Internet of Things

After Hour 2021

Redis Stream and Cache for IoT Devices

1hn0va

after hour

MVP Microsoft® Most Valuable Professional

ANDREA TOSATO

ATosato86

andreatosato

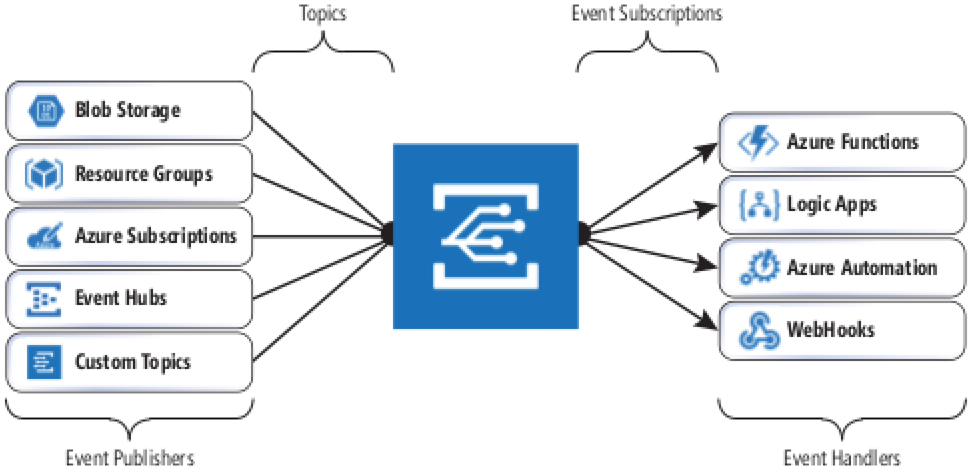
andreatosato

Di cosa non parleremo

RabbitMQ

50K msg per second
AMQP

Azure Service Bus
molti meno



Only Azure
N Million messages
per second

Messages in IoT – Rabbit Stream (new)

- RabbitMQ 3.9 introduces a new type of data structure: *streams*.
- A RabbitMQ stream models an *append-only log* with *non-destructive consuming semantics*. This means that – contrary to traditional queues in RabbitMQ – **consuming from a stream does not remove messages.**

Messages in IoT – Redis PubSub vs Stream

- Redis 5 introduce Streams
- Redis have already Pub/Sub in memory
- Redis add Stream feature

million messages per second.

Messages in IoT – Choice (Stream)

