Indian Institute of Science
Bangalore, India
भारतीय विज्ञान संस्थान
बंगलौर, भारत

Department of Computational and Data Sciences

SE256:Jan16 (2:1)

L13:Distributed Stream Processing

Yogesh Simmhan

30 Mar, 2016

Announcements

- Cluster is down until **Mar 31**, 6PM
- Assignment B deadline extended till **Apr 3**, by midnight
  - NO Extension will be given!
- Assignment B submission instructions posted
  - There will not be any demo. We should be able to run the program ourselves. The report should be self-contained. Equal weightage will be given to the program and the report.
  - You will be evaluated on correctness of the program and its outputs, its speed/scalability, accuracy of the results, and analysis of the algorithm/results/scalability in the report.
- Final Exam on **Apr 27**, Wed in the AM
- Project topics posted
  - Topic proposal by **Apr 4**
  - Project due by **Apr 28**
  - Project demos on **Apr 30**

Stream are Commonplace (too)

- Web & Social Networks
  - Twitter, Facebook, Internet packets
- Cybersecurity
  - Telecom call logs, financial transactions, Malware
- Internet of Things
  - Smart Transport/Power/Water networks
  - Smart watch/phone/TV/…

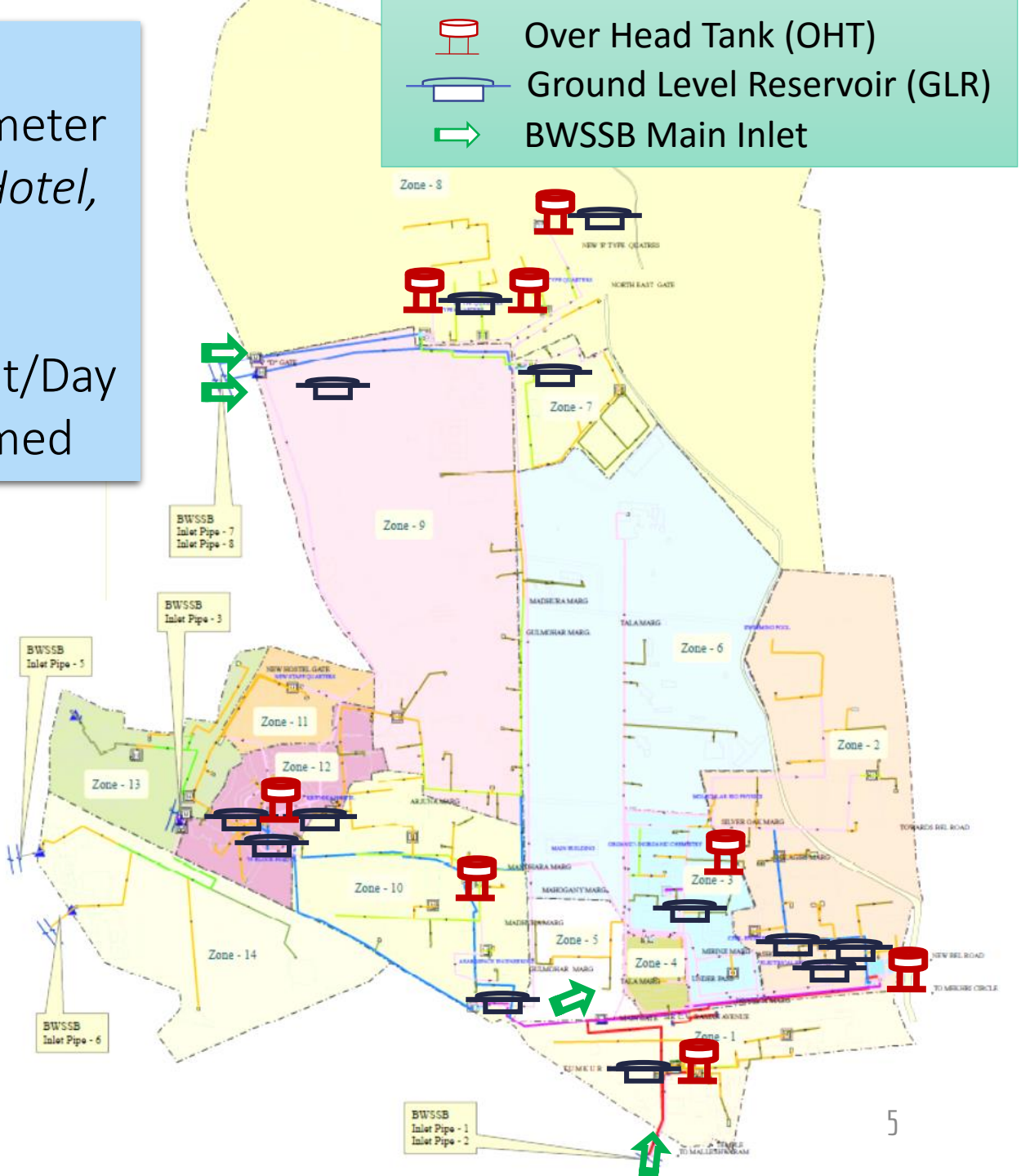
IISc Smart Campus: Water Management

- Plan pumping operations for reliability
  - Avoid underflow/overflow of water
  - 12 hrs to fill a large OHT, scarcity in summer weeks
- Provide safer water
  - Leakages, contamination from decades old network
- Reduce water usage for sustainability
  - IISc average: **400 Lit/day**, Global standard: **135 Lit/day**
  - Lack of visibility on usage footprint, sources
  - Opportunities for water harvesting, recycling
- Lower the cost
  - Reduce cost for water use & electricity for pumping

IISc Campus

- 440 Acres, 8 Km Perimeter
- 50 buildings: *Office, Hotel, Residence, Stores*
- 10,000 people
- Water Use: 40 Lakh Lit/Day
- 10MW Power Consumed

OHT	8
GLR	13
Inlet	4+3



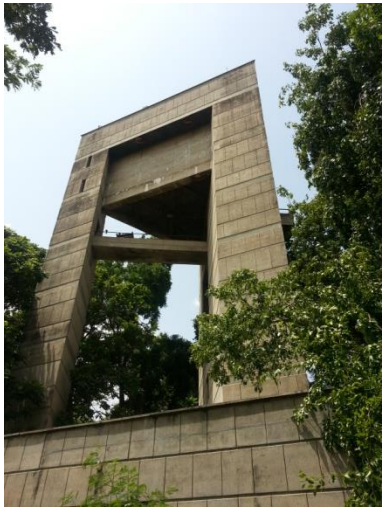
Over Head Tanks (OHT)

TPH (near Mechanical)

