ARC306

Event Handling at Scale

Designing an Auditable Ingestion and Persistence Architecture for 10K+ Events/Second

Terri Sage, VP Engineering, McGraw-Hill Education

Benjamin Feldon, Solutions Architect, AWS

November 2016

amazon web services

What to Expect from the Session

- Business Background: Reporting and event-driven analytics for hundreds of thousands of concurrent learners in a reliable, secure, and auditable manner that is cost effective
- Learning events
- Reporting & analytics architecture
- Architecture tradeoffs
- Challenges, built confidence, and lessons learned

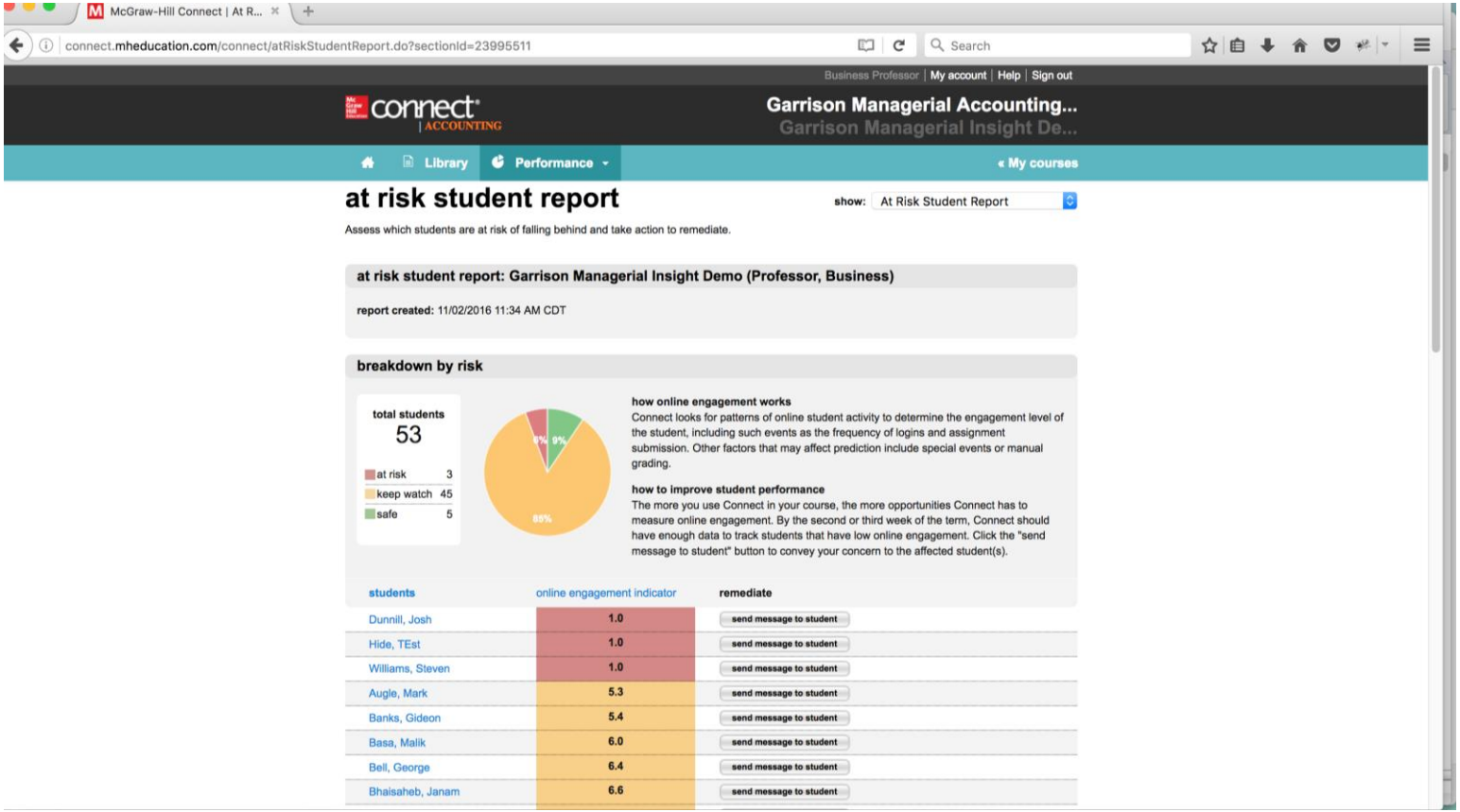
Background

- McGraw-Hill Education is a digital learning company
- Learning Management Systems and Learning Analytics
- 2 million to 14 million students, initial load 10 thousand events per second to scale to 15 million per second
- Cyclical and cost conscious nature of business – low price point per student
- Service Level Agreements
- ‘Just-In-Time’ Insights; Example is Connect Insights

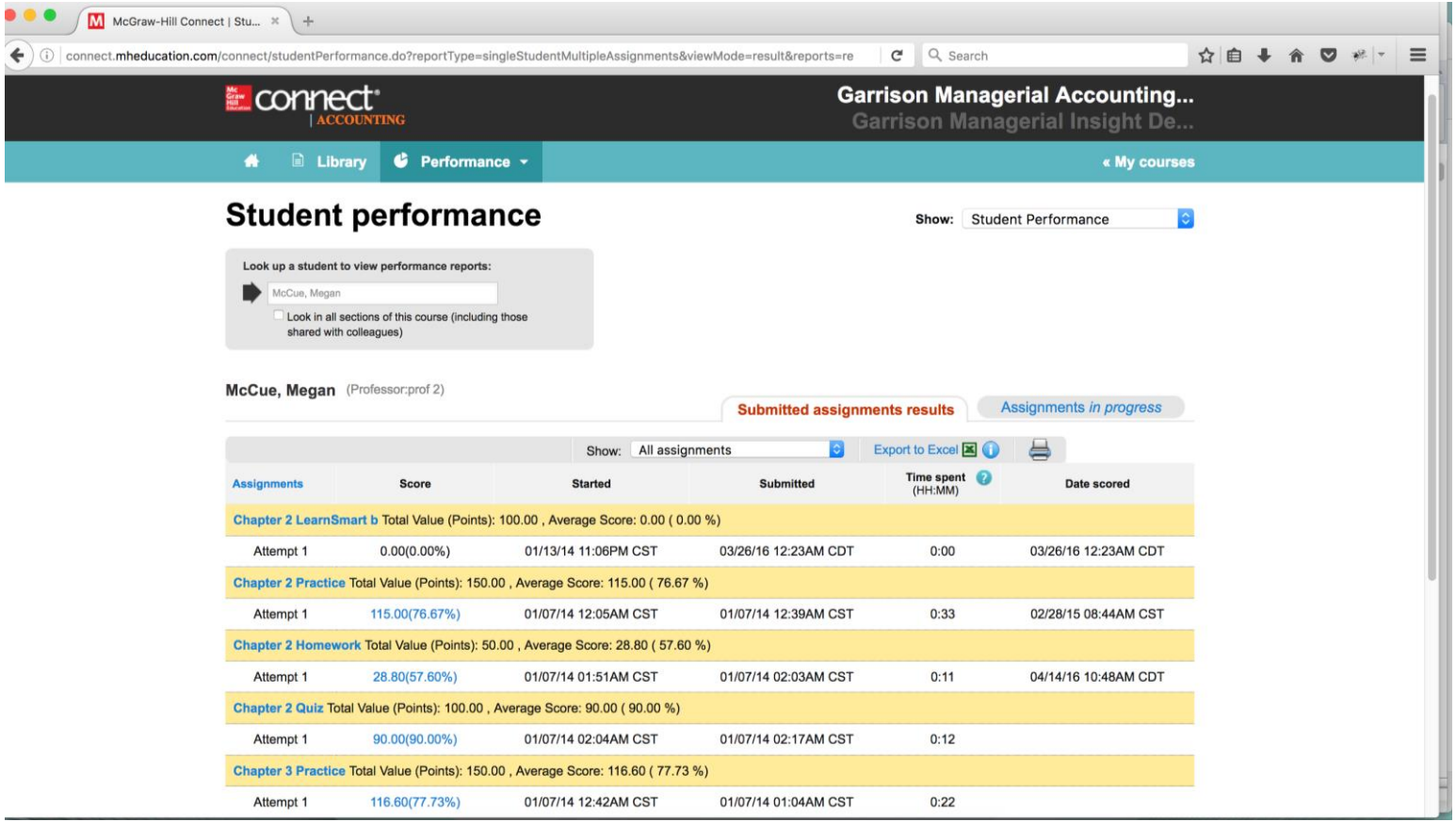
Background

- McGraw-Hill Education is a digital learning company
- Learning Management Systems and Learning Analytics
- 2 million to 14 million students, initial load 10 thousand events per second to scale to 15 million per second
- Cyclical and cost conscious nature of business – low price point per student
- Service Level Agreements
- ‘Just-In-Time’ Insights; Example is Connect Insights

