AEROSPIKE

Natalie Pistunovich

@NataliePis @aerospikeDB

DIGITAL:LAB BERLIN Volkswagen Group

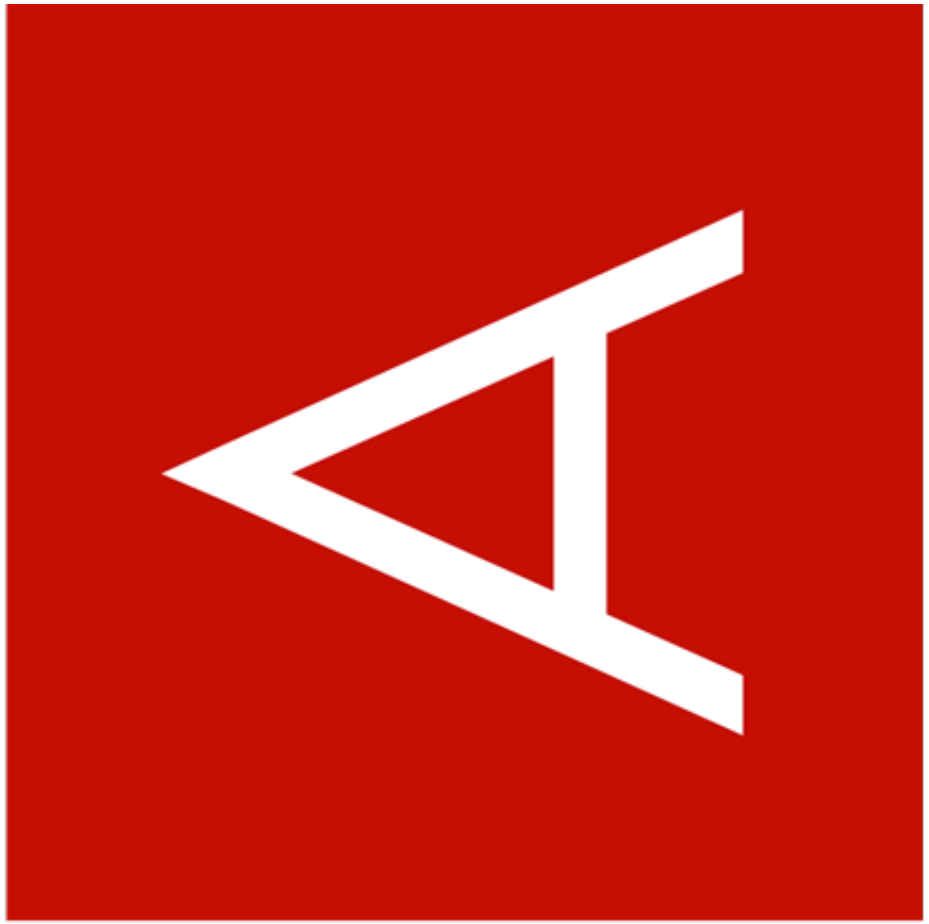
The **aerospike engine** is a type of rocket engine that maintains its aerodynamic efficiency across a wide range of altitudes.

It belongs to the class of altitude compensating nozzle engines. A vehicle with an aerospike engine uses 25–30% less fuel at low altitudes, where most missions have the greatest need for thrust.

Aerospike engines have been studied for a number of years and are the baseline engines for many single-stage-to-orbit designs and Space Shuttle main engine.

- wikipedia

AEROSPIKE



AEROSPIKE

Terminology

SQL vs. NoSQL

Parameter	SQL	NOSQL
Definition	SQL databases are primarily called RDBMS or Relational Databases	NoSQL databases are primarily called as Non-relational or distributed database
Design for	Traditional RDBMS uses SQL syntax and queries to analyze and get the data for further insights. They are used for OLAP systems.	NoSQL database system consists of various kind of database technologies. These databases were developed in response to the demands presented for the development of the modern application.
Query Language	Structured query language (SQL)	No declarative query language
Type	SQL databases are table based databases	NoSQL databases can be document based, key-value pairs, graph databases

SQL vs. NoSQL

Parameter	SQL	NOSQL
Schema	SQL databases have a predefined schema	NoSQL databases use dynamic schema for unstructured data.
Ability to scale	SQL databases are vertically scalable	NoSQL databases are horizontally scalable
Hierarchical data storage	SQL databases are not suitable for hierarchical data storage.	More suitable for the hierarchical data store as it supports key-value pair method.
Variations	One type with minor variations.	Many different types which include key-value stores, document databases, and graph databases.
Development Year	It was developed in the 1970s to deal with issues with flat file storage	Developed in the late 2000s to overcome issues and limitations of SQL databases.

When to NoSQL?

- Low-latency, low-overhead API to access data
- A few database features with high scale
- Avoid data/schema pre-design altogether for simple applications and agility
- Fast lookup by primary key

CAP

CAP Theorem is a concept that a distributed database system can only have 2 of the 3:

1. Consistency

2. Availability

3. Partition Tolerance

AEROSPIKE

CAP

1. Consistency

All nodes see the same data at the same time.

Performing a read operation will return the value of the most recent write operation causing all nodes to return the same data.

A system has consistency if a transaction starts with the system in a consistent state, and ends with the system in a consistent state.

In this model, a system can (and does) shift into an inconsistent state during a transaction, but the entire transaction gets rolled back if there is an error during any stage in the process.

CAP

2. Availability

Every request gets a response on success/failure.

Achieving availability in a distributed system requires that the system remains operational 100% of the time.

Every client gets a response, regardless of the state of any individual node in the system.

This metric is trivial to measure: either you can submit read/write commands, or you cannot. Hence, the databases are time independent as the nodes need to be available online at all times.

CAP

3. Partition Tolerance

The system continues to run, despite the number of messages being delayed by the network between nodes.

A system that is partition-tolerant can sustain any amount of network failure that doesn’t result in a failure of the entire network.

Data records are sufficiently replicated across combinations of nodes and networks to keep the system up through intermittent outages.

ACID

The principles that database transactions should adhere to,
to ensure that data doesn't become corrupt as a result of a failure

1. Atomicity
2. Consistency
3. Isolation
4. Durability

AEROSPIKE

ACID

Atomicity

Means that you guarantee that either all of the transaction succeeds or none of it does. You don’t get part of it succeeding and part of it not. If one part of the transaction fails, the whole transaction fails. With atomicity, it’s all or nothing.

Consistency

Guarantees that all data will be consistent. All data will be valid according to all defined rules, including any constraints, cascades, and triggers that have been applied on the database.

ACID

Isolation

Guarantees that all transactions will occur in isolation. No transaction will be affected by any other transaction. So a transaction cannot read data from any other transaction that has not yet completed.

Durability

Means that once a transaction is committed, it will remain in the system – even if there's a system crash immediately following the transaction. Any changes from the transaction must be stored permanently. If the system tells the user that the transaction has succeeded, the transaction must have, in fact, succeeded.

JEPSEN

Analyses | Talks | Consistency | Services | Ethics

Analyses

Over the past six years, Jepsen has analyzed over two dozen databases, coordination services, and queues—and we've found replica divergence, data loss, stale reads, read skew, lock conflicts, and much more. Here's every analysis we've published.

Aerospike	2015-05-04	3.5.4
	2018-03-07	3.99.0.3
Cassandra	2013-09-24	2.0.0
Chronos	2015-08-10	2.4.0
CockroachDB	2017-02-16	beta-20160829
Crate	2016-06-28	0.54.9
Dgraph	2018-08-23	1.0.2
Elasticsearch	2014-06-15	1.1.0
	2015-04-27	1.5.0
etcd	2014-06-09	0.4.1
	2020-01-30	3.4.3
FaunaDB	2019-03-05	2.5.4
Hazelcast	2017-10-06	3.8.3
Kafka	2013-09-24	0.8 beta
MariaDB Galera	2015-09-01	10.0

https://jepsen.io

ACID

JEPSEN

Analyses | Talks | Consistency | Services | Ethics

Analyses

Over the past six years, Jepsen has analyzed over two dozen databases, coordination services, and queues—and we've found replica divergence, data loss, stale reads, read skew, lock conflicts, and much more. Here's every analysis we've published.

