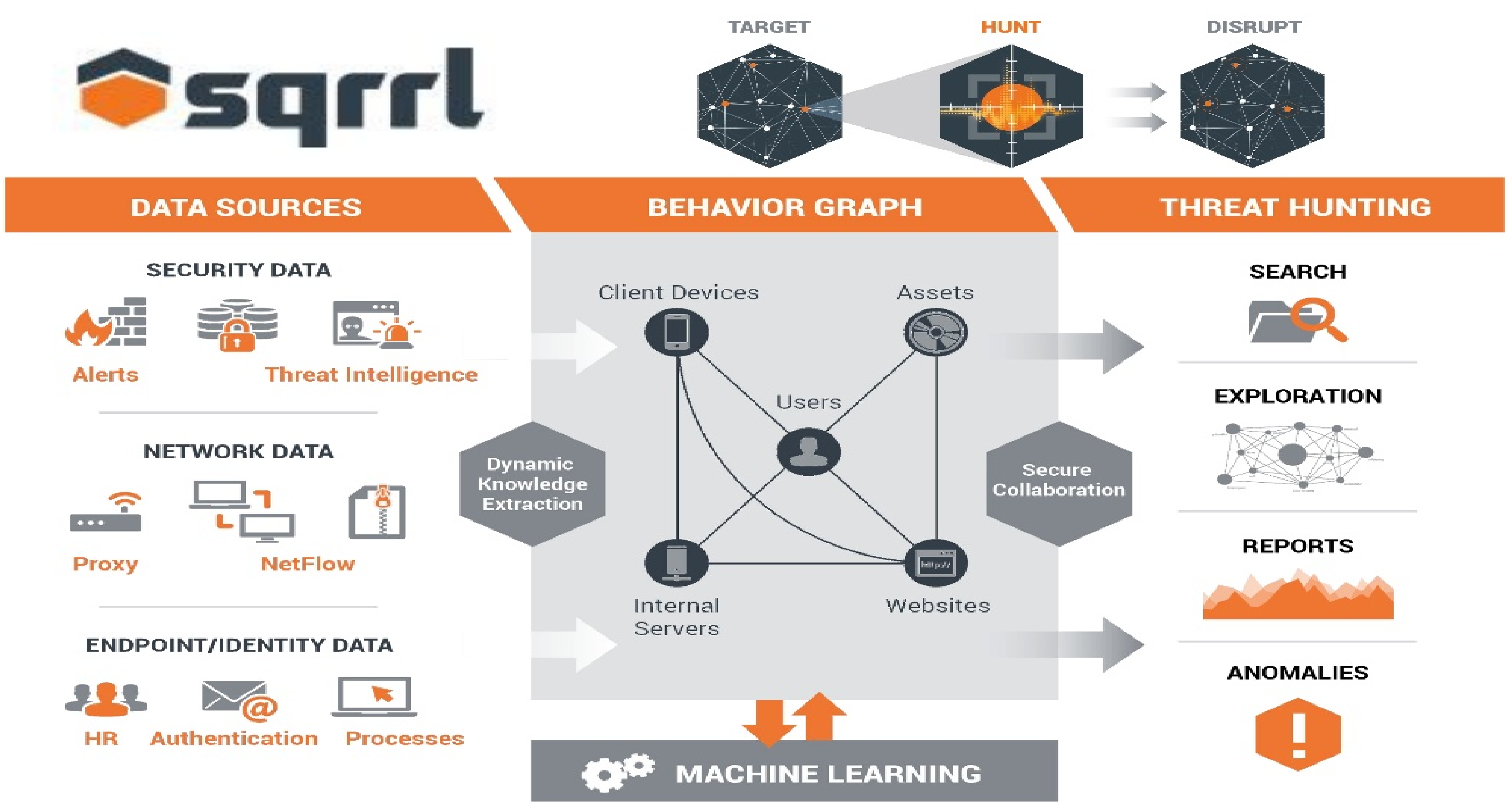
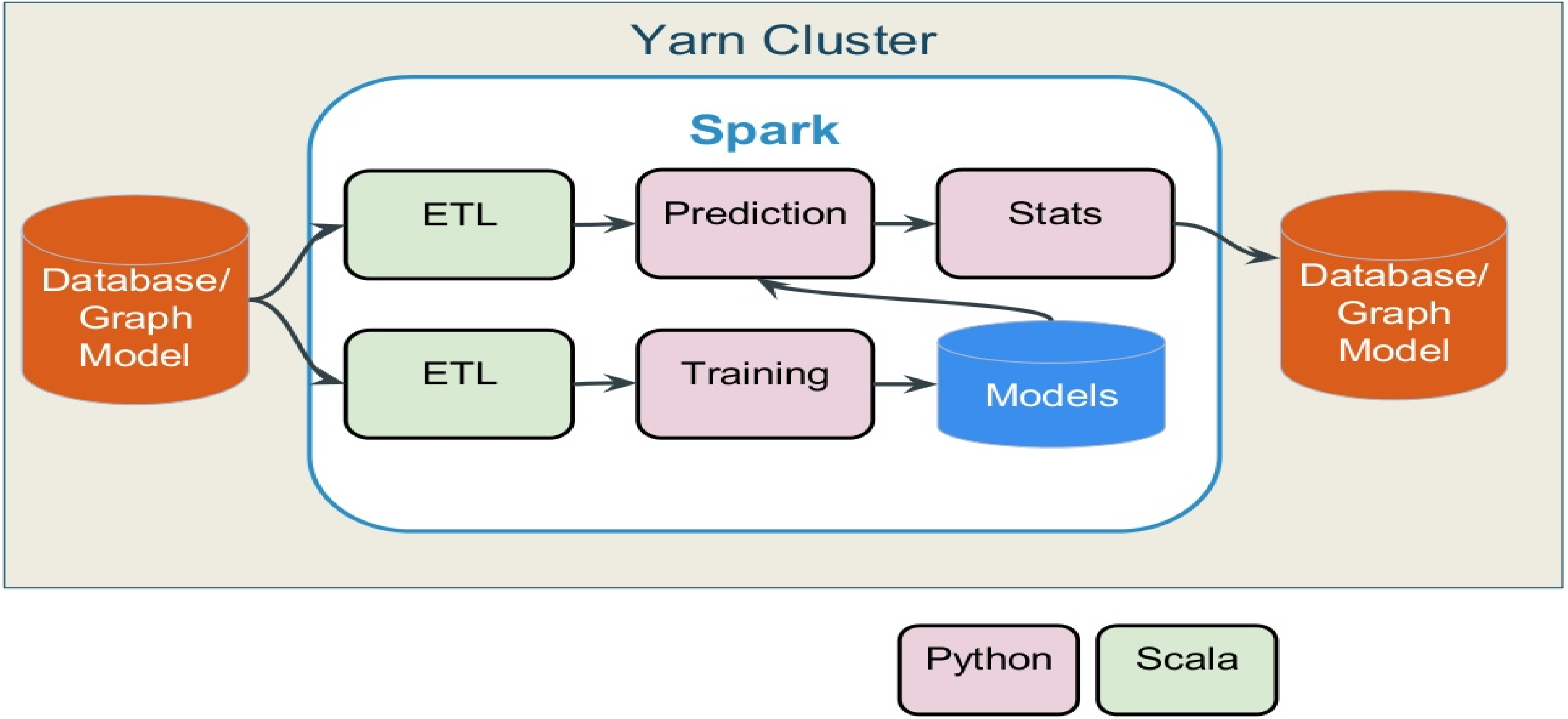
Monitoring the Dynamic Resource Usage of Scala and Python Spark Jobs in Yarn

Ed Barnes, Ruslan Vaulin and Chris McCubbin
Sqrrl Data

SPARK SUMMIT EAST 2017


sqrrl
TARGET
HUNT
DISRUPT
DATA SOURCES
BEHAVIOR GRAPH
THREAT HUNTING
SECURITY DATA
Alerts
Threat Intelligence
NETWORK DATA
Proxy
NetFlow
ENDPOINT/IDENTITY DATA
HR
Authentication
Processes
Client Devices
Assets
Users
Internal Servers
Websites
Dynamic Knowledge Extraction
Secure Collaboration
MACHINE LEARNING
SEARCH
EXPLORATION
REPORTS
ANOMALIES

