Lessons learned from designing a QA automation event streaming platform (IoT & Big Data)

Omid Vahdaty, Big Data ninja.

Agenda

- Introduction to Big Data Testing
- Challenges of Big Data Testing.
- Methodological approach for QA Automation road map
- True Story
- The Solution?
- Performance and Maintenance

Introduction to Big Data Testing

How Big is Big Data?

- 100M ?
- 1B ?
- 10B?
- 100B?
- 1Tr?

How Hard Can it be (aggregation)?

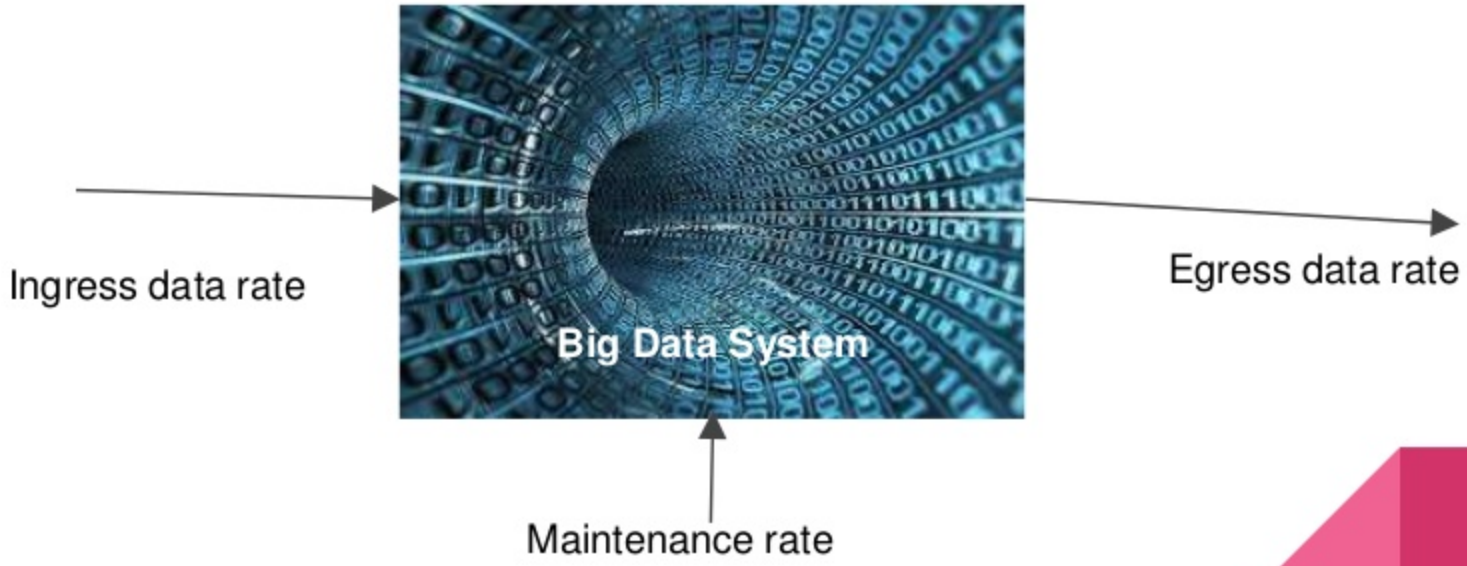
- Select Count(*) from t;
  - Assume
    - 1 Trillion records
    - ad-hoc query (no indexes)
    - Full Scan (no cache)
  - Challenges
    - Time it takes to compute?
    - IO bottleneck? What is IO pattern?
    - CPU bottleneck?

How Hard Can it be (join)?

- Select * from t1 join t2 where t1.id =t2.id
  - Assume
    - 1 Trillion records both tables.
    - ad-hoc query (no indexes)
    - Full Scan (no cache)
  - Challenges
    - Time it takes to compute?
    - IO bottleneck? What is IO pattern?
    - CPU bottleneck?

