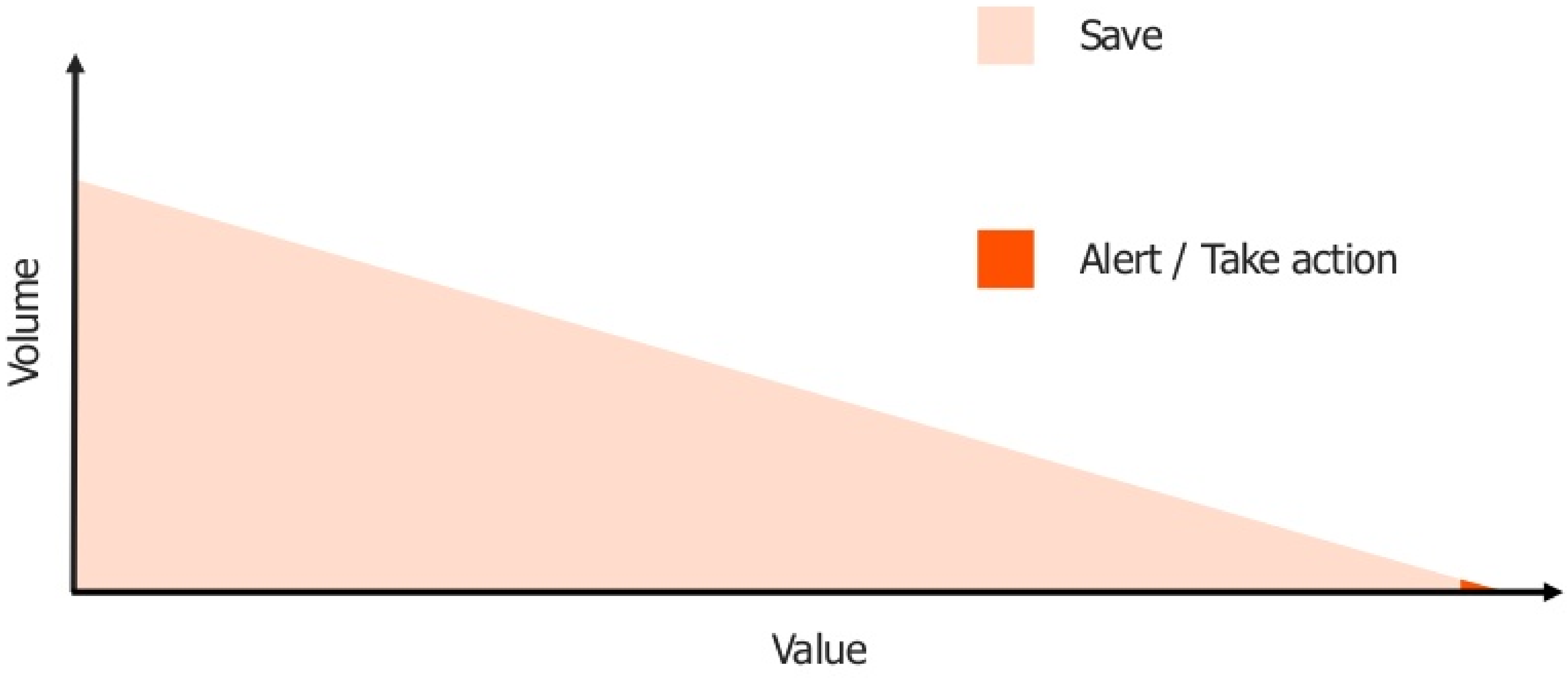
Log Analysis Dream

1. Automate triaging of failures
2. Extract performance metrics
3. Save our logs for future use
4. Do all of this in a scalable system
5. Real-time results!