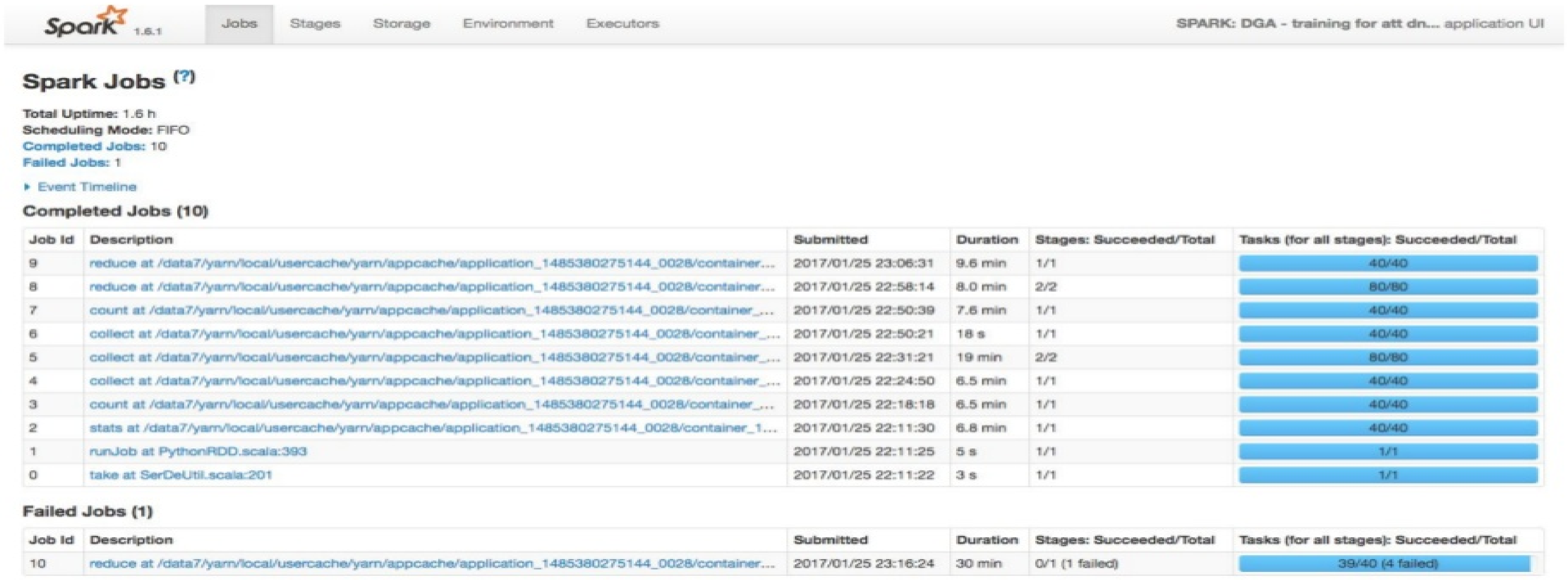
ML Application Workflow



Taking Spark Applications into Production

- Requires execution framework
- Scalable, Robust, Tested
- Test at scale
- Many issues show up only at scale
  - Performance
  - Memory requirements
  - Failures
  - Scaling
- Debugging distributed applications is really hard!

Spark UI: job level



Spark Jobs (?)

Total Uptime: 1.6 h
Scheduling Mode: FIFO
Completed Jobs: 10
Failed Jobs: 1

▸ Event Timeline

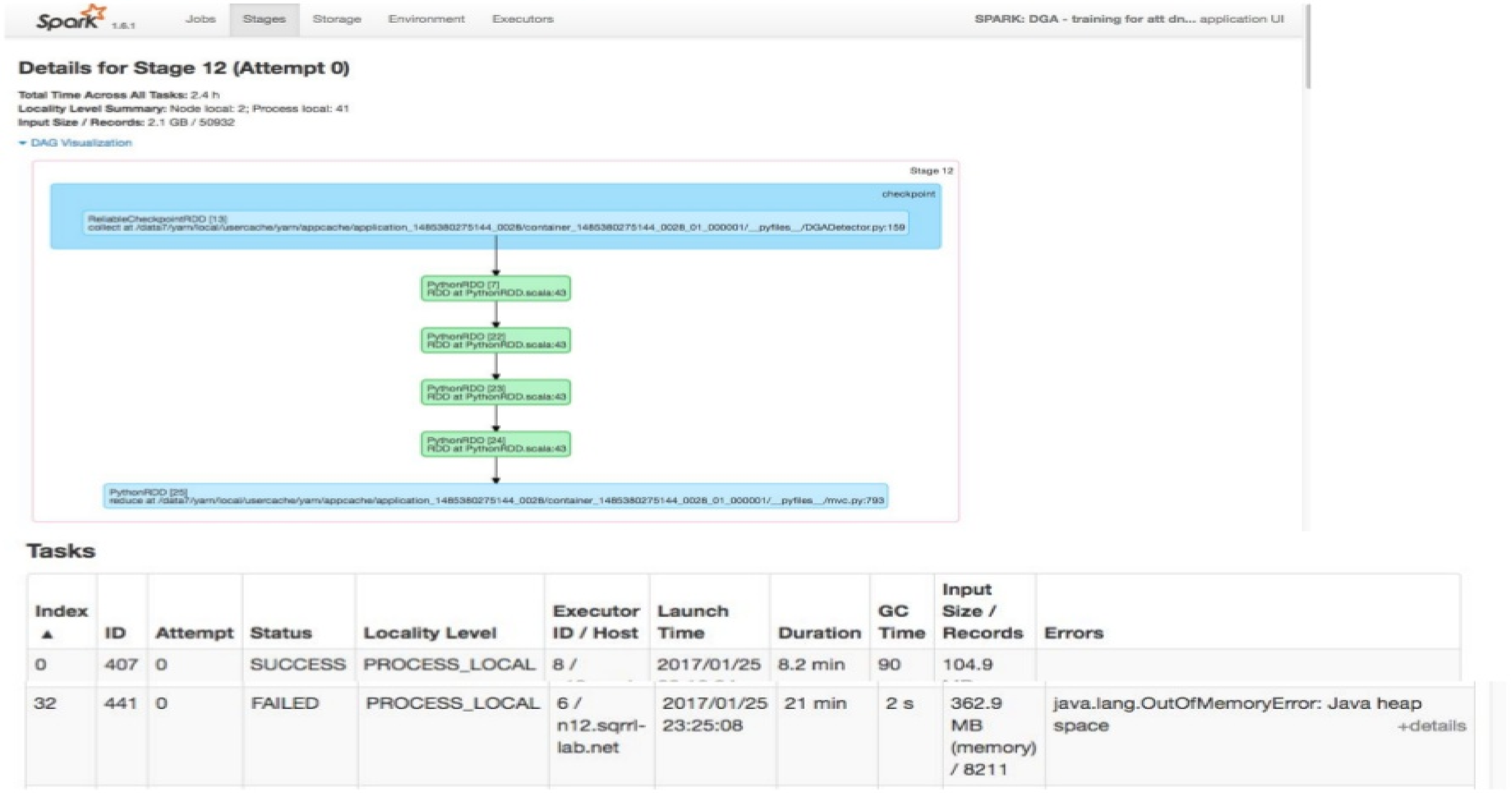
Completed Jobs (10)

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
9	reduce at /data7/yarn/local/usercache/yarn/appcache/application_1485380275144_0028/container...	2017/01/25 23:06:31	9.6 min	1/1	40/40
8	reduce at /data7/yarn/local/usercache/yarn/appcache/application_1485380275144_0028/container...	2017/01/25 22:58:14	8.0 min	2/2	80/80
7	count at /data7/yarn/local/usercache/yarn/appcache/application_1485380275144_0028/container_...	2017/01/25 22:50:39	7.6 min	1/1	40/40
6	collect at /data7/yarn/local/usercache/yarn/appcache/application_1485380275144_0028/container_...	2017/01/25 22:50:21	18 s	1/1	40/40
5	collect at /data7/yarn/local/usercache/yarn/appcache/application_1485380275144_0028/container_...	2017/01/25 22:31:21	19 min	2/2	80/80
4	collect at /data7/yarn/local/usercache/yarn/appcache/application_1485380275144_0028/container_...	2017/01/25 22:24:50	6.5 min	1/1	40/40
3	count at /data7/yarn/local/usercache/yarn/appcache/application_1485380275144_0028/container_...	2017/01/25 22:18:18	6.5 min	1/1	40/40
2	stats at /data7/yarn/local/usercache/yarn/appcache/application_1485380275144_0028/container_1...	2017/01/25 22:11:30	6.8 min	1/1	40/40
1	runJob at PythonRDD.scala:393	2017/01/25 22:11:25	5 s	1/1	1/1
0	take at SerDeUtil.scala:201	2017/01/25 22:11:22	3 s	1/1	1/1

Failed Jobs (1)

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
10	reduce at /data7/yarn/local/usercache/yarn/appcache/application_1485380275144_0028/container...	2017/01/25 23:16:24	30 min	0/1 (1 failed)	39/40 (4 failed)

Spark UI: task level



Details for Stage 12 (Attempt 0)