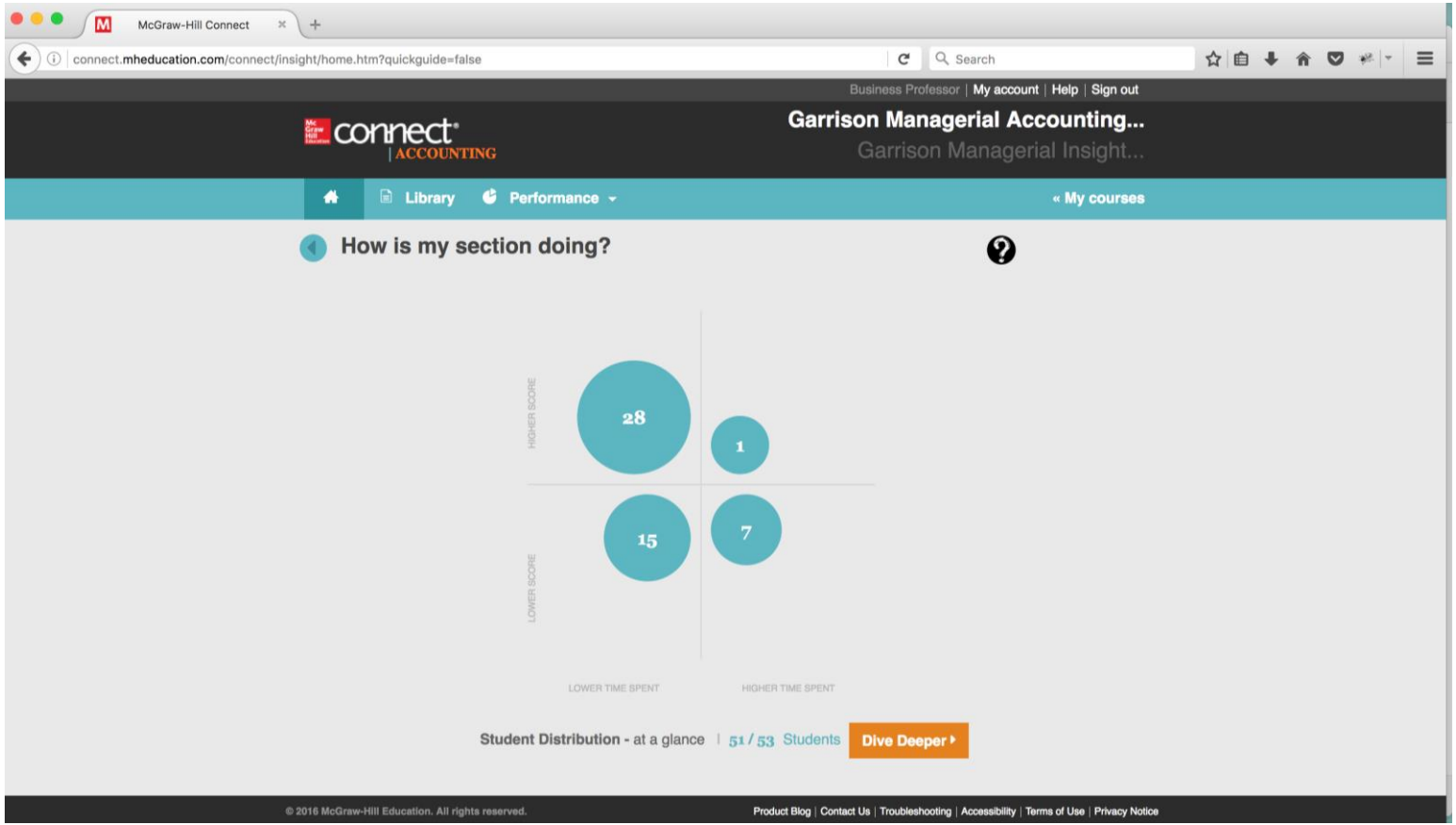
Connect Insights



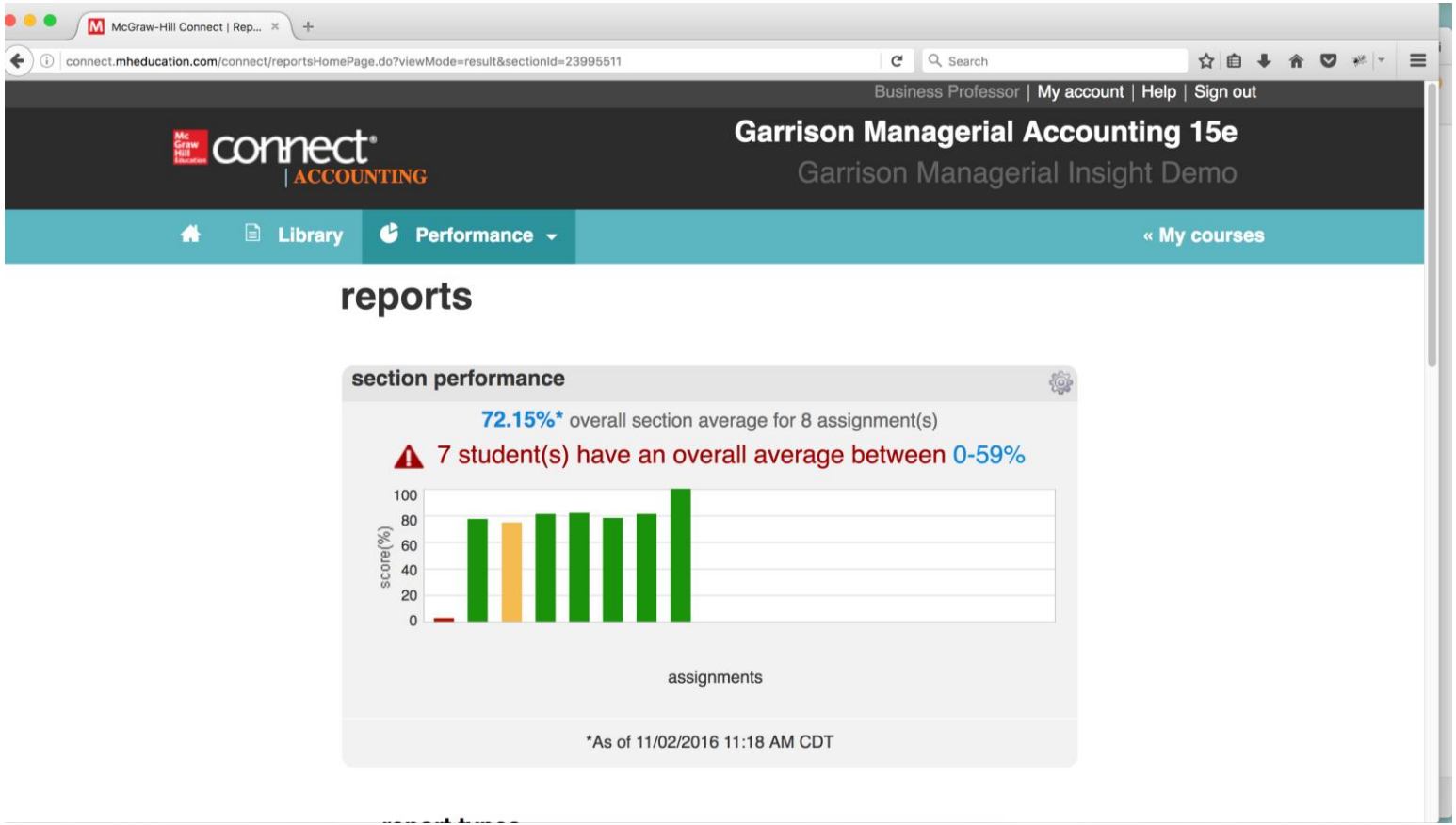
Connect Insights



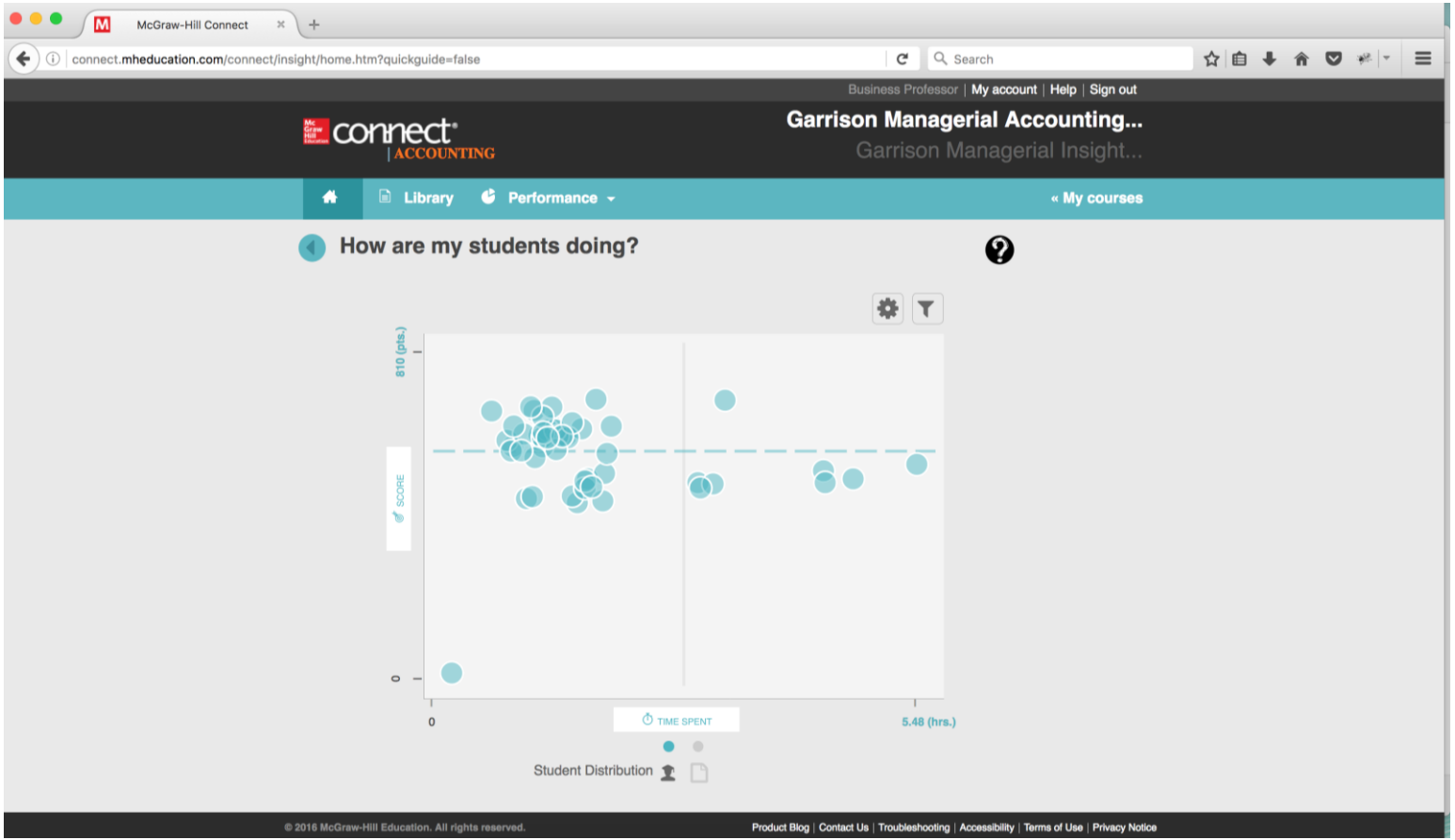
Connect Insights



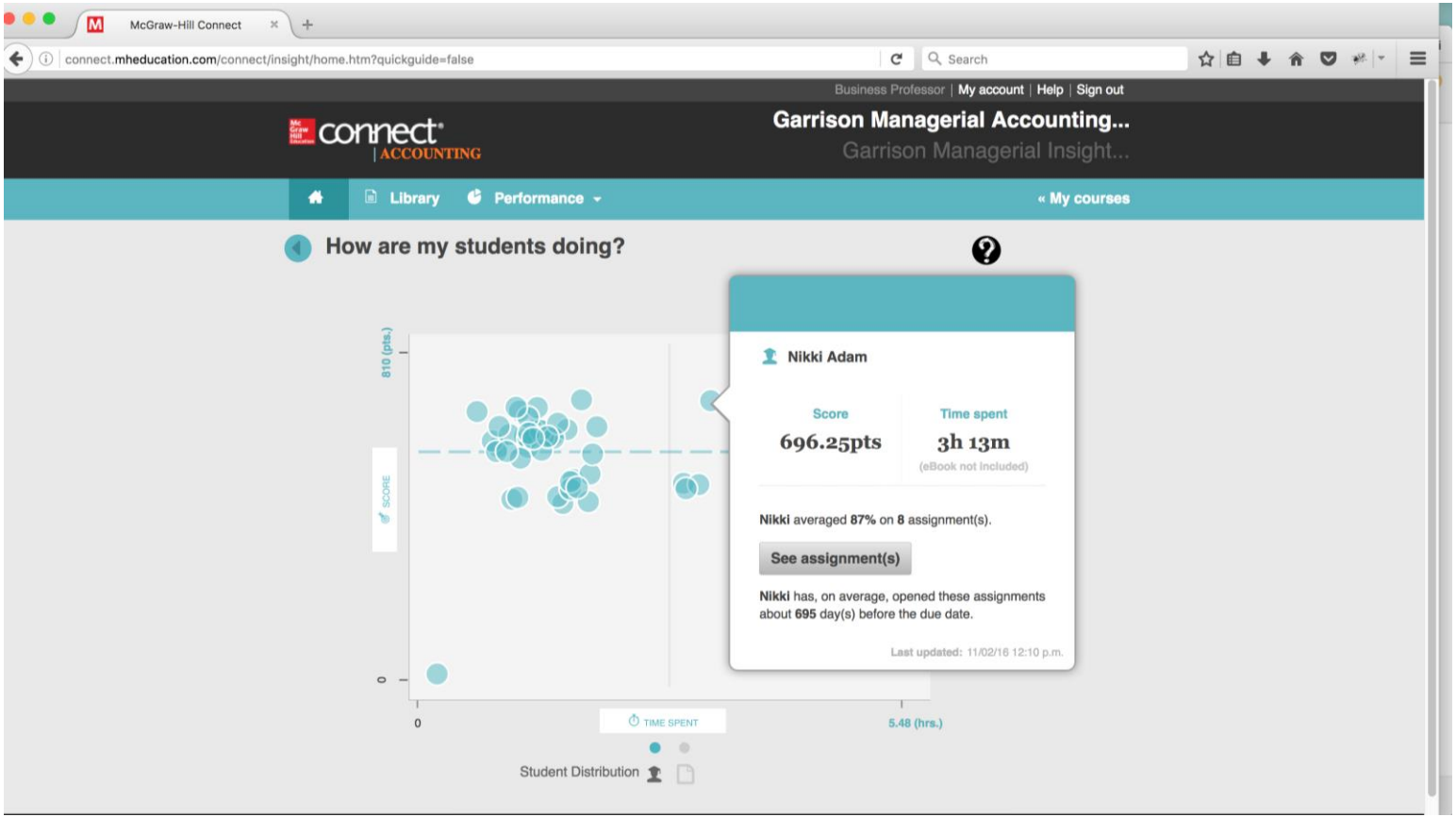
Connect Insights



Connect Insights



Connect Insights



IMS Global Learning Consortium

- Ingest 47 Caliper events
- Domain events
- https://www.imsglobal.org

Membership Events Global Login

IMS GLOBAL Learning Consortium
Better Learning From Better Learning Technology

INITIATIVES DEVELOPERS PRODUCT DIRECTORY LEARNING IMPACT LEADERSHIP

IMS GLOBAL Learning Consortium

IMS Caliper Analytics™ Implementation Guide

Version 1.0 Final
Date Issued: 15 October 2015

Latest version: http://www.imsglobal.org/caliper

IPR and Distribution Notices

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the specification set forth in this document, and to provide supporting documentation.

IMS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on IMS's procedures with respect to rights in IMS specifications can be found at the IMS Intellectual Property Rights web page: http://www.imsglobal.org/ipr/imsipr_policyFinal.pdf.

Copyright © 2015 IMS Global Learning Consortium. All Rights Reserved.

Use of this specification to develop products or services is governed by the license with IMS found on the IMS website: http://www.imsglobal.org/license.html.

Permission is granted to all parties to use excerpts from this document as needed in producing requests for proposals.

The limited permissions granted above are perpetual and will not be revoked by IMS or its successors or assigns.

THIS SPECIFICATION IS BEING OFFERED WITHOUT ANY WARRANTY WHATSOEVER, AND IN PARTICULAR, ANY WARRANTY OF NONINFRINGEMENT IS EXPRESSLY DISCLAIMED. ANY USE OF THIS SPECIFICATION SHALL BE MADE ENTIRELY AT THE IMPLEMENTER'S OWN RISK, AND NEITHER THE CONSORTIUM, NOR ANY OF ITS MEMBERS OR SUBMITTERS, SHALL HAVE ANY LIABILITY WHATSOEVER TO ANY IMPLEMENTER OR THIRD PARTY FOR ANY DAMAGES OF ANY NATURE WHATSOEVER, DIRECTLY OR INDIRECTLY, ARISING FROM THE USE OF THIS SPECIFICATION.

© 2015 IMS Global Learning Consortium, Inc.
All Rights Reserved.
Trademark information: http://www.imsglobal.org/copyright.html
Document Name: *IMS Global Caliper Analytics Implementation Guide – Final v1.0*
Revision: 15 October 2015

Example of Caliper Event


```
{
    "assessmentItem": {
        "xmlns": "http://www.imsglobal.org/xsd/imsqti_v2p1",
        "xmlns:xsi": "http://www.w3.org/2001/XMLSchema-instance",
        "xsi:schemaLocation": "…",
        "identifier": "MACHINE-ITEM04",
        "title": "Chopper is a type of?",
        "label": "Fill-In Blank Question on a Chopper",
        "xml:lang": "en-US",
        "timeDependent": "true",
        "adaptive": "false",
        "responseDeclaration": {
            "identifier": "RESPONSE",
            "cardinality": "single",
            "baseType": "identifier",
            "correctResponse": {
                "value": "motorcycle"
            }
        },
        "outcomeDeclaration": [
            {
                "identifier": "SCORE",
                "cardinality": "single",
                "baseType": "float",
                "masteryValue": "0.0",
                "defaultValue": {
                    "value": "0.0"
                }
            },
            {
                "identifier": "MAXSCORE",
                "cardinality": "single",
                "baseType": "float",
                "defaultValue": {
                    "value": "1.0"
                }
            },
```

Example of Caliper Event

```
{
    "assessmentItem": {
        "xmlns": "http://www.imsglobal.org/xsd/imsqti_v2p1",
        "xmlns:xsi": "http://www.w3.org/2001/XMLSchema-instance",
        "xsi:schemaLocation": "…",
        "identifier": "MAGNUM_ITEM01",
        "title": "Chopper is a type of?",
        "label": "Fill In Blank Question on a Chopper",
        "xml:lang": "en-US",
        "timeDependent": "true",
        "adaptive": "false",
        "responseDeclaration": {
            "identifier": "RESPONSE",
            "cardinality": "single",
            "baseType": "identifier",
            "correctResponse": {
                "value": "motorcycle"
            }
        },
        "outcomeDeclaration": [
            {
                "identifier": "SCORE",
                "cardinality": "single",
                "baseType": "float",
                "masteryValue": "0.0",
                "defaultValue": {
                    "value": "0.0"
                }
            },
            {
                "identifier": "MAXSCORE",
                "cardinality": "single",
                "baseType": "float",
                "defaultValue": {
                    "value": "1.0"
                }
            },
            {
```

Example of Caliper Event

```
{
    "assessmentItem": {
        "xmlns": "http://www.imsglobal.org/xsd/imsqti_v2p1",
        "xmlns:xsi": "http://www.w3.org/2001/XMLSchema-instance",
        "xsi:schemaLocation": "…",
        "identifier": "MACHINE-ITEM04",
        "title": "Chopper is a type of?",
        "label": "Fill-In Blank Question on a Chopper",
        "xml:lang": "en-US",
        "timeDependent": "true",
        "adaptive": "false",
        "responseDeclaration": {
            "identifier": "RESPONSE",
            "cardinality": "single",
            "baseType": "identifier",
            "correctResponse": {
                "value": "motorcycle"
            }
        },
        "outcomeDeclaration": [
            {
                "identifier": "SCORE",
                "cardinality": "single",
                "baseType": "float",
                "masteryValue": "0.0",
                "defaultValue": {
                    "value": "0.0"
                }
            },
            {
                "identifier": "MAXSCORE",
                "cardinality": "single",
                "baseType": "float",
                "defaultValue": {
                    "value": "1.0"
                }
            },
```

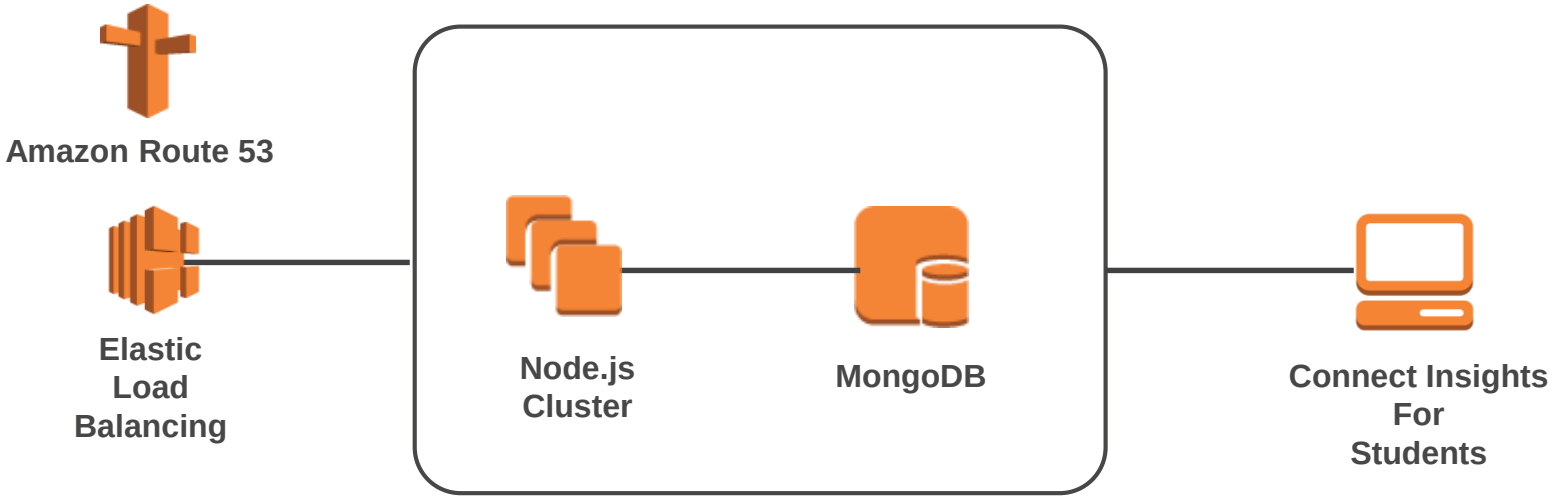
Example of Caliper Event

```
{
    "assessmentItem": {
        "xmlns": "http://www.imsglobal.org/xsd/imsqti_v2p1",
        "xmlns:xsi": "http://www.w3.org/2001/XMLSchema-instance",
        "xsi:schemaLocation": "…",
        "identifier": "MACHINE-ITEM04",
        "title": "Chopper is a type of?",
        "label": "Fill-In Blank Question on a Chopper",
        "xml:lang": "en-US",
        "timeDependent": "true",
        "adaptive": "false",
        "responseDeclaration": {
            "identifier": "RESPONSE",
            "cardinality": "single",
            "baseType": "identifier",
            "correctResponse": {
                "value": "motorcycle"
            }
        },
        "outcomeDeclaration": [
            {
                "identifier": "SCORE",
                "cardinality": "single",
                "baseType": "float",
                "masteryValue": "0.0",
                "defaultValue": {
                    "value": "0.0"
                }
            },
            {
                "identifier": "MAXSCORE",
                "cardinality": "single",
                "baseType": "float",
                "defaultValue": {
                    "value": "1.0"
                }
            },
```