Aerospike	2015-05-04	3.5.4
	2018-03-07	3.99.0.3
	2015-04-27	1.5.0
etcd	2014-06-09	0.4.1
	2020-01-30	3.4.3
FaunaDB	2019-03-05	2.5.4
Hazelcast	2017-10-06	3.8.3
Kafka	2013-09-24	0.8 beta
MariaDB Galera	2015-09-01	10.0

"Aerospike does appear to provide linearizability through network partitions and process crashes"

- Kyle Kingsbury, Jepsen.io (https://jepsen.io/analyses/aerospike-3-99-0-3)

https://jepsen.io

About Aerospike

AEROSPIKE

The Hummingbird

- Takes ~20ms to flap its wings, 50-70 wing flaps every second
- The smallest bird in the world, weigh less than a penny (2 grams)

AEROSPIKE

Aerospike

- ~300K read/writes every second, 99.9% are <1ms latency
- Unmatched reliability and uptime and deployable anywhere

AEROSPIKE

Deployment

Cloud

AWS · Google Cloud Platform · Azure · Alibaba Cloud · IBM Cloud

30% of license revenue
45% by customer count

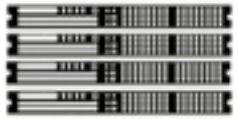
Managed Service

AEROSPIKE + AWS · Google Cloud Platform · Azure · Pivotal Cloud Foundry · Kubernetes

Orchestration Platforms

Pivotal Cloud Foundry · docker · Kubernetes

Traditional Linux Packages

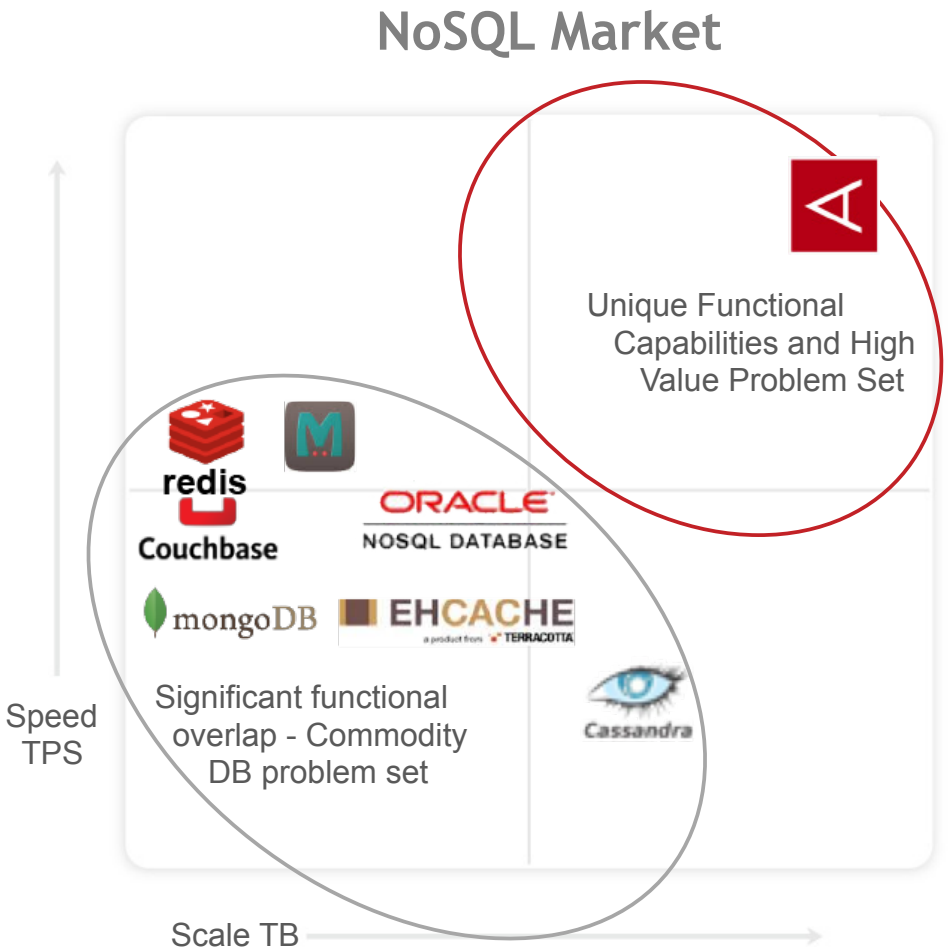


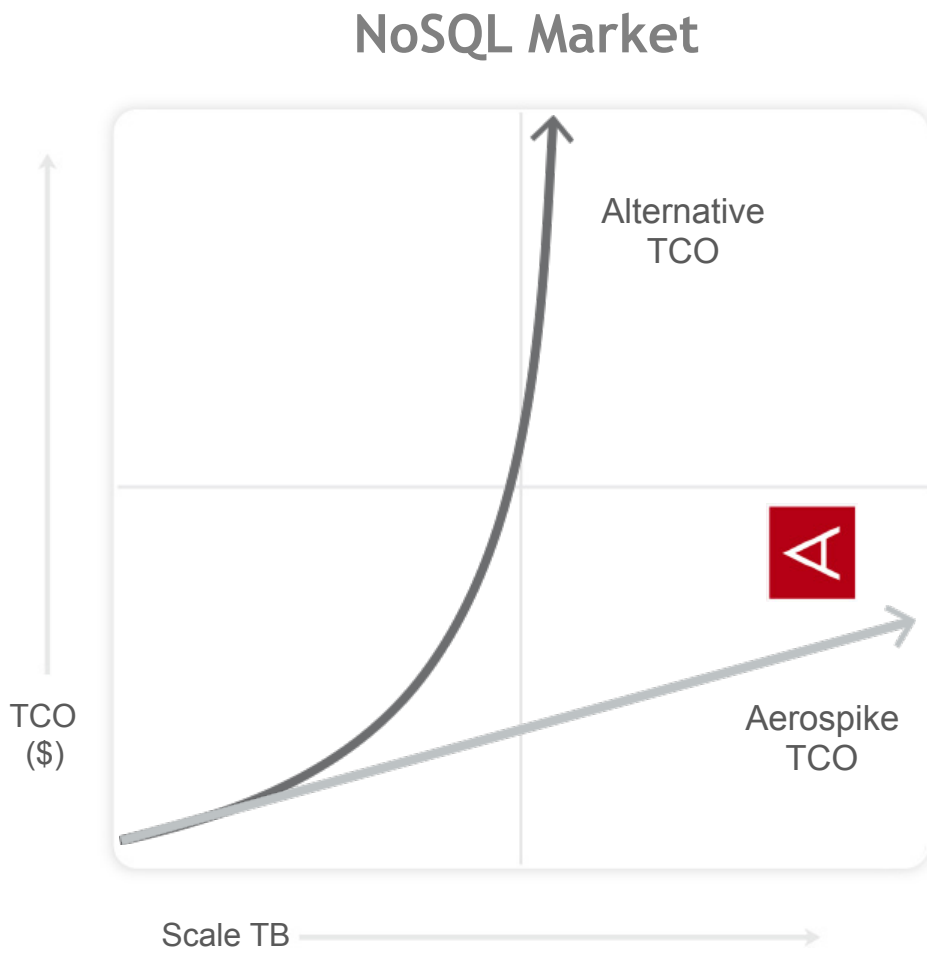
Bare Metal Commodity Infrastructure

70% of license revenue

AEROSPIKE

Multiple NoSQL Technologies





Customers

ADTECH	ECOMMERCE	FINANCIAL	NEWTECH	TELCO	GAMING
AppNexus	wayfair	American Express	snapchat	KYOCERA	Sony Interactive Entertainment
nielsen	KAYAK	charles SCHWAB	cybereason	airtel	Playtika
Oath: A Verizon company	Rakuten	Fidelity	McAfee	NOKIA	DRAFTKINGS
SIGNAL	WILLIAMS SONOMA	BARCLAYS	Adobe	Jio	SBTech we know sports
AppsFlyer	bookmyshow my show my time my seat	BANCA D'ITALIA EUROSISTEMA	Baidu 百度	amdocs	DREAM11
INMOBI	Flipkart	PayPal	AmerisourceBergen	COMCAST	FACEOFF
adform	snapdeal	LexisNexis RISK SOLUTIONS ThreatMetrix	Roku Now this is TV.™	VIETTEL Say it your way	
YAHOO! JAPAN	ifood	neustar	OLA		
APPLOVIN		paytm			
PubMatic		PhonePe INDIA'S PAYMENTS APP			
theTradeDesk		London STOCK EXCHANGE			