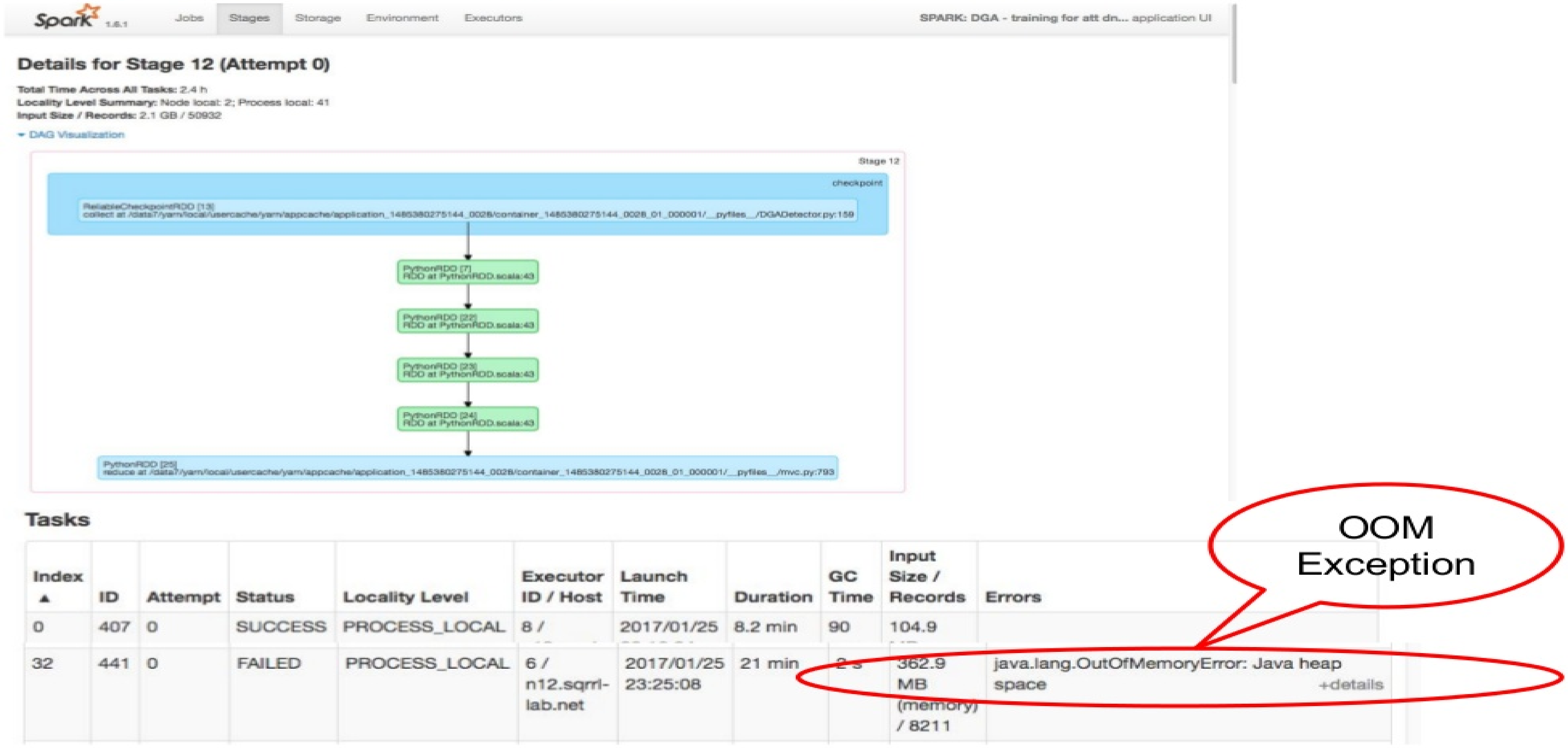
Total Time Across All Tasks: 2.4 h
Locality Level Summary: Node local: 2; Process local: 41
Input Size / Records: 2.1 GB / 50932

```
ReliableCheckpointRDD [13]
collect at /data7/yarn/local/usercache/yarn/appcache/application_1485380275144_0028/container_1485380275144_0028_01_000001/__pyfiles__/DGADetector.py:159

PythonRDD [7]
RDD at PythonRDD.scala:43

PythonRDD [22]
RDD at PythonRDD.scala:43

PythonRDD [23]
RDD at PythonRDD.scala:43

PythonRDD [24]
RDD at PythonRDD.scala:43

PythonRDD [25]
reduce at /data7/yarn/local/usercache/yarn/appcache/application_1485380275144_0028/container_1485380275144_0028_01_000001/__pyfiles__/mvc.py:793
```

Tasks

Index ▲	ID	Attempt	Status	Locality Level	Executor ID / Host	Launch Time	Duration	GC Time	Input Size / Records	Errors
0	407	0	SUCCESS	PROCESS_LOCAL	8 /	2017/01/25	8.2 min	90	104.9	
32	441	0	FAILED	PROCESS_LOCAL	6 / n12.sqrrl-lab.net	2017/01/25 23:25:08	21 min	2 s	362.9 MB (memory) / 8211	java.lang.OutOfMemoryError: Java heap space +details

Spark UI: task level



Details for Stage 12 (Attempt 0)

Total Time Across All Tasks: 2.4 h
Locality Level Summary: Node local: 2; Process local: 41
Input Size / Records: 2.1 GB / 50932

▾ DAG Visualization

Stage 12

checkpoint

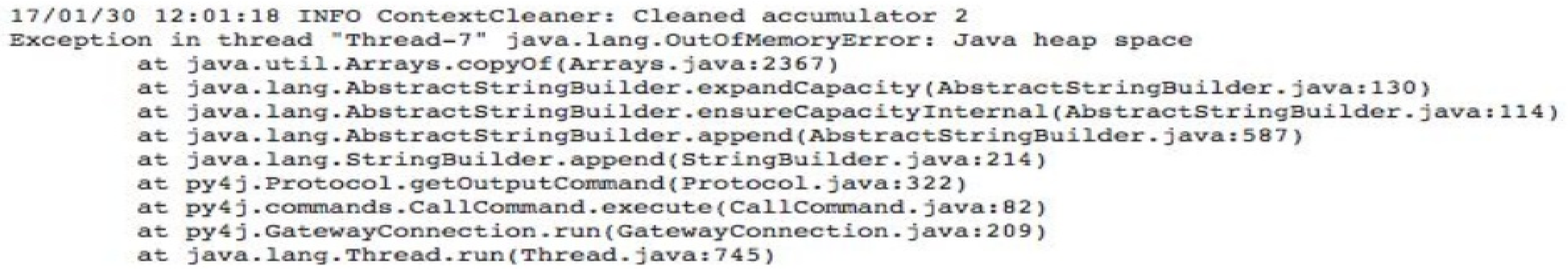
```
ReliableCheckpointRDD [13]
collect at /data7/yarn/local/usercache/yarn/appcache/application_1485380275144_0028/container_1485380275144_0028_01_000001/__pyfiles__/DGADetector.py:159

PythonRDD [7]
RDD at PythonRDD.scala:43

PythonRDD [22]
RDD at PythonRDD.scala:43

PythonRDD [23]
RDD at PythonRDD.scala:43

PythonRDD [24]
RDD at PythonRDD.scala:43

PythonRDD [25]
reduce at /data7/yarn/local/usercache/yarn/appcache/application_1485380275144_0028/container_1485380275144_0028_01_000001/__pyfiles__/mvc.py:793
```

Tasks

Index ▲	ID	Attempt	Status	Locality Level	Executor ID / Host	Launch Time	Duration	GC Time	Input Size / Records	Errors
0	407	0	SUCCESS	PROCESS_LOCAL	8 /	2017/01/25	8.2 min	90	104.9	
32	441	0	FAILED	PROCESS_LOCAL	6 / n12.sqrrl-lab.net	2017/01/25 23:25:08	21 min	2 s	362.9 MB (memory) / 8211	java.lang.OutOfMemoryError: Java heap space +details

Case Study: Py4J Issue

- Testing engineer - “Your code blows up with OOM when processing X amount of data!”

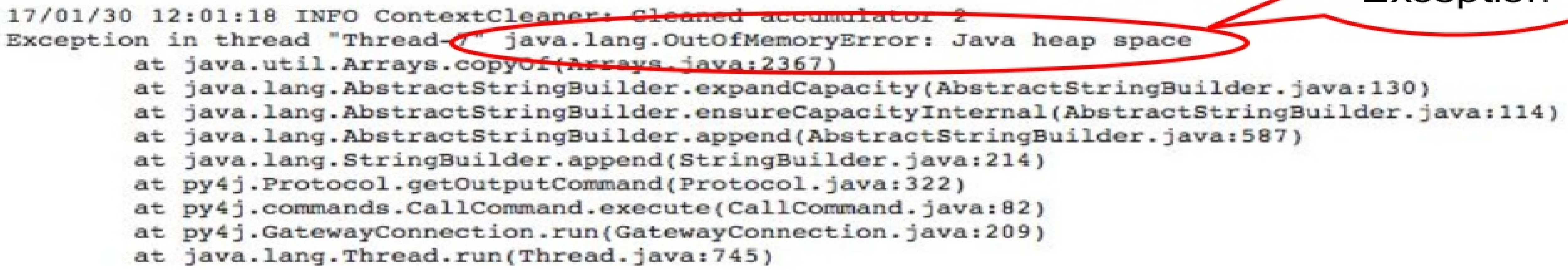


```
17/01/30 12:01:18 INFO ContextCleaner: Cleaned accumulator 2
Exception in thread "Thread-7" java.lang.OutOfMemoryError: Java heap space
        at java.util.Arrays.copyOf(Arrays.java:2367)
        at java.lang.AbstractStringBuilder.expandCapacity(AbstractStringBuilder.java:130)
        at java.lang.AbstractStringBuilder.ensureCapacityInternal(AbstractStringBuilder.java:114)
        at java.lang.AbstractStringBuilder.append(AbstractStringBuilder.java:587)
        at java.lang.StringBuilder.append(StringBuilder.java:214)
        at py4j.Protocol.getOutputCommand(Protocol.java:322)
        at py4j.commands.CallCommand.execute(CallCommand.java:82)
        at py4j.GatewayConnection.run(GatewayConnection.java:209)
        at java.lang.Thread.run(Thread.java:745)
```

- Why?!!!!
- Nothing obvious in Spark UI!

Case Study: Py4J Issue

- Testing engineer - “Your code blows up with OOM when processing X amount of data!”