Example of Caliper Event

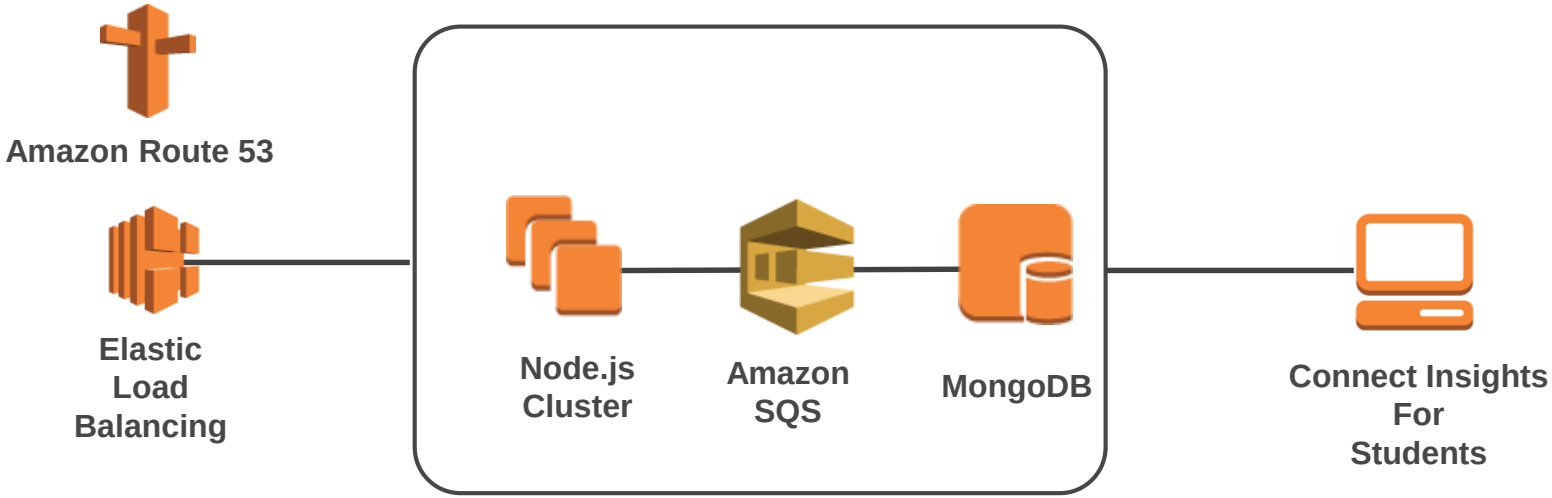
```
{
    "assessmentItem": {
        "xmlns": "http://www.imsglobal.org/xsd/imsqti_v2p1",
        "xmlns:xsi": "http://www.w3.org/2001/XMLSchema-instance",
        "xsi:schemaLocation": "…",
        "identifier": "MACHINE-ITEM04",
        "title": "Chopper is a type of?",
        "label": "Fill-In Blank Question on a Chopper",
        "xml:lang": "en-US",
        "timeDependent": "true",
        "adaptive": "false",
        "responseDeclaration": {
            "identifier": "RESPONSE",
            "cardinality": "single",
            "baseType": "identifier",
            "correctResponse": {
                "value": "motorcycle"
            }
        },
        "outcomeDeclaration": [
            {
                "identifier": "SCORE",
                "cardinality": "single",
                "baseType": "float",
                "masteryValue": "0.0",
                "defaultValue": {
                    "value": "0.0"
                }
            },
            {
                "identifier": "MAXSCORE",
                "cardinality": "single",
                "baseType": "float",
                "defaultValue": {
                    "value": "1.0"
                }
            },
```

Evolution of the Learning Analytics Platform



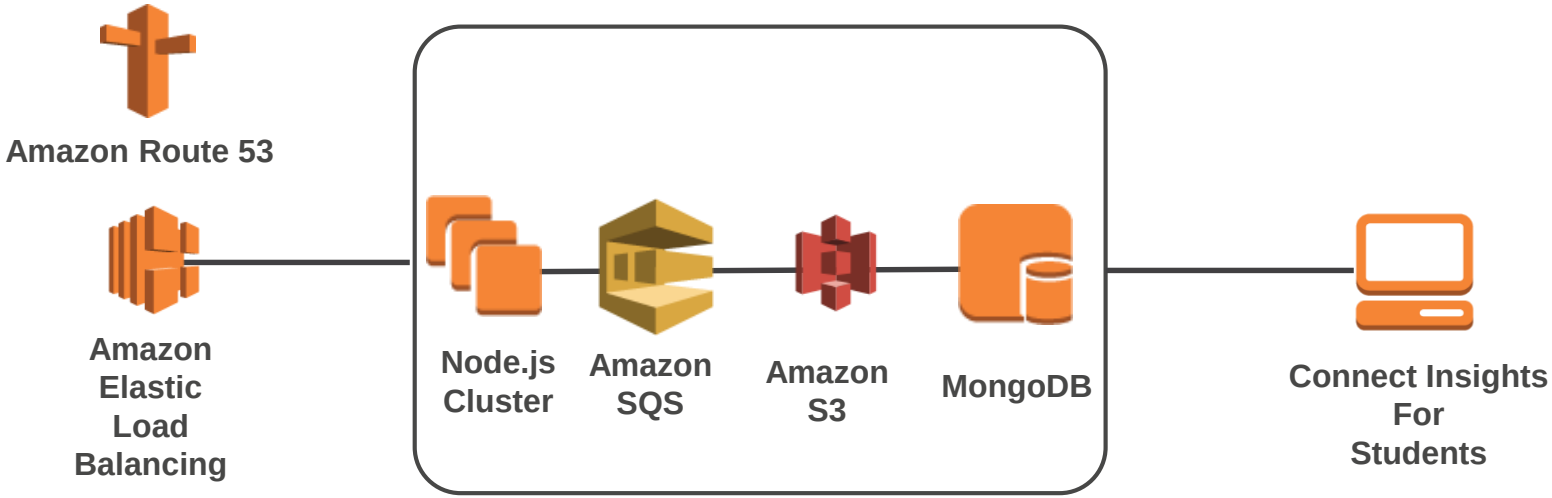
LAP 1.0: Cluster of Node.js servers writing aggregations to MongoDB.

Evolution of the Learning Analytics Platform



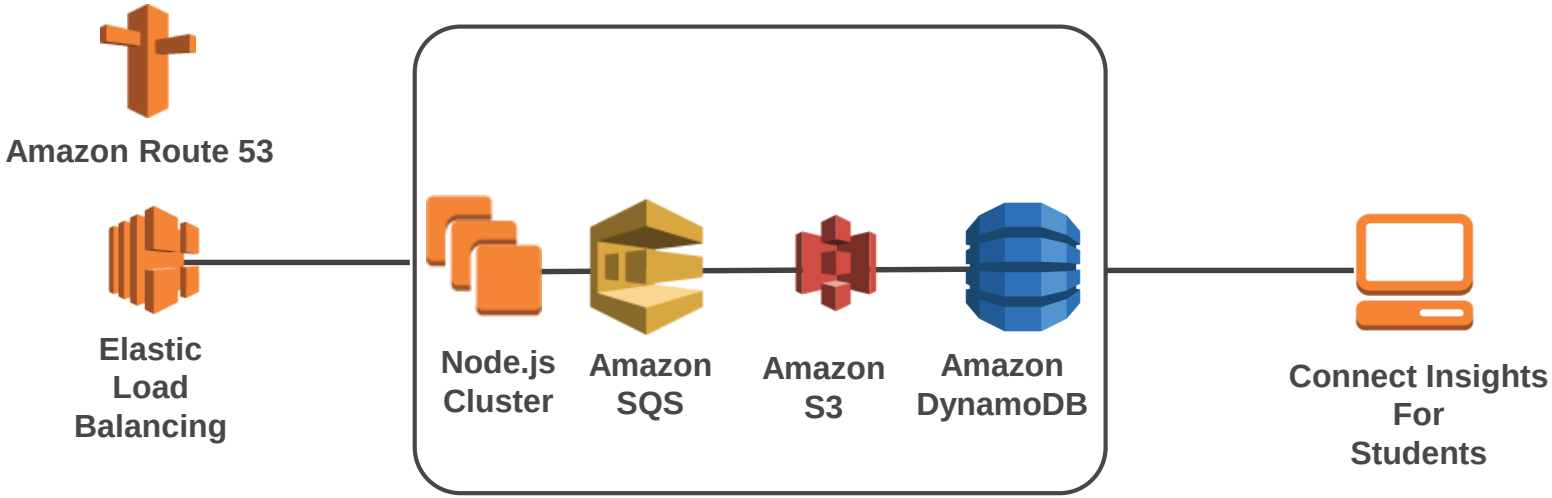
LAP 1.1: Put a queue in the middle.

Evolution of the Learning Analytics Platform



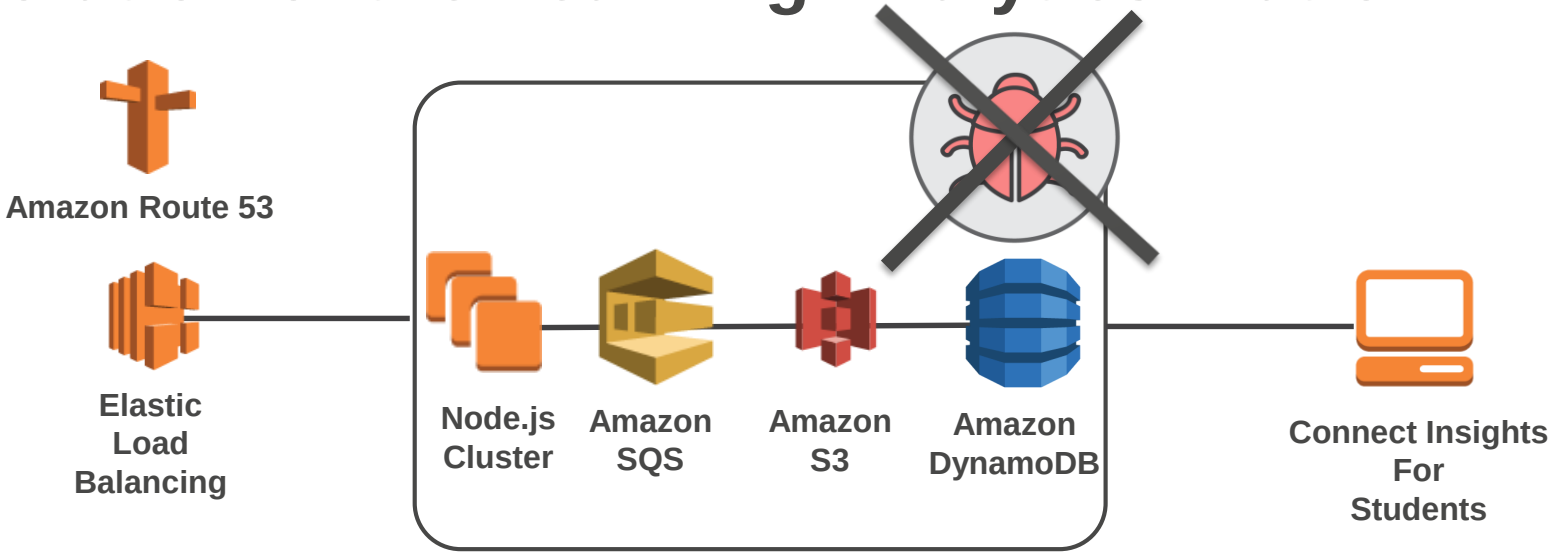
LAP 1.2: Added S3 to pre-aggregate events and then load into MongoDB.

Evolution of the Learning Analytics Platform



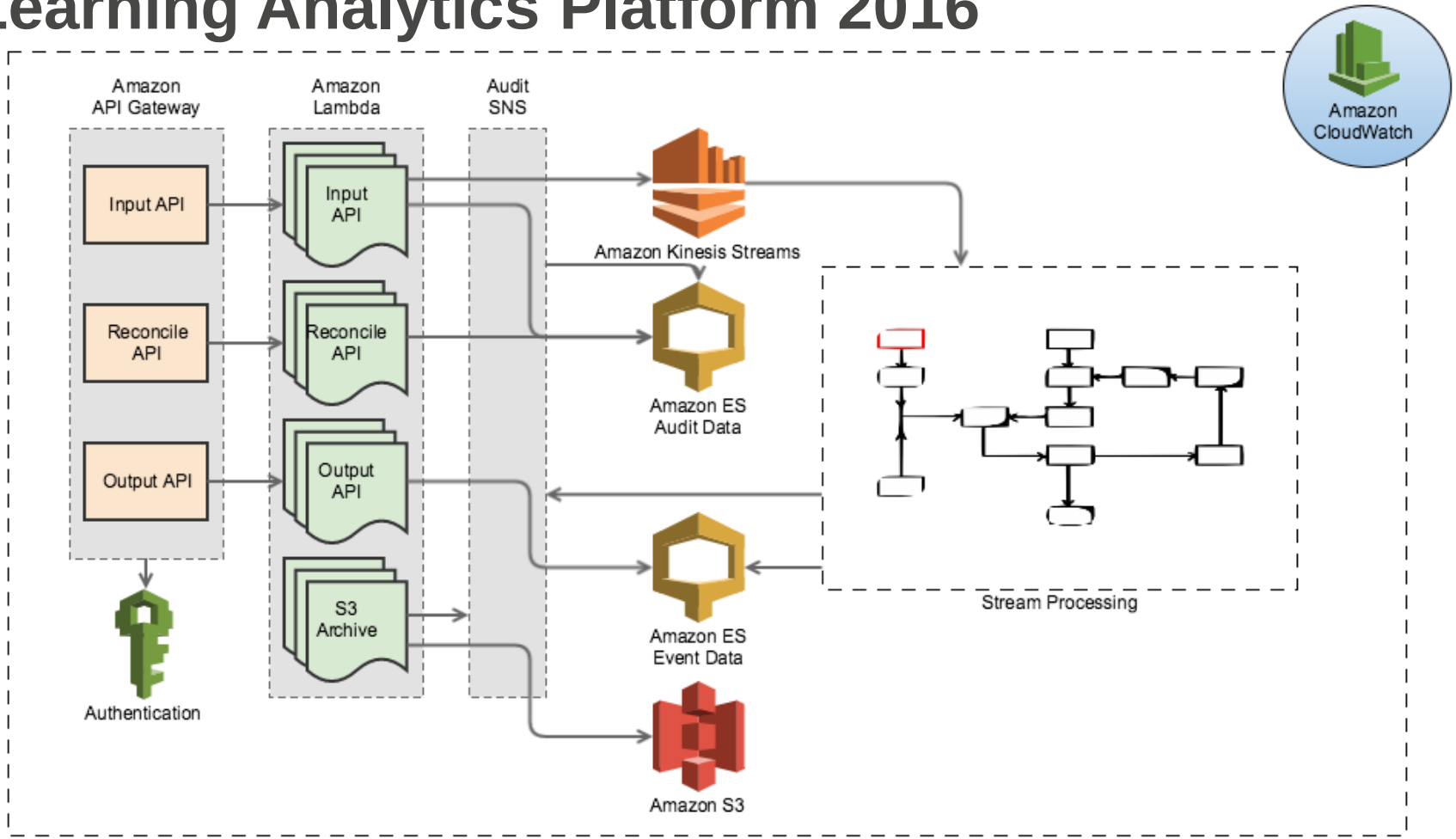
LAP 1.3: Replaced MongoDB with DynamoDB.

Evolution of the Learning Analytics Platform

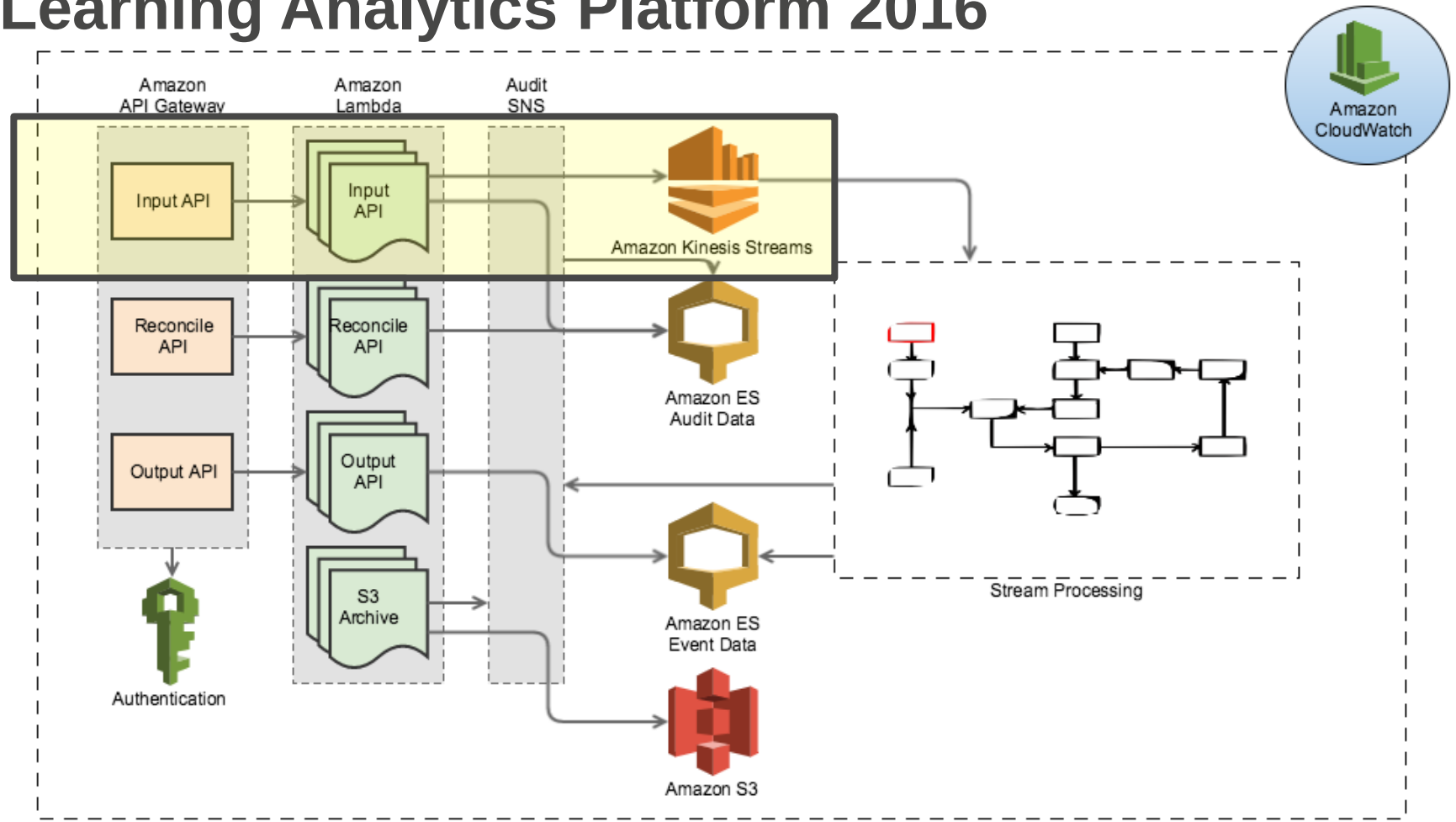


LAP 1.4 & LAP 1.5: Stabilized and fixed bugs.