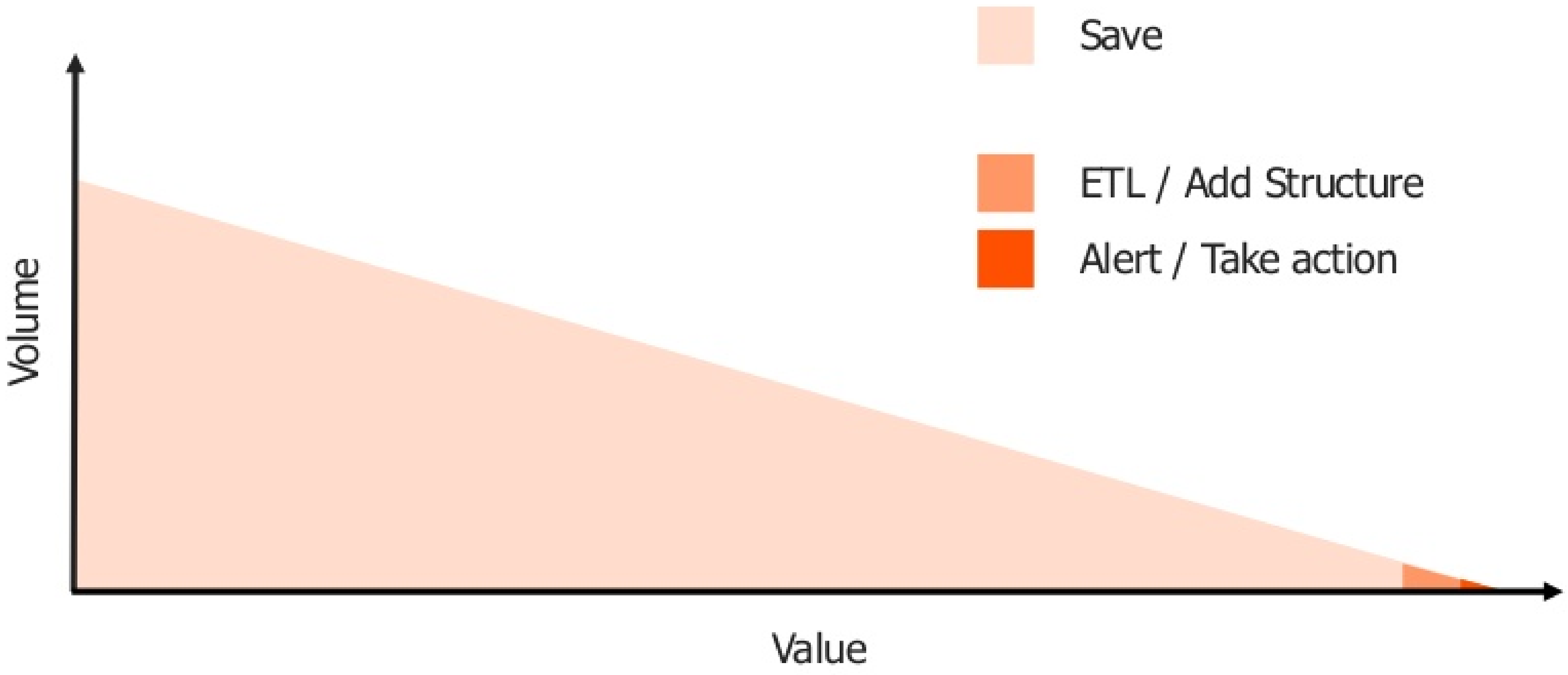
Log Analysis



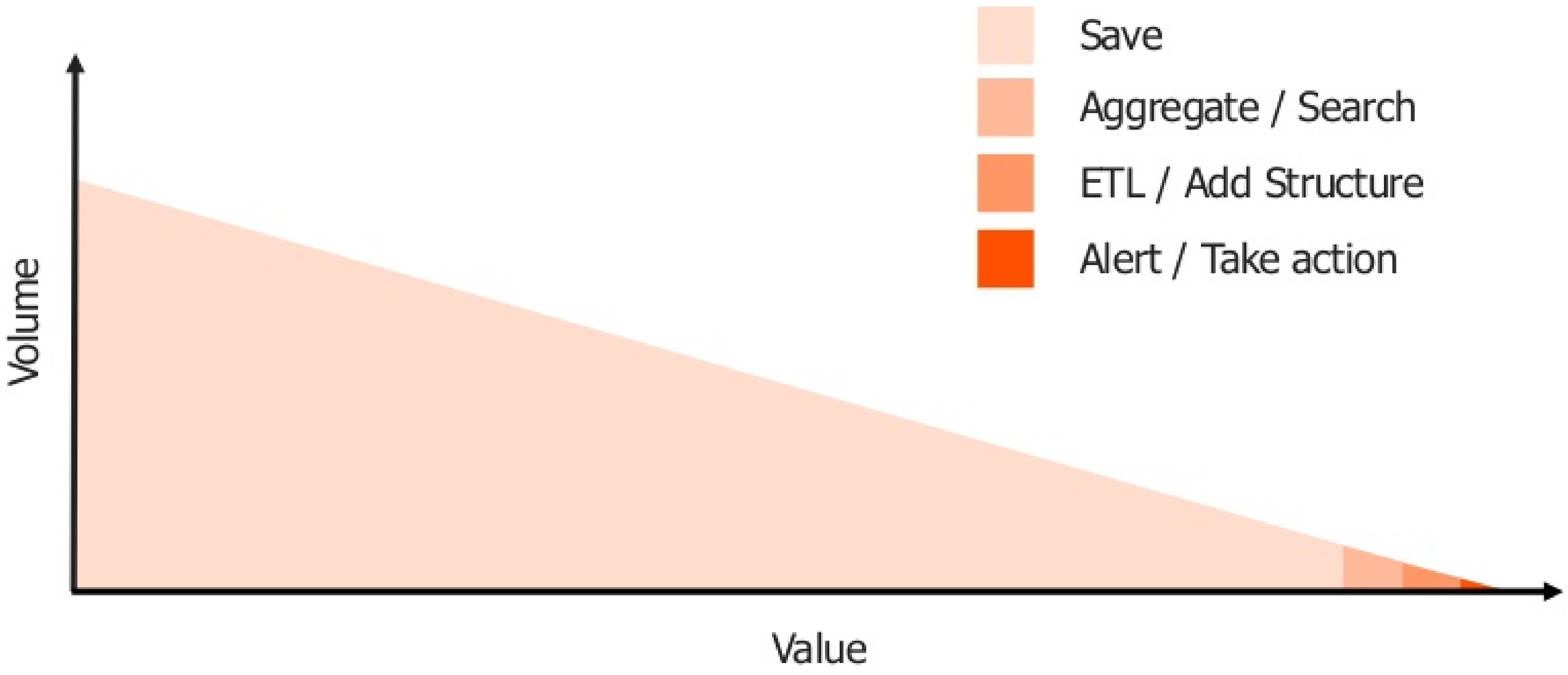
Log Analysis v1



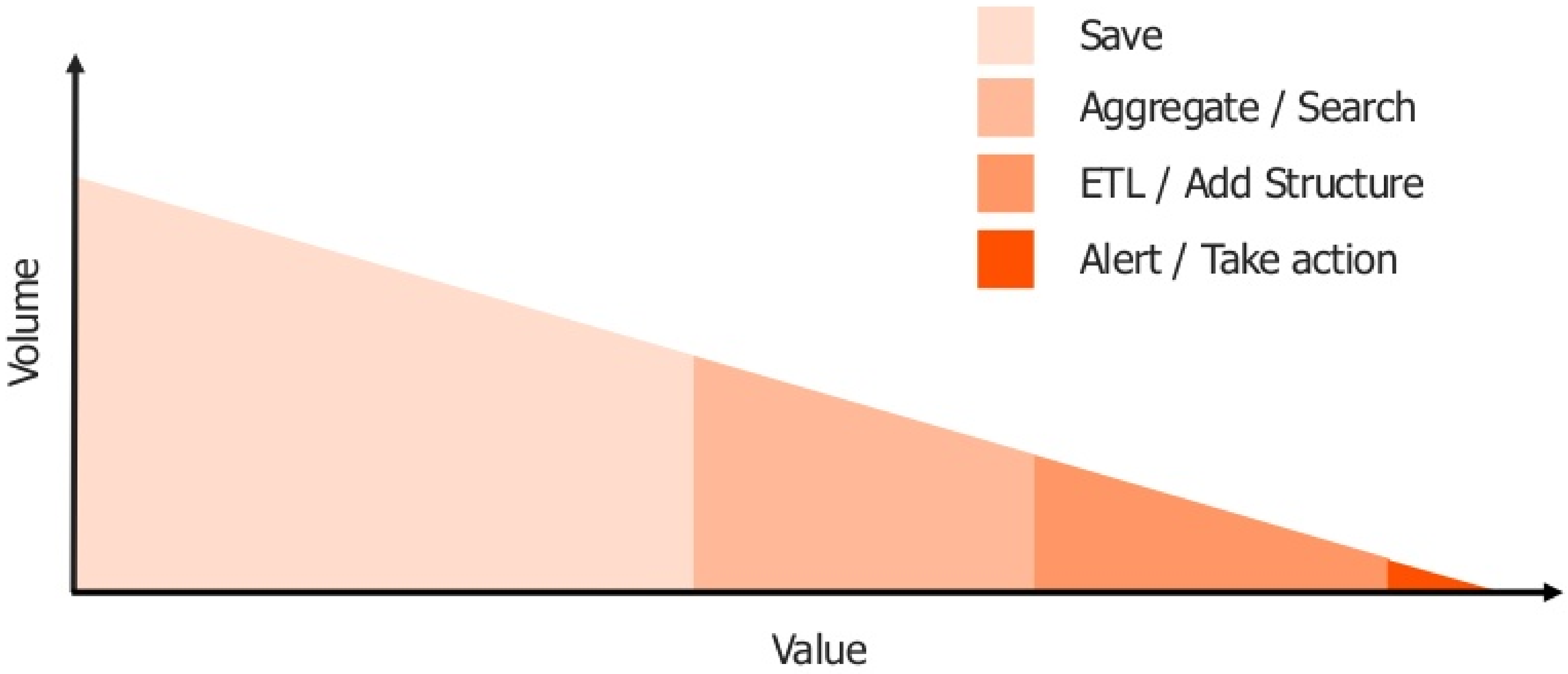
Log Analysis v2



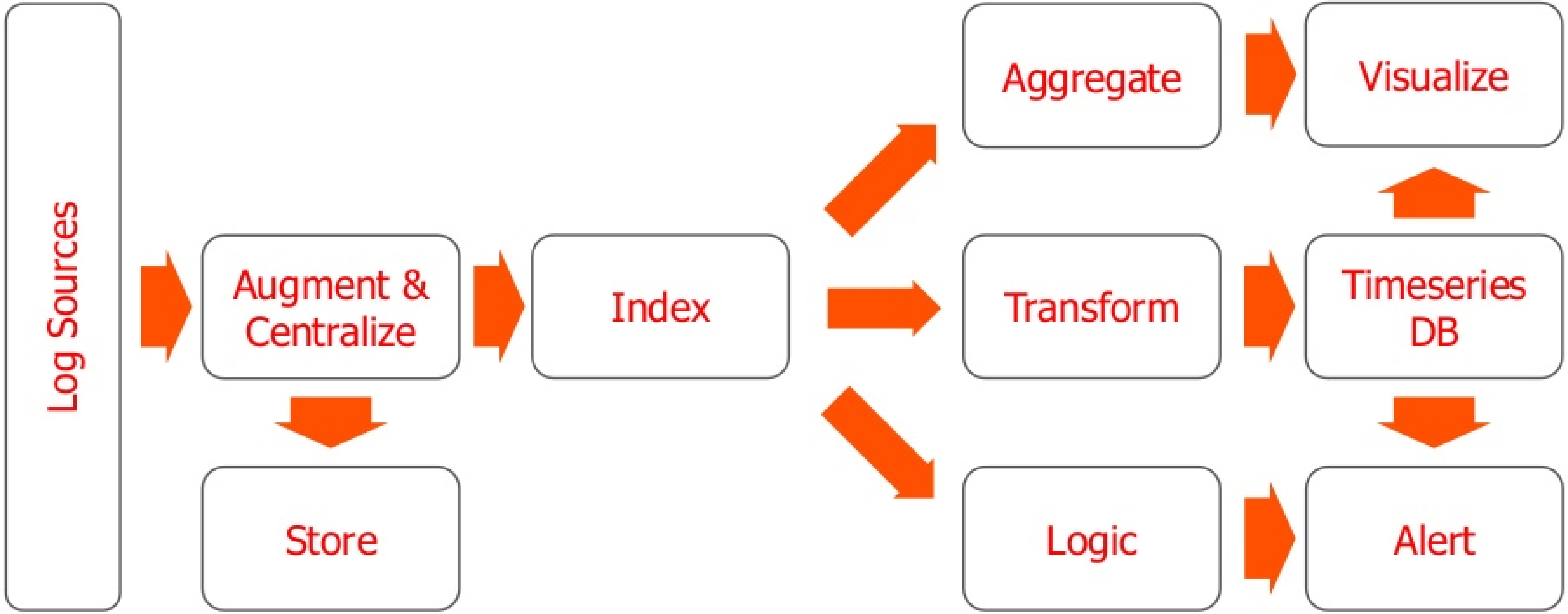
Log Analysis v3



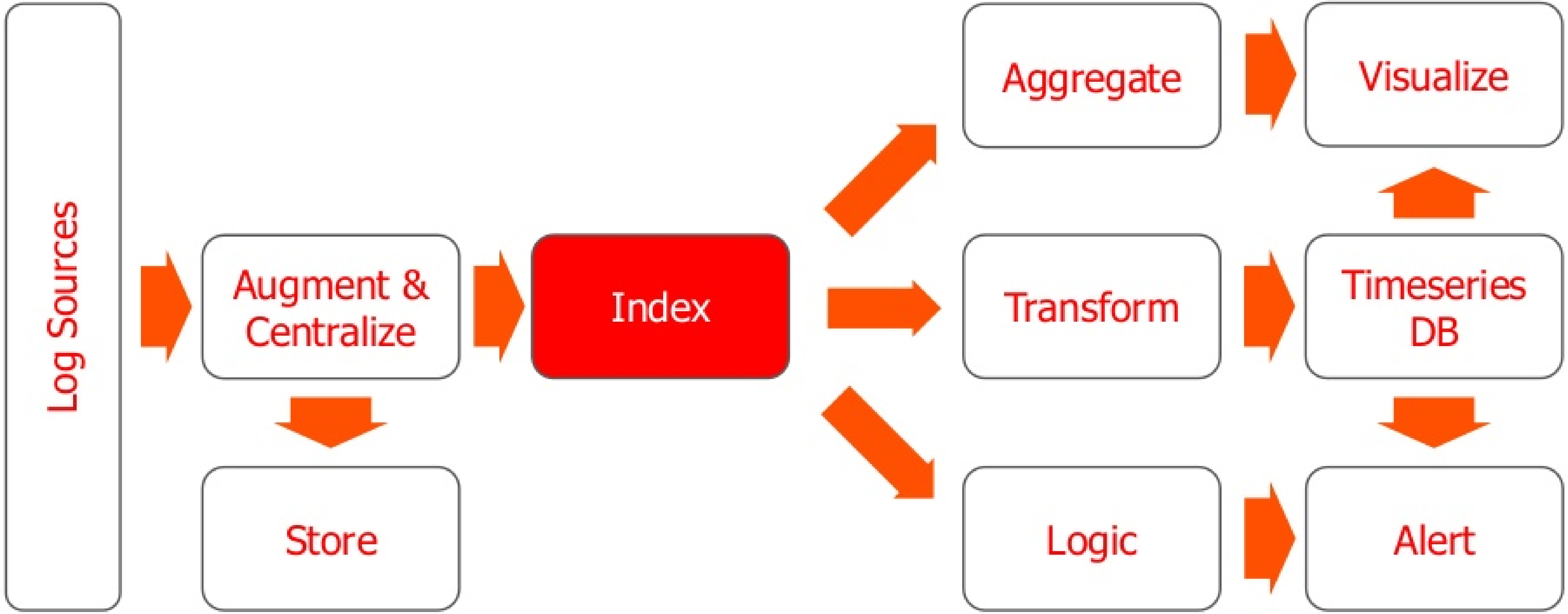
Log Analysis v10



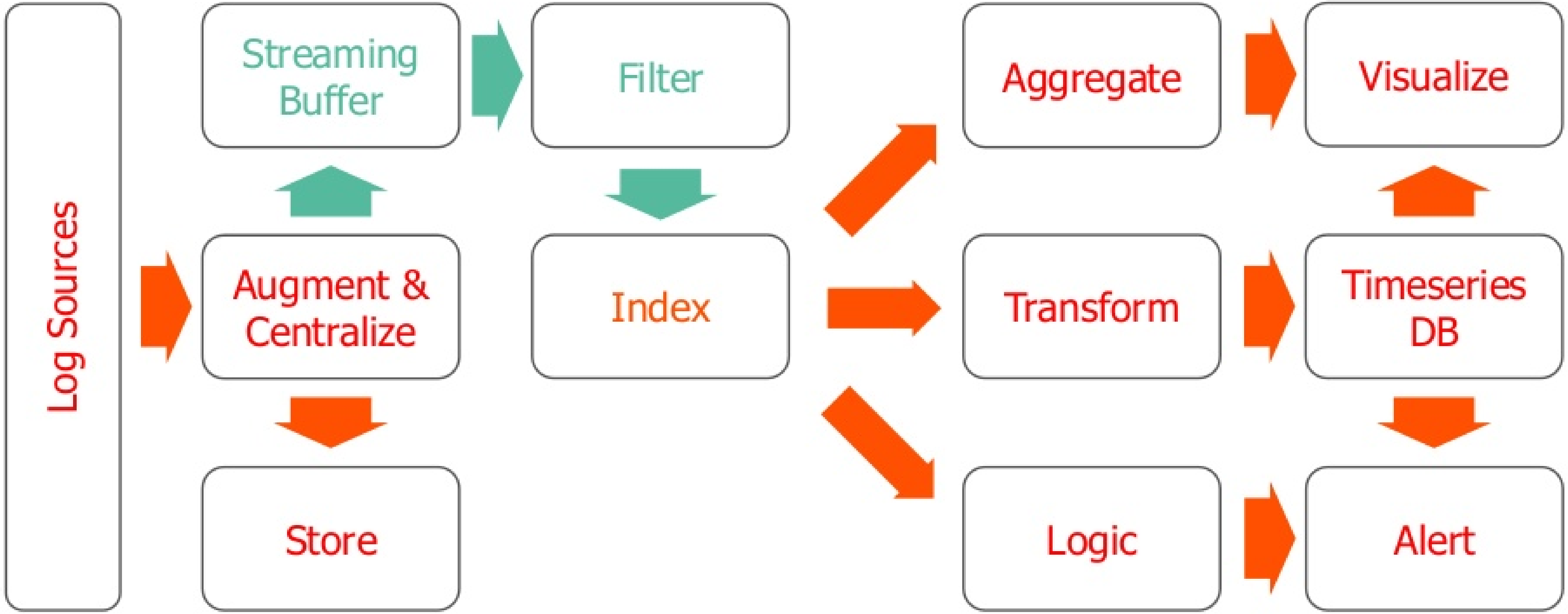
Log Analysis Pipeline