JNT Auditorium

Chemical Stores

Opposite to CENSE



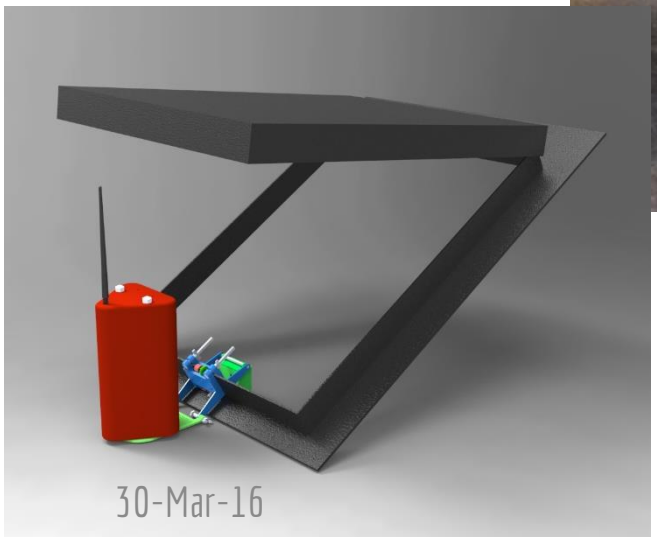
Opposite to NESARA

Behind old C-Mess

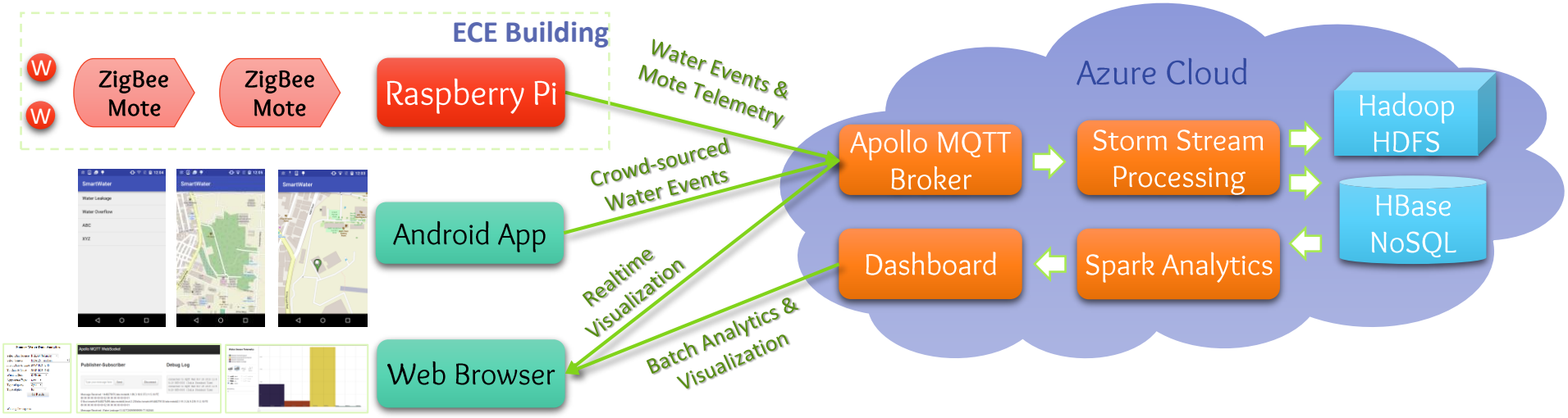
Opposite to Cense (new)

E Type Quarters

Custom Level + Quality Sensor



Backend


ECE Building
ZigBee Mote
ZigBee Mote
Raspberry Pi
Water Events & Mote Telemetry
Crowd-sourced Water Events
Android App
Realtime Visualization
Batch Analytics & Visualization
Web Browser
Azure Cloud
Apollo MQTT Broker
Storm Stream Processing
Hadoop HDFS
HBase NoSQL
Dashboard
Spark Analytics

Aadhaar Enrolment

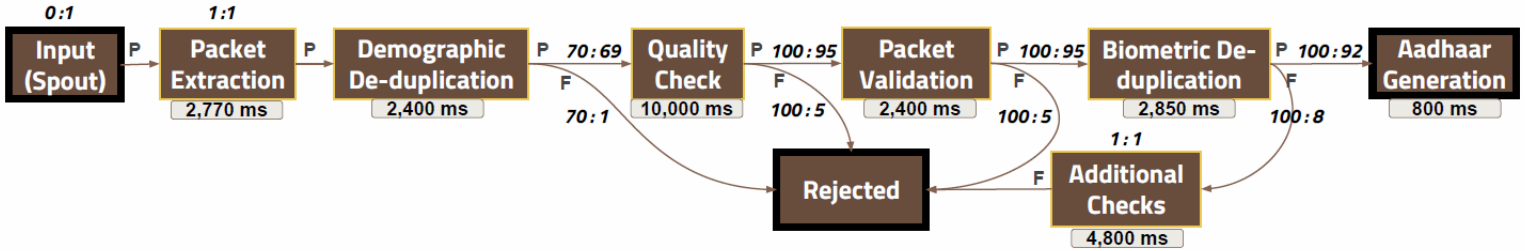


Fig. 1. *Enrollment dataflow.* Tasks are labeled with the average latency time in milliseconds. The selectivity is given for each outgoing edge. "P" edges are taken by events that pass the check at a task, while "F" edges are taken by events that fail a check.

- Input is stream of identity enrolment packets
- Output is a *UIDAI ID (success)* or *rejection*
- Each task tagged with Latency (ms)
- Each edge tagged with Selectivity
  - Input:output rate, probability of path taken

Benchmarking Fast Data Platforms for the 'Aadhaar' Biometric Database, Shukla, et al, Workshop on Big Data Benchmarking (WBDB), 2015

Aadhaar Authentication

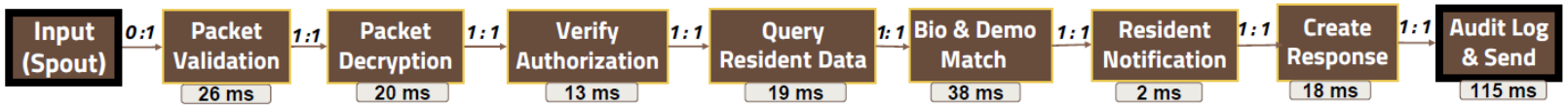


Fig. 3. *Authentication dataflow.* Tasks are labeled with the average latency time in milliseconds. Selectivity for all tasks is 1:1.