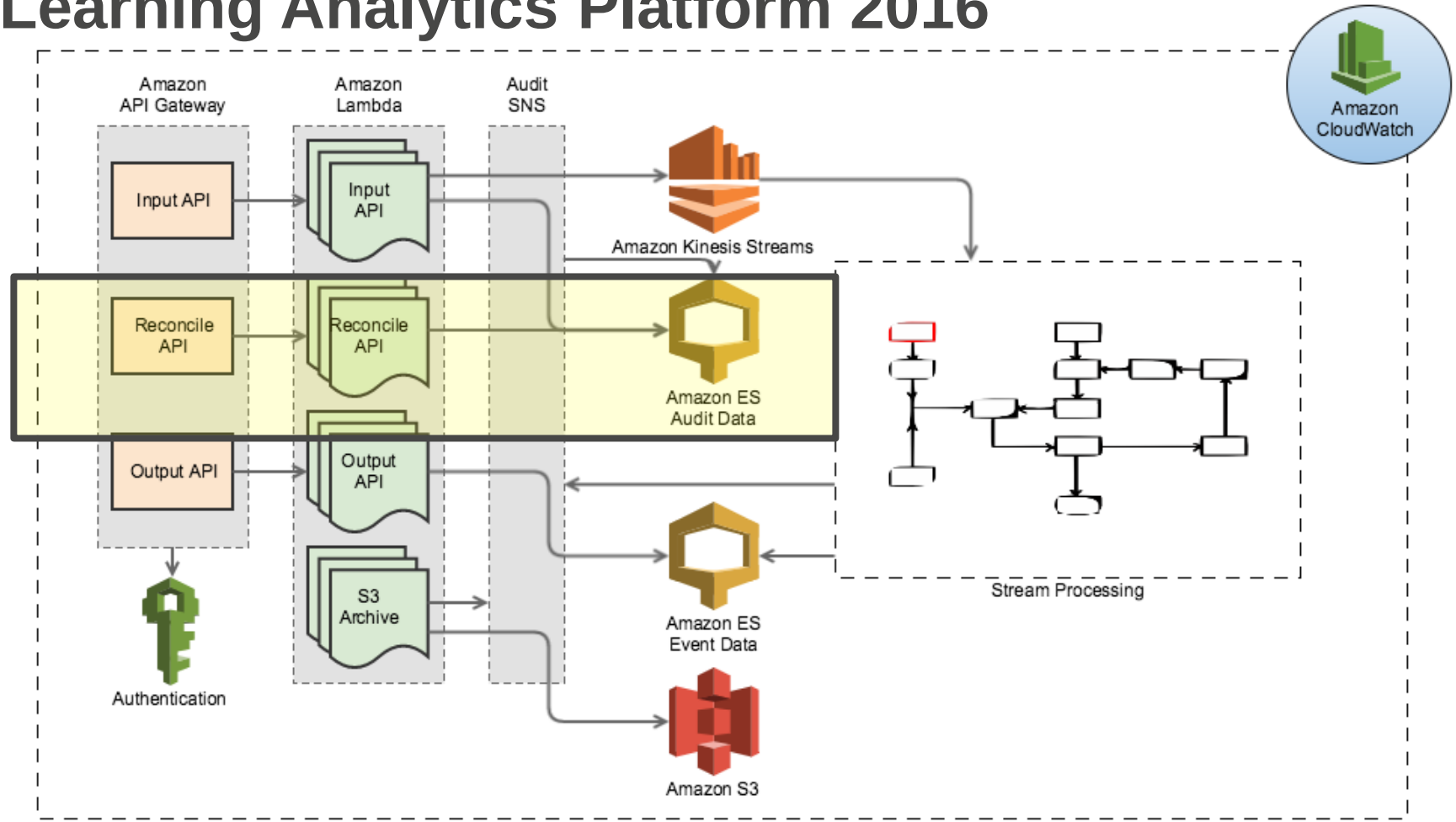
Learning Analytics Platform 2016



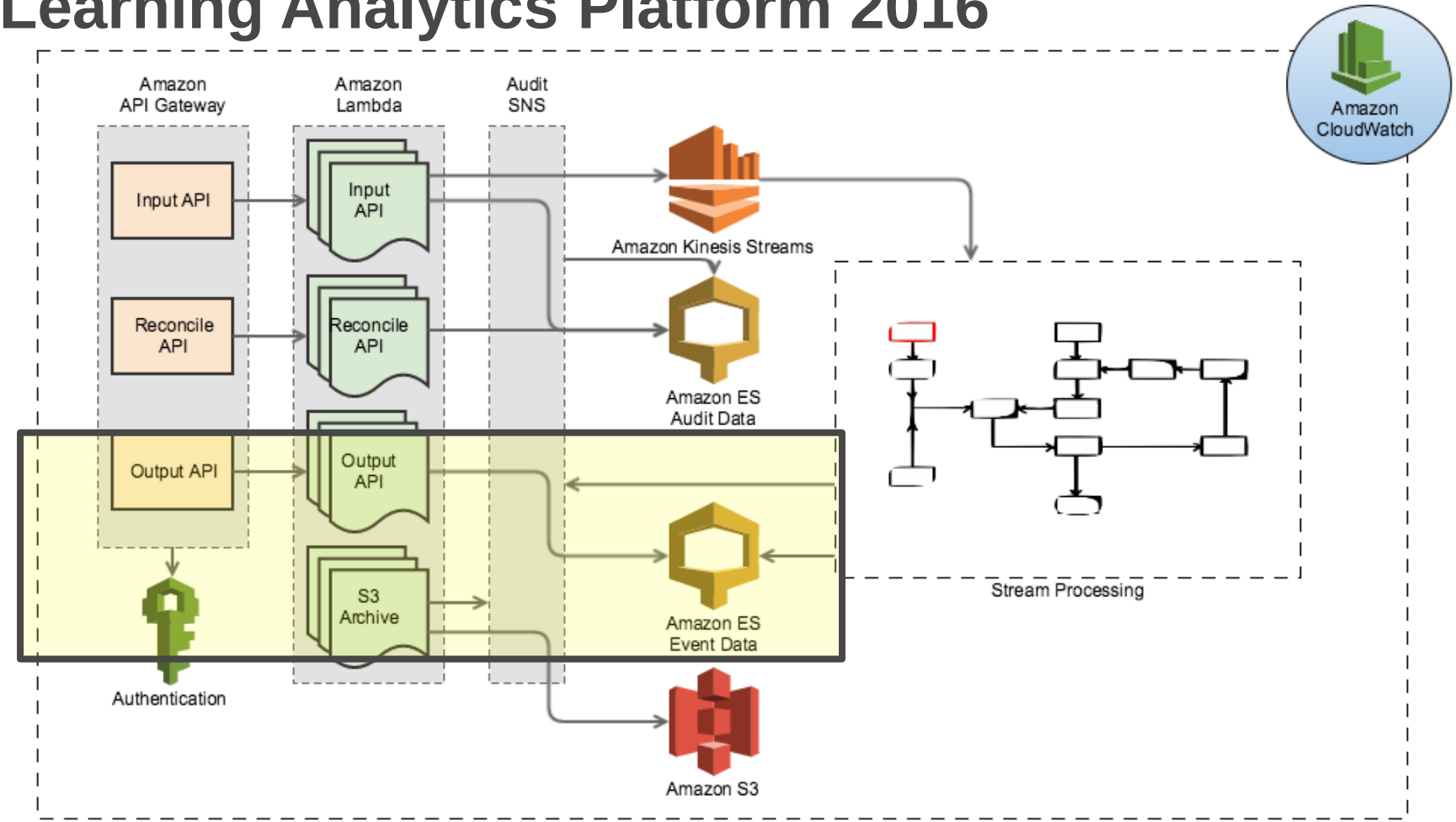
Learning Analytics Platform 2016



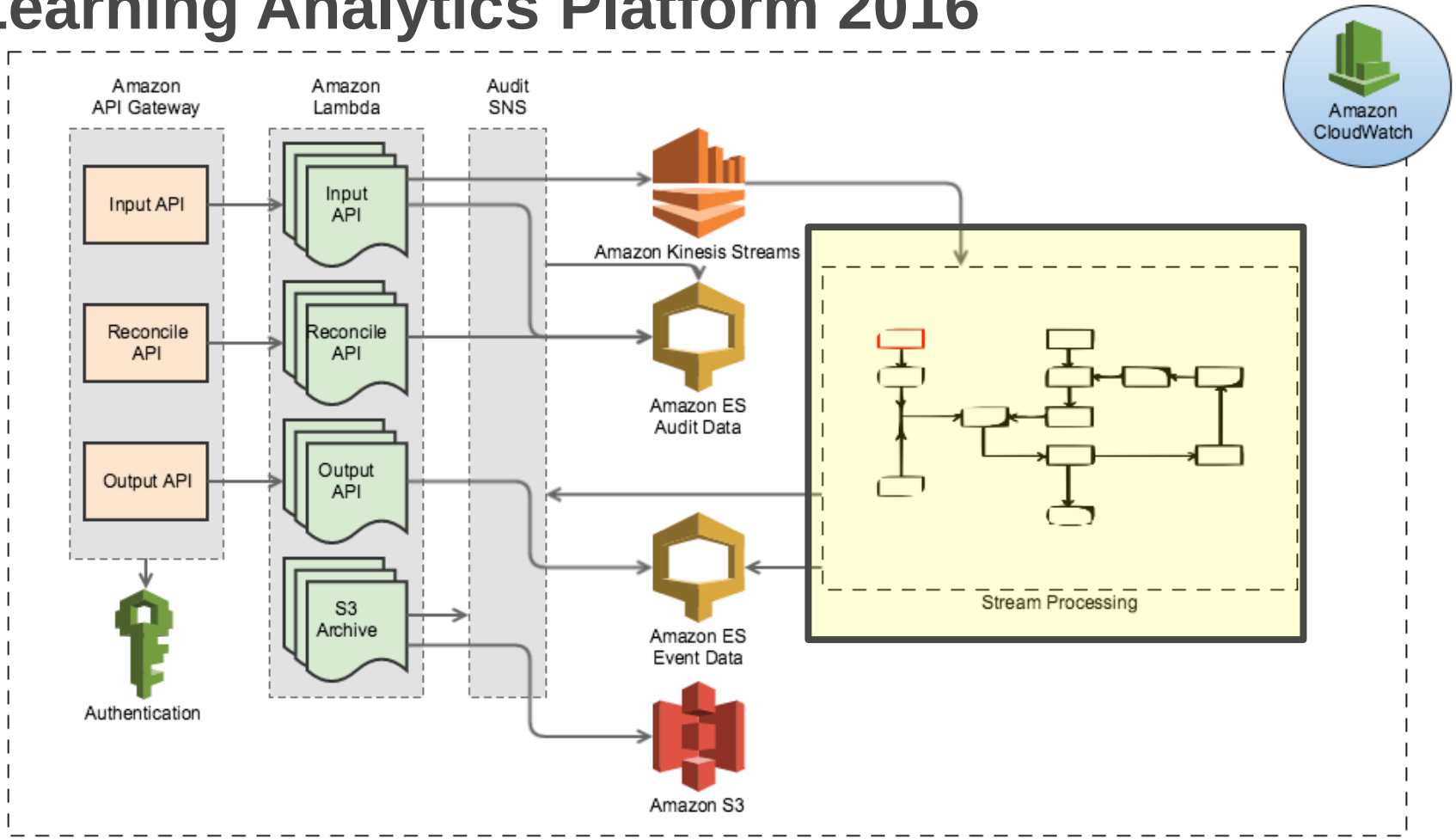
Learning Analytics Platform 2016



Learning Analytics Platform 2016



Learning Analytics Platform 2016



Learning Analytics Platform - Services

Amazon API Gateway

AWS Lambda

Amazon Kinesis Streams

Amazon Elasticsearch Service

Amazon Simple Storage Service (S3)

Amazon Relational Database Service (RDS)

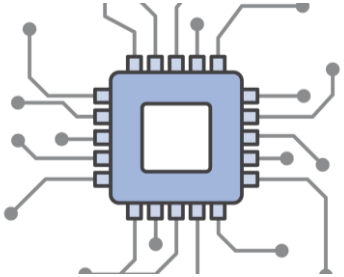
Amazon DynamoDB

Amazon API Gateway

Fully managed service for hosting HTTPS APIs on top of AWS

- Support for standard HTTP methods
- Authenticate and authorize requests
- Highly scalable parallel processing
- DDoS protection and throttling for back-end systems
- Support for standard HTTP methods
- Swagger Import/Export
- Custom domains

Benefits of Amazon API Gateway



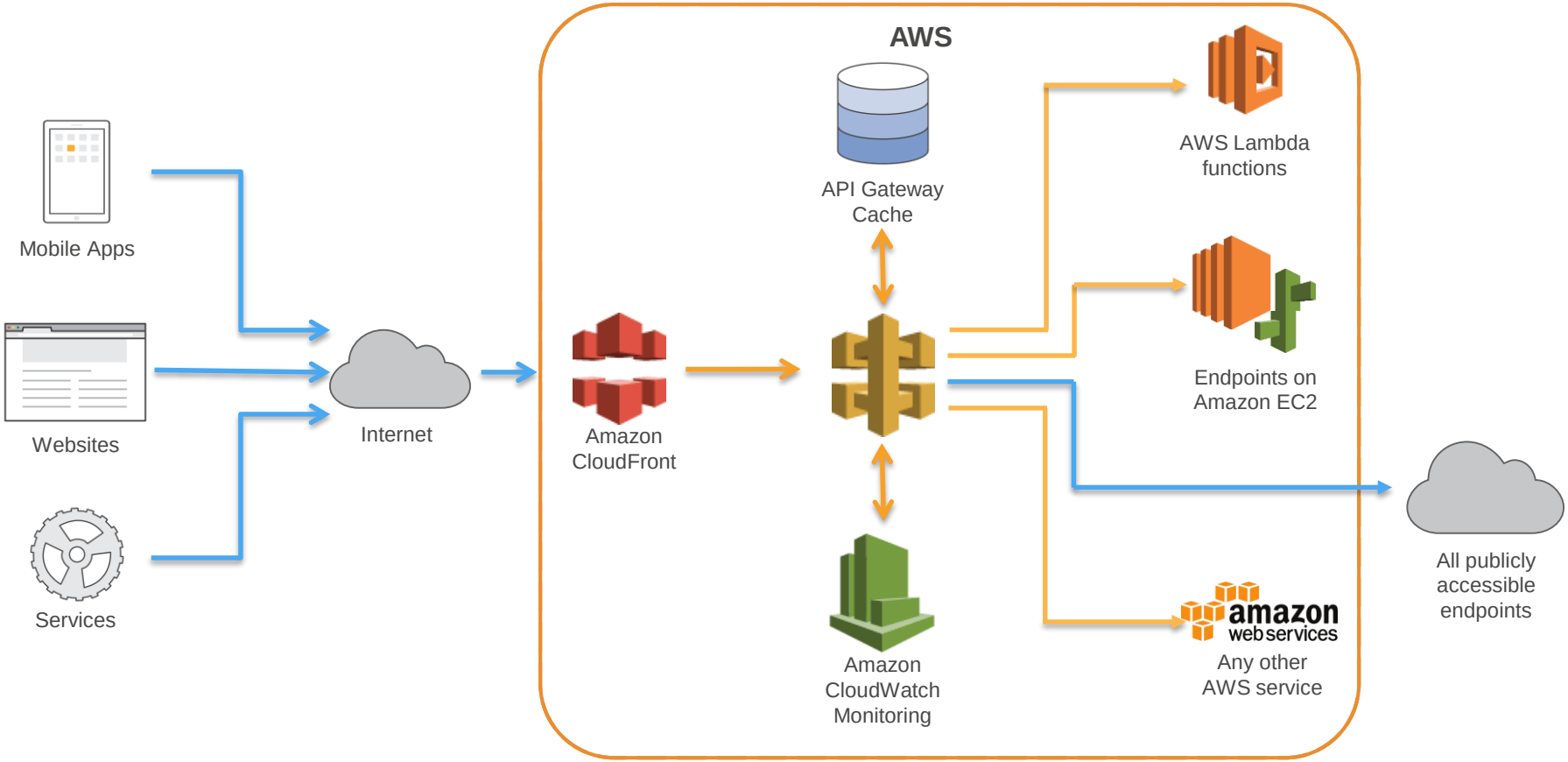
Create a unified API frontend for multiple micro-services

DDoS protection and throttling for your backend

Authenticate and authorize requests to a backend

Throttle, meter, and monetize API usage by 3rd party developers

API Gateway integrations



AWS Lambda

Serverless, event-driven compute service

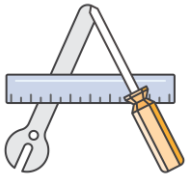
- Runs your function code without you managing or scaling servers
- Provides an API to trigger the execution of your function
- Ensures function is executed when triggered, in parallel, regardless of scale
- Provides additional capabilities for your function (logging, monitoring).