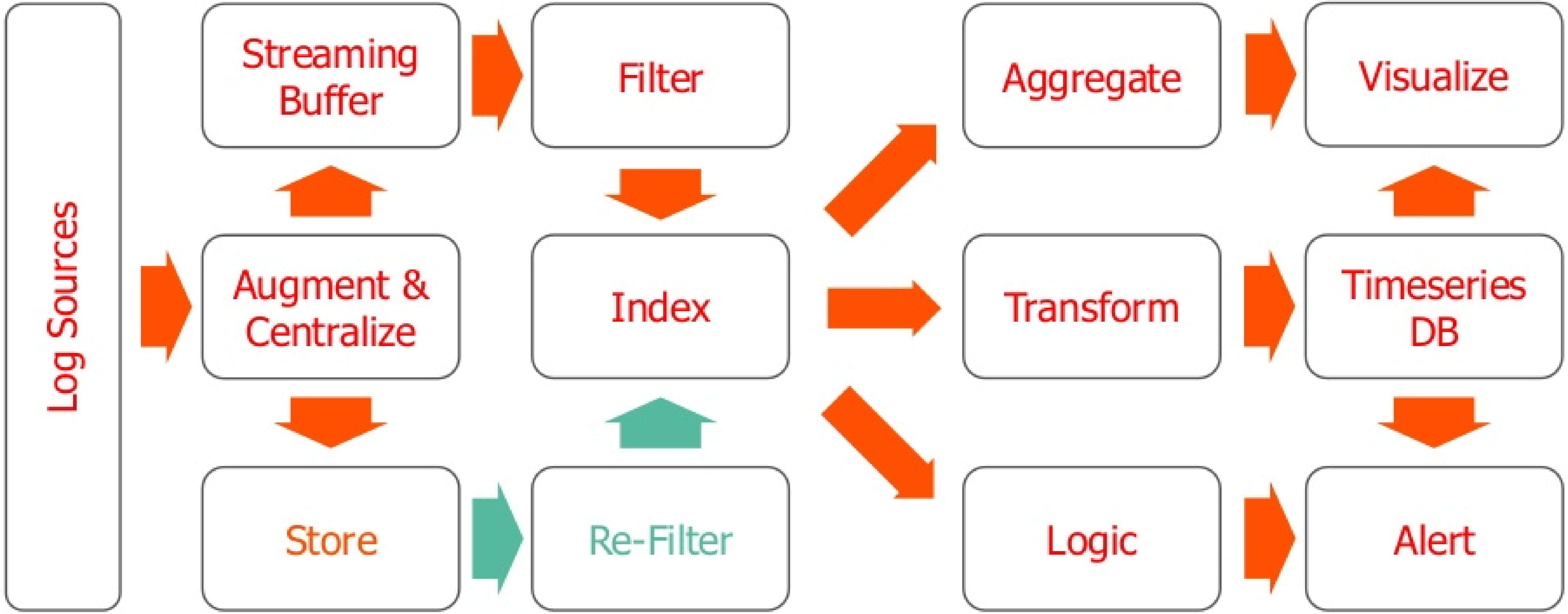
Log Analysis Pipeline



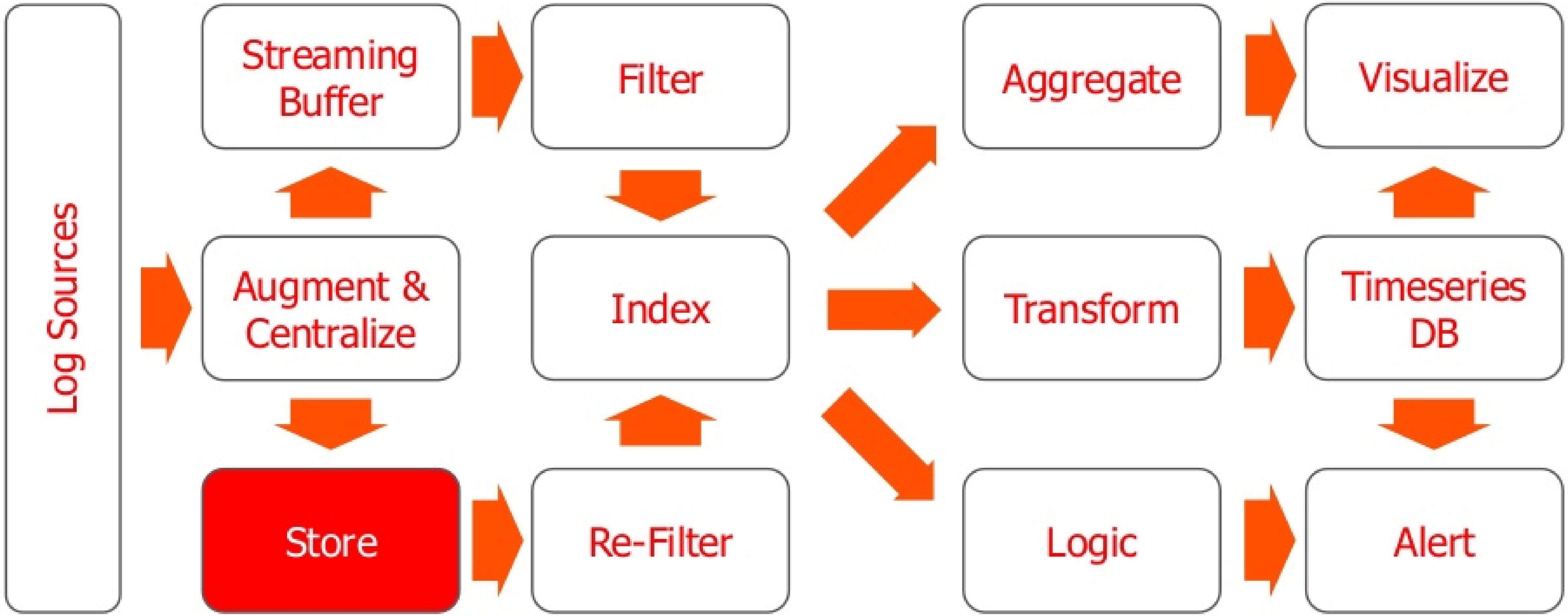
Log Analysis Pipeline



Log Analysis Pipeline



Log Analysis Pipeline



Log Sources

Streaming Buffer

Augment & Centralize

Store

Filter

Index

Re-Filter

Aggregate

Transform

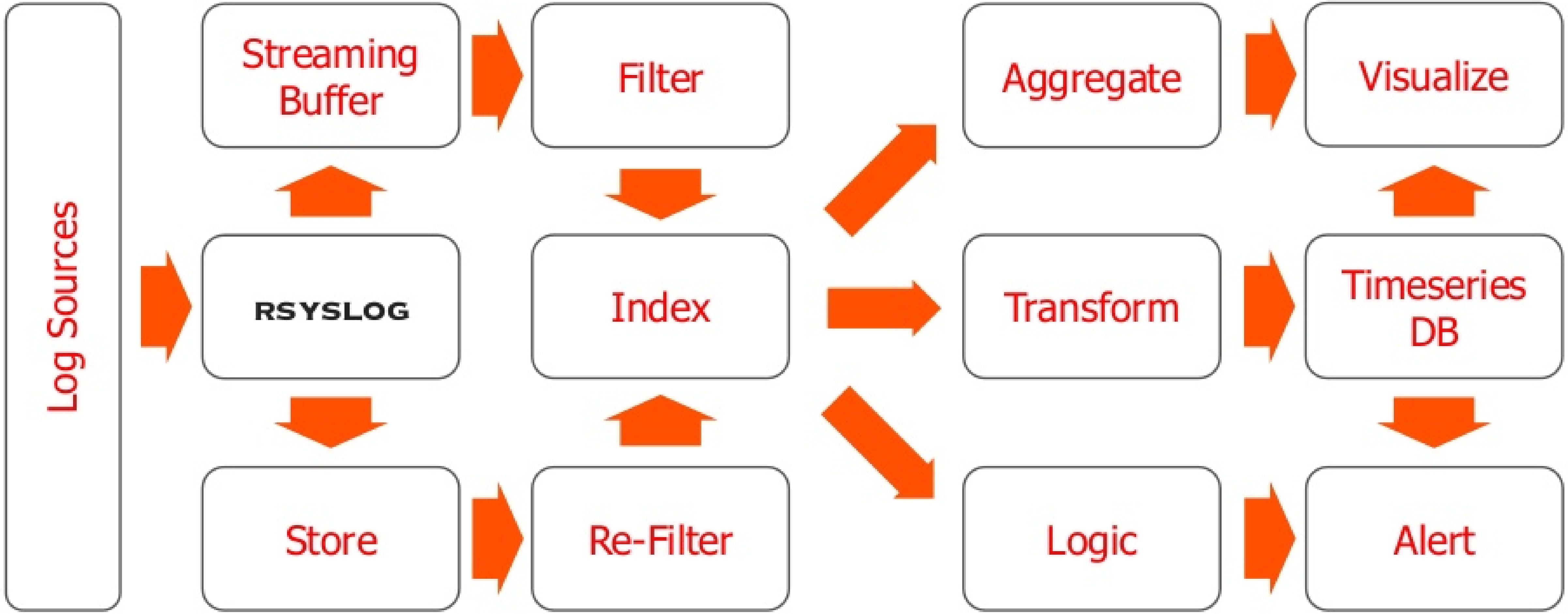
Logic

Visualize