AEROSPIKE

A Use Case

AEROSPIKE

Real-time Fraud Prevention for Digital Payments

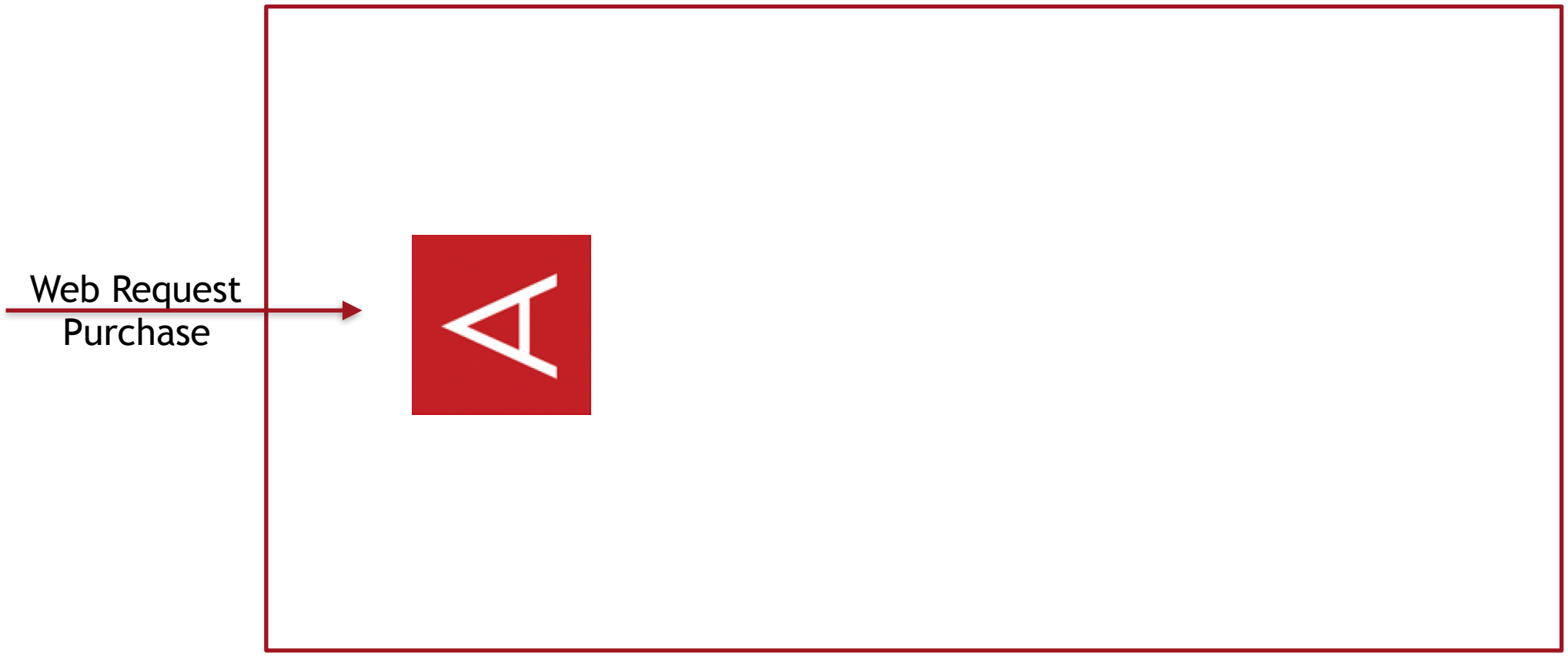
Business Challenge

- Tracking 150 million customers enrollment, payment and invoicing preferences, and profile data (e.g. IP address, location)
- Frequent user/device movement opens door for fraud
- Every payment transaction requires hundreds of DB reads/writes
- Missed latency SLA leads to lost business – fraud algorithm has a process window during payments
- Caching solution too expensive

Need to scale up

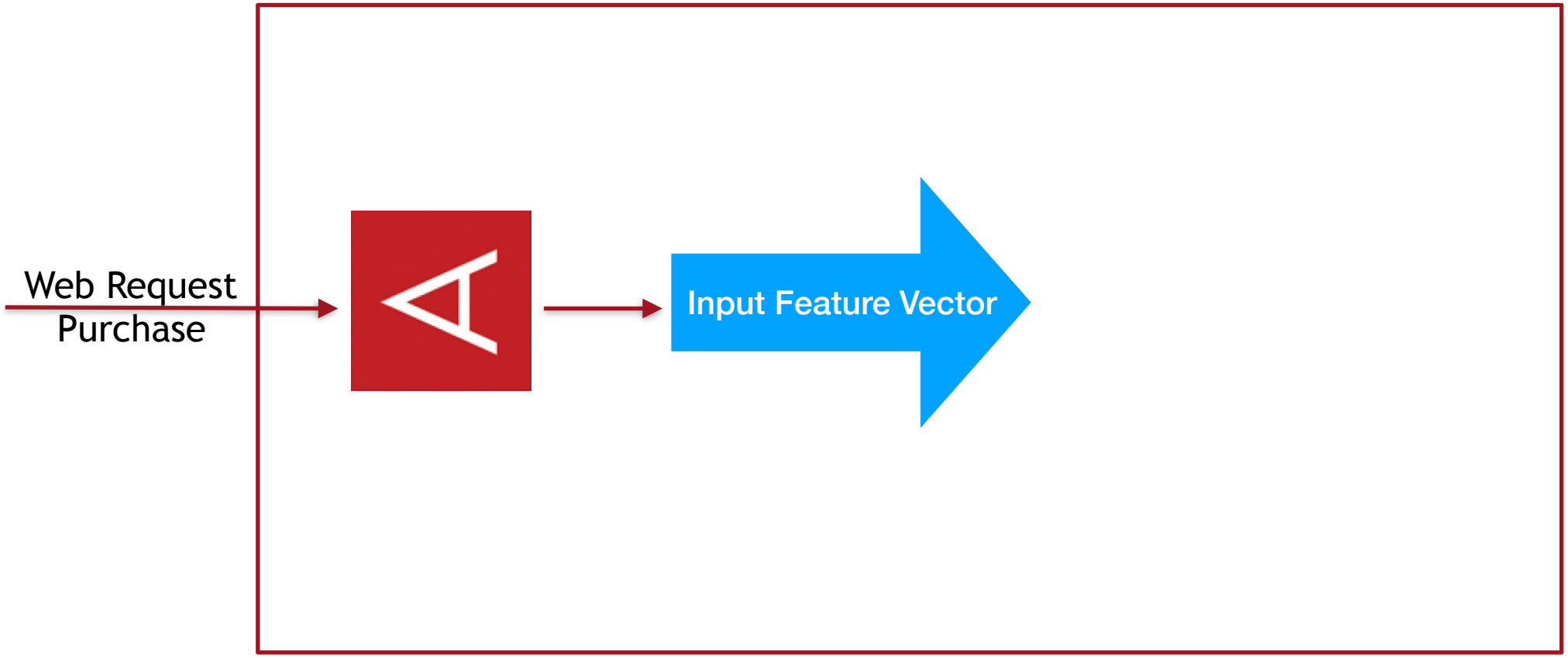
- 10 → 100 TB
- 10B → 100 B objects
- 200k → 1 Million+ TPS