AWS Lambda Overview

Lambda functions: Stateless, trigger-based code execution

1

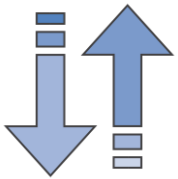
No Infrastructure to manage



Focus on business logic, not infrastructure. You upload code; AWS Lambda handles everything else.

2

High performance at any scale; Cost-effective and efficient



Pay only for what you use: Lambda automatically matches capacity to your request rate. Purchase compute in 100ms increments.

3

Bring Your Own Code

Run code in a choice of standard languages. Use threads, processes, files, and shell scripts normally.

How Lambda works

Invoked in response to events

- **Changes in data**
- **Changes in state**


S3 event notifications

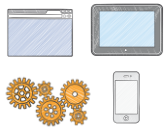

DynamoDB Streams


Amazon Kinesis events


SNS events


CloudTrail events


Cognito events




Custom events

Author in familiar language using any libraries; Execute only when needed, automatic scale

“Lambda functions”

Access any service, including your own

Any custom

Any AWS

Such as…

SNS

DynamoDB

Lambda

Amazon Redshift

Amazon Kinesis

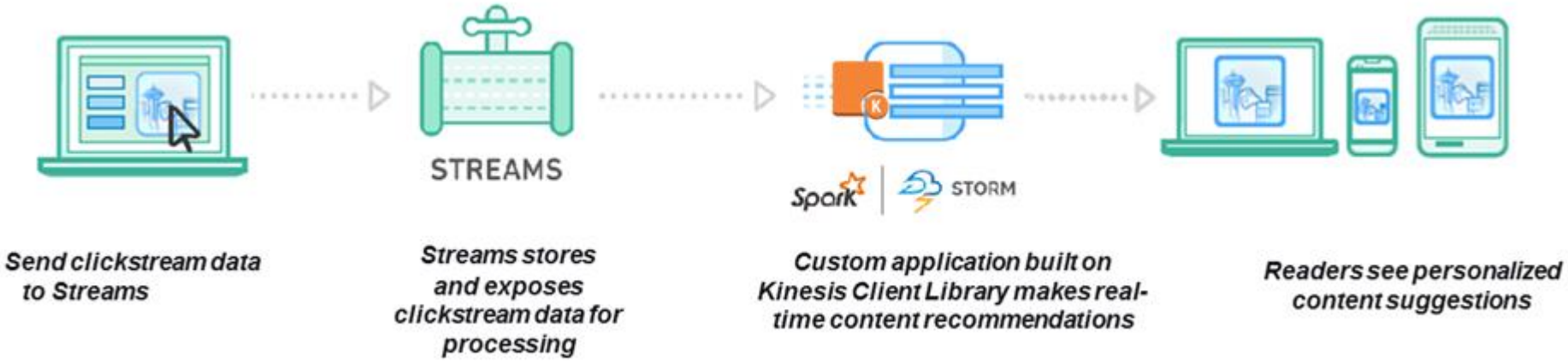
S3

Amazon Kinesis Streams

Fully managed service for real-time processing of high-volume, streaming data

- Processes data in real-time
- Highly scalable parallel processing
- Open source libraries for sending data to and reading data from a stream
- Synchronously replicates your data across 3 facilities
- Integrated with many AWS & third party technologies
- Supports SSL and automatic encryption of data once it is uploaded

Amazon Kinesis Streams

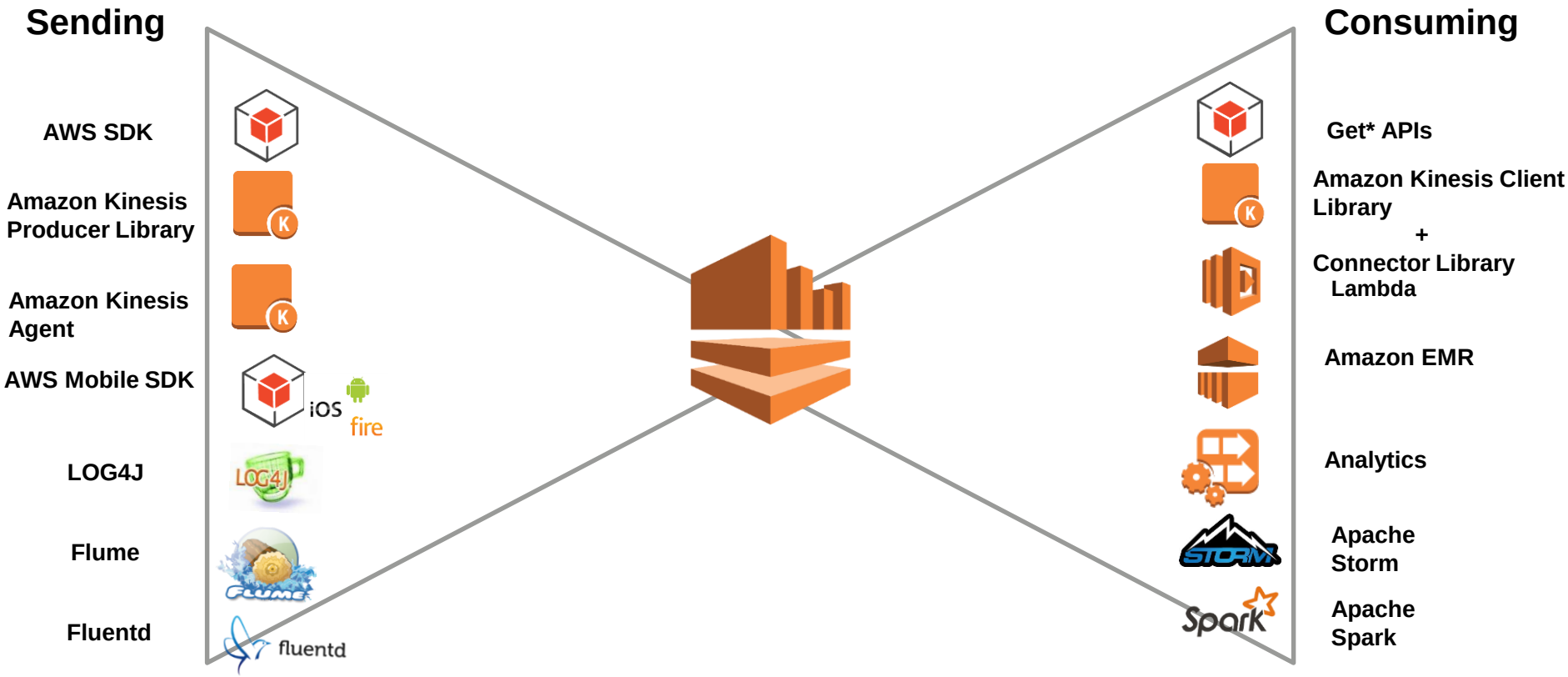


Easy administration: Simply create a new stream and set the desired level of capacity with shards. Scale to match your data throughput rate and volume.

Build real-time applications: Perform continual processing on streaming big data using Amazon Kinesis Client Library (KCL), Apache Spark/Storm, AWS Lambda, and more.

Low cost: Cost-efficient for workloads of any scale.

Sending & reading data from Amazon Kinesis Streams



Elasticsearch

elasticsearch.

A powerful, real-time, distributed, open-source search and analytics engine:

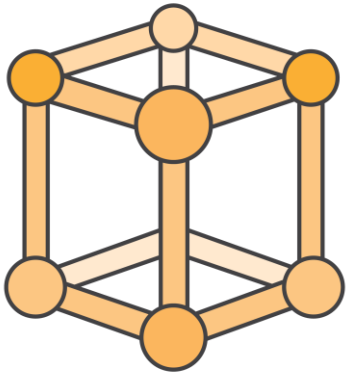
- Built on top of Apache Lucene
- Schema-free
- Developer-friendly RESTful API

Amazon Elasticsearch Service

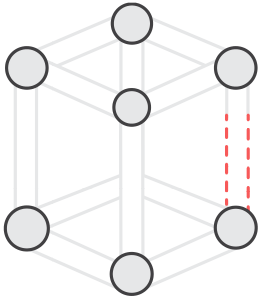
Managed service that makes it easy to set up, operate, and scale Elasticsearch clusters in the cloud

- Built-in Kibana and Logstash plugin
- Modify clusters with no downtime
- Integrated with many AWS services like CloudWatch Logs, Lambda, DynamoDB, etc.
- Supports the ES API and is a drop in replacement for your existing Elasticsearch clusters
- Only pay for what you use

Ease of operation



Easy cluster creation and configuration management

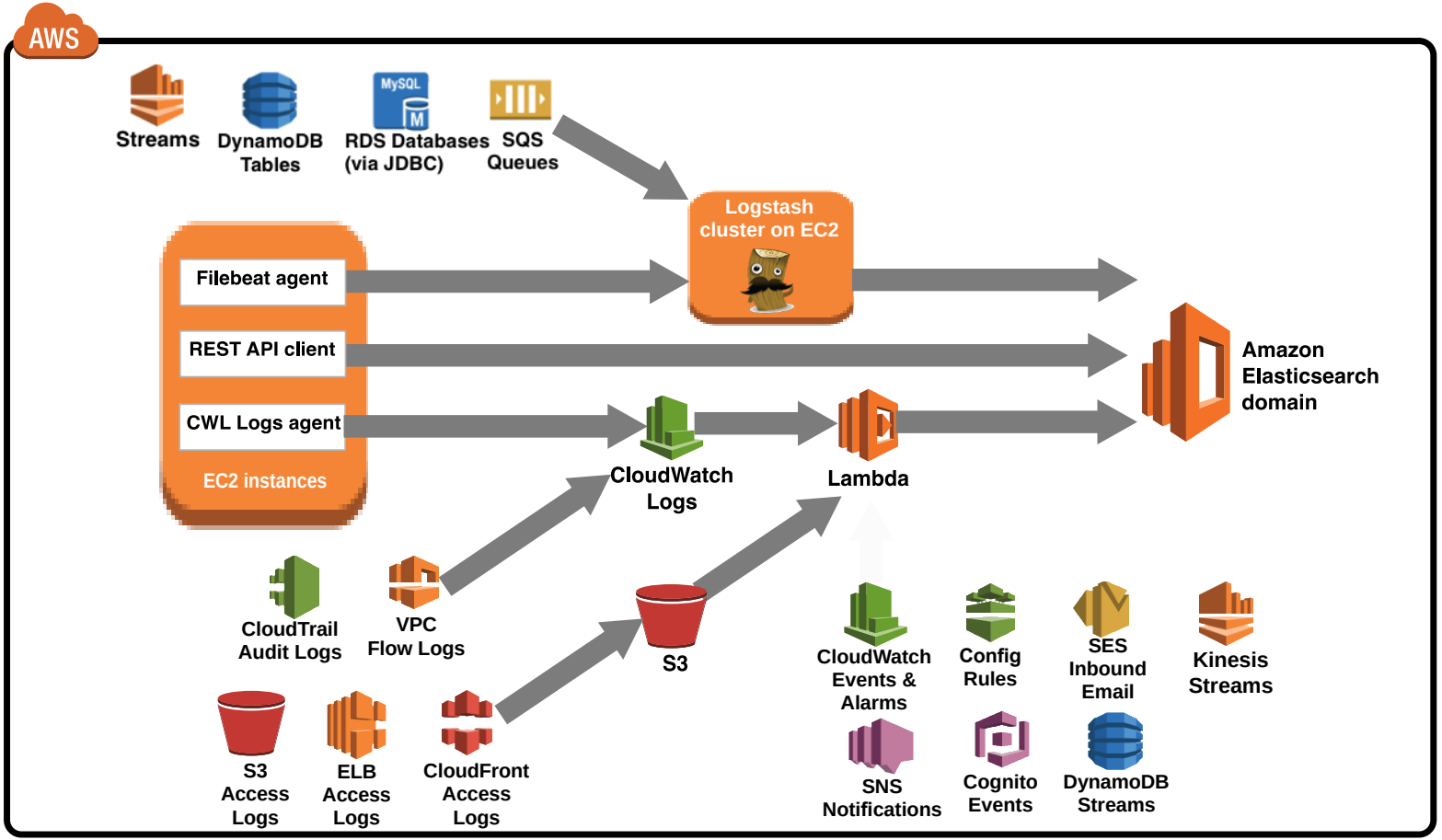


Self-healing clusters

High availability

Data durability

Integration with the AWS ecosystem

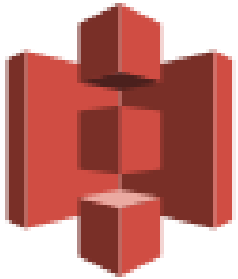


Arrow direction indicates general direction of data flow

Amazon Simple Storage Service (S3)

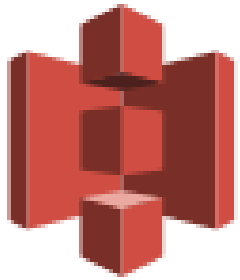
Secure, durable, low cost, highly-scalable object storage

- Easy to scale
- Designed for 99.999999999% durability and up to 99.99% availability of objects over a given year
- Cost-effective, pay only for the storage you actually use
- Lifecycle policies can move objects to long term storage or lower cost S3 Standard-IA
- Integrated with many AWS & third party technologies
- Supports SSL and automatic encryption of data once it is uploaded



Amazon Simple Storage Service (S3) for Big Data

- Scalable
  - Virtually unlimited number of objects
  - Very high bandwidth – no aggregate throughput limit



- Cost-Effective
  - No need to run compute clusters for storage (unlike HDFS)
  - Can run transient Hadoop clusters & Amazon EC2 Spot Instances
  - Tiered storage (Standard, IA, Amazon Glacier) via life-cycle policy

- Flexible Access
  - Direct access by big data frameworks (Spark, Hive, Presto)
  - Shared access: Multiple (Spark, Hive, Presto) clusters can use the same data

Amazon Relational Database Service

Fully managed relational database service

- Simple and fast to deploy
- Fully managed = low admin
- Fast, predictable performance
- Easy to scale
- Cost-effective
- Open source engines: MySQL, PostgreSQL, MariaDB
- Commercial engines: Oracle, SQLServer
- MySQL compatible engine: Aurora

Amazon RDS

RDS PostgreSQL

Amazon RDS makes it easy to set up, operate, and scale PostgreSQL deployments in the cloud. With Amazon RDS, you can deploy scalable PostgreSQL databases in minutes with cost-efficient and resizable hardware capacity.

Key Features

Read Replicas (Same region and cross region)
High Availability with Multi-AZ
VPC and private subnet groups
Geospatial capabilities
Syntactically similar to Oracle

Amazon RDS

Amazon DynamoDB

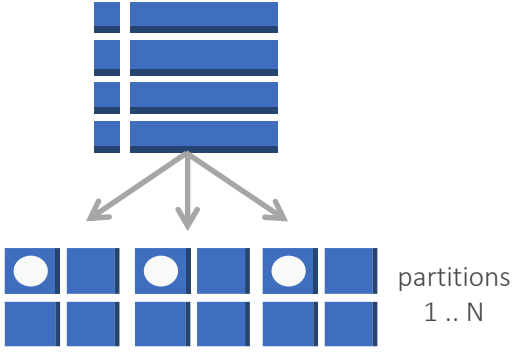
Non-Relational Managed NoSQL Database Service

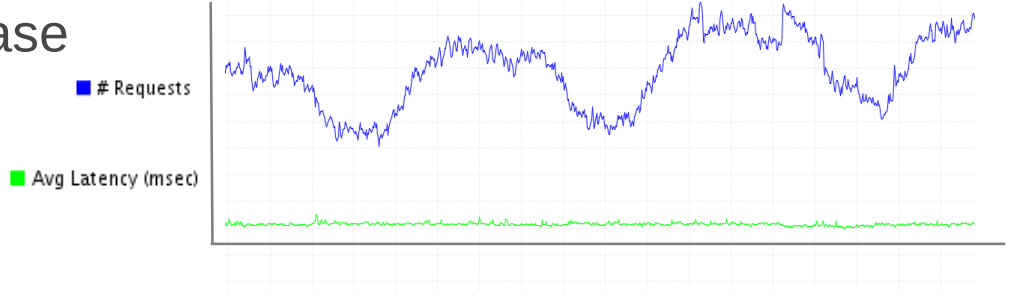
- Schemaless data model
- Consistent, low-latency performance (single digit ms)
- Predictable provisioned throughput
- Seamless scalability
- Practically no storage limits
- High durability and availability (replication between 3 facilities)
- Easy administration – we scale for you!
- Low cost
- Cost modelling on throughput and size

