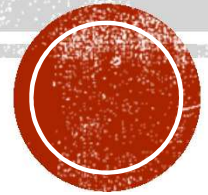


SOCIAL FEED INGESTION PIPELINE

Sandeep Bhat



ABOUT ME

- EXTREME NETWORKS(THROUGH ACQUISITION FROM BROCADE)
- KU PNNI SIMULATOR (MS PROJECT)
- Represent company at AT & T Capacity expansion discussion
- MANAGE and TRAIN a PROTOCOL team of 6
- Delivered many software features which are deployed in networks.
- I try to keep them busy



AGENDA

- ASSUMPTIONS/LIMITATIONS
- INPUT/OUTPUT
- INGESTION PIPELINE
- DEMO - AWS-LAMBDA
- JUNIT
- CODE-COVERAGE/CODE WALKTHROUGH



ASSUMPTIONS

Assumption	Comment
Simulate Twitter/FB events	100 Ids each along with 100 userIds
2 Twitter Feeds types	RETWEET, FWD
2 FB Posts Types	LIKE, REPLY
Producer produces messages serially in a thread context.	Messages are produced serially, using random generator
User ID is present in Twitter and FB posts	Same UserId in Tweet and a FB post means the same user.
Controlled environment. We can just specify number of messages to inject	Messages will be processed and stored in database.



LIMITATIONS / DISCLOSURE

Assumption	Comment
Processing Twitter/FB from 3 rd party APIs not supported	Simulations of events
Handling of document changing retroactively	Java Lambda functions used to process messages to store in DB.
No high availability support	
No worker threads for parallelism	
No multiple consumer based queue.	



FEEDS

Twitter Feed	FB post
TWITTER ID	FB ID:
USERID	USERID
TYPE: TWEET, RETWEET	TYPE:LIKE,REPLY
Text: "This is tweet"	Text: "This is FB post"



STATS

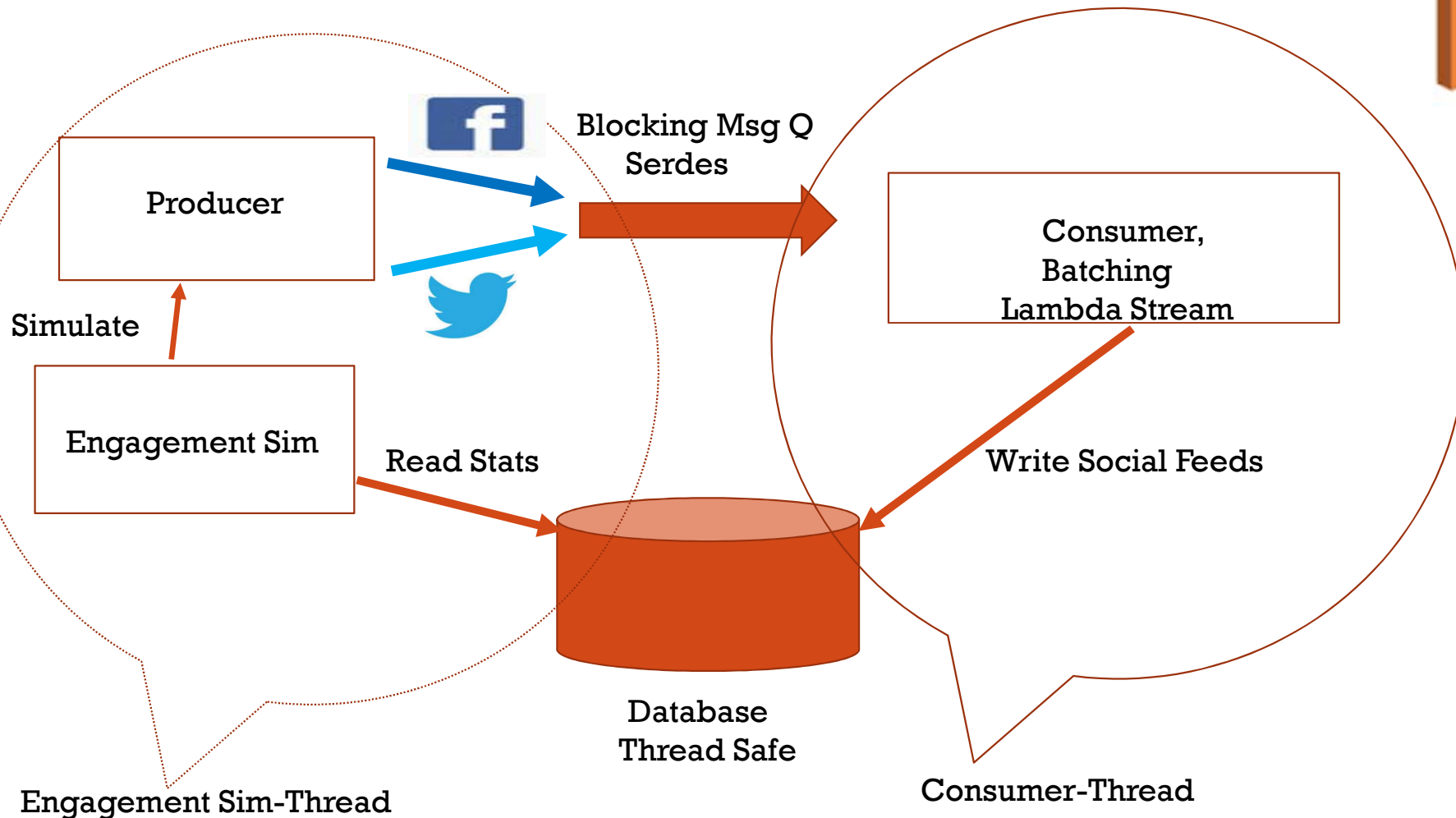
TWTS	100	FBPOSTS	100	USERIDS	100
TWTR-ID	0	RETWEET	122	FWD	129
FB-ID	0	LIKES	120	REPLY	144