AEROSPIKE

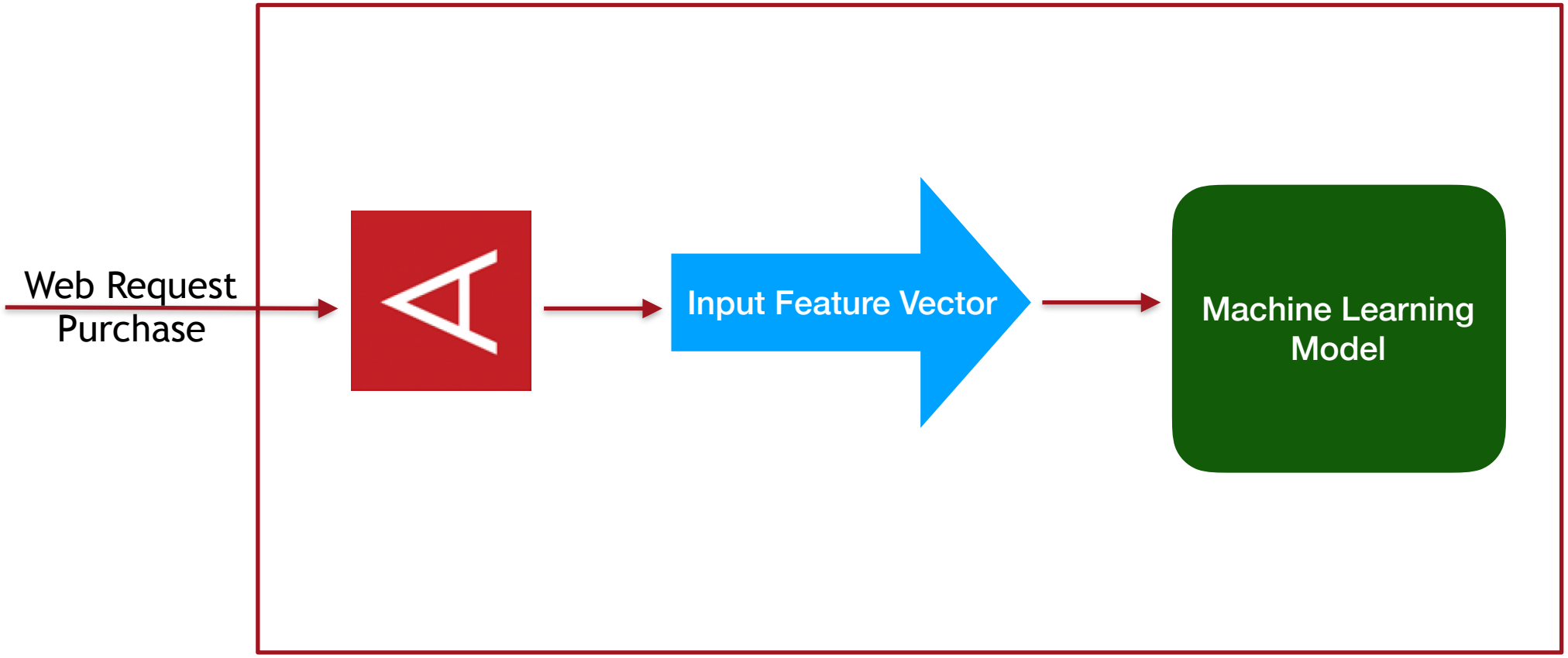


AEROSPIKE

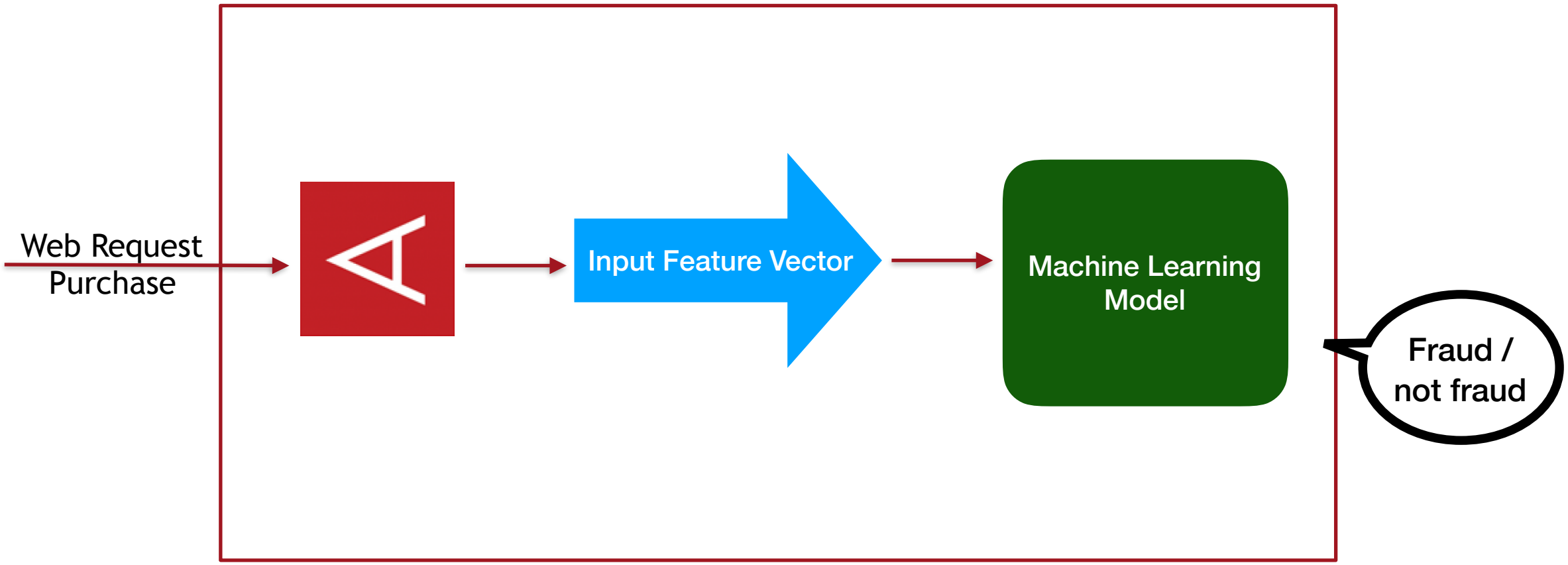
Presence of a recognised header	1
How long have the user been registered	0.78
Frequency of purchase	0
Frequency of reported issues	0.9
In the average purchase price range	0.61



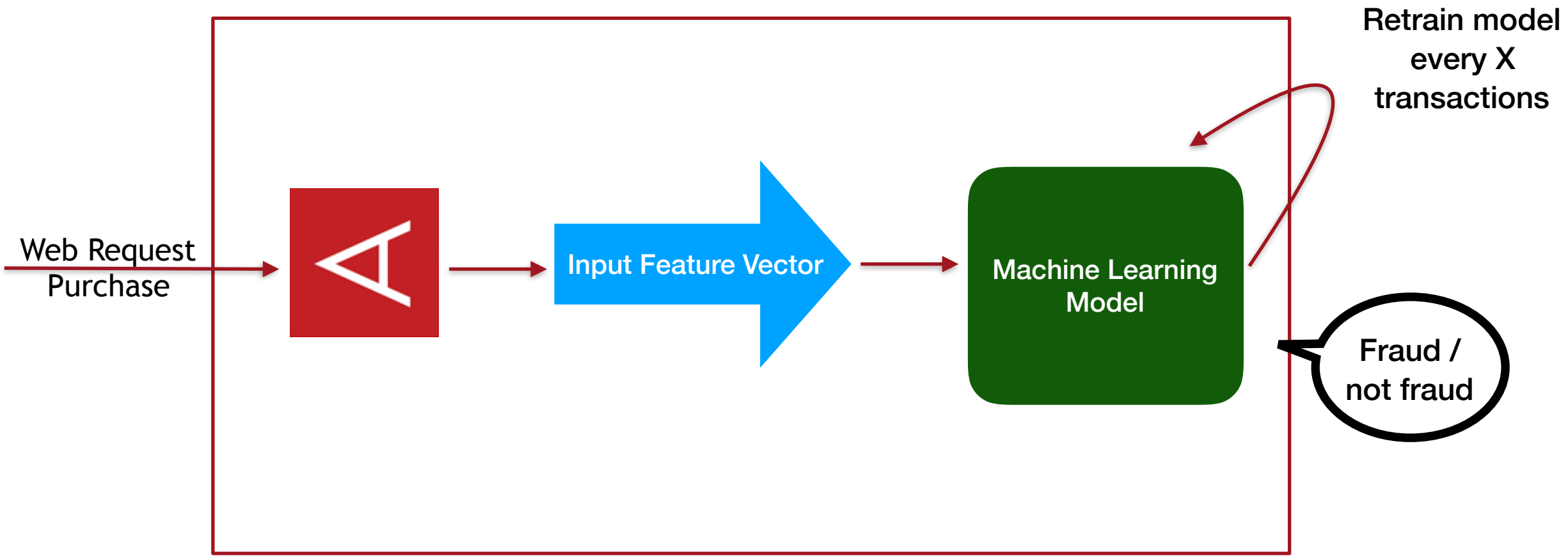
AEROSPIKE



AEROSPIKE



AEROSPIKE



AEROSPIKE

Real-time Fraud Prevention for Digital Payments

PayPal

Powering Global Fraud Prevention Network

✓ $280B+ payments annually

Replaced Oracle + Terracota

✓ Reduced server footprint 15x

Improved SLAs

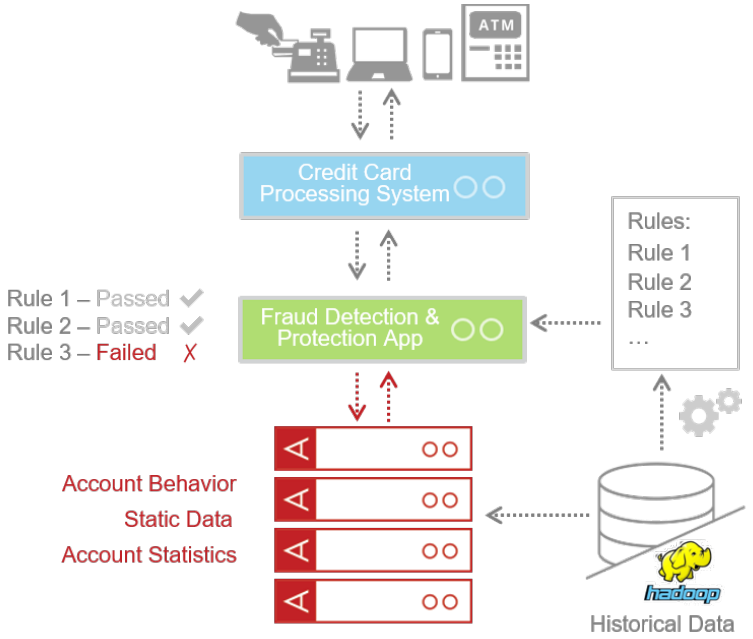
✓ 30x reduction in false positives

Increased revenue

✓ 10x improvement in fraud calculation data used

"PayPal is innovating deep analytics to rapidly respond to emerging fraud patterns**, then deploying into an event-driven, fast data, in-memory architecture to **accelerate detection, reduce losses and achieve near-continuous availability."