DynamoDB

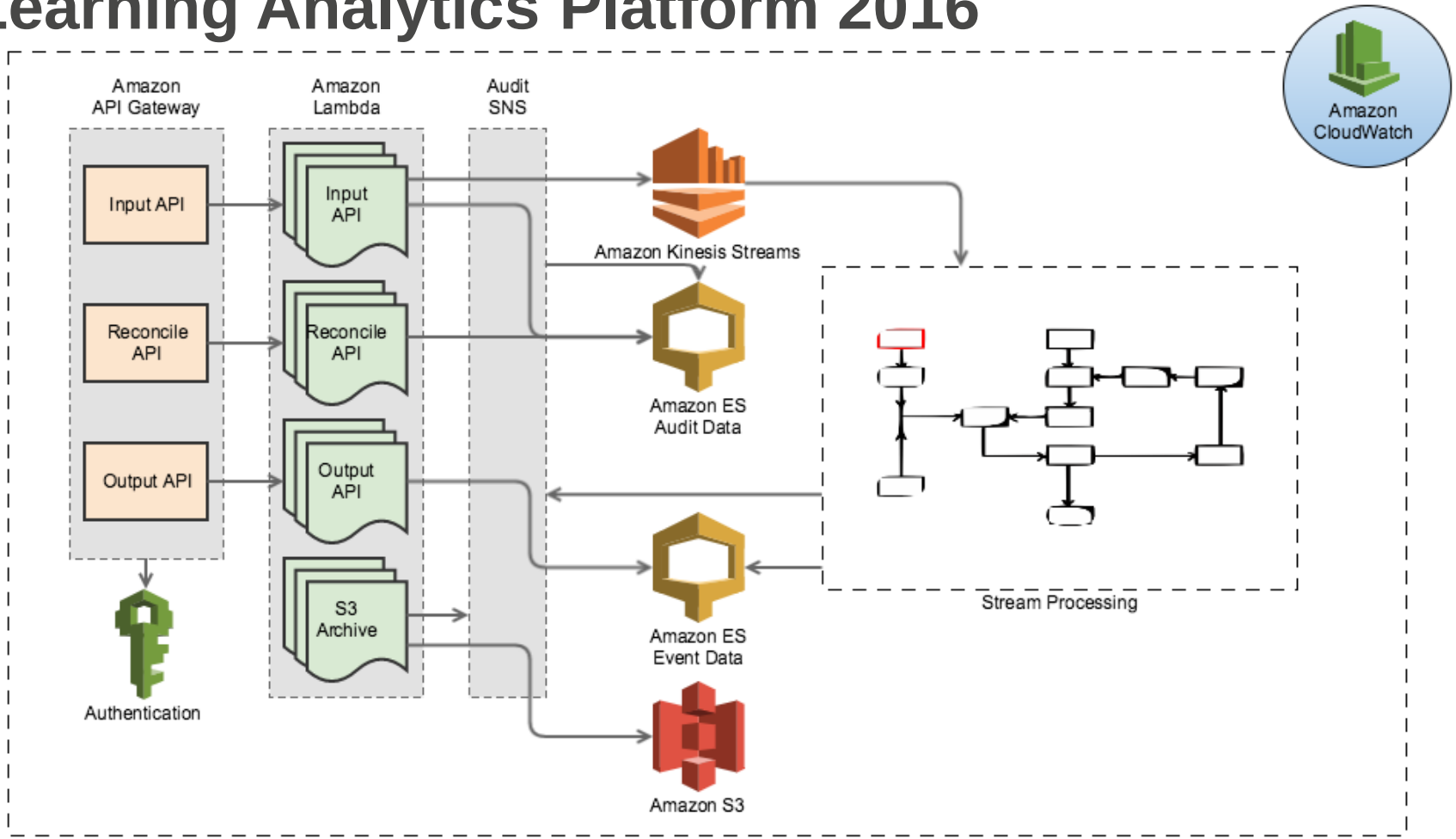
Amazon DynamoDB Scalability

- Virtually no limit in throughput (reads/writes per second)
- Virtually no limit in storage
- DynamoDB automatically partitions data
- Auto-partitioning occurs when:
  - Data set growth
  - Provisioned capacity increase

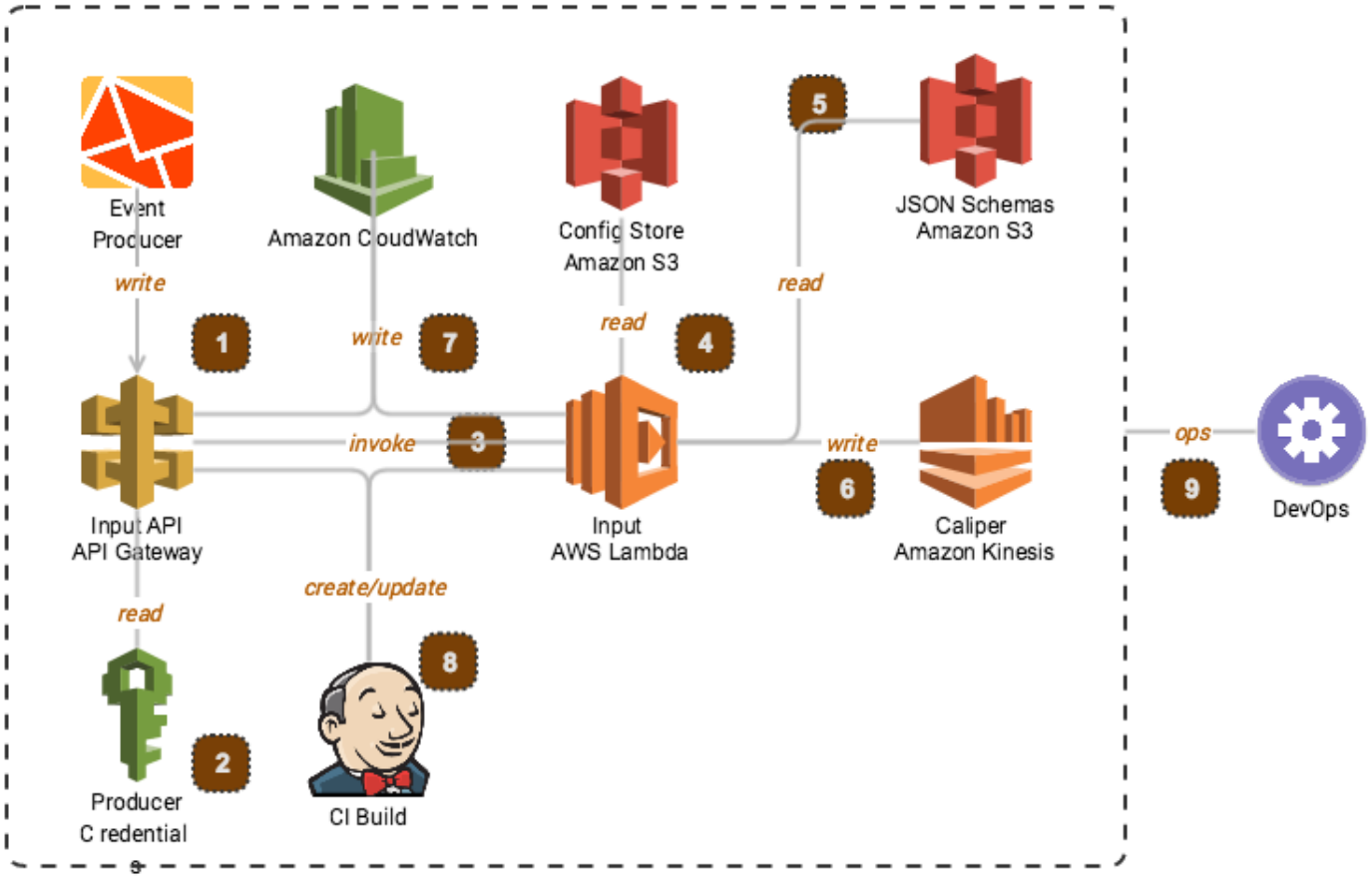




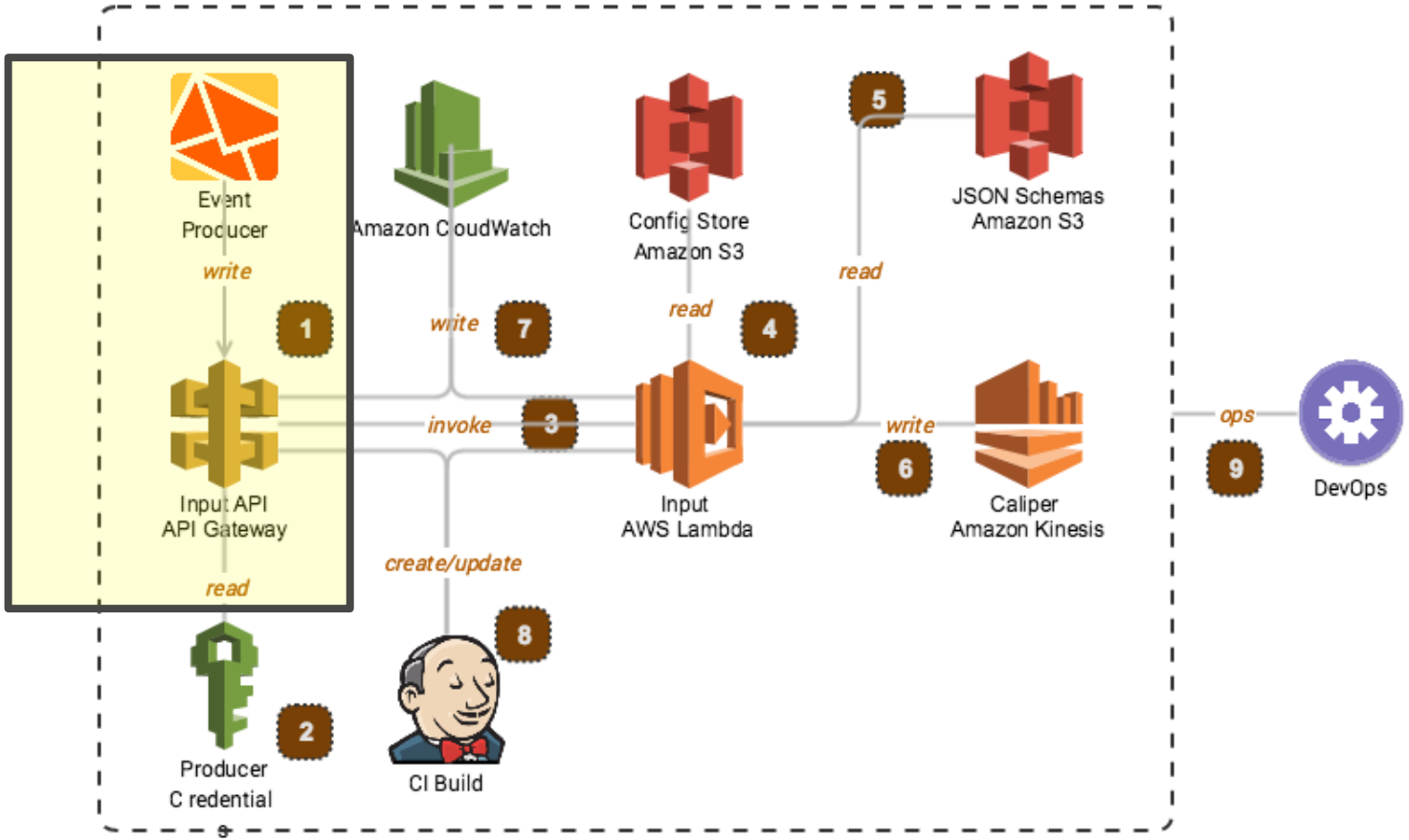
Learning Analytics Platform 2016



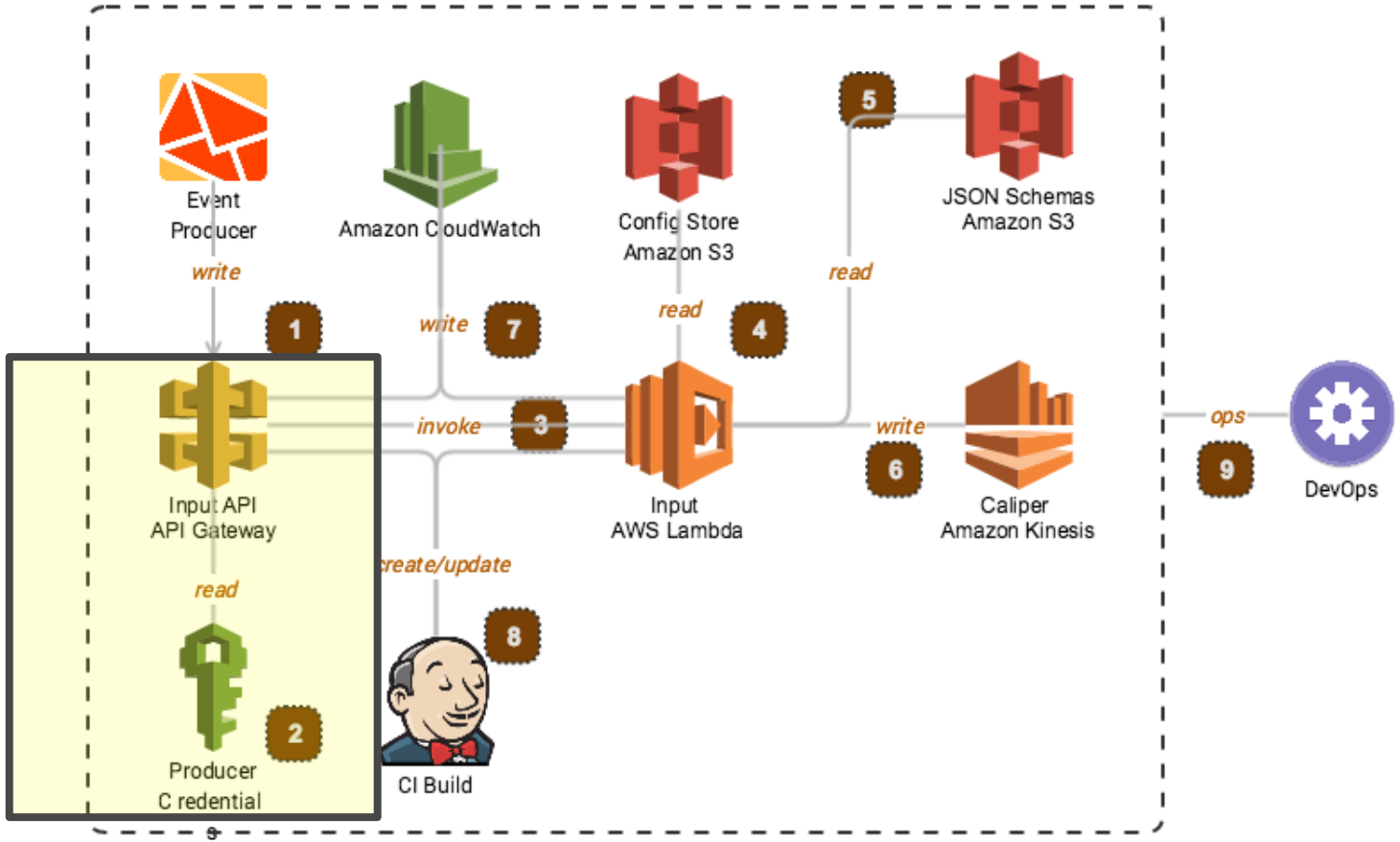
Input API


Event
Producer
write
1
Input API
API Gateway
read
2
Producer
C redential
Amazon CloudWatch
write
7
invoke
3
create/update
8
CI Build
Config Store
Amazon S3
read
4
Input
AWS Lambda
5
JSON Schemas
Amazon S3
read
write
6
Caliper
Amazon Kinesis
ops
9
DevOps

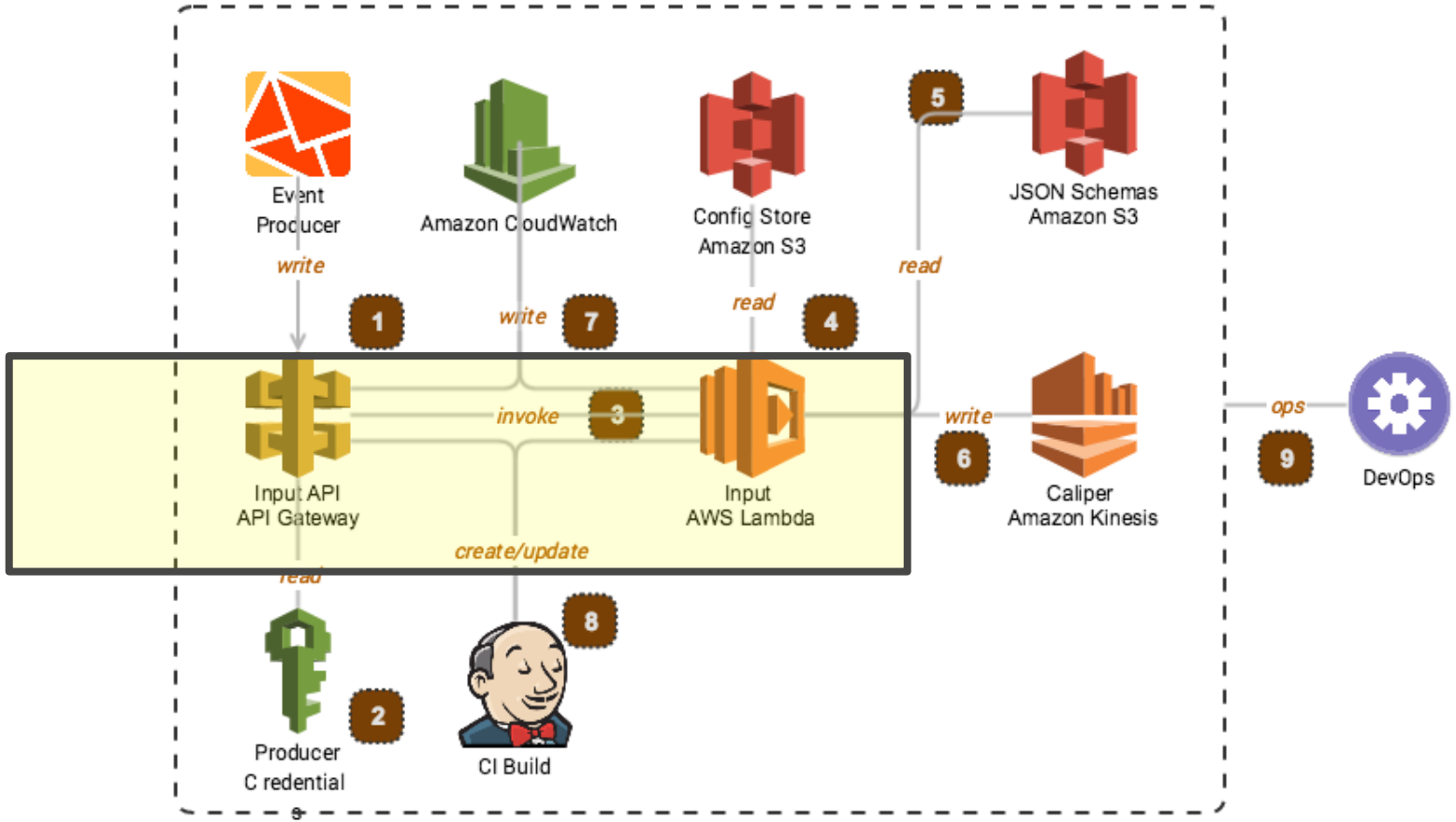
Input API


Event
Producer
write
1
Input API
API Gateway
read
2
Producer
C redential
Amazon CloudWatch
write
7
invoke
3
create/update
8
CI Build
Config Store
Amazon S3
read
4
Input
AWS Lambda
5
JSON Schemas
Amazon S3
read
write
6
Caliper
Amazon Kinesis
ops
9
DevOps