Big Data Challenges

3 Pain Points in Big Data DBMS



Big Data Ingress Challenges

- Parsing
- Rate and Amount of data coming into the system
- ACID: **Atomicity, Consistency, Isolation, Durability**
- Compression on the fly (non unique values)
- On the fly analytics
- Time constraints (target: x rows per sec/hour)
- High Availability Scenarios
  - **Can you afford to lost evetns/data?**

Big Data egress challenges

- Sort
- Group by
- Reduce
- Join (sortMergeJoin)
- Data distribution
- Compression
- Theoretical Bottlenecks of hardware.

What is your engine algorithm doing?
What is the impact on the system?
CPU?
IO?
RAM?
Network?
What is the impact on the cluster?

Egress while Ingress Challenges

- Input never stops
- Input rate rate oscillations
- Bottlenecks
- System gets stuck at different places in a complex cluster
- Ingress performance impact on egress.
- High Availability Scenarios
    - Can you afford to lost data?
    - How much downtime can you sustain ?

Methodological Approach

Event Streaming Platform testing

Business constraints?

- Budget?
- Time to market?
- Resources available?
- Skill set available ?

Product requirements?

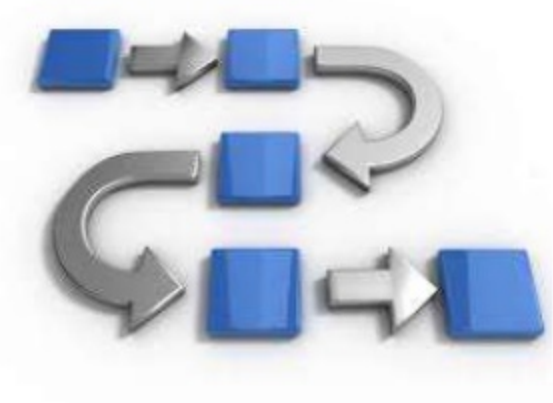
- What are the product's supported use cases?
- Supported Scale?
- Supported rate of ingress?
- Supported rate on egress?
- Complexity of Cluster?
- High availability?
- Cloud or onsite?
- Regulation? GMP?

The Automation: Must have requirements?

- Scale out/up?
- Fault tolerance!
- Reproducibility from scratch!
- Reporting!
- “Views” to avoid duplication for testing?
- Orchestration of same environment per Developer!
- Debugging options
- versioning!!

The method

- Phase 0: **get ready**
  - Product requirements & business constraints
  - Automation requirements
  - Creating a testing matrix based on insight and product features.
- Phase 1: **Get insights (some baselines)**
  - Test **Ingress** separately, find your **baseline**.
  - Test **Egress** separately, find your **baseline**.
  - Test **Ingress while Egress** in progress, find your **baseline**.
  - Stability and Performance
- Phase 2: Agile: **Design an automation system** that satisfies the requirements



True Story

Hadoop based Event Streaming Platform

(Peta Scale)

True story

- Product: Hadoop based event streaming platform. [peta scale]
- Technical Constraints
  - What is the expected **impact of big data** on the platform?
  - How to **debug scenarios** issues?
  - **QA???**
  - **Running time** - how much it takes to run all the tests?
- Company Challenges
  - **Cost** of Hardware? High end 6U Server
  - **Expertise** ? skillset?