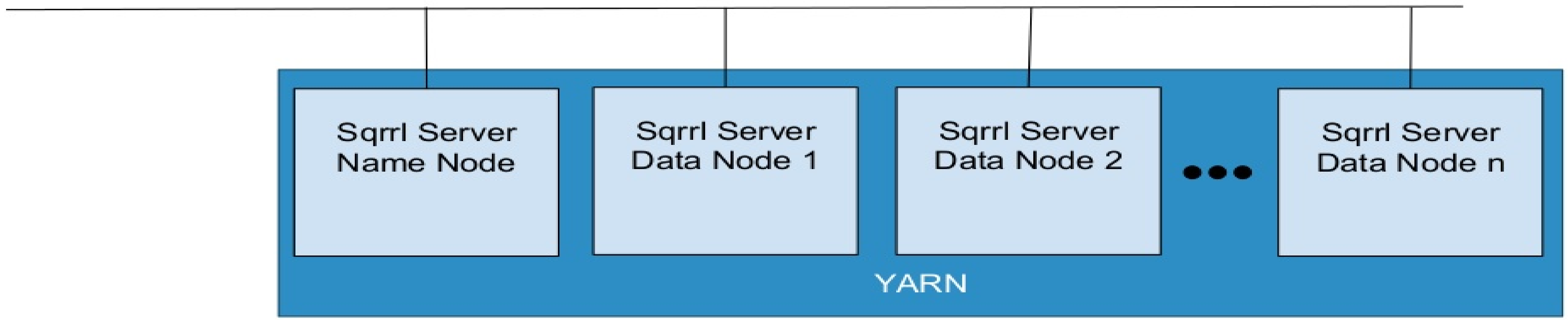
```
17/01/30 12:01:18 INFO ContextCleaner: Cleaned accumulator 2
Exception in thread "Thread-7" java.lang.OutOfMemoryError: Java heap space
        at java.util.Arrays.copyOf(Arrays.java:2367)
        at java.lang.AbstractStringBuilder.expandCapacity(AbstractStringBuilder.java:130)
        at java.lang.AbstractStringBuilder.ensureCapacityInternal(AbstractStringBuilder.java:114)
        at java.lang.AbstractStringBuilder.append(AbstractStringBuilder.java:587)
        at java.lang.StringBuilder.append(StringBuilder.java:214)
        at py4j.Protocol.getOutputCommand(Protocol.java:322)
        at py4j.commands.CallCommand.execute(CallCommand.java:82)
        at py4j.GatewayConnection.run(GatewayConnection.java:209)
        at java.lang.Thread.run(Thread.java:745)
```

OOM Exception

- Why?!!!!
- Nothing obvious in Spark UI!

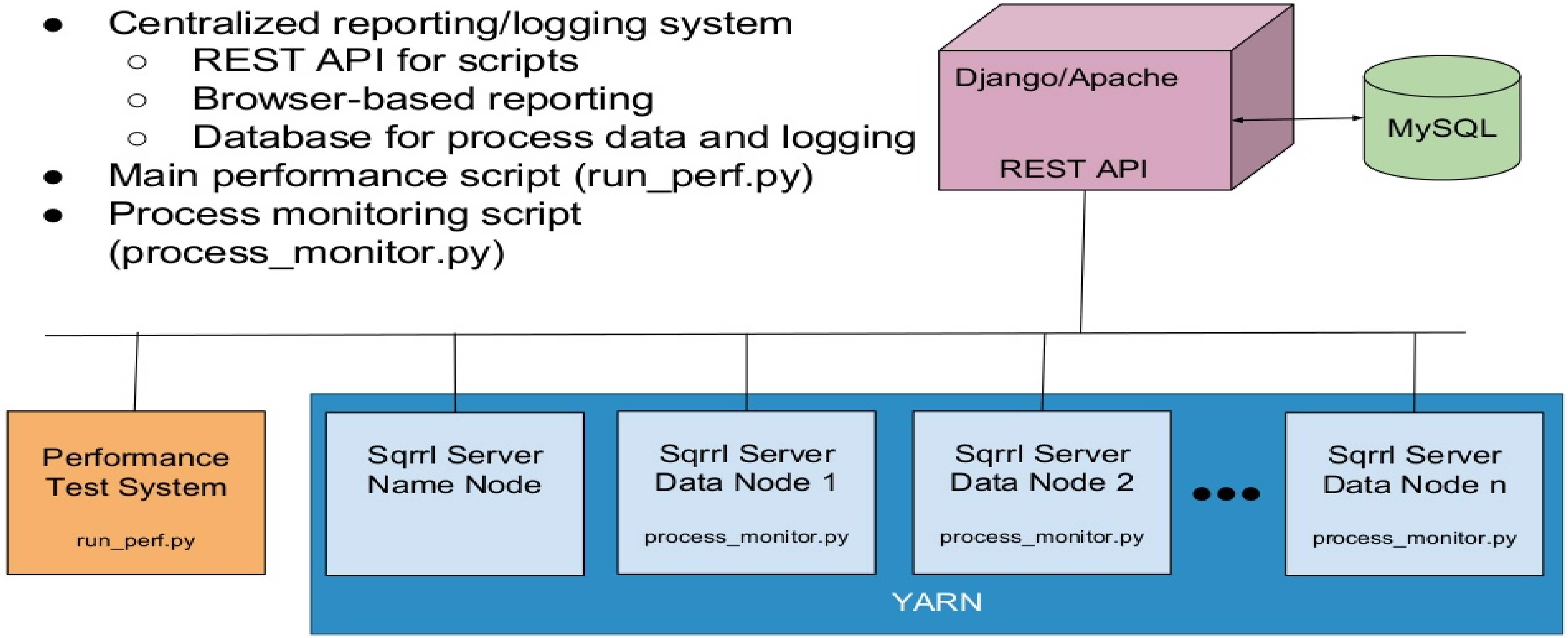
Tooling Approach Requirements

- Cluster-wide process monitoring
- Per-node, per-process statistics
- Identification of high CPU, memory usage
- Record process hierarchy/timing of Spark jobs under YARN
- Characterize scaling behaviors for production/releases
- Integrate with internal processes/harnesses for general development and test use



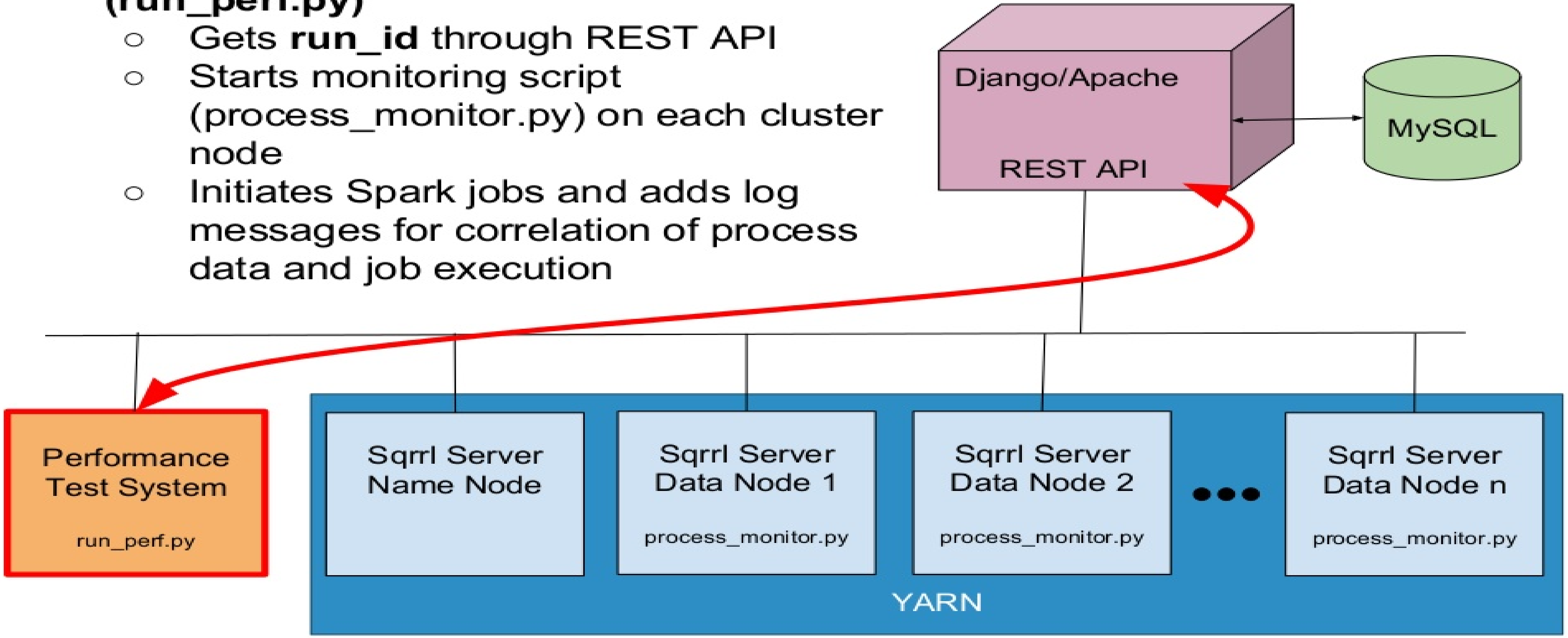
Overall System Design

- Centralized reporting/logging system
  - REST API for scripts
  - Browser-based reporting
  - Database for process data and logging
- Main performance script (run_perf.py)
- Process monitoring script (process_monitor.py)