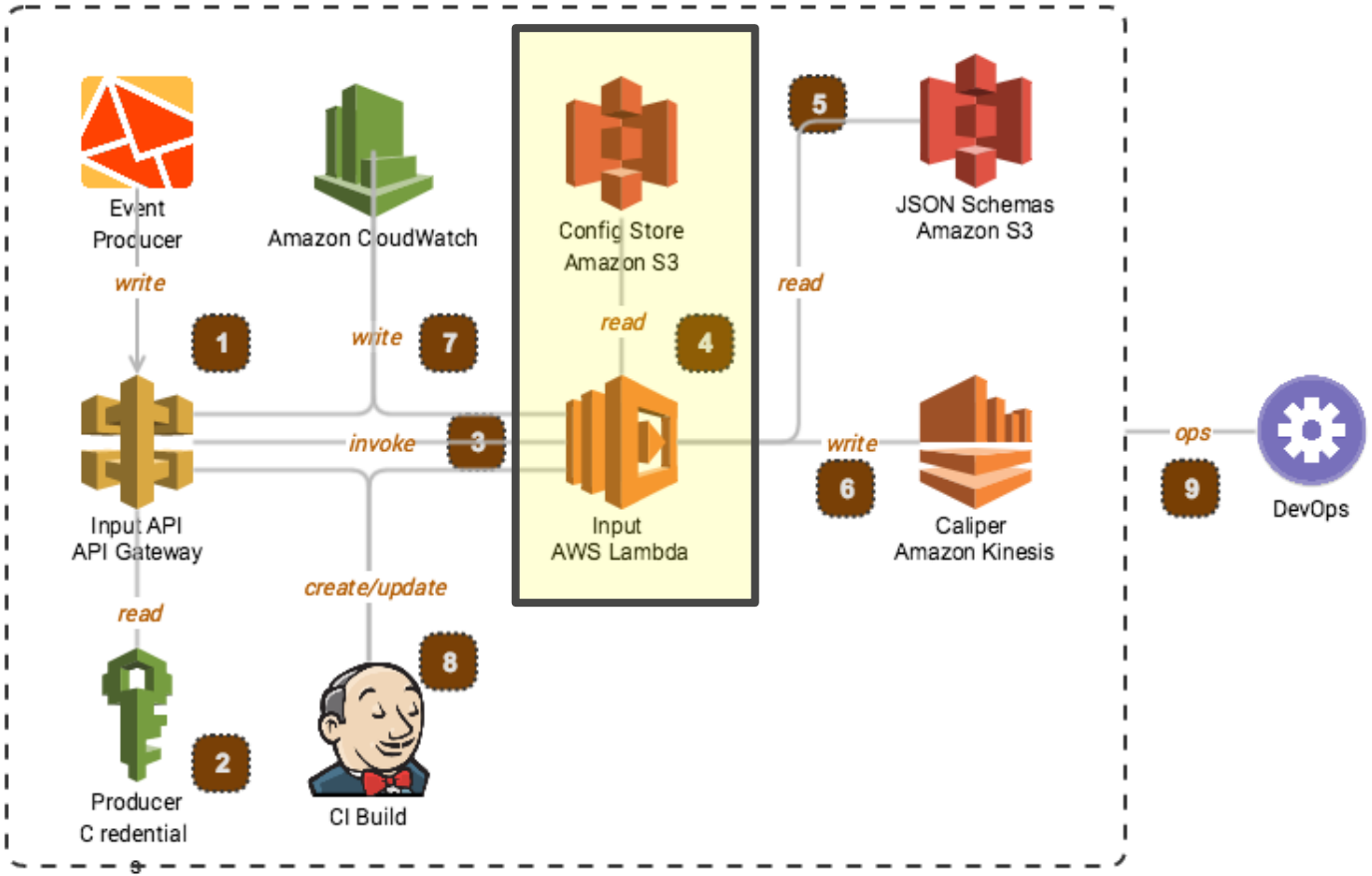
Input API


Event
Producer
write
Amazon CloudWatch
Config Store
Amazon S3
5
JSON Schemas
Amazon S3
read
1
write
7
read
4
Input API
API Gateway
invoke
3
Input
AWS Lambda
write
6
Caliper
Amazon Kinesis
ops
9
DevOps
read
create/update
8
2
Producer
C redential
CI Build

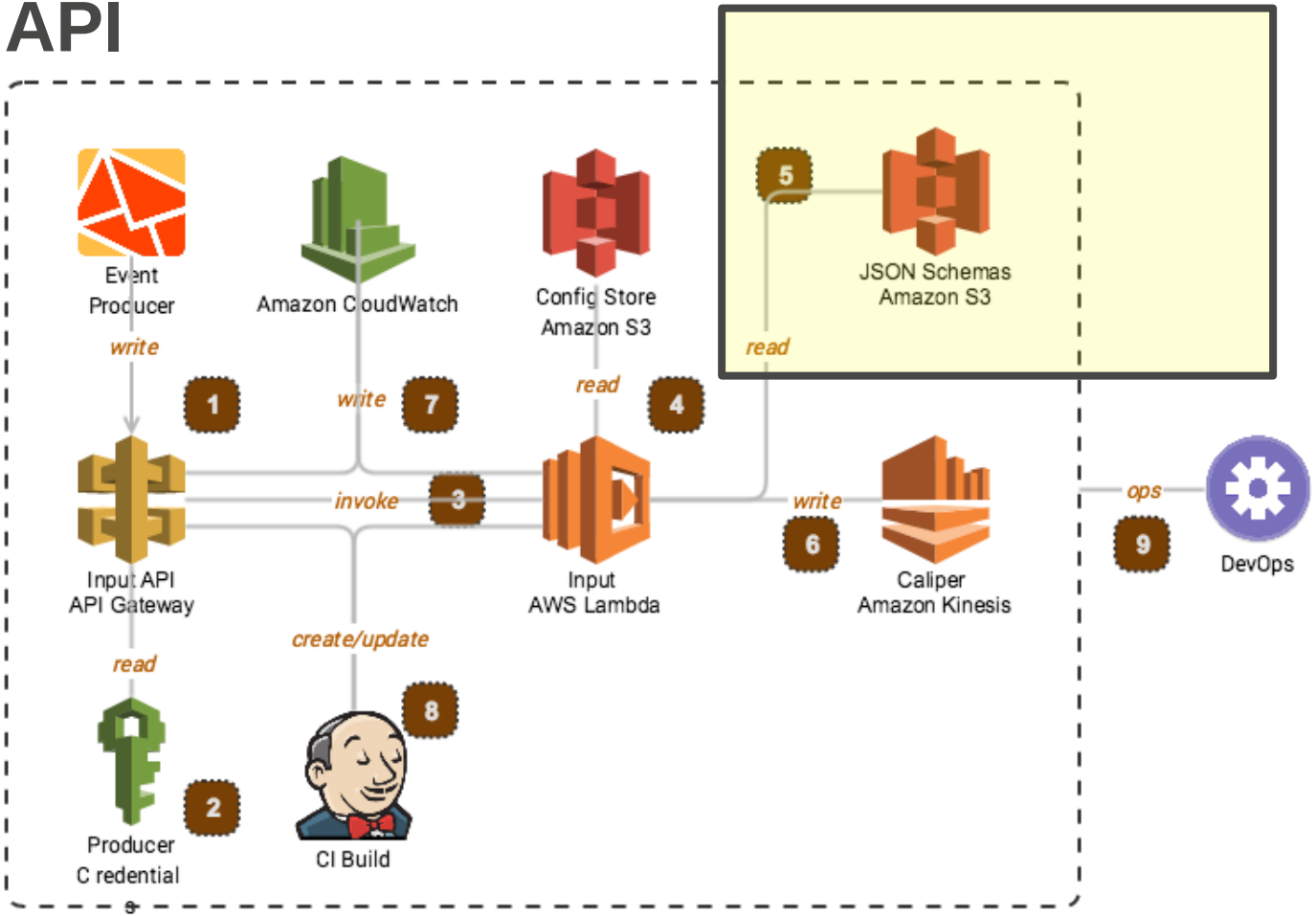
Input API


Event
Producer
write
Amazon CloudWatch
Config Store
Amazon S3
read
5
JSON Schemas
Amazon S3
read
1
write
7
read
4
Input API
API Gateway
invoke
3
Input
AWS Lambda
write
6
Caliper
Amazon Kinesis
ops
9
DevOps
create/update
read
8
2
Producer
C redential
CI Build

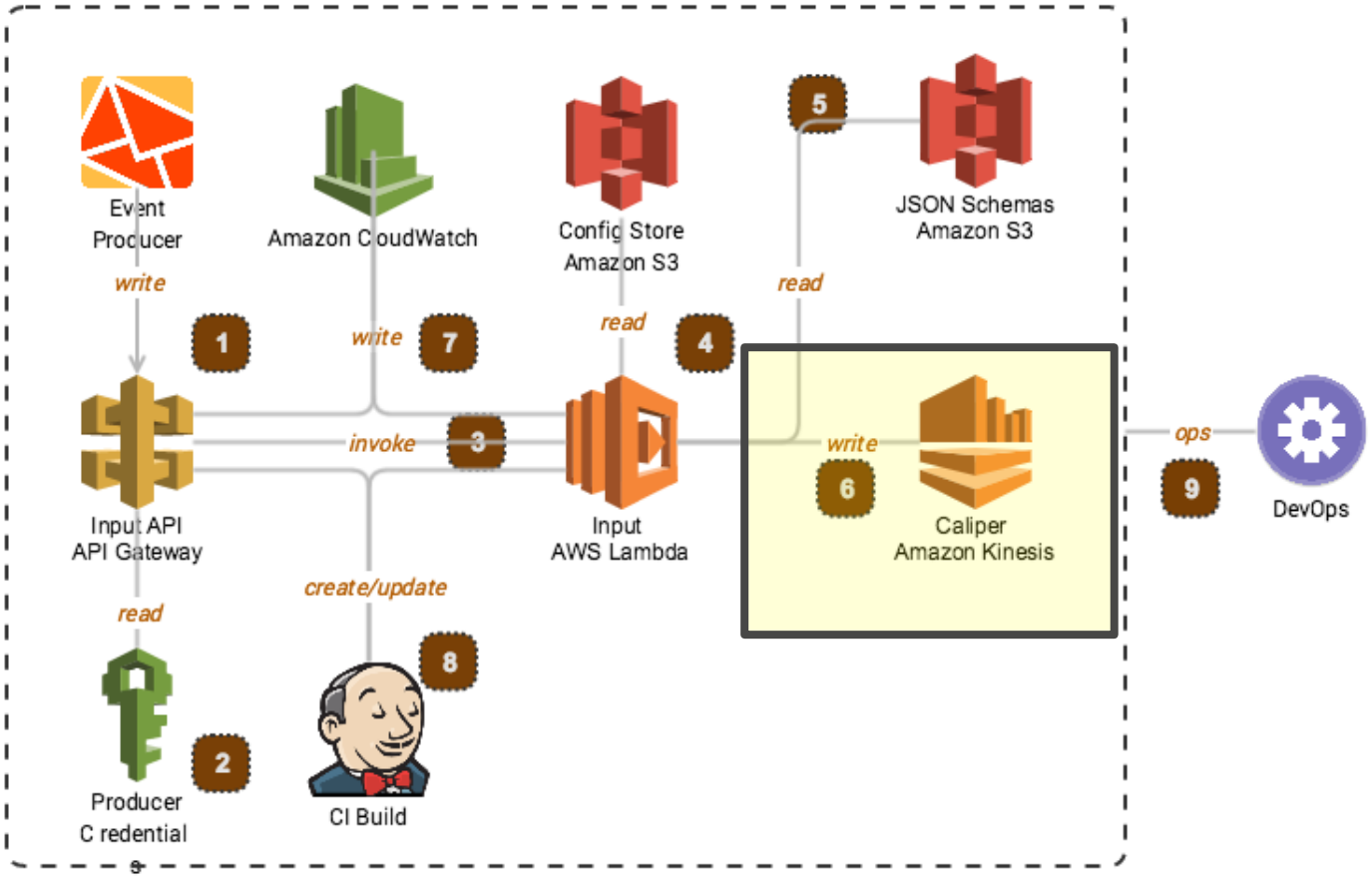
Input API


Event
Producer
write
1
Input API
API Gateway
read
2
Producer
C redential
Amazon CloudWatch
write
7
invoke
3
create/update
8
CI Build
Config Store
Amazon S3
read
4
Input
AWS Lambda
5
read
JSON Schemas
Amazon S3
write
6
Caliper
Amazon Kinesis
ops
9
DevOps

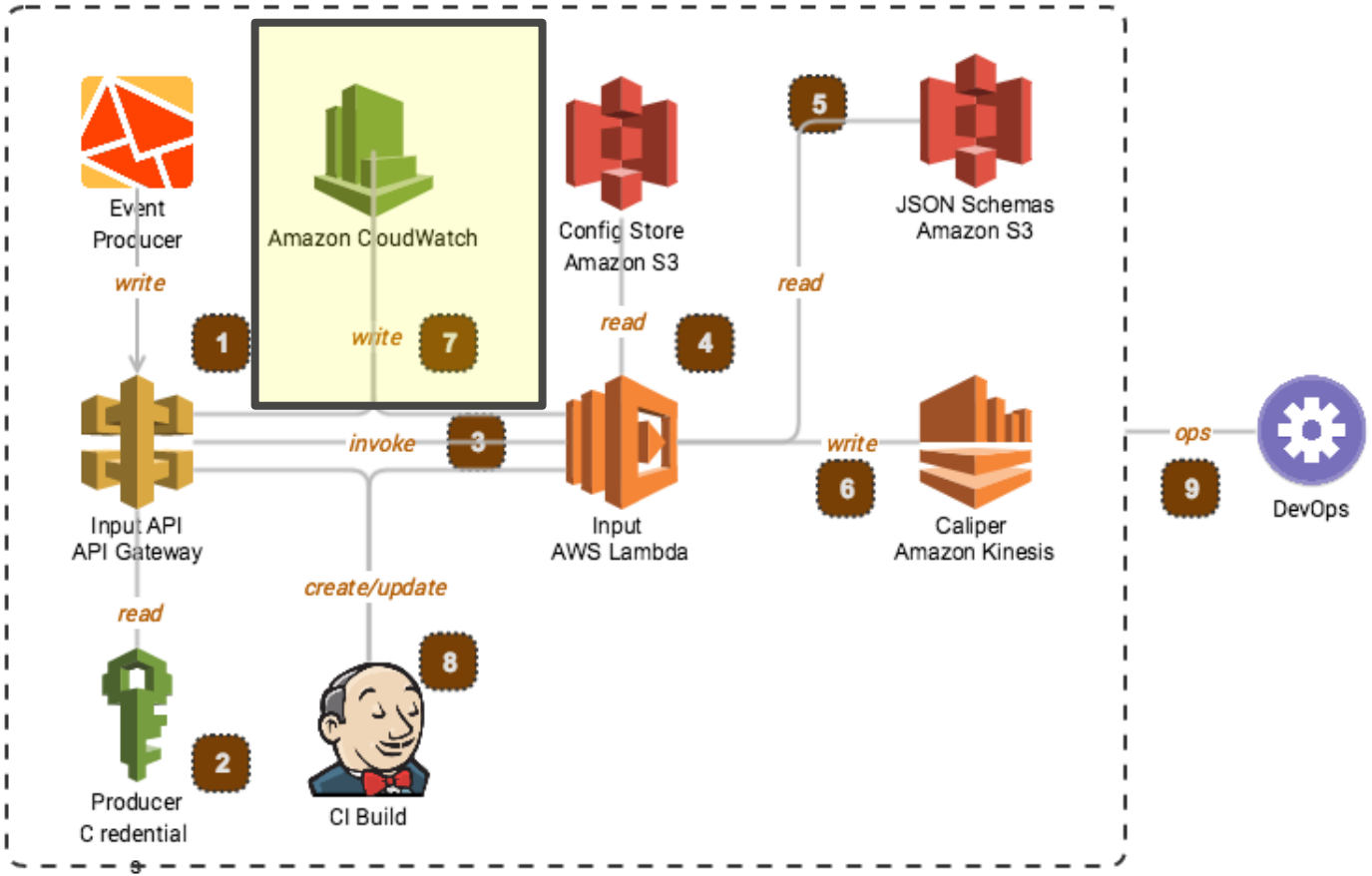
Input API


Event
Producer
write
Amazon CloudWatch
Config Store
Amazon S3
5
JSON Schemas
Amazon S3
read
1
write
7
read
4
Input API
API Gateway
invoke
3
Input
AWS Lambda
write
6
Caliper
Amazon Kinesis
ops
9
DevOps
read
create/update
8
2
Producer
C redential
CI Build