Benchmarking Fast Data Platforms for the 'Aadhaar' Biometric Database, Shukla, et al, Workshop on Big Data Benchmarking (WBDB), 2015

Stream Processing ➲ Continuous Dataflows

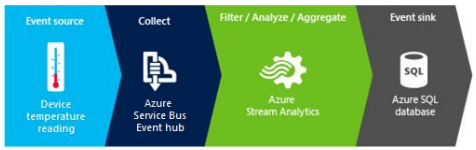
- How do you "compose" analytics that run continuously over streaming data?
- Application defined as Directed Acyclic Graph (DAG)
  - Vertices are *Tasks*
  - Edges are *streams*
  - Streams carry *tuples/events (name:value)* or *messages (opaque)*
  - Tasks process one or more messages/tuples and generate zero or more messages/tuples
  - Message routing?

Event Processing Programming Models

- Query Based
  - Complex Event processing
  - SQL like languages
- Programming APIs
- Queries or the Programs run on a continuous stream, unlike Hadoop where your data is static for the Batch processor
- Need to address diverse streams – Unbounded sequence of events
- Examples
  - Video Camera frames
  - Tweets
  - Laser scans from a robot
  - Log data

Distributed Stream Processing Systems

- Aurora – Early Research System
- Borealis – Early Research System
- **Apache Storm**
- Apache S4
- Apache Samza
- Google MillWheel
- Amazon Kinesis
- LinkedIn Databus
- Facebook Puma/Ptail/Scribe/ODS
- Azure Stream Analytics

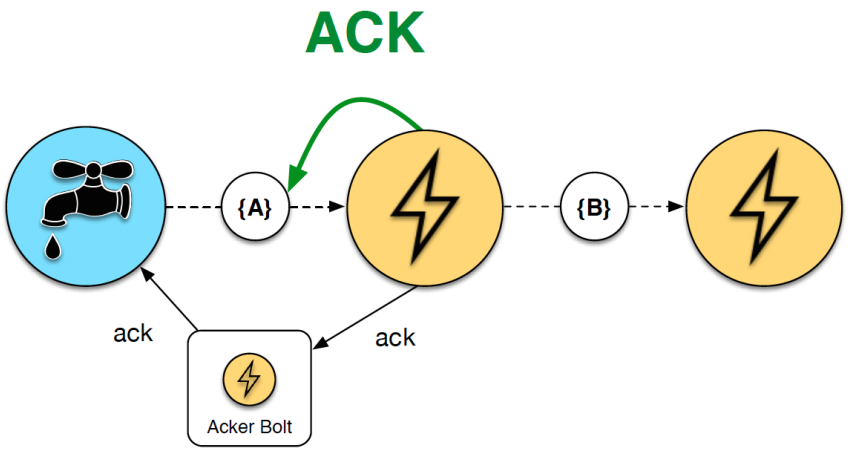


Apache Storm

See Nathan Marz's Slides

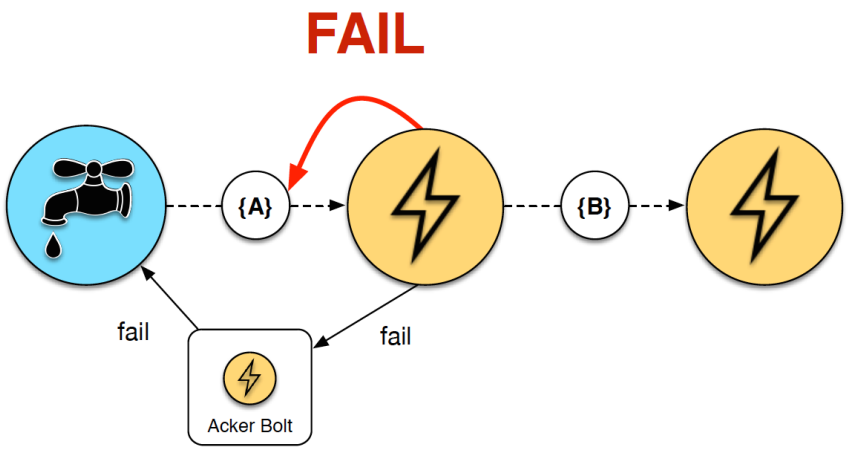
http://nathanmarz.com/blog/history-of-apache-storm-and-lessons-learned.html