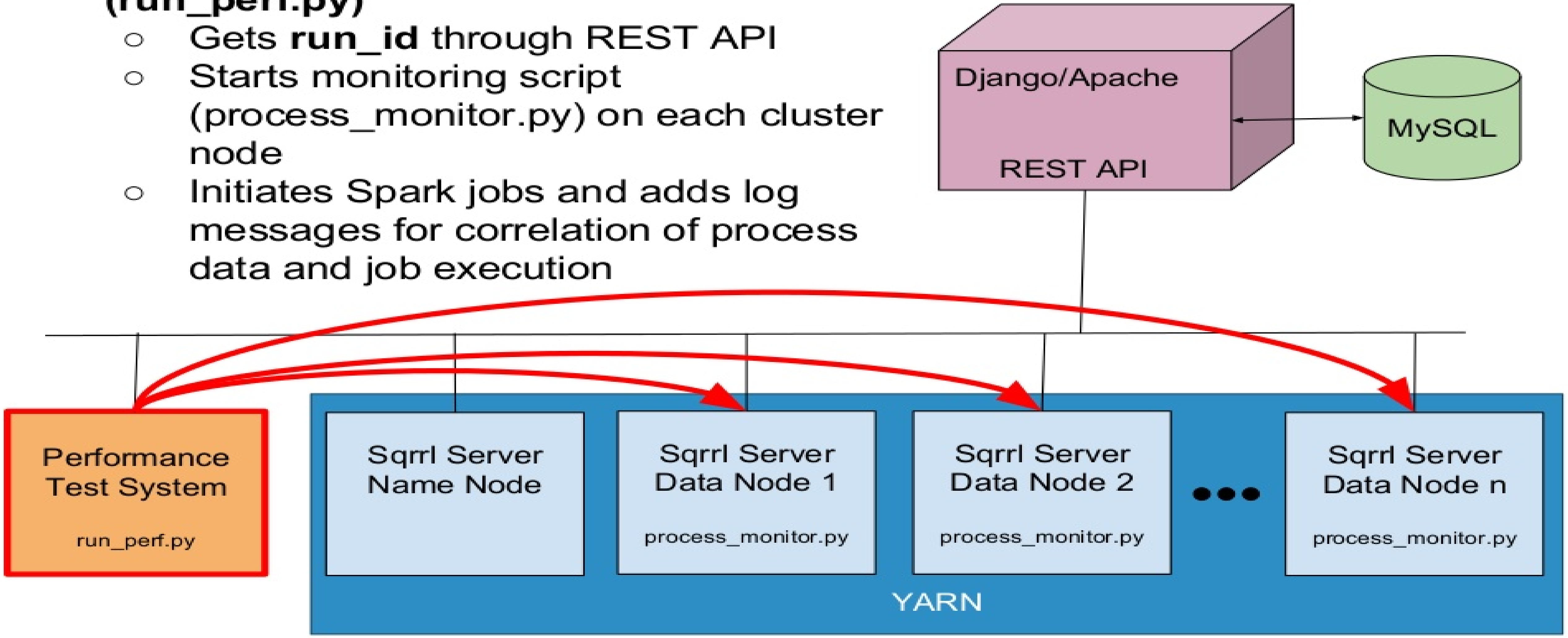
Scripts/Operation

- **Main performance harness (run_perf.py)**
  - Gets **run_id** through REST API
  - Starts monitoring script (process_monitor.py) on each cluster node
  - Initiates Spark jobs and adds log messages for correlation of process data and job execution



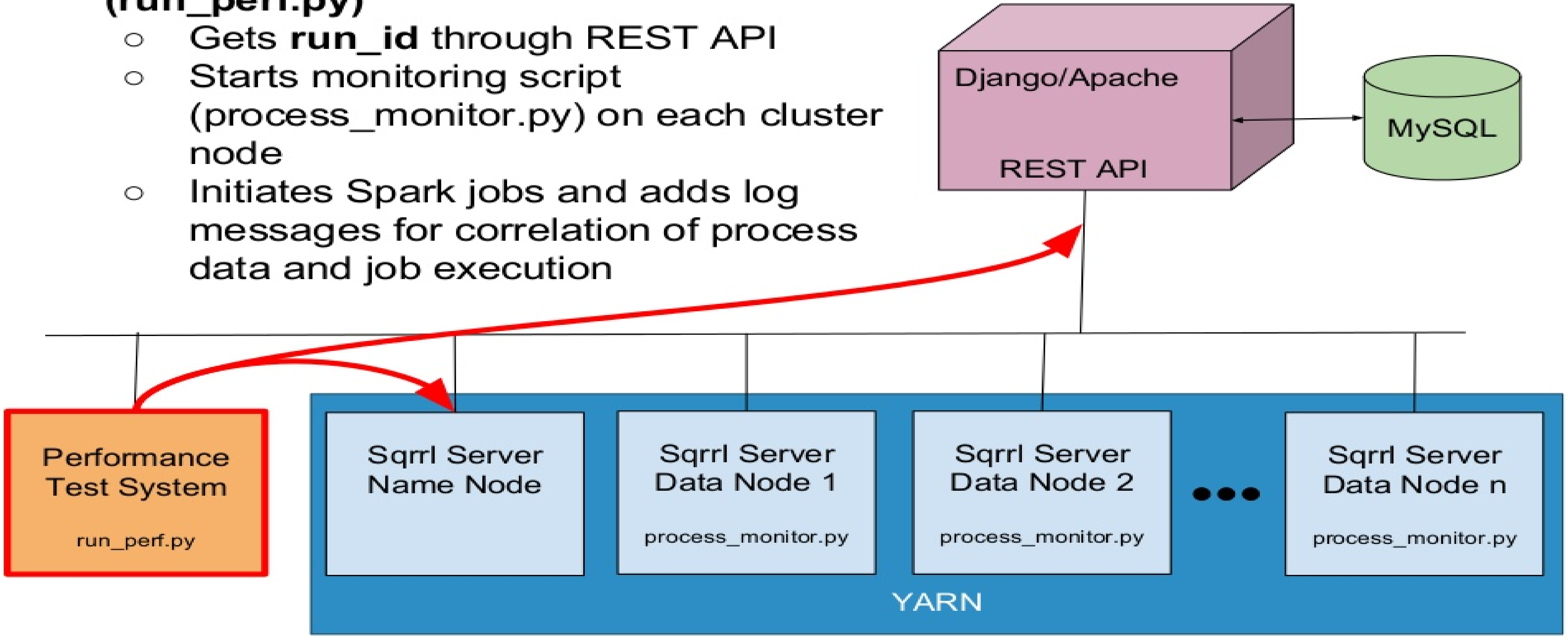
Scripts/Operation

- **Main performance harness (run_perf.py)**
  - Gets **run_id** through REST API
  - Starts monitoring script (process_monitor.py) on each cluster node
  - Initiates Spark jobs and adds log messages for correlation of process data and job execution



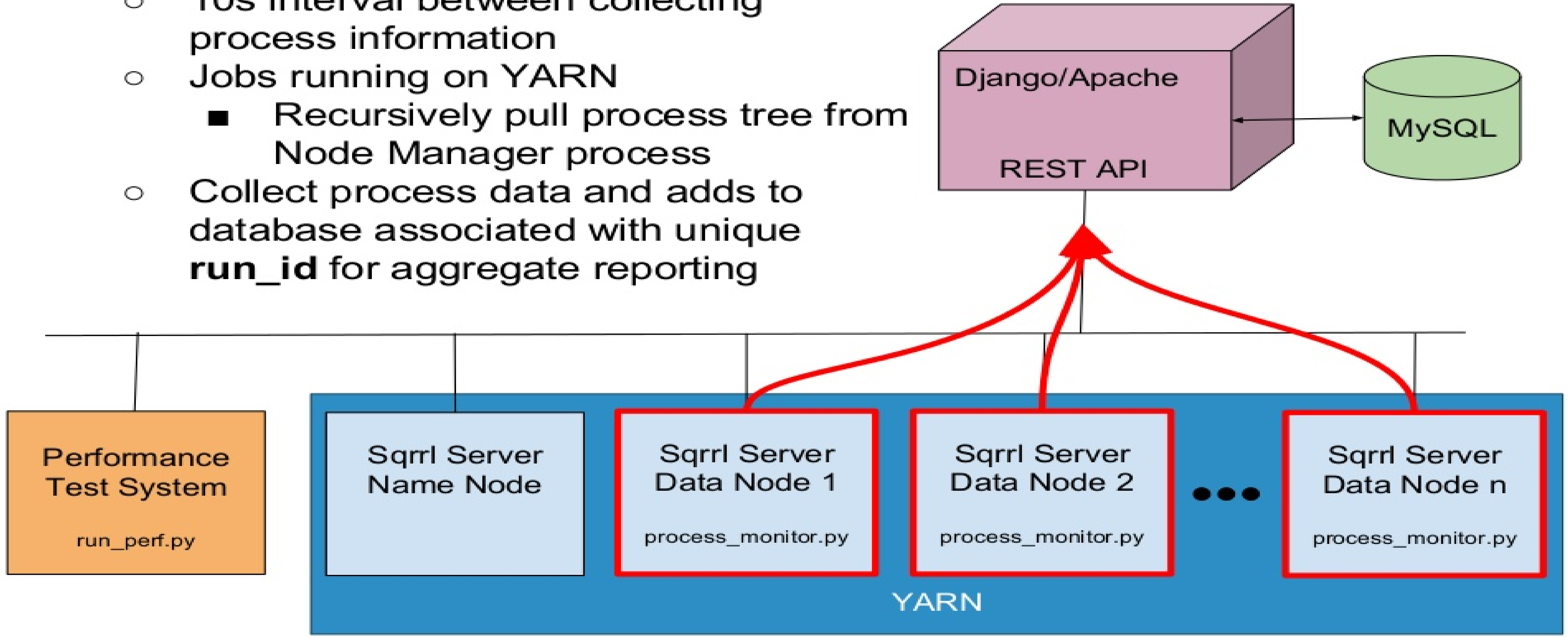
Scripts/Operation

- **Main performance harness (run_perf.py)**
  - Gets **run_id** through REST API
  - Starts monitoring script (process_monitor.py) on each cluster node
  - Initiates Spark jobs and adds log messages for correlation of process data and job execution