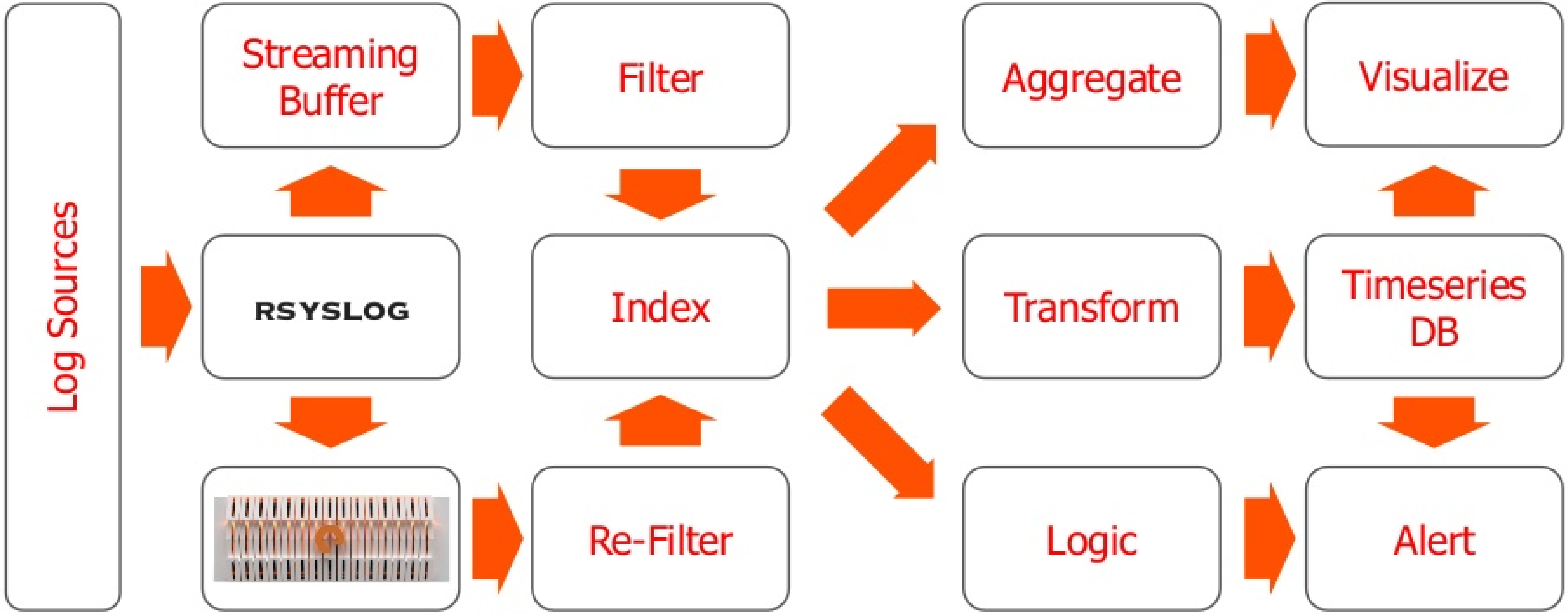
Timeseries DB

Alert

Log Analysis Pipeline



Log Analysis Pipeline



Log Sources

RSYSLOG

Streaming Buffer

Filter

Index

Re-Filter

Aggregate

Transform

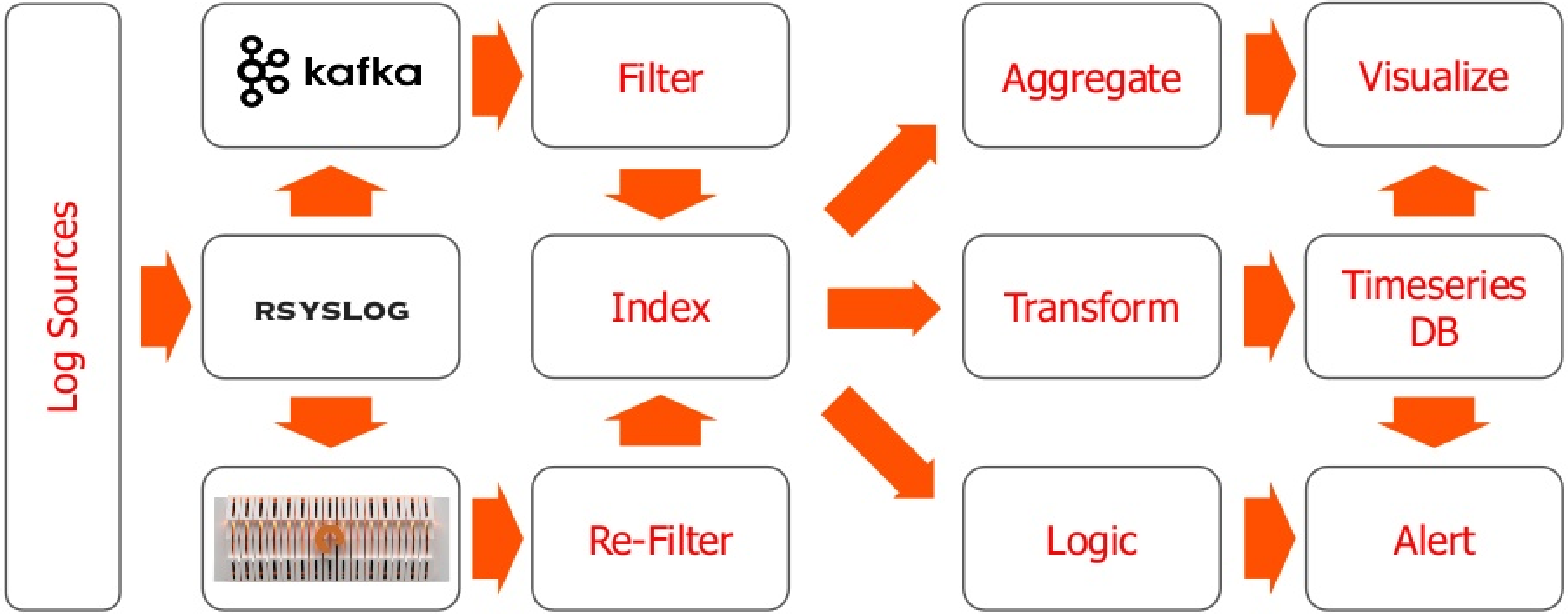
Logic

Visualize

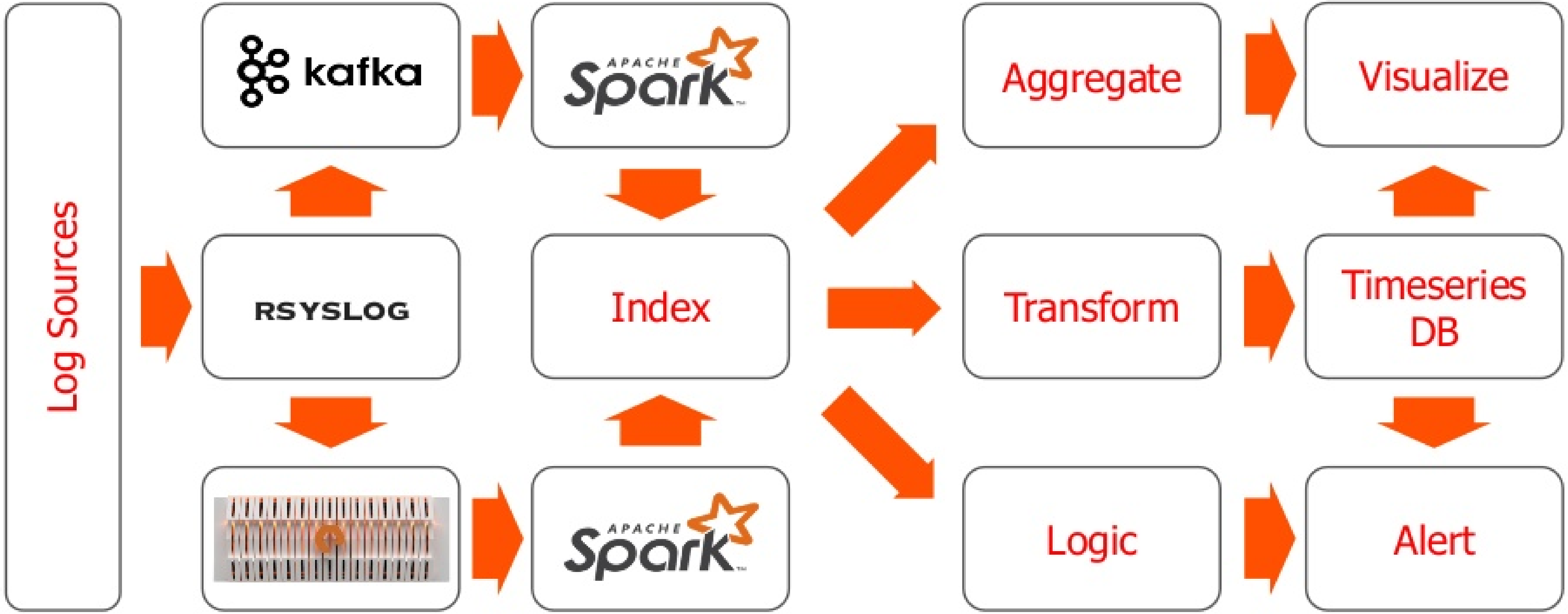
Timeseries DB

Alert

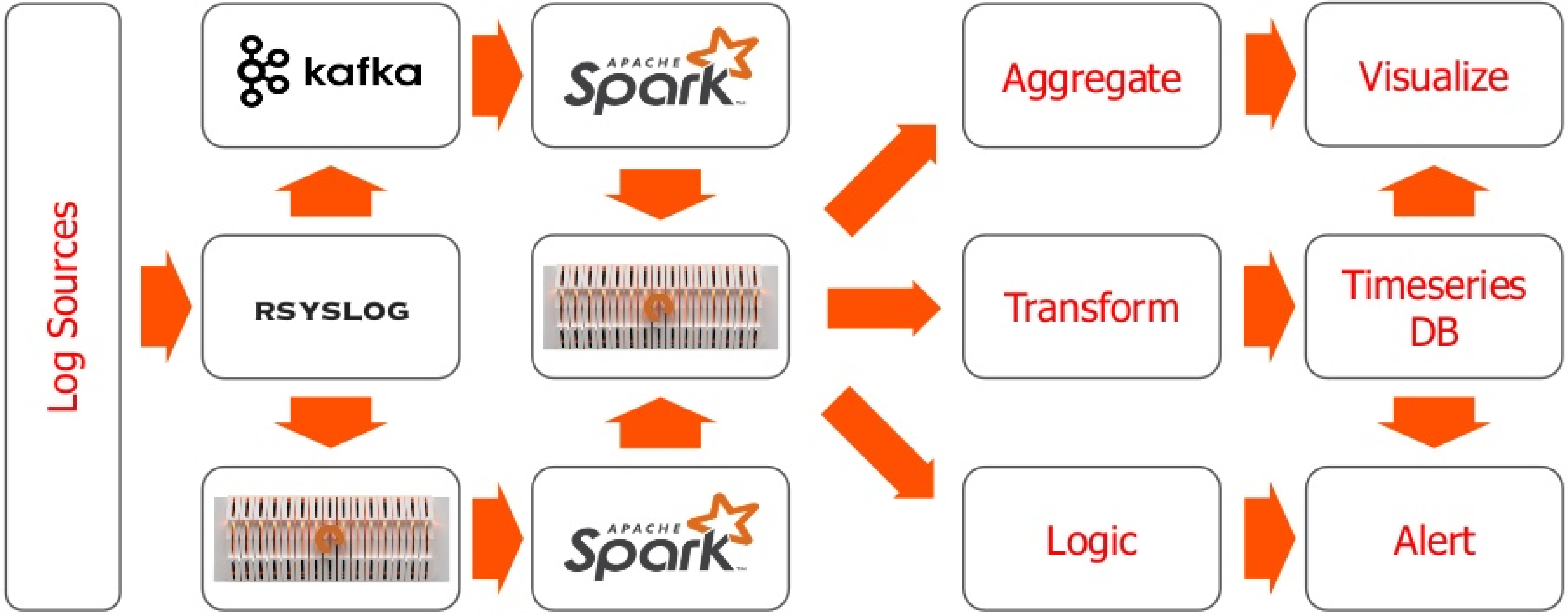