Reliable Processing



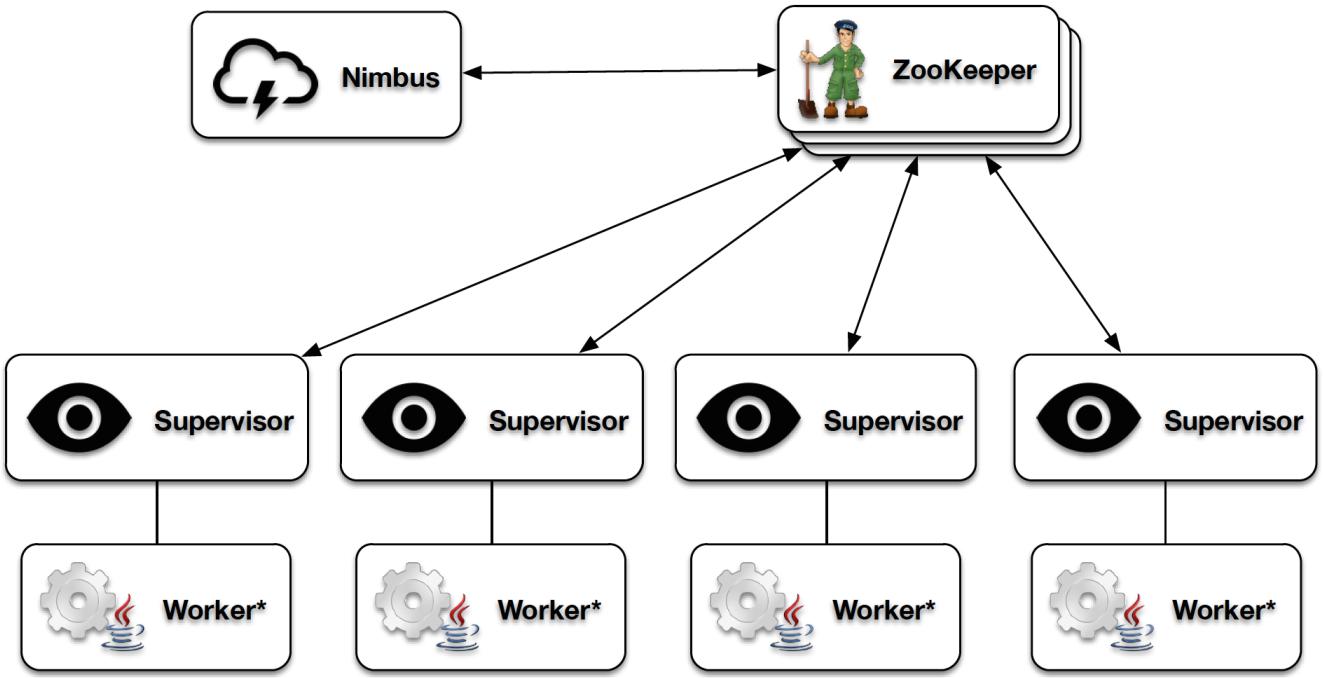
Acks are delivered via a system-level bolt