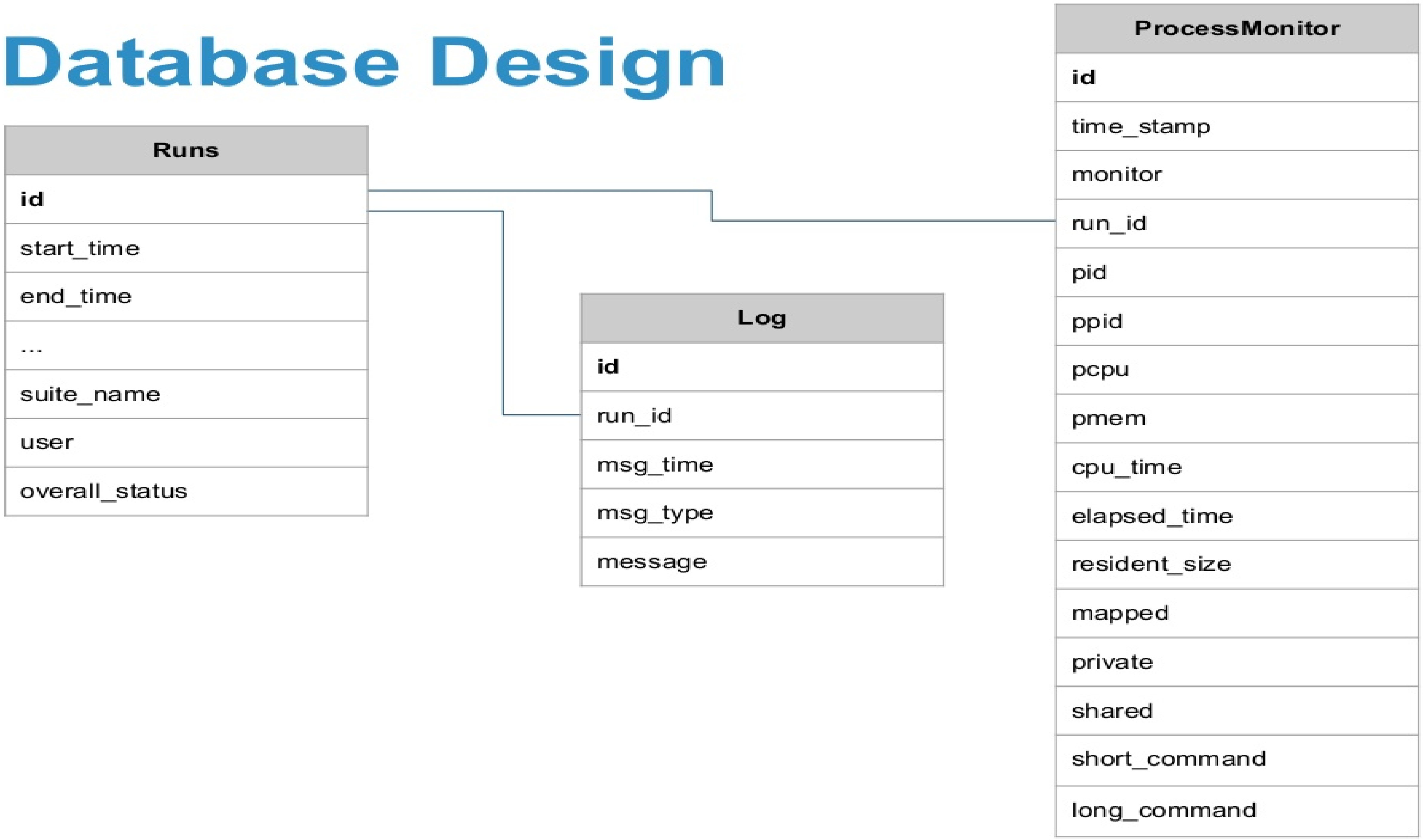


Scripts/Operation

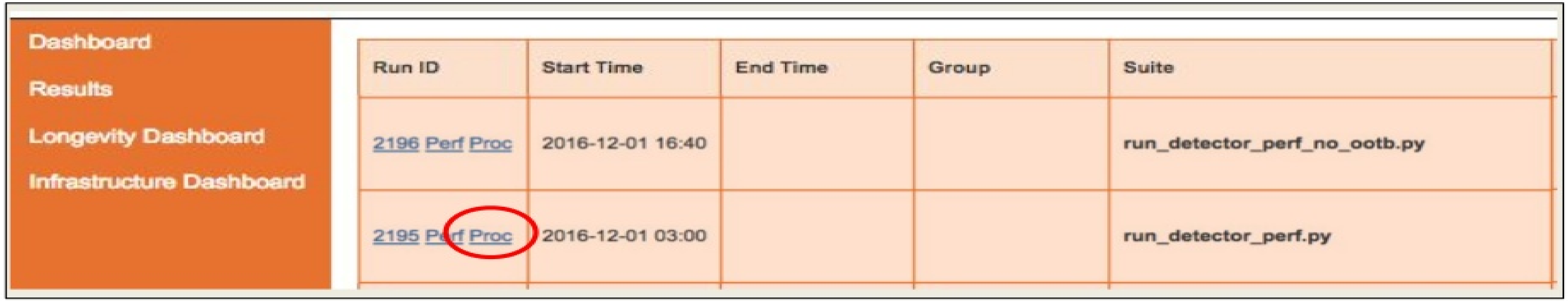
- **Monitoring script (process_monitor.py)**
  - 10s interval between collecting process information
  - Jobs running on YARN
    - Recursively pull process tree from Node Manager process
  - Collect process data and adds to database associated with unique **run_id** for aggregate reporting



Database Design

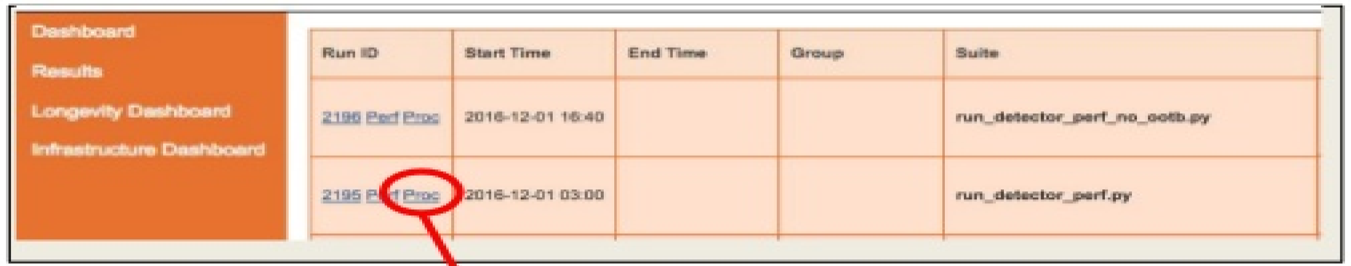


User Experience

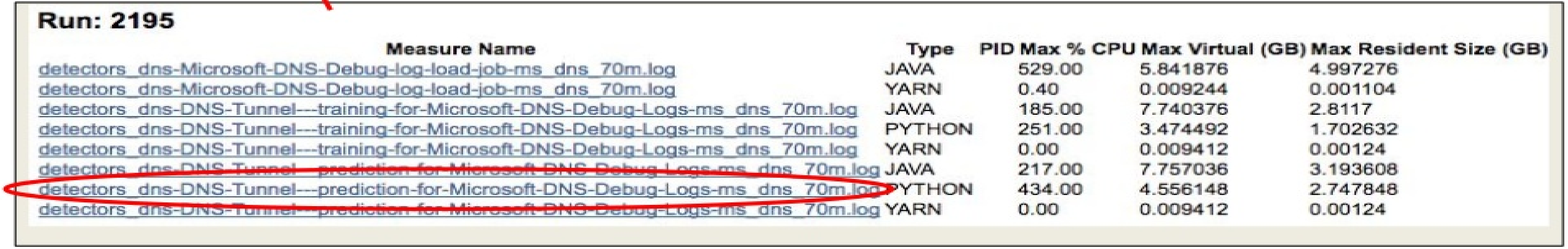


Run ID	Start Time	End Time	Group	Suite
2196 Perf Proc	2016-12-01 16:40			run_detector_perf_no_ootb.py
2195 Perf Proc	2016-12-01 03:00			run_detector_perf.py

User Experience



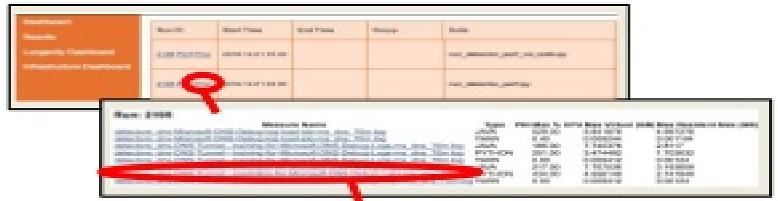
Run ID	Start Time	End Time	Group	Suite
2196 Perf Proc	2016-12-01 16:40			run_detector_perf_no_ootb.py
2195 Perf Proc	2016-12-01 03:00			run_detector_perf.py