Log Analysis Pipeline



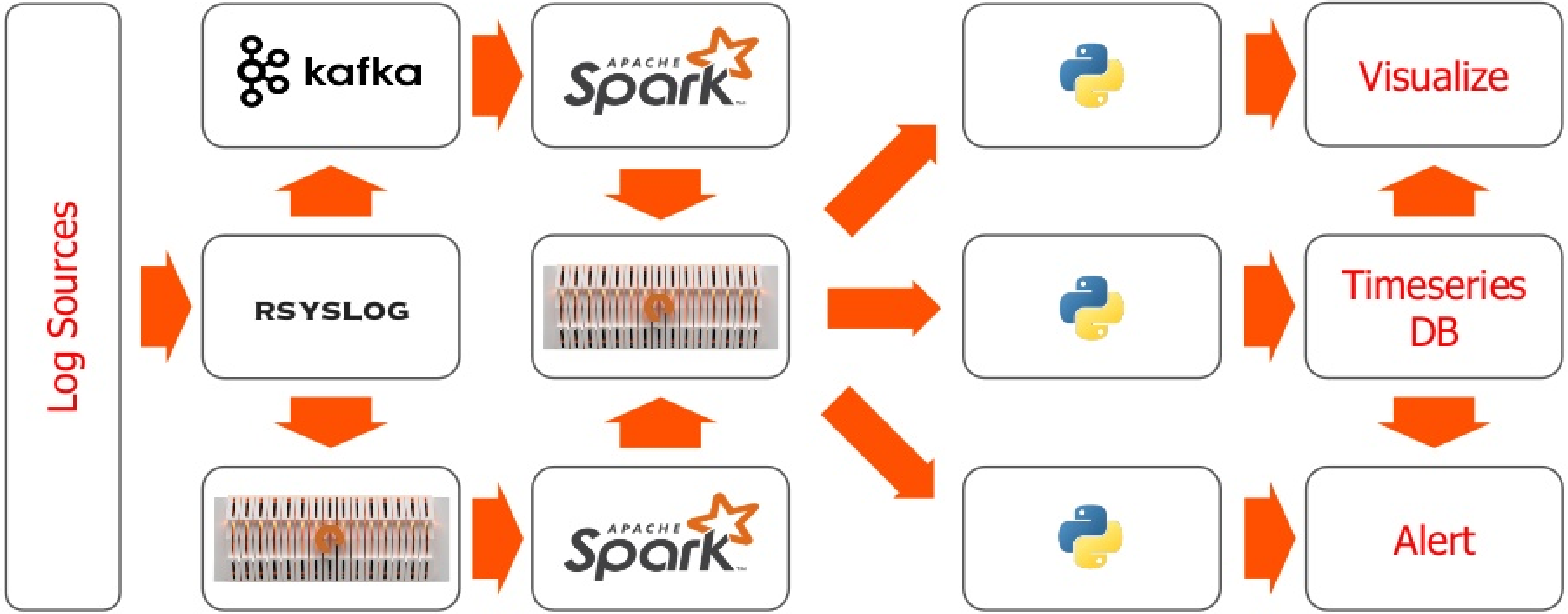
Log Analysis Pipeline



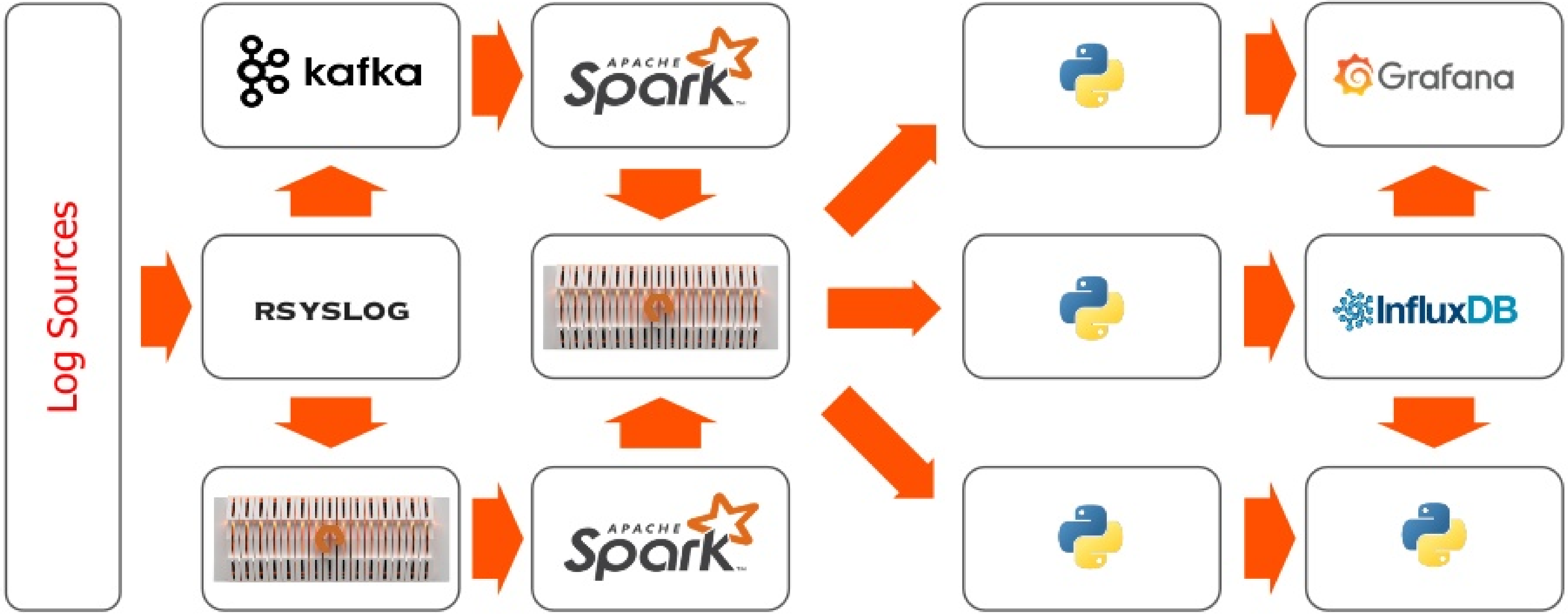
Log Analysis Pipeline



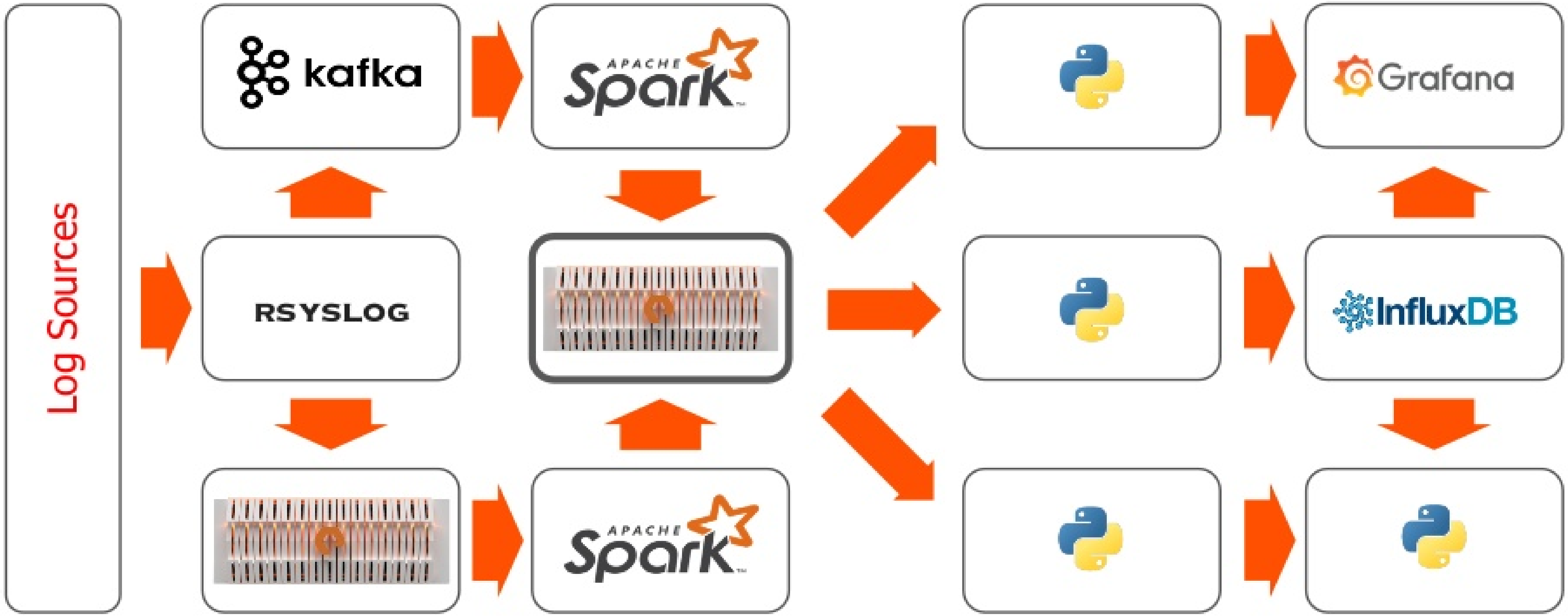
Log Analysis Pipeline



Log Analysis Pipeline



Log Analysis Pipeline



Indexing

Use filesystem directory structure to encode metadata

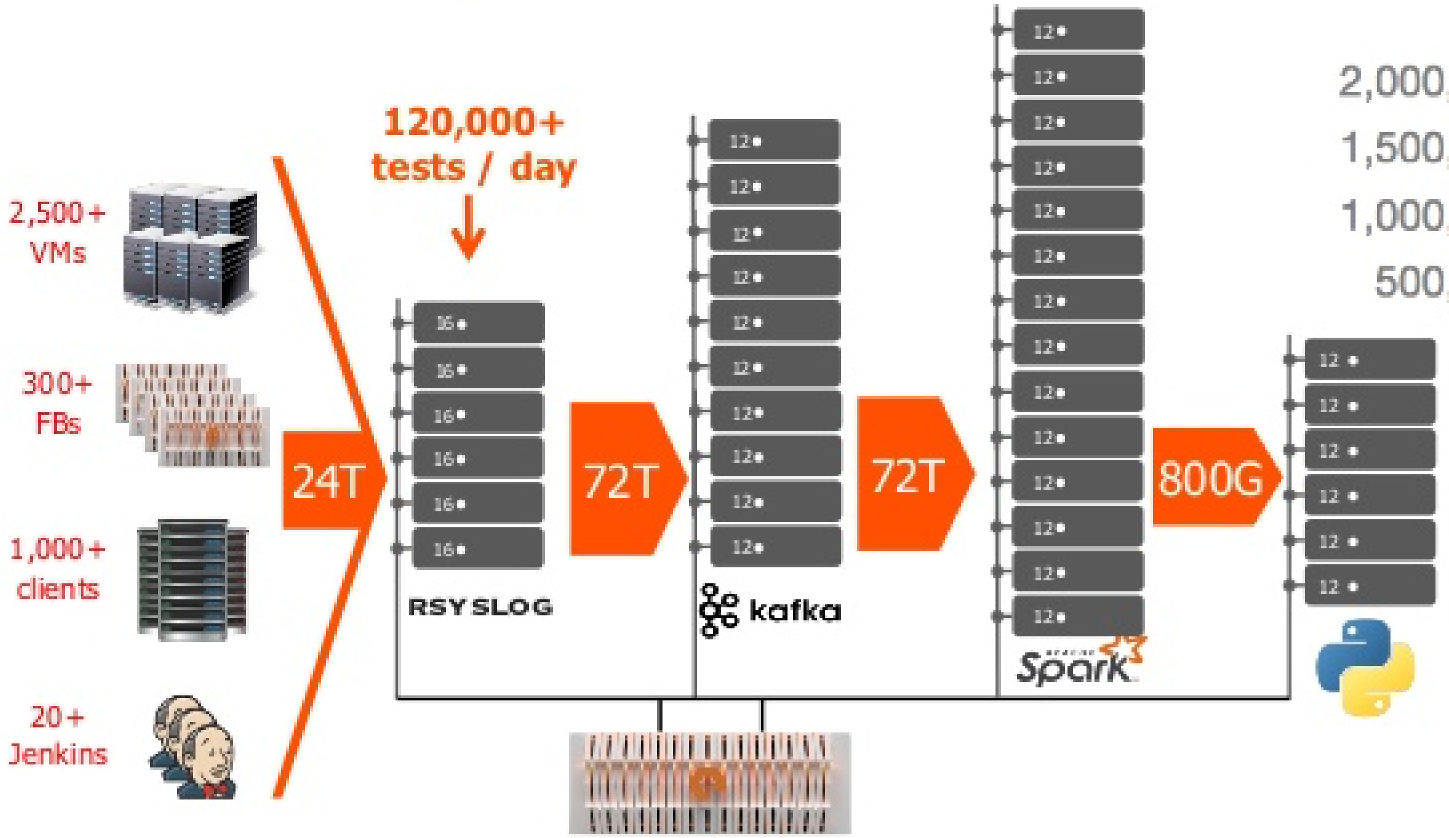
- **Raw data:** <host>/<year>/<month>/<day>/<flat files>
  - Producer: Rsyslog
  - Consumer: Spark batch (re-filter or custom lookbacks)
- **Indexed data:** <pattern>/<year>/<month>/<day>/<hour>/<host>/<flat files>
  - Producer: Spark streaming (filter)
  - Consumer: Python services (e.g. ETL, alert, searchability)

Querying

Find and load data

- FlashBlade NFS protocol. < 1ms latency
- **Listing**
  - “ls -alR” is still SLOW
  - NFS client in kernel sequentially discovers filesystem structure.
  - Solution: Skip the kernel. Use libnfs to create our own parallelized discovery. 1000x faster for 1M files
- **Reading**
  - Buffering: Create input pipeline to optimize for throughput and hide latency away