Mikhail Kourjanski, PhD. - Lead Data Architect, Risk and Compliance Management Platform, PayPal



AEROSPIKE

What's So Special About the Architecture?

AEROSPIKE

Hybrid Memory Architecture

- Flash Optimized Storage Layer
- Multi-threaded & Massively Parallel
- Self-healing Clusters

Hybrid Memory Architecture

- **Flash Optimized Storage Layer**
- Multi-threaded & Massively Parallel
- Self-healing Clusters

Flash Optimized Storage Layer: Indexes in DRAM, Data on SSD

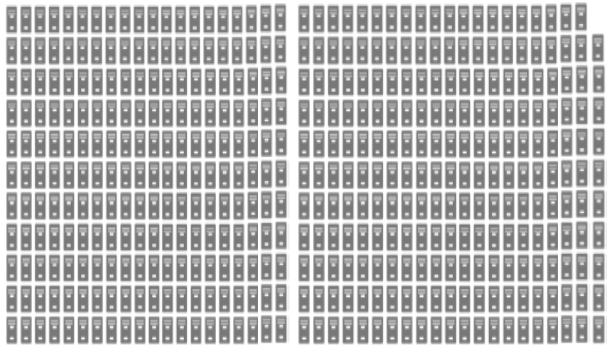
- Small amount of DRAM
- No cache so no cache misses
- Optimized for SSDs
- Continual Defragmentation
- Direct Device Access

Total Cost of Ownership

AdTech Use Case - Signal

TCO savings of 1 app $7.75M over 3 yrs

450 Cassandra Servers



cassandra

Only 60 Aerospike Servers

AEROSPIKE

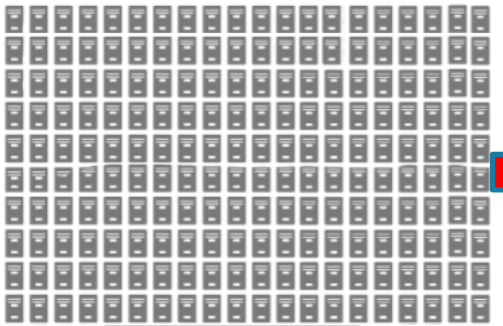
	Year 1 ($M USD)	Year 2 ($M USD)	Year 3 ($M USD)	Total ($M USD)
Cassandra	$2.39	$3.29	$4.53	$10.21
Aerospike	$1.69	$ .24	$.52	$2.46
Total OpEx Savings from Aerospike ($M USD)	$.70	$3.05	$4.01	$7.75

"Free is even too expensive compared to Aerospike."
- Jason Yanowitz, EVP and CTO, Signal

Gaming Use Case - Playtika

TCO savings of app $1.4M/year & $4.2M over 3 yrs

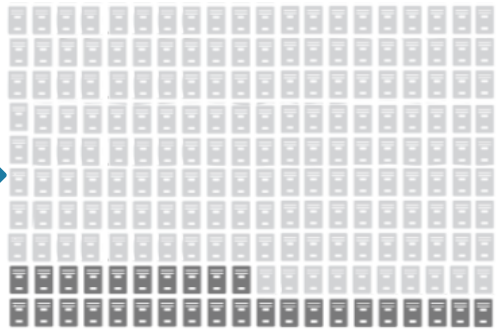
200 Couchbase Servers



Couchbase

DRAM & SSD

Only 30 Aerospike Servers



AEROSPIKE

SSD & DRAM

200K/s Reads+Writes → 600K/s Reads+Writes

"Simplicity of IT Ops frees teams up for future-looking initiatives."
- Guy Almog, Director of Engineering, Playtika

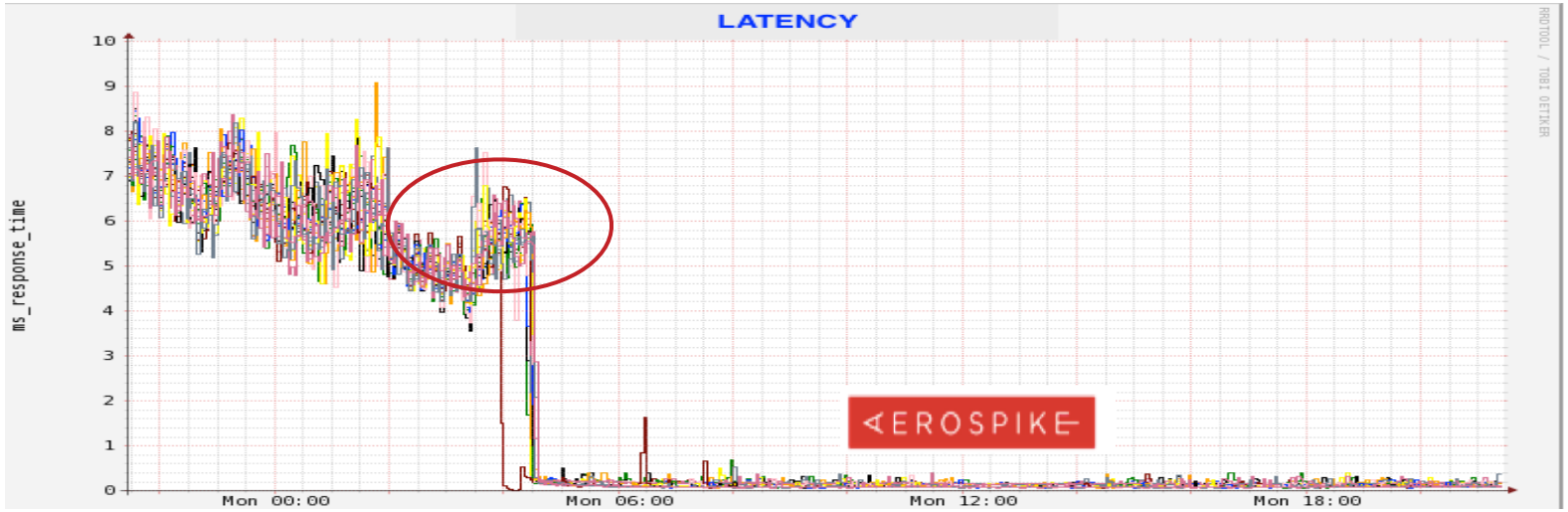
Aerospike's cacheless architecture is simpler, faster and cheaper than cache-based architectures

AEROSPIKE

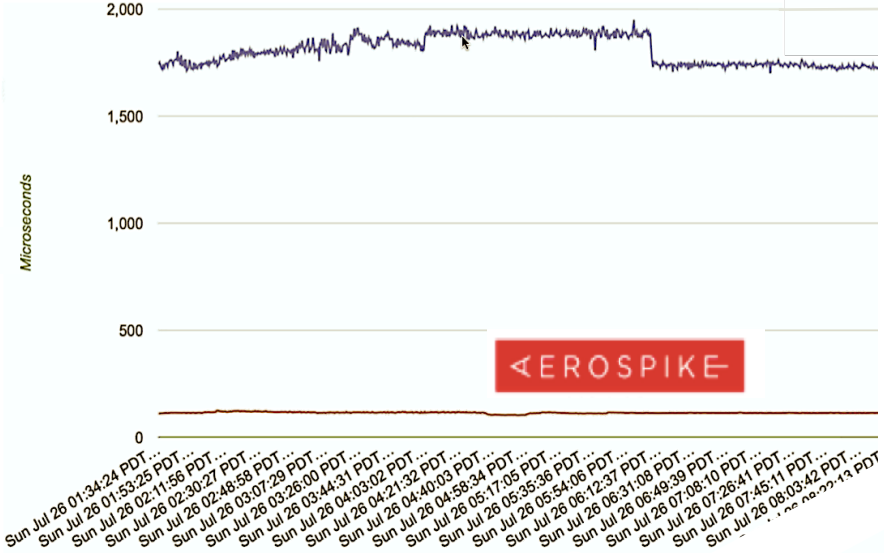
Actual Customer Comparisons when moving to Aerospike

Cassandra to Aerospike

- Digital identity management



Relational + Cache to Aerospike (Fraud detection)