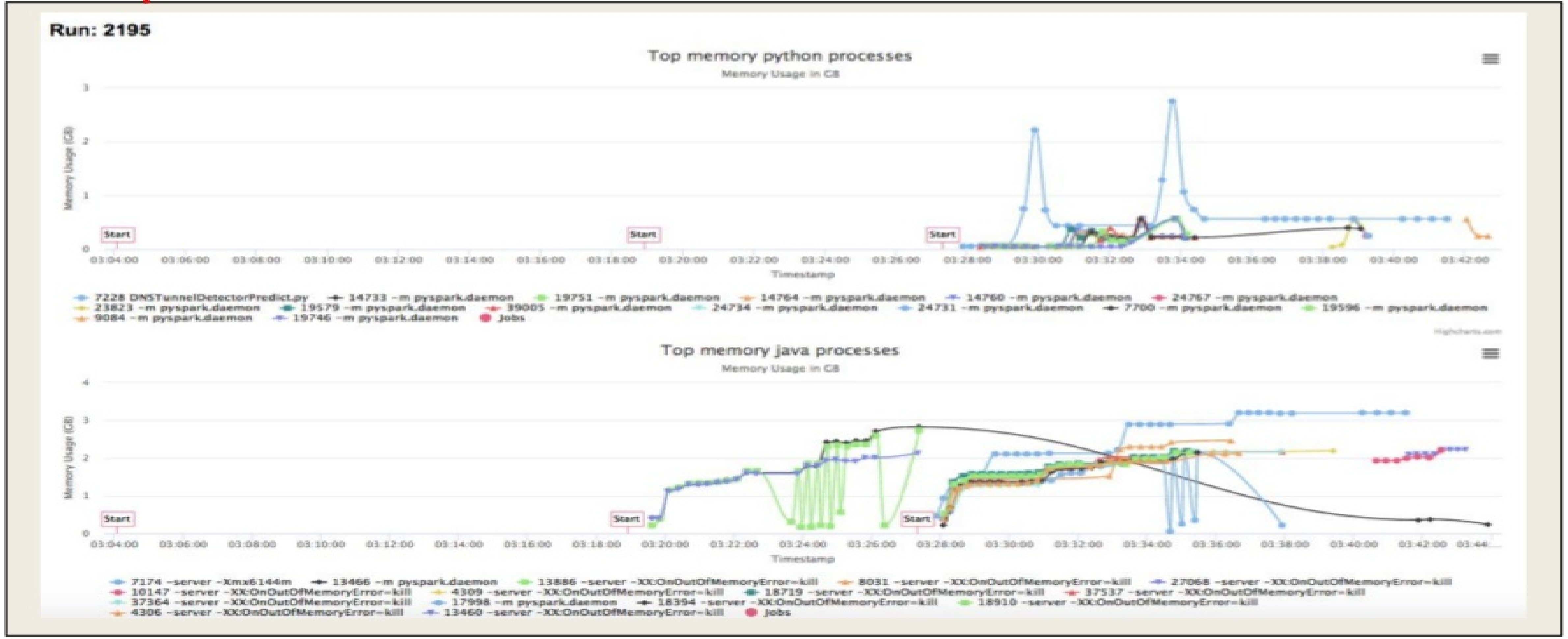


Run: 2195

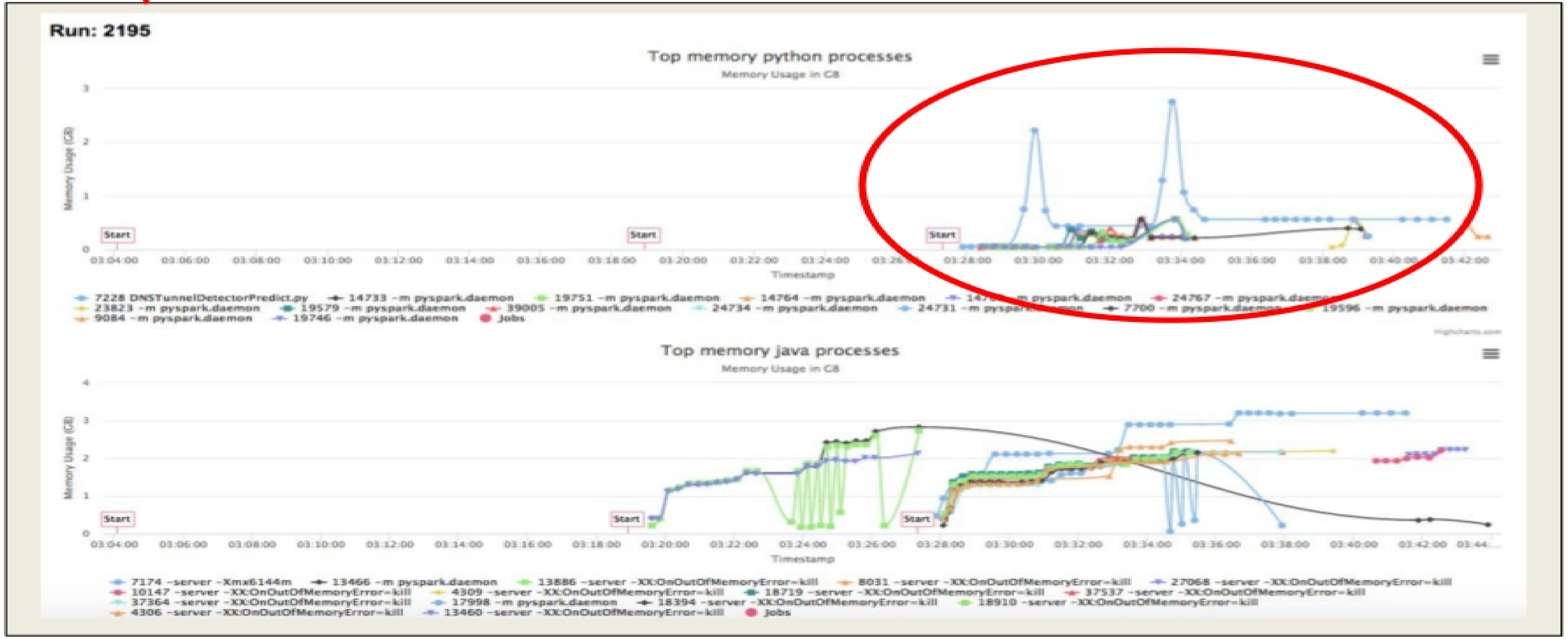
Measure Name	Type	PID Max % CPU	Max Virtual (GB)	Max Resident Size (GB)
detectors_dns-Microsoft-DNS-Debug-log-load-job-ms_dns_70m.log	JAVA	529.00	5.841876	4.997276
detectors_dns-Microsoft-DNS-Debug-log-load-job-ms_dns_70m.log	YARN	0.40	0.009244	0.001104
detectors_dns-DNS-Tunnel---training-for-Microsoft-DNS-Debug-Logs-ms_dns_70m.log	JAVA	185.00	7.740376	2.8117
detectors_dns-DNS-Tunnel---training-for-Microsoft-DNS-Debug-Logs-ms_dns_70m.log	PYTHON	251.00	3.474492	1.702632
detectors_dns-DNS-Tunnel---training-for-Microsoft-DNS-Debug-Logs-ms_dns_70m.log	YARN	0.00	0.009412	0.00124
detectors_dns-DNS-Tunnel---prediction-for-Microsoft-DNS-Debug-Logs-ms_dns_70m.log	JAVA	217.00	7.757036	3.193608
detectors_dns-DNS-Tunnel---prediction-for-Microsoft-DNS-Debug-Logs-ms_dns_70m.log	PYTHON	434.00	4.556148	2.747848
detectors_dns-DNS-Tunnel---prediction-for-Microsoft-DNS-Debug-Logs-ms_dns_70m.log	YARN	0.00	0.009412	0.00124