Reliable Processing

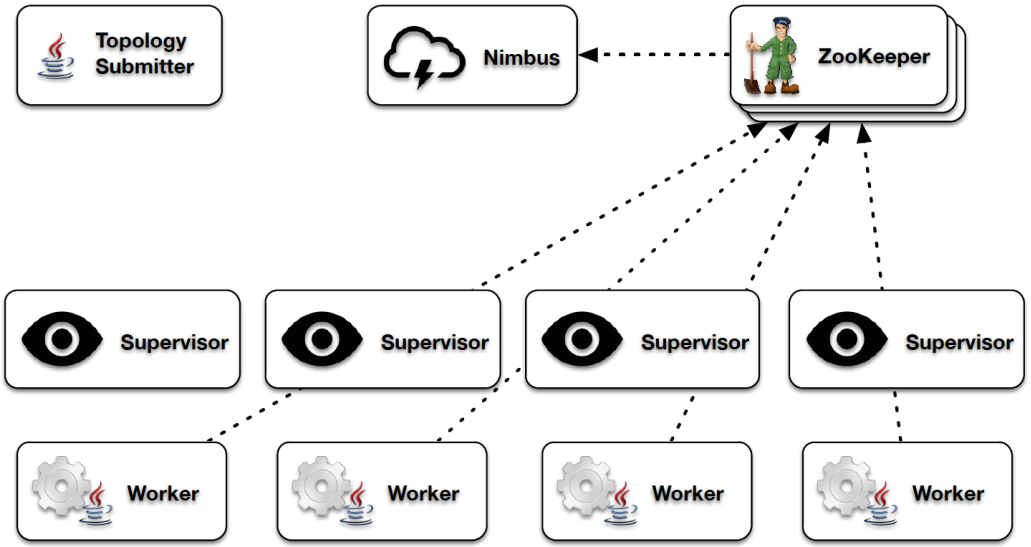


Bolts can also ***Fail*** a tuple to trigger a spout to replay the original.

Storm Cluster View

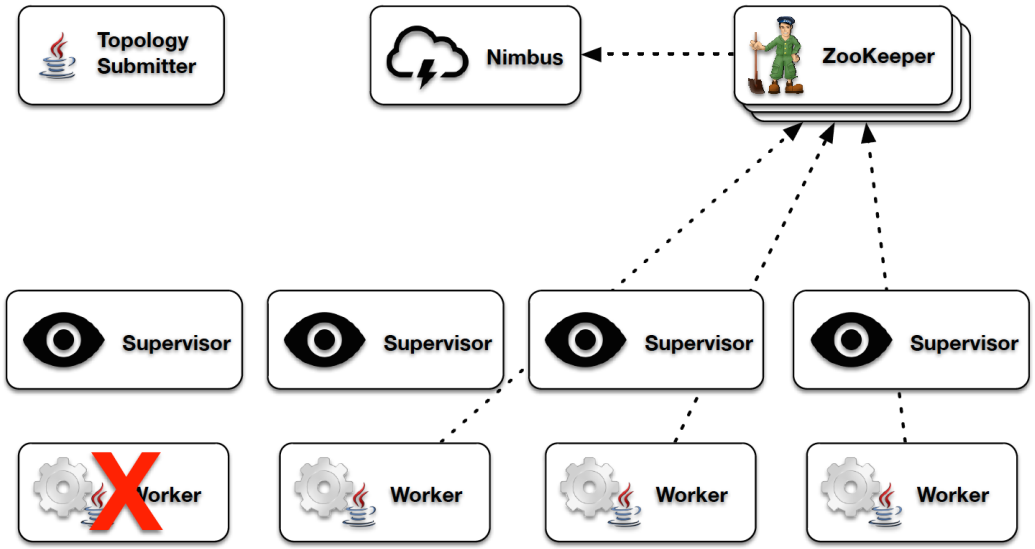


Fault Tolerance



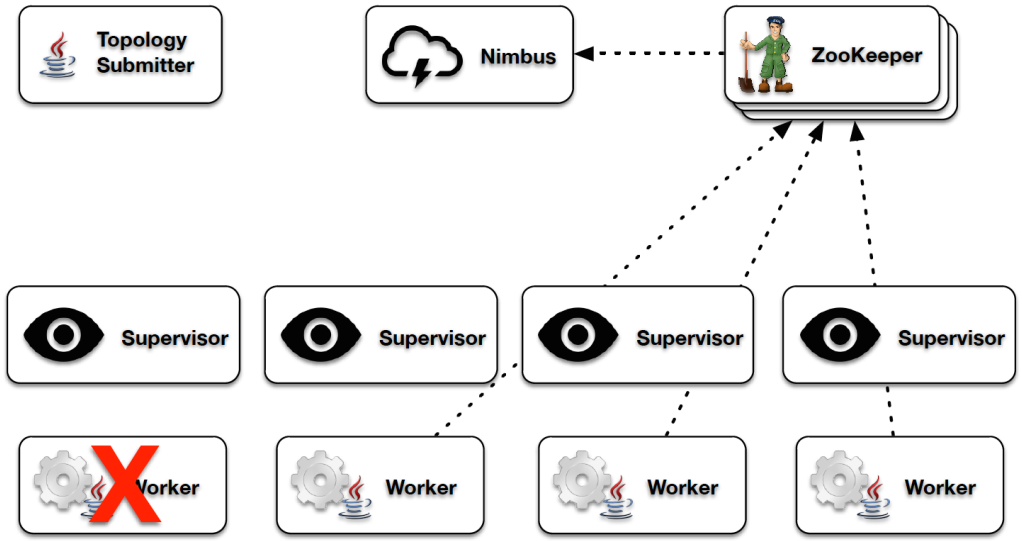
Workers heartbeat back to Supervisors and Nimbus via ZooKeeper, as well as locally.