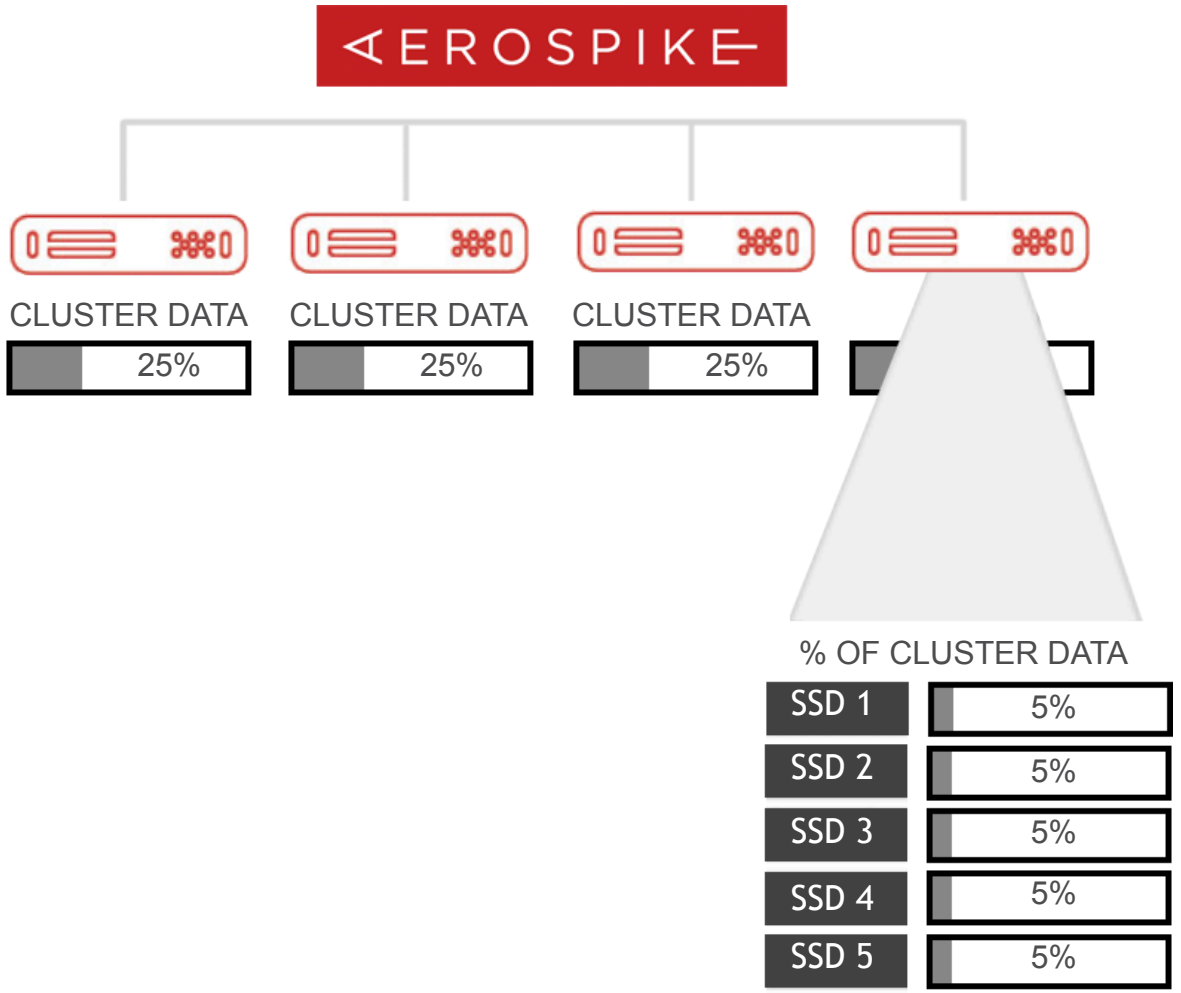
AEROSPIKE

Hybrid Memory Architecture

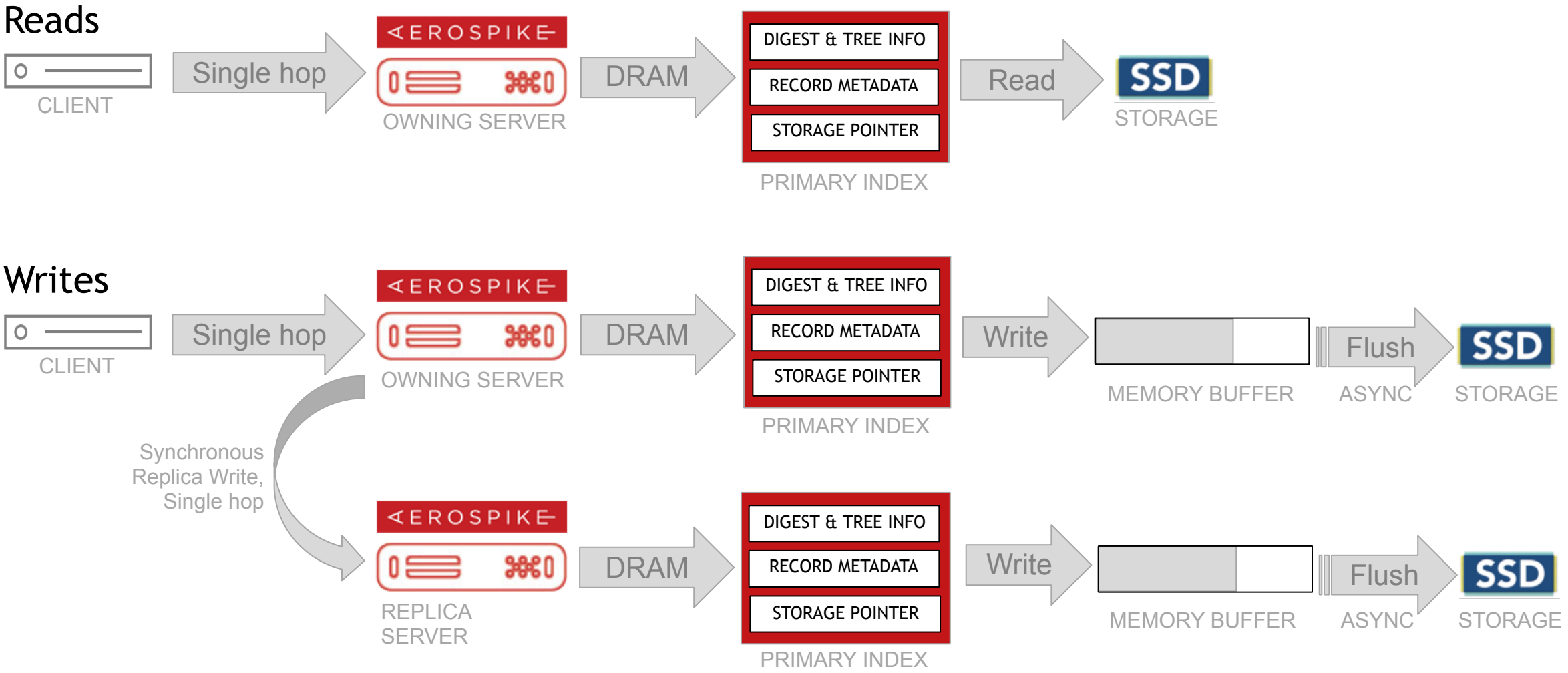
- Flash Optimized Storage Layer: Indexes in DRAM, Data on SSD
- **Multi-threaded & Massively Parallel**
- Self-healing Clusters