Technical Flow

1. Start by testing the **simplest flow,** see that behaves as expected, and then another **layer of complexity**. for this step you must understand the products state machine from A to Z.
2. start with **single thread** (one of each component)
3. continue with **small data**. gradual increments. understand **machine metrics** of each component. and all the different **fine tune of configuration**.
4. get to the point you are convinced you have **big data** (what does your product support?)
    a. 0 data, 1M , 10M, 100M, 1GB, 10GB, 100G, 1TB, 10TB, 100TB ,1PB
5. start over , this time using **multi threads/components**
6. start over , this time try **negative testing** (fail over scenarios, data node failures, flume failures, namenode failures etc, spark failures)
7. Make sure **Logs** reflect the accurate status of the product. (error/warn/info/debug)

Testing matrix dimensions

- Multiple components: scale out of of same component:
- Data
  - different data rates: slow/fast /bursts (delay b/w packets slow/fast)
  - what happens if we change data distribution?
  - different input size (small , large, changing)

Performance & Metrics

  - what are the machine metrics: collector/flume/HDFS over time? get a rule of thumb of processing power
  - ingress rate?
  - Heap allocations per Hadoop ecosystem?

- stability and persistence:
  a. what happen if Flume fail?data loss/stuck on flume?
  b. recovery from HA
  c. data node loss?
  d. NameNode recovery? persistence of Flume

Ingress Testing matrix

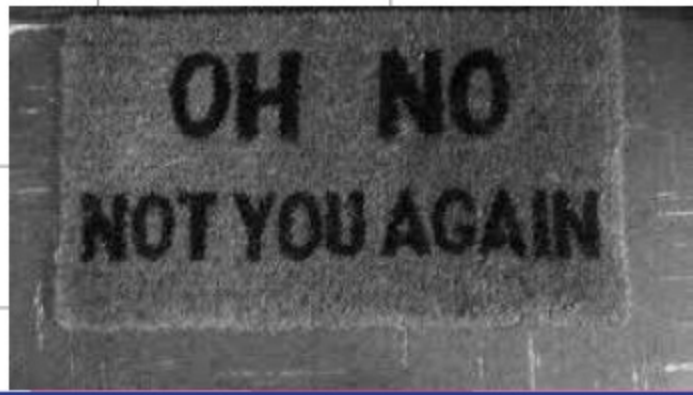
	single collector, single Flume,	multi collector, single Flume,	multi collector, multi Flume	HA HDFS
New cluster				
Existing data small cluster				
Existing cluster Big data				
Stability (failure of components)				

Egress Testing Matrix

1. Hadoop
    a. HDFS performance/failover/data replication/block size
    b. Yarn availability/metrics of jobs
    c. spark availability/ metrics of jobs/ performance tuning
2. Assuming some sort of Engine
    a. State (marks which files were already processed)
    b. Poller (polling of new evetns)

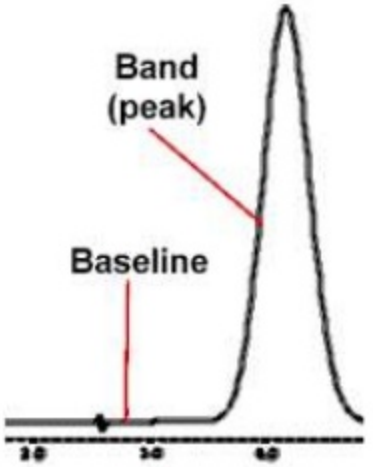
Egress Testing Matrix

	HDFS	Yarn	Spark	Engine state & poller	Any other layers
New cluster					
Existing data small cluster					
Existing cluster Big data					
Stability (failure of components)					



The baseline concept

- Requirements
  - Events data (simulated or real)
  - Cluster
  - Metrics - to understand if behaviour is expected.
  - Do i need to QA hadoop ecosystem?
- Steps
  - Create a golden standard - per event type - per building block
  - permutate



The solution

The Solution: Event Flow testing framework.

- **TBD**

Challenges with Ingress

- Amount of collectors
- Events streaming cluster
  - Kafka VS. Flume
  - Hadoop tuning and maintenance
  - Data loss - how to handle
  - Performance of HDFS sink
  - Channel persistency
  - Channel performance
  - Downtime + recovery time

Challenges with Egress

- Hadoop tuning
  - Spark Tuning
  - Engine Tuning
    - Size of file on HDFS - bigger is better?
    - Processing
      - Recovery from Crushes
      - Delayed data
        - In order /out of order

Challenges with Big Data testing.