USERID	0	LIKES	136	REPLY	115	RETWEET	136	FWD	114
--------	---	-------	-----	-------	-----	---------	-----	-----	-----



INGESTION PIPELINE

AWS Lambda



JUNIT

The screenshot shows an IDE's JUnit test runner interface. At the top, a toolbar contains icons for Markers, Properties, Servers, Data Source, Snippets, Console, EC2 Instance, EC2 AMIs, Error Log, JUnit, and Coverage. Below the toolbar, a status bar indicates "Finished after 8.511 seconds". A summary row shows "Runs: 3/3", "Errors: 0", and "Failures: 0", followed by a green progress bar. The test results list includes:














- `com.example.lamda.demo.ConsumerTest` [Runner: JUnit 4] (2.131 s)
- `com.example.lamda.demo.HelloTest` [Runner: JUnit 4] (3.051 s)
- `com.example.lamda.demo.EngagementSimTest` [Runner: JUnit 4] (3.101 s)

Each item is preceded by a green icon with a white checkmark. To the right of the list is a "Failure Trace" section with a hamburger menu icon and a "Failure Trace" label. The bottom of the window is empty.



CODE-COVERAGE

Merged (Jan 16, 2018 6:21:42 PM)

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
▸ 📁 src/test/java	 36.2 %	161	284	445
▼ 📁 src/main/java	 80.2 %	864	213	1,077
▼ 📁 com.example.lamda.demo	 80.2 %	864	213	1,077
▸ 📄 SerDes.java	 41.2 %	61	87	148
▸ 📄 DataBase.java	 89.2 %	264	32	296
▸ 📄 Consumer.java	 88.0 %	184	25	209
▸ 📄 FeedSource.java	 58.0 %	29	21	50
▸ 📄 EngagementSim.java	 84.7 %	83	15	98
▸ 📄 FBFeed.java	 79.6 %	39	10	49
▸ 📄 TwitterFeed.java	 79.6 %	39	10	49
▸ 📄 MsgQ.java	 90.7 %	68	7	75
▸ 📄 FeedType.java	 90.0 %	45	5	50
▸ 📄 Hello.java	94.1 %	16	1	17
▸ 📄 Producer.java	 100.0 %	36	0	36

ENGAGEMENT STATS

TWTS	100	FBPOSTS	100	USERS	100		
TWTR-ID	0	RETWEET	122	TWTFWD	129		
FB-ID	0	LIKES	120	REPLY	144		
USERID	0	LIKES	136	REPLY	115	RETWEET	136
						TWTFWD	114



THANK YOU

<https://github.com/cosmos342/Engagement>