Multi-threaded & Massively Parallel

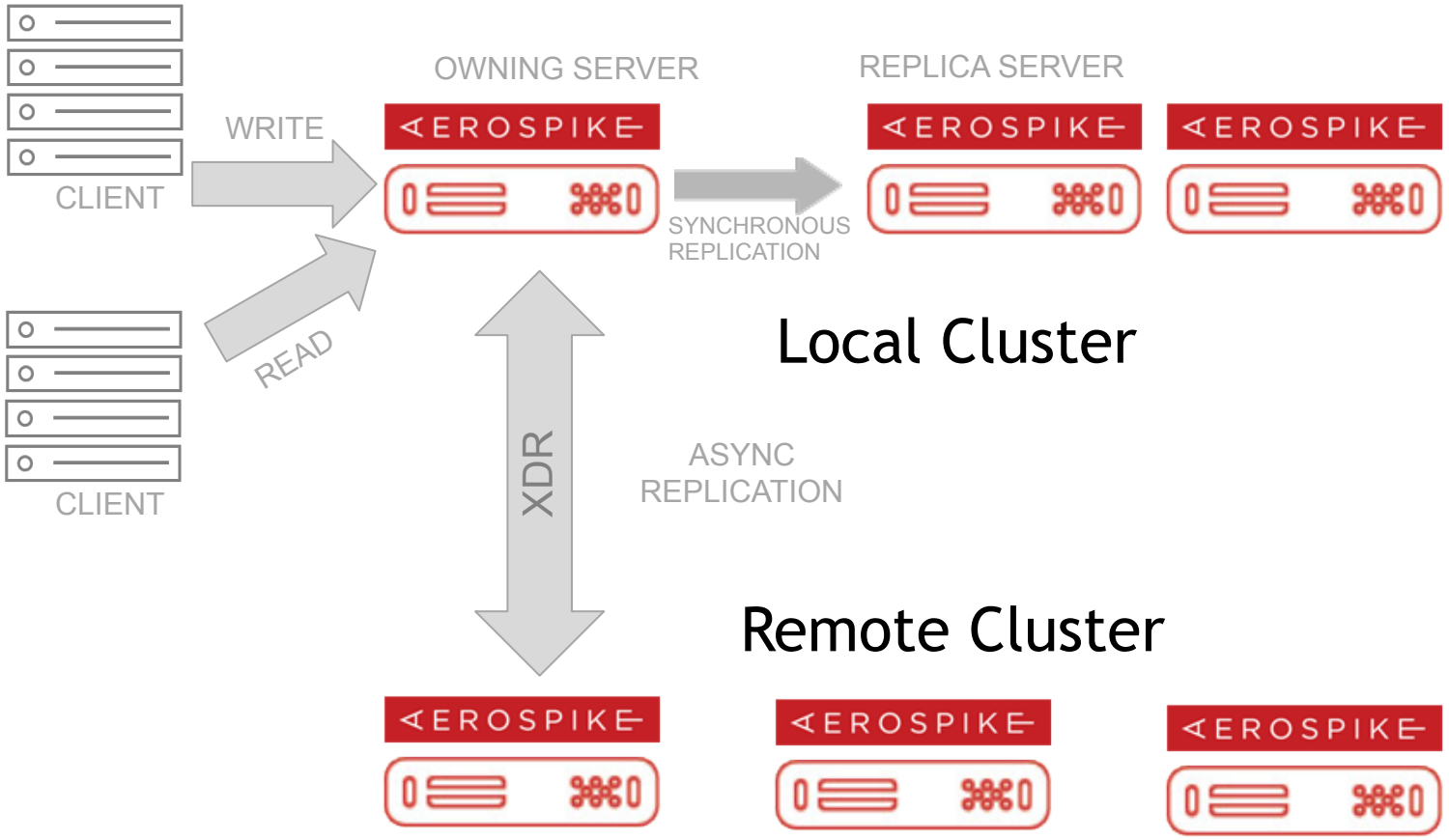
- Linear scaling
- Automatic Distribution of Data Using
- Smart Clients



Smart Clients



Data Correctness



Cacheless Architecture

✓ Reduce server count

✓ No cache inconsistencies

Immediate Consistency within a Cluster

✓ No stale reads

✓ No quorum reads

Workload Agnostic

✓ Read heavy

✓ Write heavy

✓ Mixed

Geographic Replication (XDR)

✓ Efficient Pipelining

✓ Asynchronous transfer

Hybrid Memory Architecture

- Flash Optimized Storage Layer: Indexes in DRAM, Data on SSD
- Multi-threaded & Massively Parallel
- **Self-healing Clusters**

Self-healing Clusters

- High Uptime
- Self-healing
- Low Management

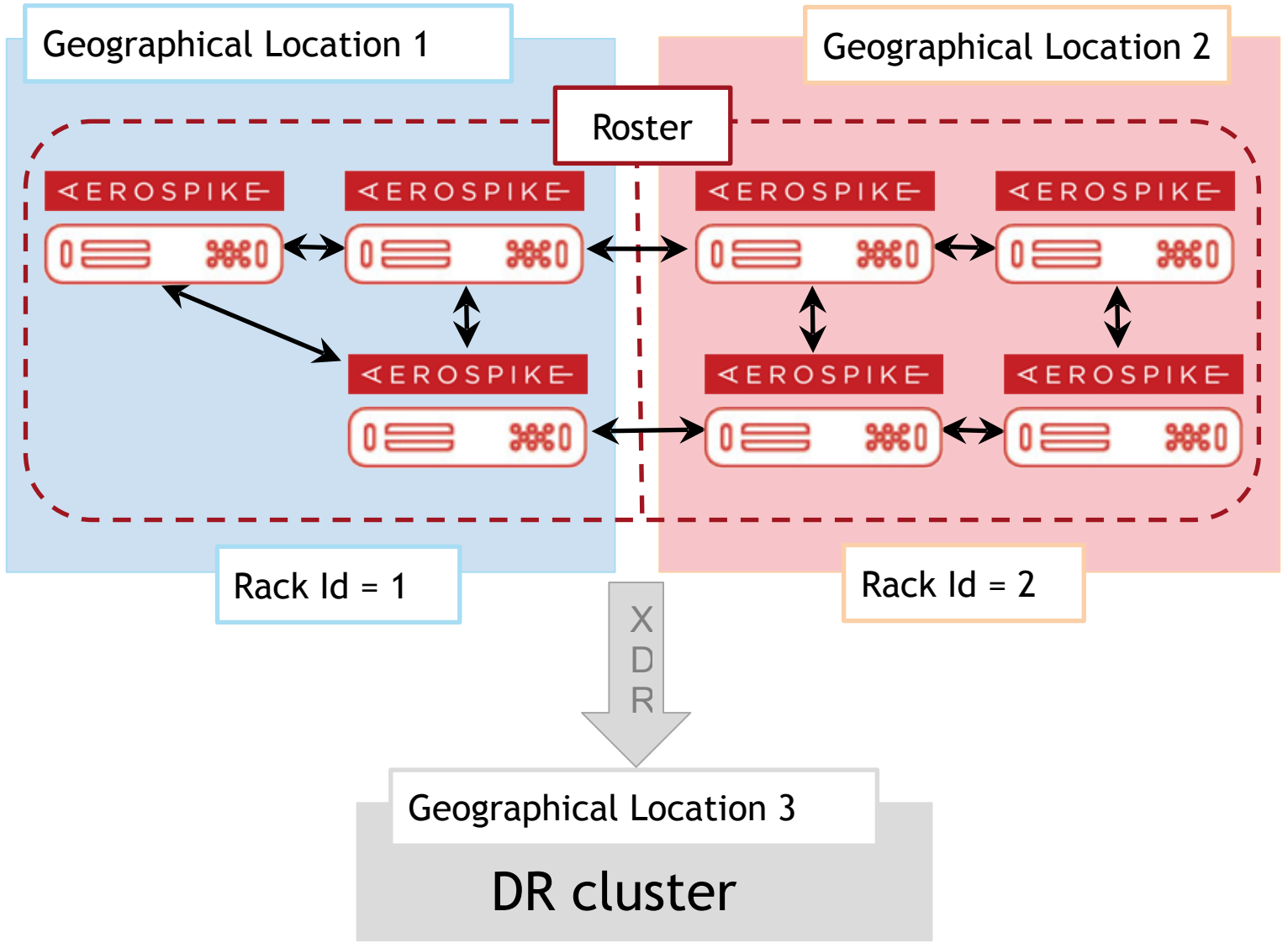
Strong Consistency with High Availability

Guaranteed Strong Consistency
✓ Even with network splits

Rack-Awareness used
✓ Master and replica in different data centers
✓ Synchronous writes

High Availability
✓ Lose any node => 100% available
✓ Lose a DC => 100% available
✓ Might require one operator command

Low cost in Cloud
✓ Reads can use local DC
✓ Minor increase in chance of stale reads



Availability vs. Strong Consistency

"Aerospike does appear to provide linearizability through network partitions and process crashes"

- Kyle Kingsbury, Jepsen.io (https://jepsen.io/analyses/aerospike-3-99-0-3)

AEROSPIKE

Availability vs. Strong Consistency

"Aerospike does appear to provide linearizability through network partitions and process crashes"

- Kyle Kingsbury, Jepsen.io (https://jepsen.io/analyses/aerospike-3-99-0-3)

	Linearizable Consistency	Sequential Consistency	Availability Mode
OPS (million)	1.87m	5.95m	6m
Read Latency	548 µs	225 µs	220 µs
Update Latency	630 µs	640 µs	640 µs

Configuration:

- In-memory
- Persistence enabled
- 5 node cluster
- 500M keys
- Replication factor 2
- 8 byte integer objects

AEROSPIKE

Integrating Aerospike

AEROSPIKE

Aerospike Clients

Find the latest Aerospike Client library for your favorite language.
Client libraries includes:

- Source or binary for the language
- Examples
- Benchmark tool

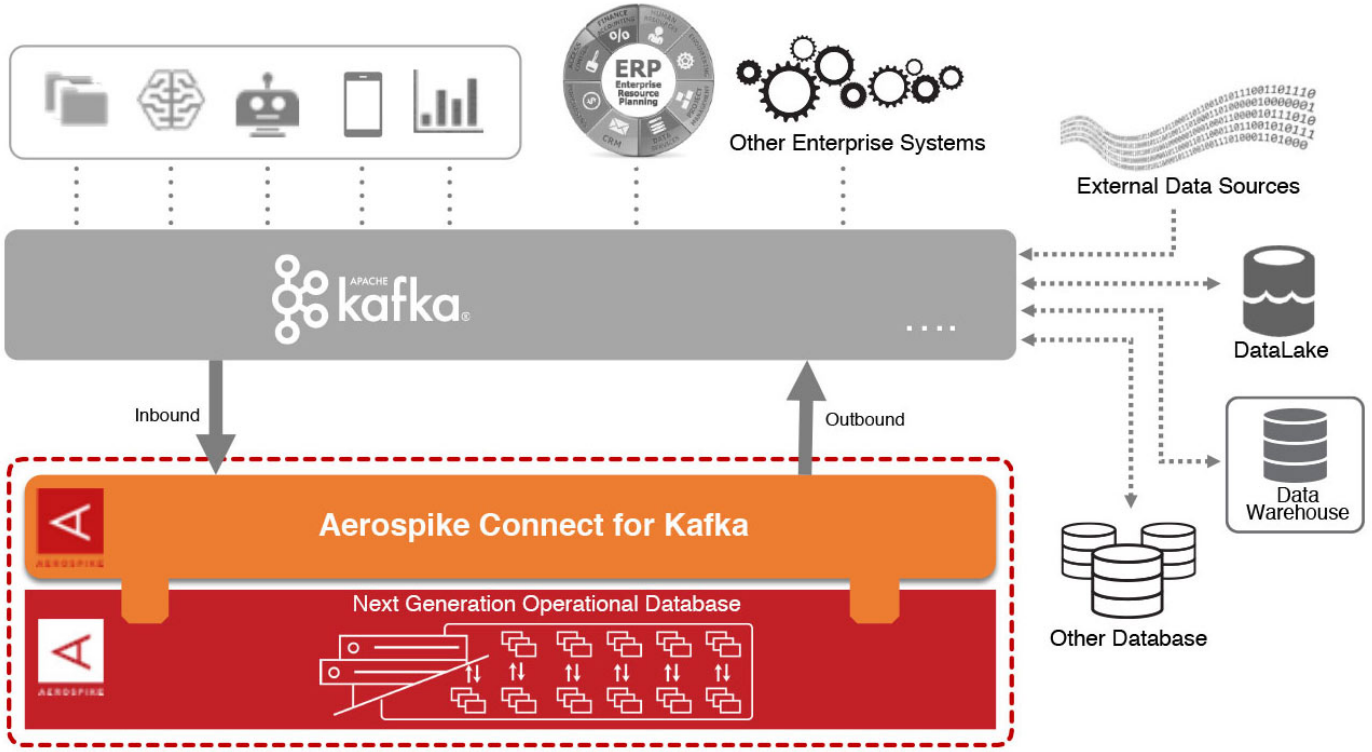
If you don't already have Aerospike Server up and running, **download and install it**.

For more information on Aerospike's functionality, visit **the feature guides**.

C / C++ 4.6.13 C Client Library	Download
Java 4.4.9 Java Client Library	Download
Go 2.8.1 Go Client Library	Download
C# / .NET 3.9.3 C# Client Library	Download
Python 3.10.0 Python Client Library	Download
REST 1.1.1 Aerospike REST Client	Download
Node.js 3.14.1 Node.js Client Library	Download
Ruby 2.11.0 Ruby Client Library	Download
PHP 7.4.2 PHP Client Library	Download
Rust 0.4.0 Rust Client Library	Download

Orchestration Platforms

Connector for Kafka

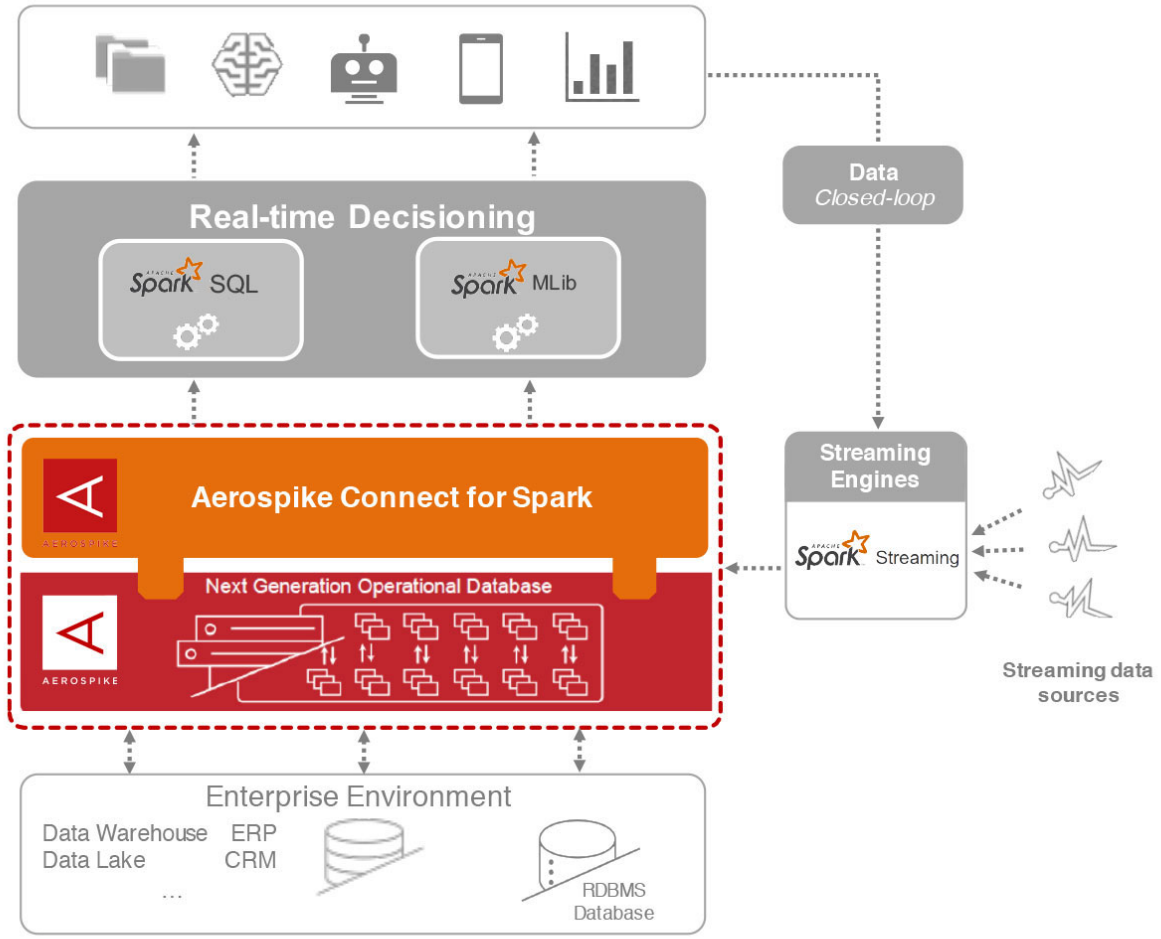


Example Use Cases

- ✓ **Edge-to-core data movement** in fraud prevention systems and financial trading systems
- ✓ **Edge-to-core data integration** in retail user behavior at the POS, checkout with Product data for propensity & dynamic pricing
- ✓ **Integration of operational data** stored in Aerospike with DataLake / EDW

AEROSPIKE

Connector for Spark



Example Use Cases

- ✓ **Fraud prevention:** transaction data via streaming and need to analyze based on historical data in real time
- ✓ **Recommendation Engines:** Real-time recommendations and targeting based on user behavior
- ✓ **Ad Tech:** Ad Fraud and real-time retargeting base on user behavior
- ✓ **Digital Identity Management**
- ✓ **Industrial Internet of Things (IIoT):** Real-time & closed loop business decisions

AEROSPIKE

Other Features

Query Capabilities

Secondary indexes provide the ability query information based on a predicate across the database. Scans allow access to all the data. Predicate Expressions allow optimized, server-side data filtering.

Data structure & Document indexing

Easy-to-program data structures such as lists, maps, which improve performance and allow indexed lookups on field values.

Simple, Clean API

Harness powerful database features with minimal developer complexity.

Enterprise Security

Full transport encryption. Support for encrypted storage and in-database transparent data encryption. Authentication, access control, exception logging. LDAP and Kerberos integration.

Primary Key Consistency, High Performance

Strong Consistency on primary key access with zero performance penalty. Commit to device support for demanding correctness environments.

High Availability (AP mode)

Used traditionally for high performance machine-to-machine analytic systems, such as fraud detection, pattern matching, and real-time pricing.

Aerospike Connect for Spark

Deep integration with Spark to allow SQL and ML based tools to operate seamlessly and efficiently

Compression At Rest

Selectable compression algorithm to optimize compression ratio based on specific data usage

AEROSPIKE

Thank You!

Natalie Pistunovich
@NataliePis @aerospikeDB

AEROSPIKE