- **Generating** Time of events → genUtil with json for user input
- **Reproducibility** → genUtil again.
- **Disk space** for testing data → peta scale product → peta scale testing data
- **Results compared** mechanism
- **Network** bottlenecks
- **ORDER** IS NOT GUAR
- **Replications**
- Cluster utilization

The solution: Continuous Testing

1. Extra large **Sanity** Testing per commit (!)
2. **Hourly** testing on latest commit (24 X 7)
3. **Weekly** testing
4. **Monthly** testing

The solution: Hardware perspective

- Cluster utilization? (mesos, dockers)?
- Developer sandbox: Dockers
- Uniform OS
- Uniform Hardware
- Get red of weak links

Performance Challenges: app perspective

- **Architecture bottlenecks [count(*), group by, compression, sort Merge Join]**
- What is **IO pattern**?
  - OLTP VS OLAP.
  - Columnar or Row based?
  - How big is your data?
  - READ % vs WRITE %.
  - Sequential? random?
  - Temporary VS permanent

Performance and Maintenance

Performance Challenges: OPS perspective

- Metrics
  - What is Expected Total **RAM required per Query**?
  - OS **RAM cache** , **CPU Cache** hits ?
  - **SWAP** ? yes/no how much?
  - OS metrics - **open files**, stack size, realtime priority?
- Theoretical limits?
  - **Disk type selection** -limitation, expected throughput
  - **Raid** controller, RAID type? Configuration? caching?
  - **File system** used? DFS? PFS? Ext4? XFS?

Maintenance challenges: OPS Perspective

How to Optimize your hardware selection ? (Analytics on your testing data)

a. **Compute intensive**
    i. GPU CORE intensive
    ii. CPUcores intensive
        1. Frequency
        2. Amount of thread for concurrency
b. **IO intensive?**
    i. SSD?
    ii. NearLine sata?

Maintenance challenges: DevOps Perspective

- Lightweight - python code
  - **Advantage** → focus on **generic skill** set (python coding)
  - **Disadvantage** → **reinventing the wheel**
- Fault tolerance - Scale out topology
  - **Cheap "desktops" (weak machines)**
    - **Advantage** - >
      - quick recovery from Image
      - Cheap, low risk, pay as you go.
      - Gets 90 % of the job DONE!

Maintenance challenges: DevOps Perspective

- Infrastructure
  - **Continuous integration**: continuous build, continuous packaging, installer.
  - **Continuous Deployment**: pre flight check, remote machine (on site, private cloud, cloud)
  - **Continuous testing**: Sanity, Hourly, Daily, weekly
- Reporting and Moni
  - **Extensive report se**
  - Monitoring: **Green/**

Maintenance challenges: Innovation Perspective

1. Data generation (using **GPU to generate data**)
2. (**hashing**)
3. Workload (one **dispatcher**, many compute nodes)
4. saving running time: data/compute time
    a. **Hashing**
    b. **Cacheing**
    c. **Smart Data Creating - no** **enerate 100B events test.**
    d. **Logs: Test results were n**
5. Dockers

Lesson learned (insights)

- Very hard to guarantee **100% coverage**
- Quality in a reasonable **cost is a huge challenge**
- Very **easy to miss milestones with QA of big data,**
- each mistake create a **huge ripple effect in timelines**
- Main **costs**:
    - **Employees** : team of 2 people working 220 hours per month over 1 years.
    - **Running Time**: Coding of innovative algorithms inside testing framework,
    - **Maintenance**: Automation, Monitoring, Analytics
    - **Hardware:** innovative DevOps , innovative OPS, research
    - Skillset and knowledge base takes time

Stay in touch...

- Omid Vahdaty
- +972-54-2384178

HALO ANALYTICS

SQREAM TECHNOLOGIES

jajah Telefonica