User Experience



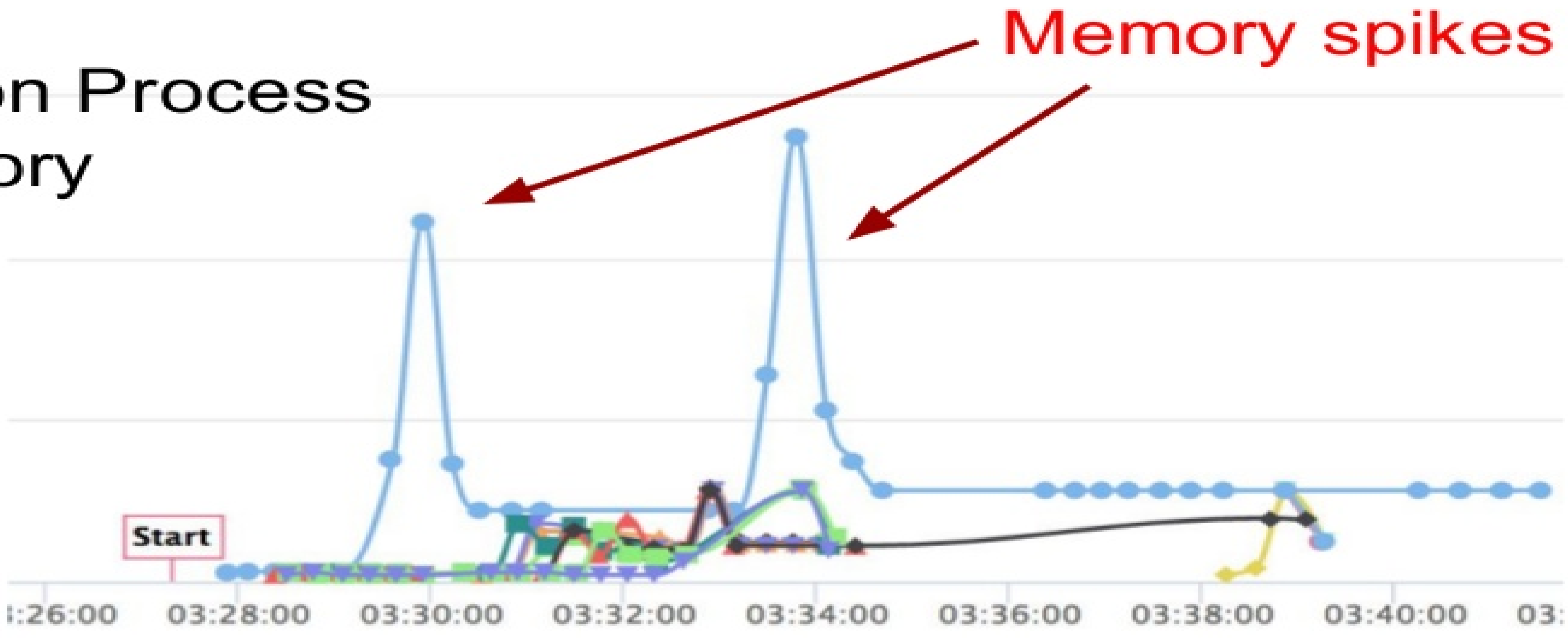

Run: 2195
Top memory python processes
Memory Usage in GB
Memory Usage (GB)
Timestamp
Start
7228 DNSTunnelDetectorPredict.py
14733 -m pyspark.daemon
19751 -m pyspark.daemon
14764 -m pyspark.daemon
14760 -m pyspark.daemon
24767 -m pyspark.daemon
23823 -m pyspark.daemon
19579 -m pyspark.daemon
39005 -m pyspark.daemon
24734 -m pyspark.daemon
24731 -m pyspark.daemon
7700 -m pyspark.daemon
19596 -m pyspark.daemon
9084 -m pyspark.daemon
19746 -m pyspark.daemon
Jobs
Highcharts.com
Top memory java processes
Memory Usage in GB
Memory Usage (GB)
Timestamp
Start
7174 -server -Xmx6144m
13466 -m pyspark.daemon
13886 -server -XX:OnOutOfMemoryError=kill
8031 -server -XX:OnOutOfMemoryError=kill
27068 -server -XX:OnOutOfMemoryError=kill
10147 -server -XX:OnOutOfMemoryError=kill
4309 -server -XX:OnOutOfMemoryError=kill
18719 -server -XX:OnOutOfMemoryError=kill
37537 -server -XX:OnOutOfMemoryError=kill
37364 -server -XX:OnOutOfMemoryError=kill
17998 -m pyspark.daemon
18394 -server -XX:OnOutOfMemoryError=kill
18910 -server -XX:OnOutOfMemoryError=kill
4306 -server -XX:OnOutOfMemoryError=kill
13460 -server -XX:OnOutOfMemoryError=kill
Jobs

User Experience



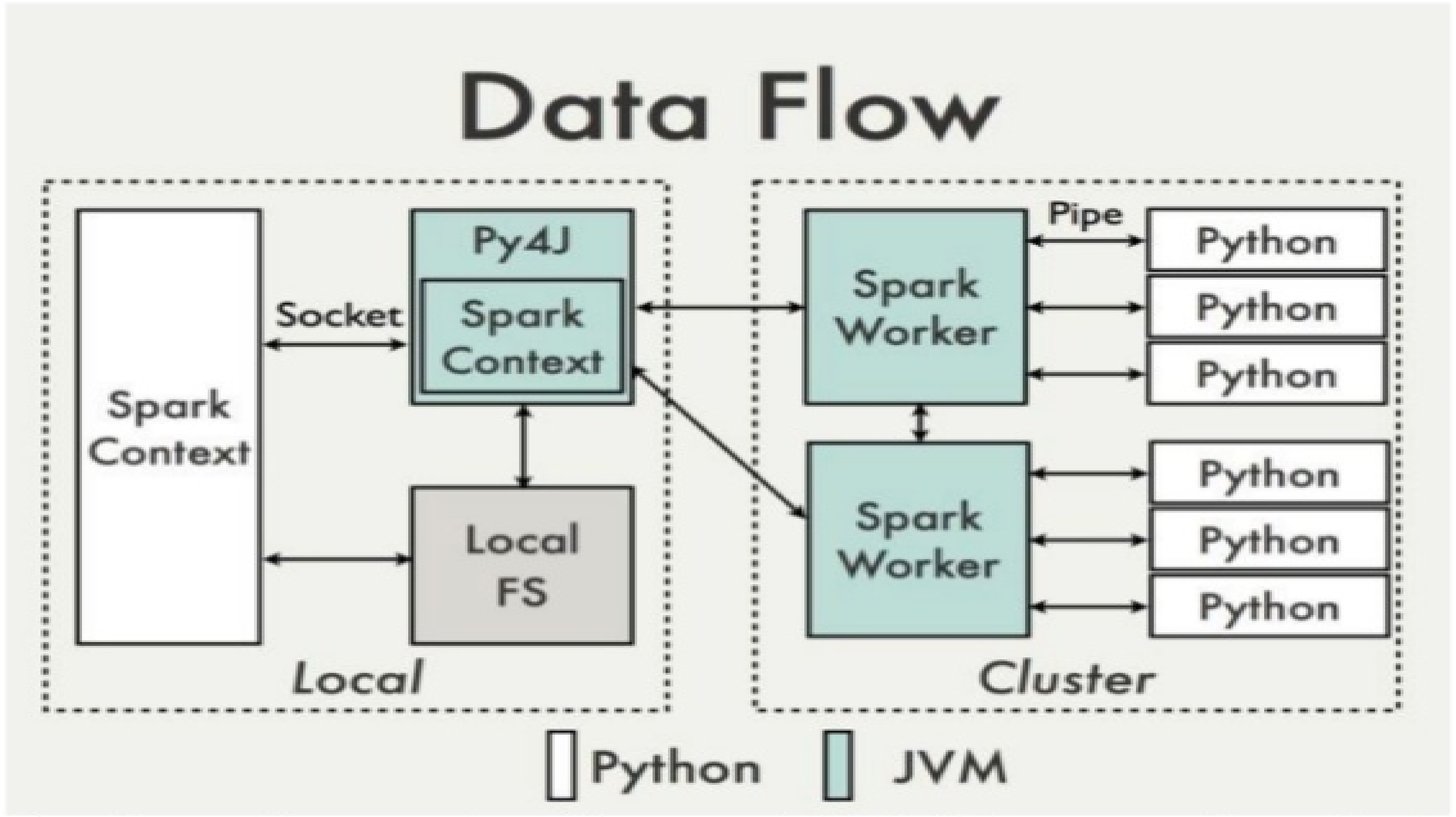
Py4J Issue: Evidence

Python Process Memory



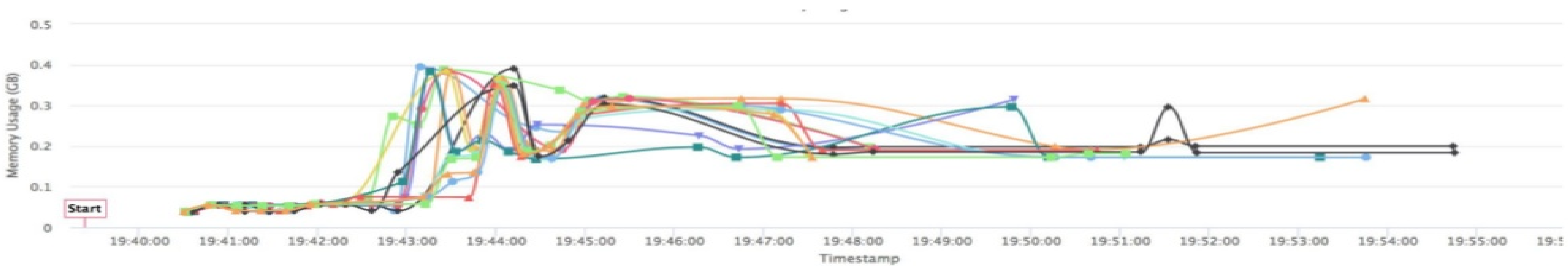
- Memory spiked during loading of trained ML models into python process.
- Only on driver!

Diagnosis and Conclusions



- Loading trained model from HDFS required decryption with java libraries and Py4J for loading into python.
- Py4J protocol inflated size by almost 100x!
- Solution: implement custom protocol for loading data from jvm to python via socket.

Py4J issue: After Fix



- Memory spikes are gone!
- Ready for production!

Recommendations & Lessons Learned

- Do not take scalability for granted!
- Understand Spark's architecture
  - Python/JVM interaction
- Follow best practices
  - Iterators not Lists
  - Careful with joins
- Understand your computing demands
- Test at scale
- Invest in tools
- Think distributed and your code will shine!

Thank You.

sqrrl

sqrrl.com
@SqrrlData

ebarnes@sqrrl.com
ruslan@sqrrl.com
chris@sqrrl.com

SPARK SUMMIT EAST 2017