Fault Tolerance



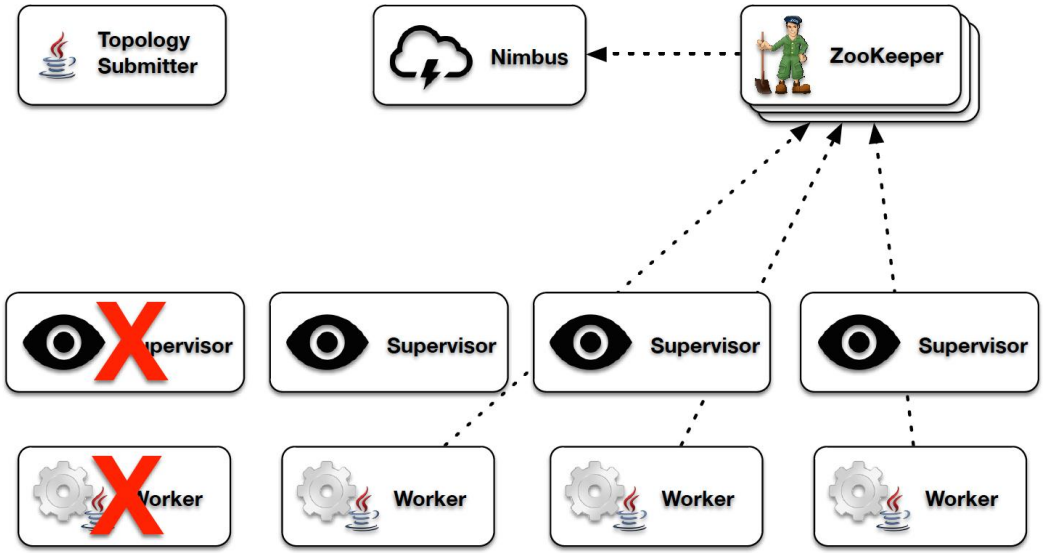
If a worker dies (fails to heartbeat), the Supervisor will restart it

Fault Tolerance



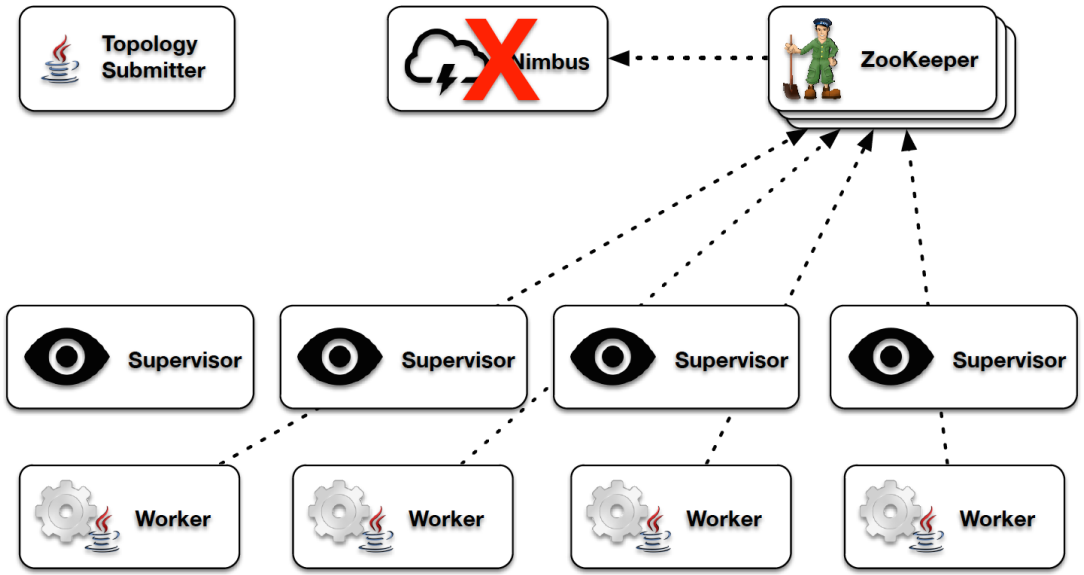
If a worker dies repeatedly, Nimbus will reassign the work to other nodes in the cluster.