Full Pipeline



2,500+ VMs

300+ FBs

1,000+ clients

20+ Jenkins

120,000+ tests / day

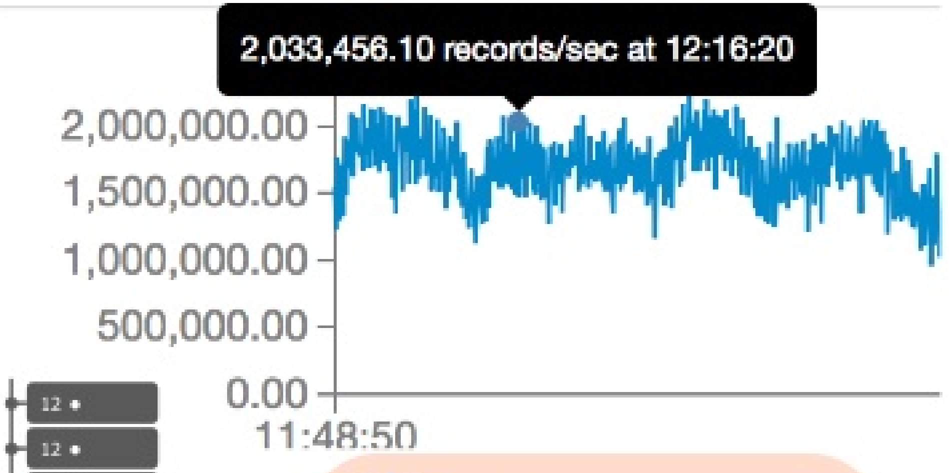
24T

72T

72T

800G

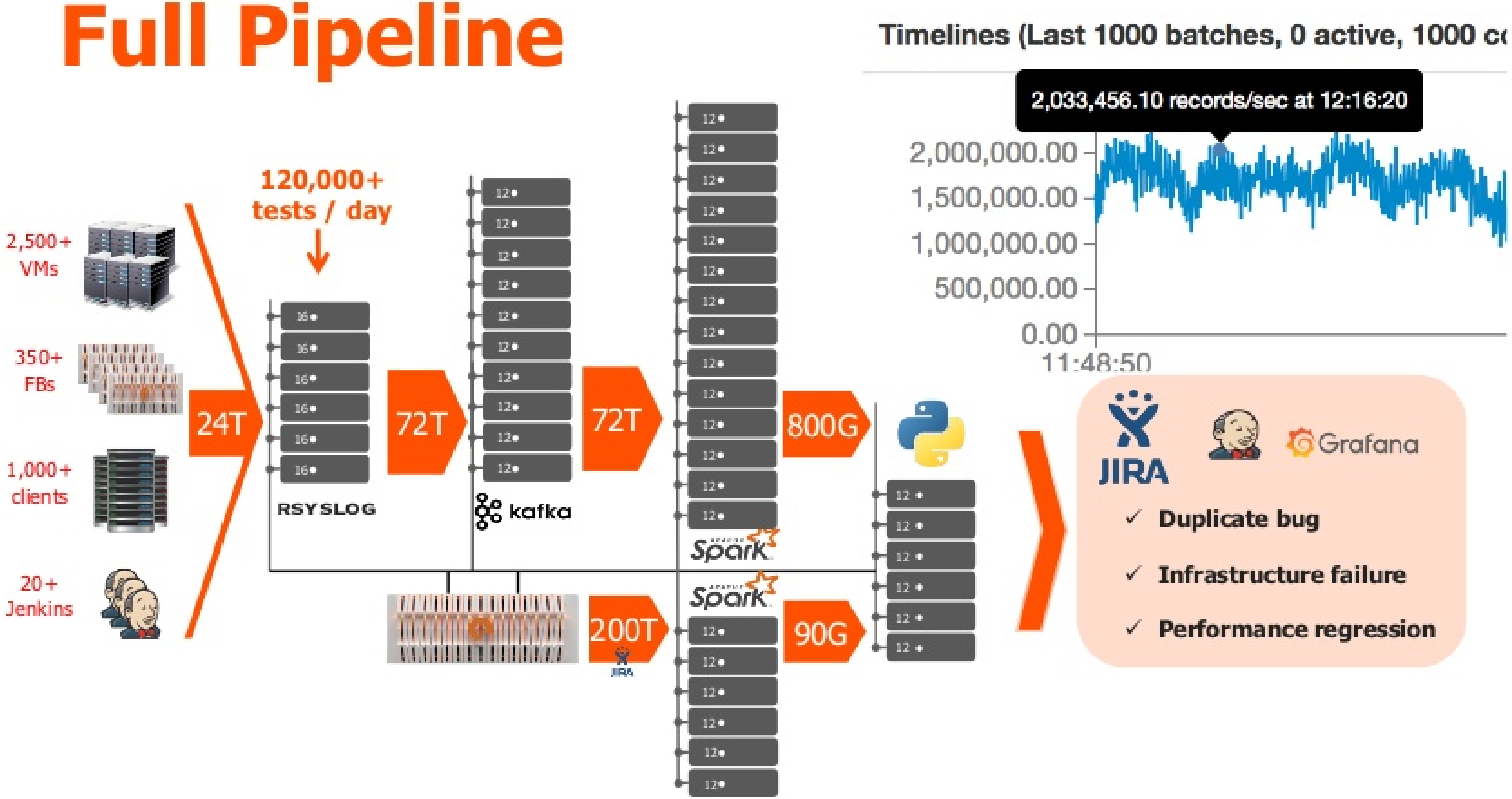
Timelines (Last 1000 batches, 0 active, 1000 c



JIRA Grafana

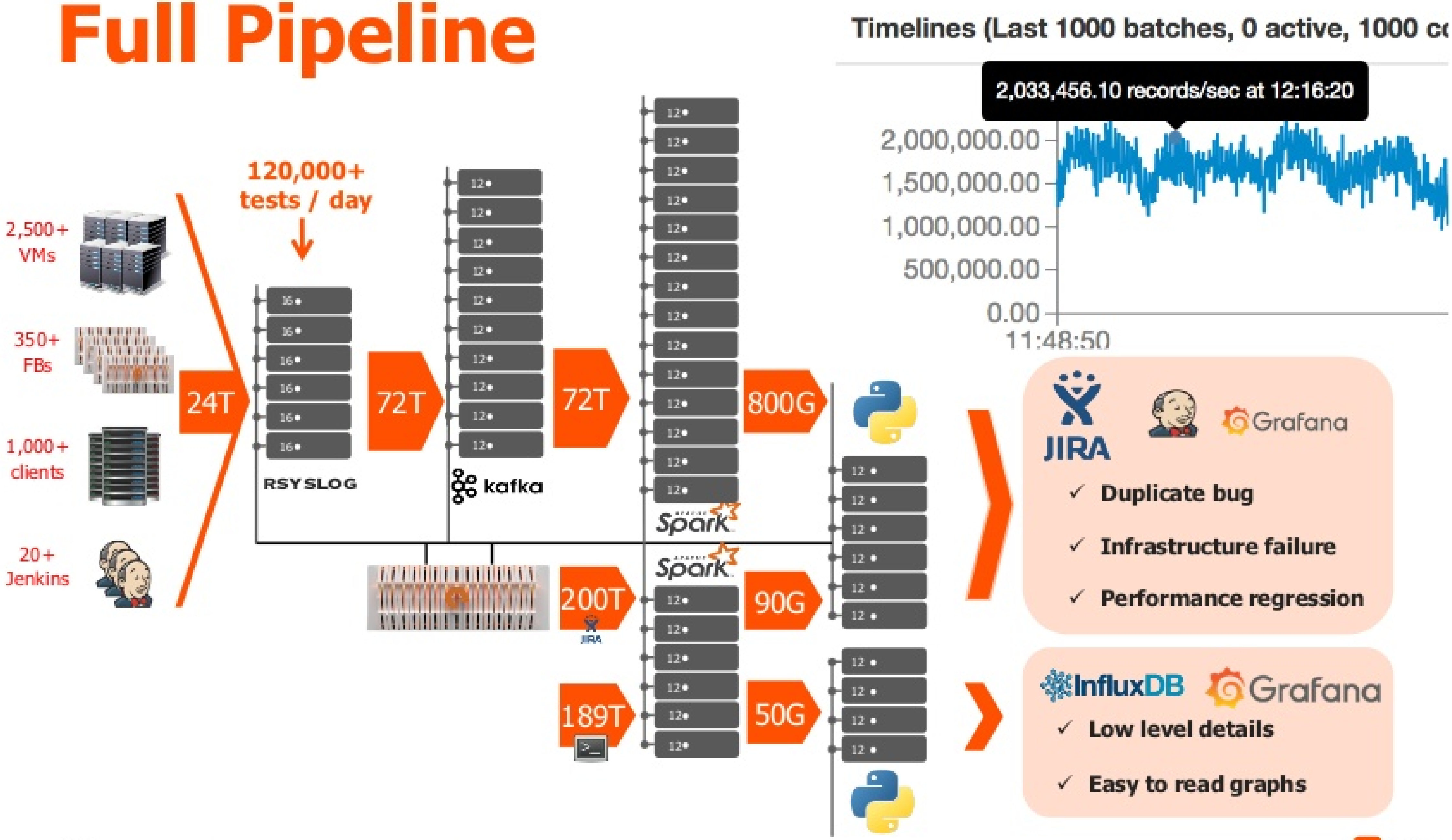
- ✓ **Duplicate bug**
- ✓ **Infrastructure failure**
- ✓ **Performance regression**

Full Pipeline



- ✓ Duplicate bug
- ✓ Infrastructure failure
- ✓ Performance regression

Full Pipeline



JIRA Grafana

- ✓ Duplicate bug
- ✓ Infrastructure failure
- ✓ Performance regression

InfluxDB Grafana

- ✓ Low level details
- ✓ Easy to read graphs

Takeaways

- ✓ **Index only what you need, store the rest**
  (in a storage layer that scales in throughput and to billions of files/objects)
- ✓ **Optimize for throughput and not latency**
- ✓ **Disaggregation of compute and storage for scalability of subsystems**

QUESTIONS?