rabbit

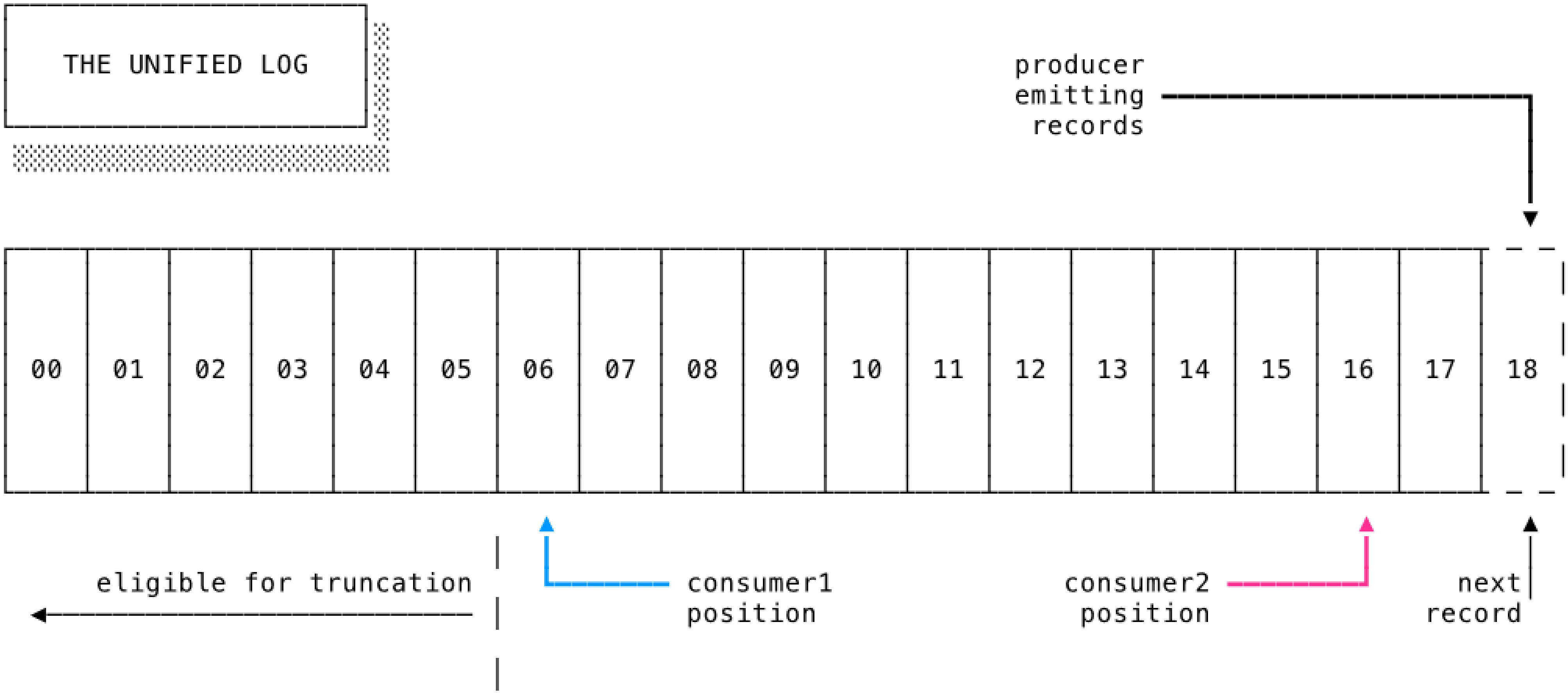
kafka

redis

What are streams good for

- **large fan-outs**: where many applications need to read the same messages
- **large backlogs**: streams store messages on disk, not in-memory, so the only limit is the disk capacity
- **replay & time-traveling**: consumers can attach anywhere in a stream, using an absolute offset or a timestamp, and they can read and re-read the same data
- **high throughput**: streams are super fast compared to traditional queues, several orders of magnitude faster

How to work stream



THE UNIFIED LOG

producer
emitting
records

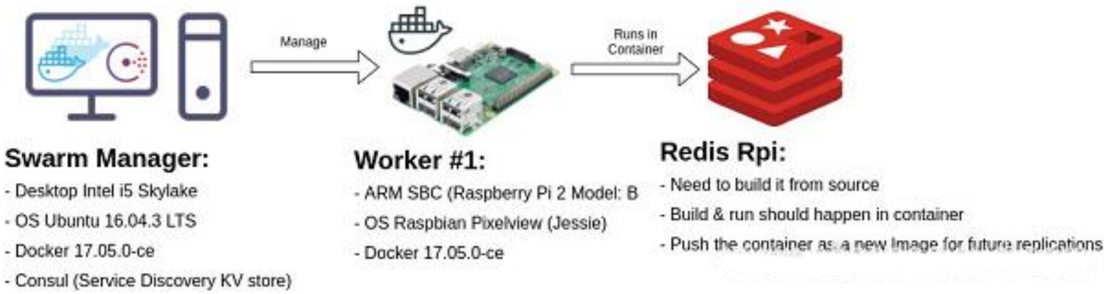
00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18

eligible for truncation

consumer1
position

consumer2
position

next
record



Swarm Manager:

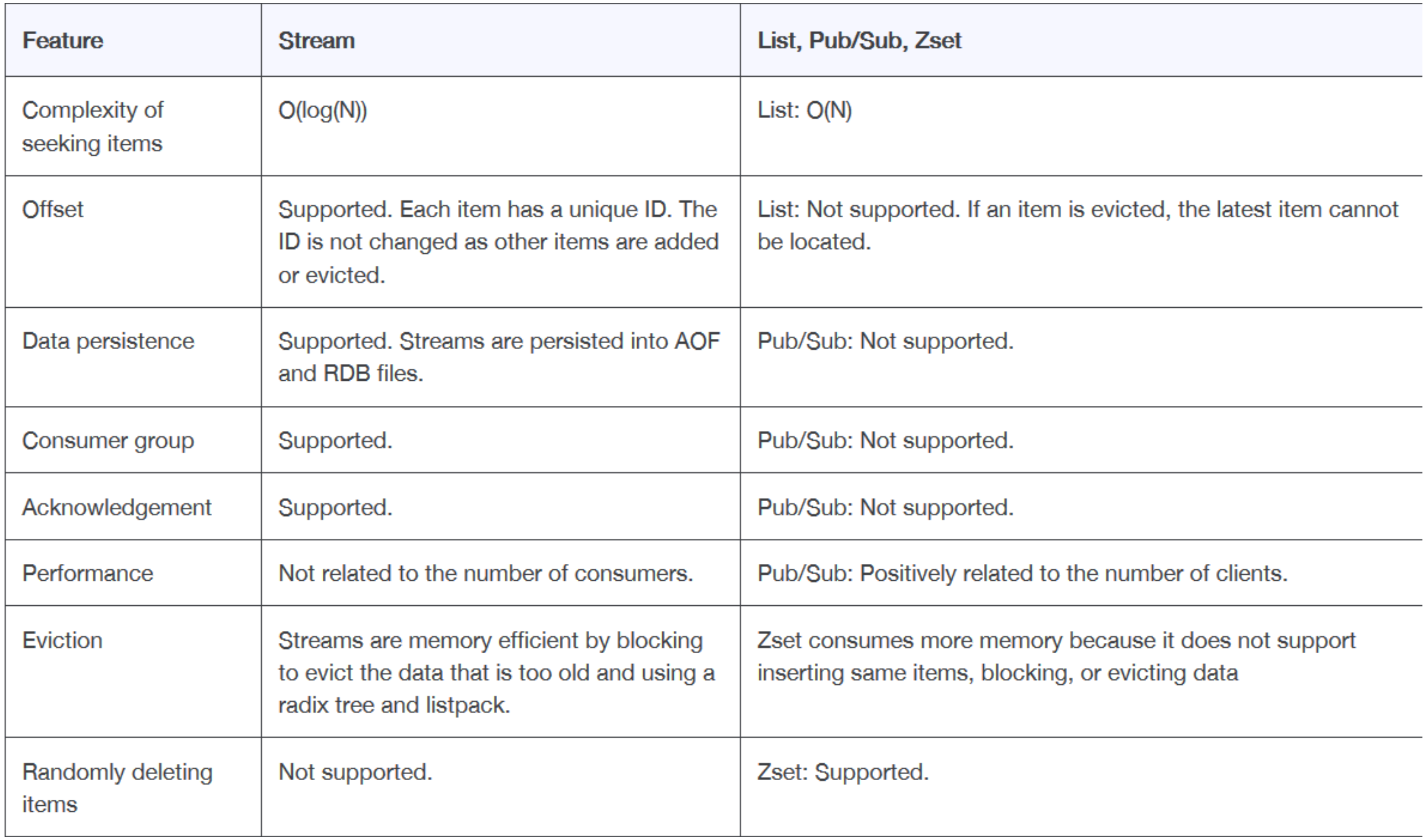
- Desktop Intel i5 Skylake
- OS Ubuntu 16.04.3 LTS
- Docker 17.05.0-ce
- Consul (Service Discovery KV store)

Worker #1:

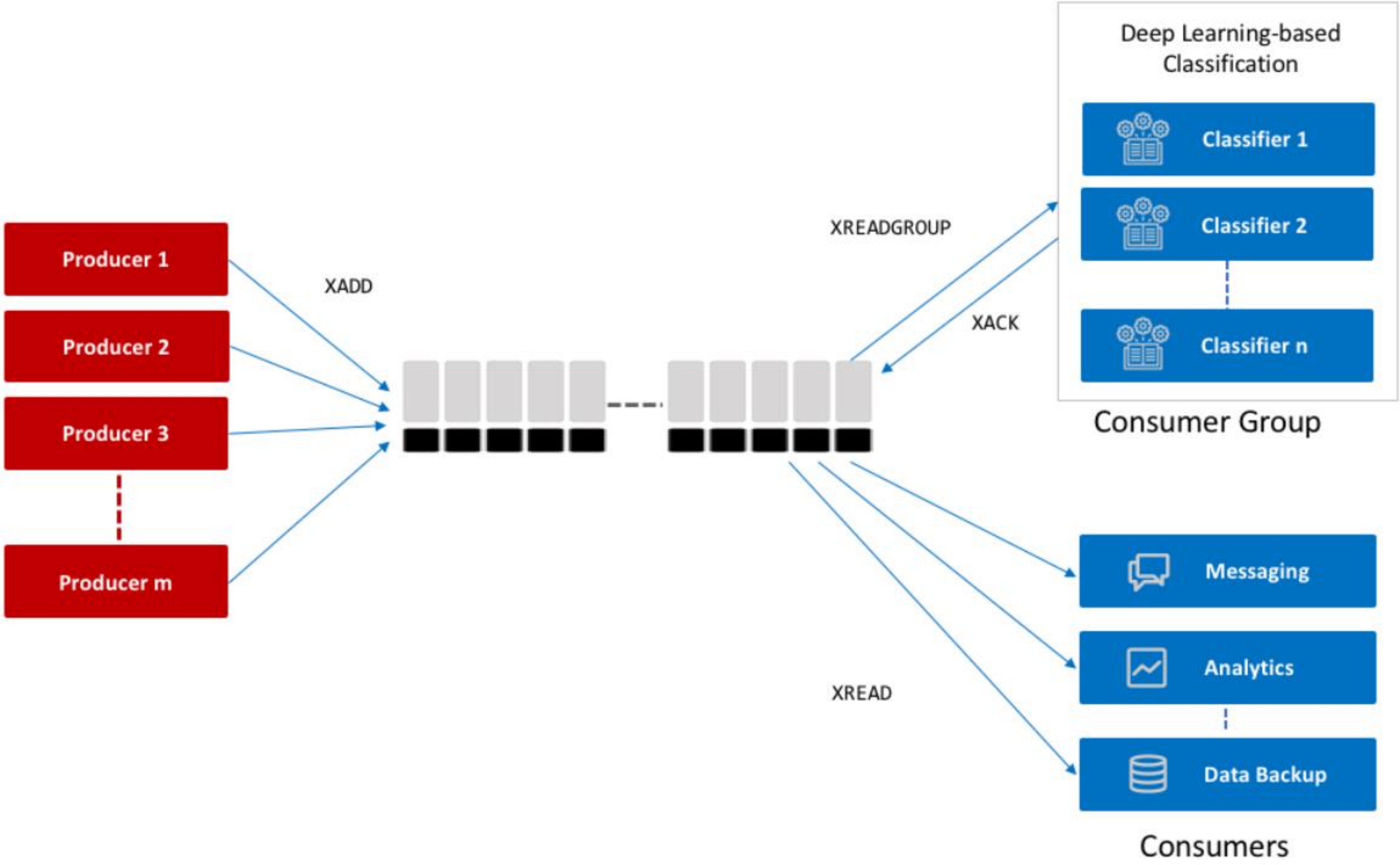
- ARM SBC (Raspberry Pi 2 Model: B
- OS Raspbian Pixelview (Jessie)
- Docker 17.05.0-ce

Redis Rpi:

- Need to build it from source
- Build & run should happen in container
- Push the container as a new Image for future replications



Feature	Stream	List, Pub/Sub, Zset
Complexity of seeking items	O(log(N))	List: O(N)
Offset	Supported. Each item has a unique ID. The ID is not changed as other items are added or evicted.	List: Not supported. If an item is evicted, the latest item cannot be located.
Data persistence	Supported. Streams are persisted into AOF and RDB files.	Pub/Sub: Not supported.
Consumer group	Supported.	Pub/Sub: Not supported.
Acknowledgement	Supported.	Pub/Sub: Not supported.
Performance	Not related to the number of consumers.	Pub/Sub: Positively related to the number of clients.
Eviction	Streams are memory efficient by blocking to evict the data that is too old and using a radix tree and listpack.	Zset consumes more memory because it does not support inserting same items, blocking, or evicting data
Randomly deleting items	Not supported.	Zset: Supported.


Producer 1
Producer 2
Producer 3
Producer m
XADD
XREADGROUP
XACK
XREAD
Deep Learning-based Classification
Classifier 1
Classifier 2
Classifier n
Consumer Group
Messaging
Analytics
Data Backup
Consumers

MVP Microsoft® Most Valuable Professional

ANDREA TOSATO

ATosato86

andreatosato

andreatosato

https://github.com/andreatosato/RedisLabDotNet