Fault Tolerance



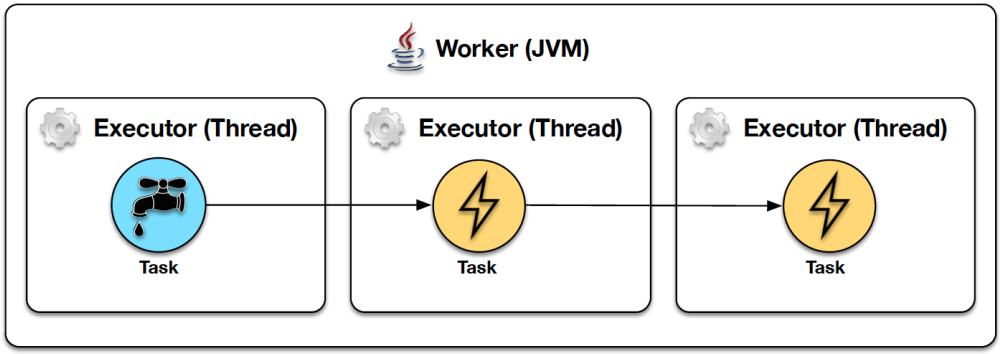
If a supervisor node dies, Nimbus will reassign the work to other nodes.

Fault Tolerance



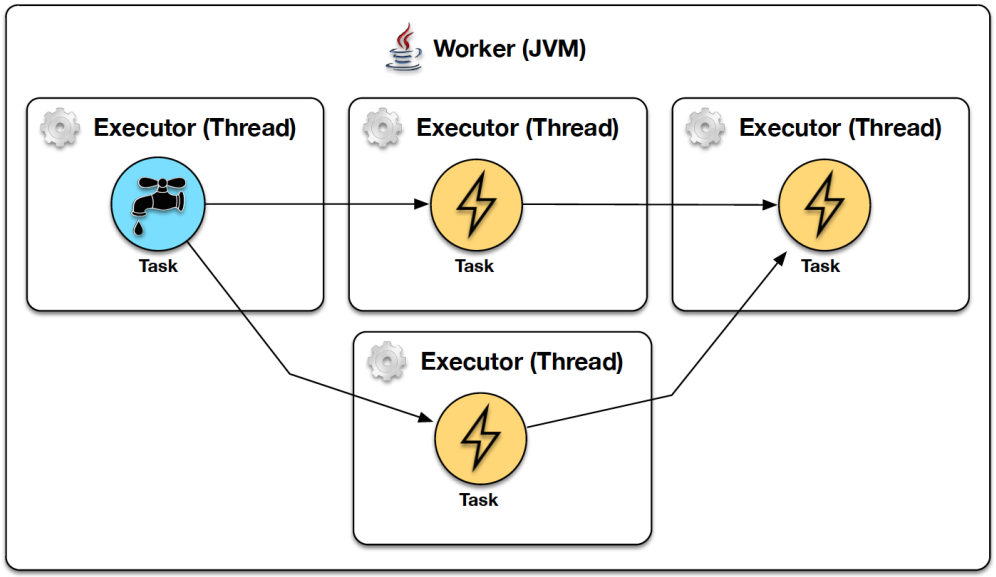
If Nimbus dies, topologies will continue to function normally, but won't be able to perform reassignments.

Parallelism



1 Worker,
Parallelism = 1

Parallelism



1 Worker,
Parallelism = 2

Other Streaming Platforms

- Apache Spark Streaming
- IBM InfoSphere Streams
- Yahoo S4

Reading

- Ankit Toshniwal, et al. Storm@twitter. In *ACM SIGMOD,* 2014
- Discretized Streams: An Efficient and Fault-Tolerant Model for Stream Processing on Large Clusters, Zaharia, et al, *USENIX HotCloud*, 2012, https://www.usenix.org/conference/hotcloud12/workshop-program/presentation/zaharia
- Leonardo Neumeyer, et al, S4: Distributed Stream Computing Platform. In *ICDMW* 2010