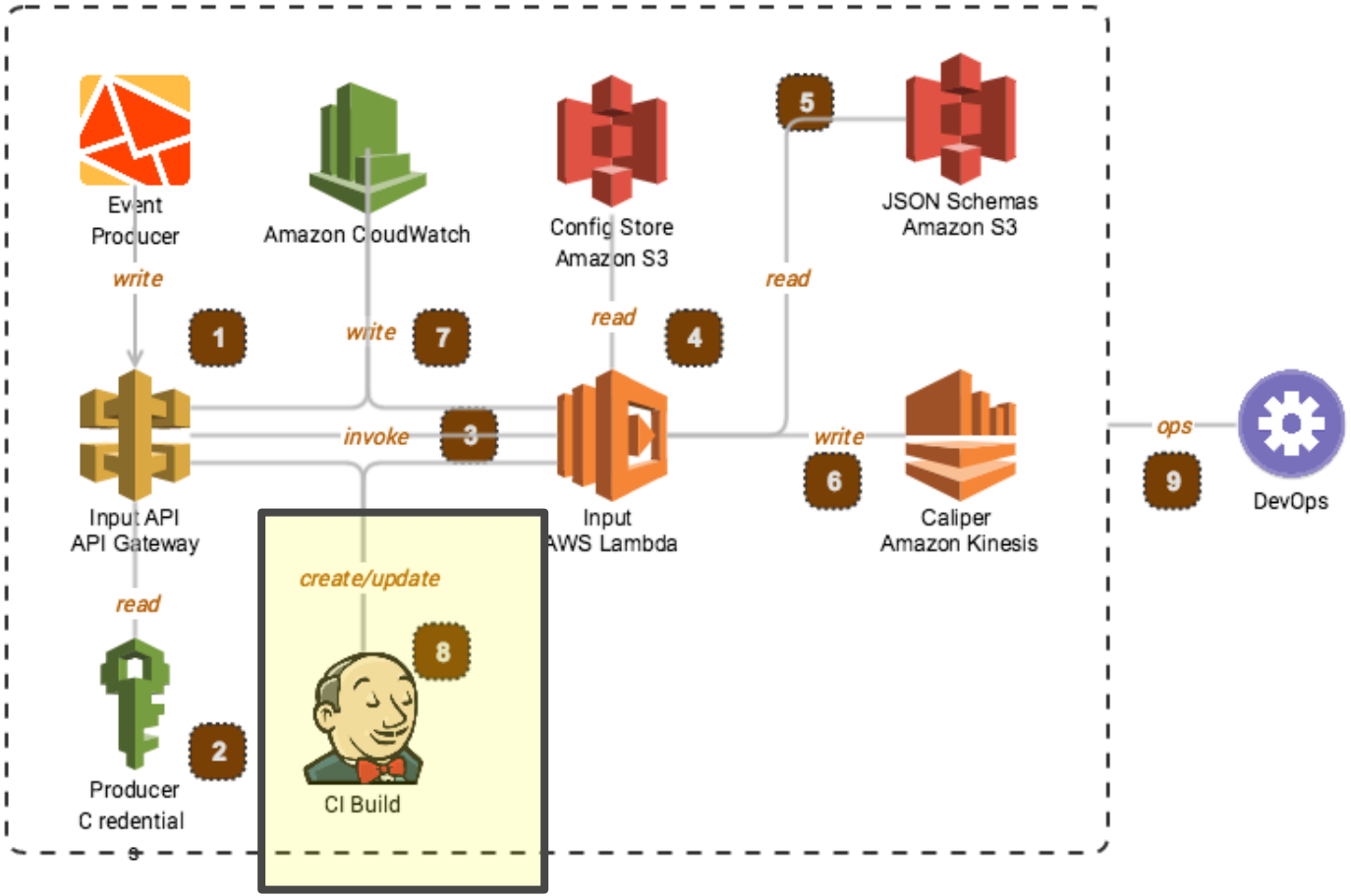
Input API


Event
Producer
write
Amazon CloudWatch
write
7
Config Store
Amazon S3
read
4
5
JSON Schemas
Amazon S3
read
1
invoke
3
Input API
API Gateway
read
2
Producer
C redential
create/update
8
CI Build
Input
AWS Lambda
write
6
Caliper
Amazon Kinesis
ops
9
DevOps

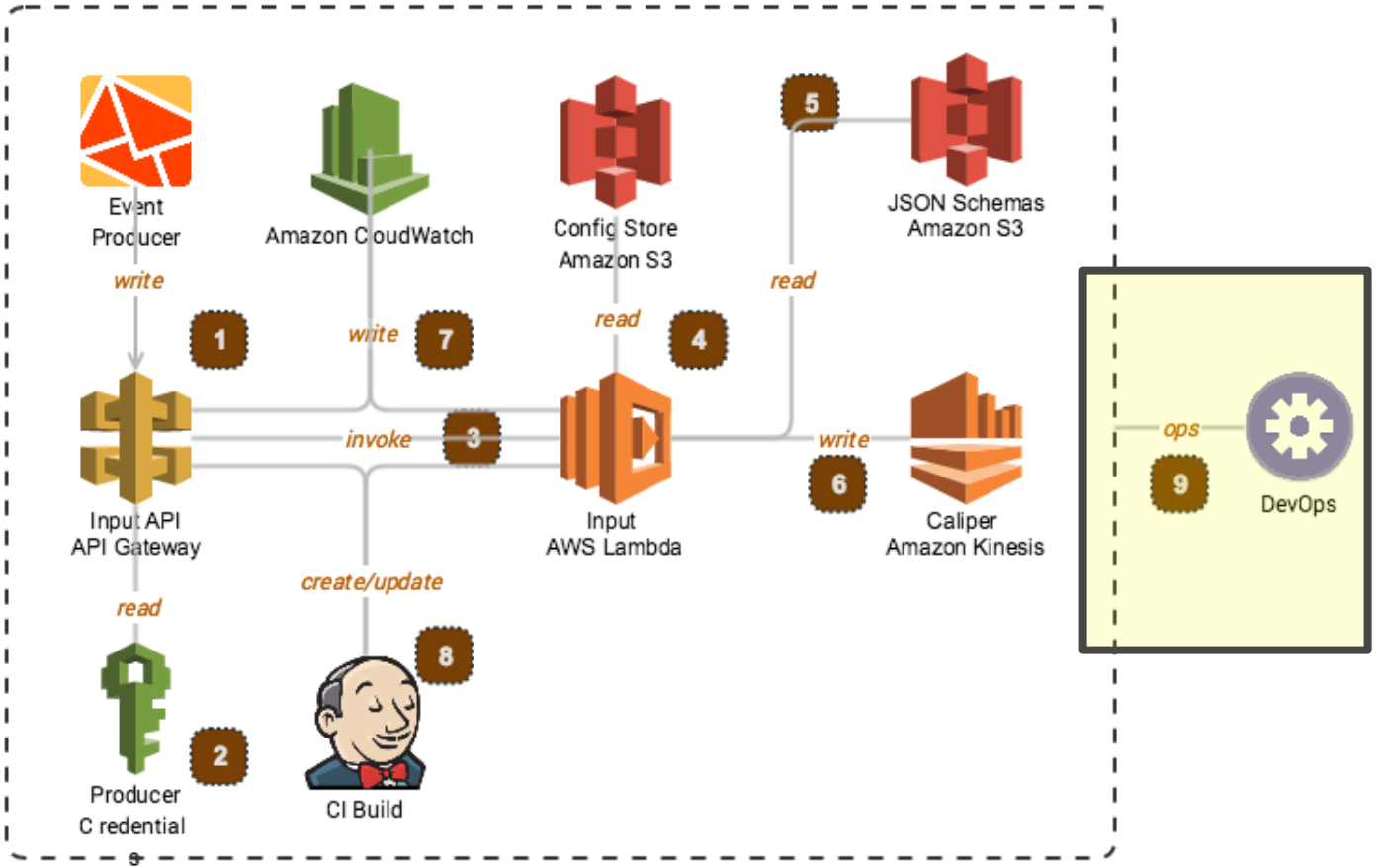
Input API


Event
Producer
write
1
Amazon CloudWatch
write
7
Config Store
Amazon S3
read
4
5
JSON Schemas
Amazon S3
read
invoke
3
Input API
API Gateway
Input
AWS Lambda
write
6
Caliper
Amazon Kinesis
ops
9
DevOps
read
2
Producer
C redential
create/update
8
CI Build

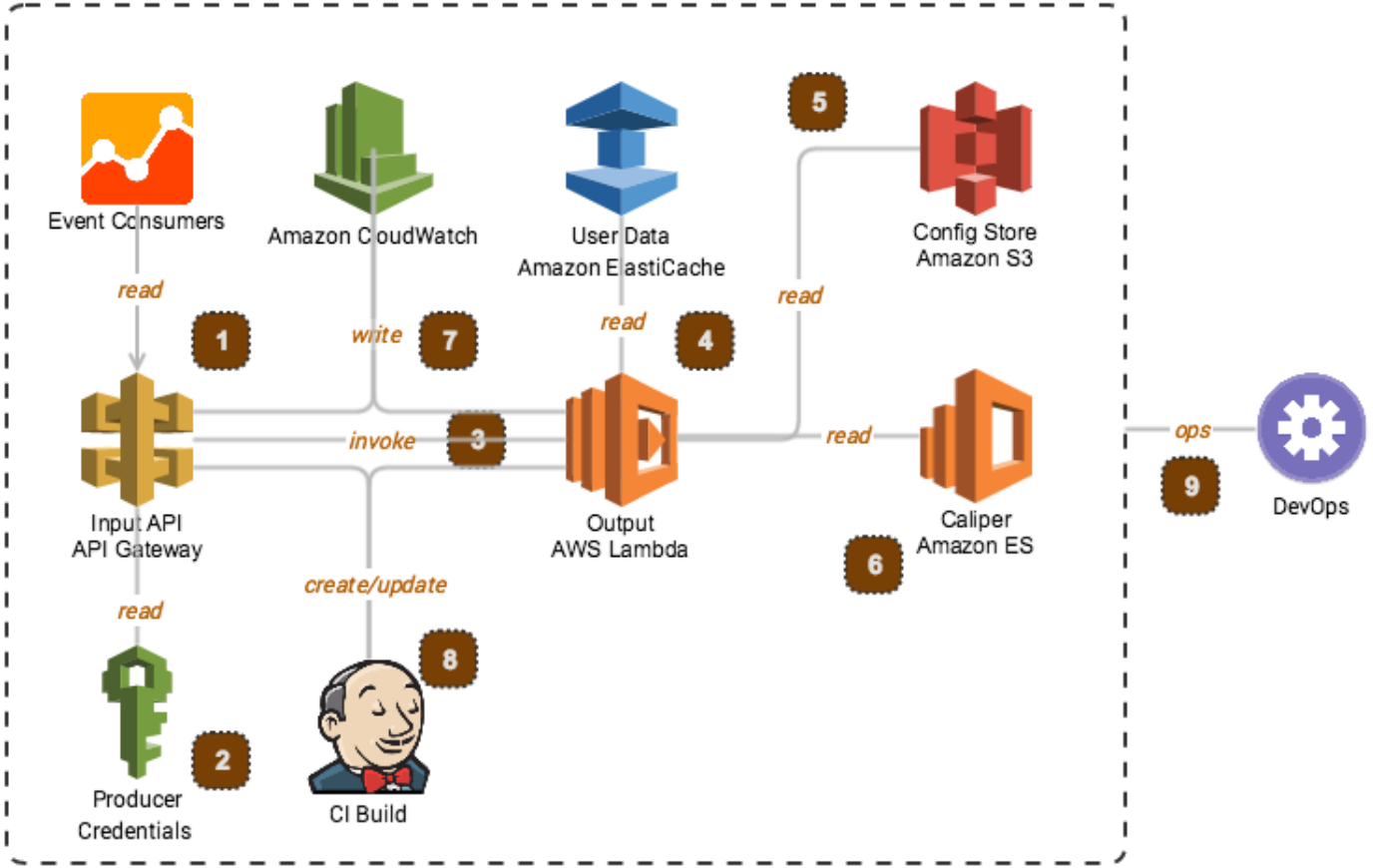
Input API


Event
Producer
write
Amazon CloudWatch
Config Store
Amazon S3
5
JSON Schemas
Amazon S3
read
1
write
7
read
4
Input API
API Gateway
invoke
3
Input
AWS Lambda
write
6
Caliper
Amazon Kinesis
ops
9
DevOps
read
create/update
8
2
Producer
C redential
CI Build

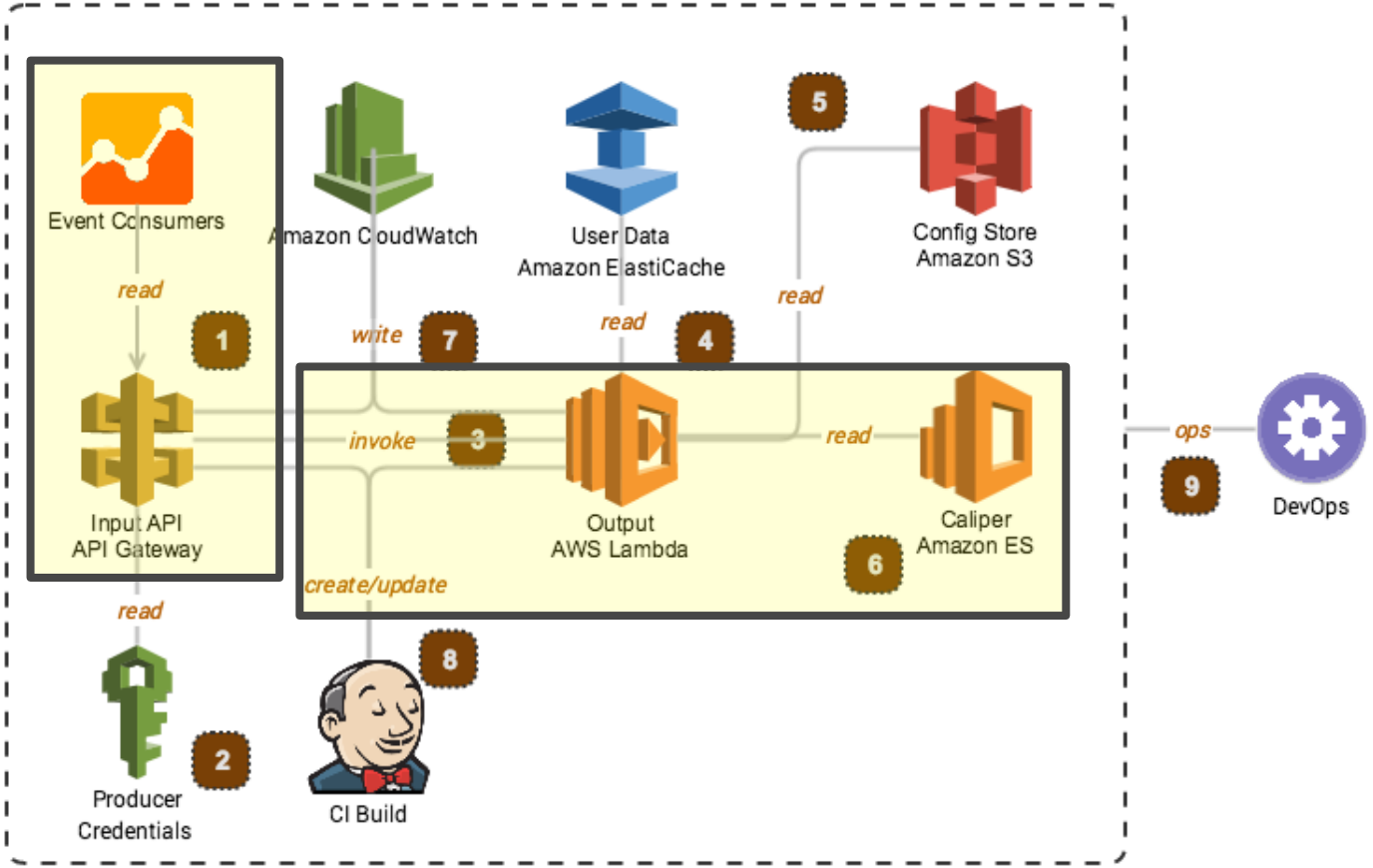
Input API


Event
Producer
write
Amazon CloudWatch
write
Config Store
Amazon S3
read
JSON Schemas
Amazon S3
read
1
7
4
5
invoke
3
Input API
API Gateway
Input
AWS Lambda
write
6
Caliper
Amazon Kinesis
ops
9
DevOps
read
create/update
8
2
Producer
C redential
CI Build

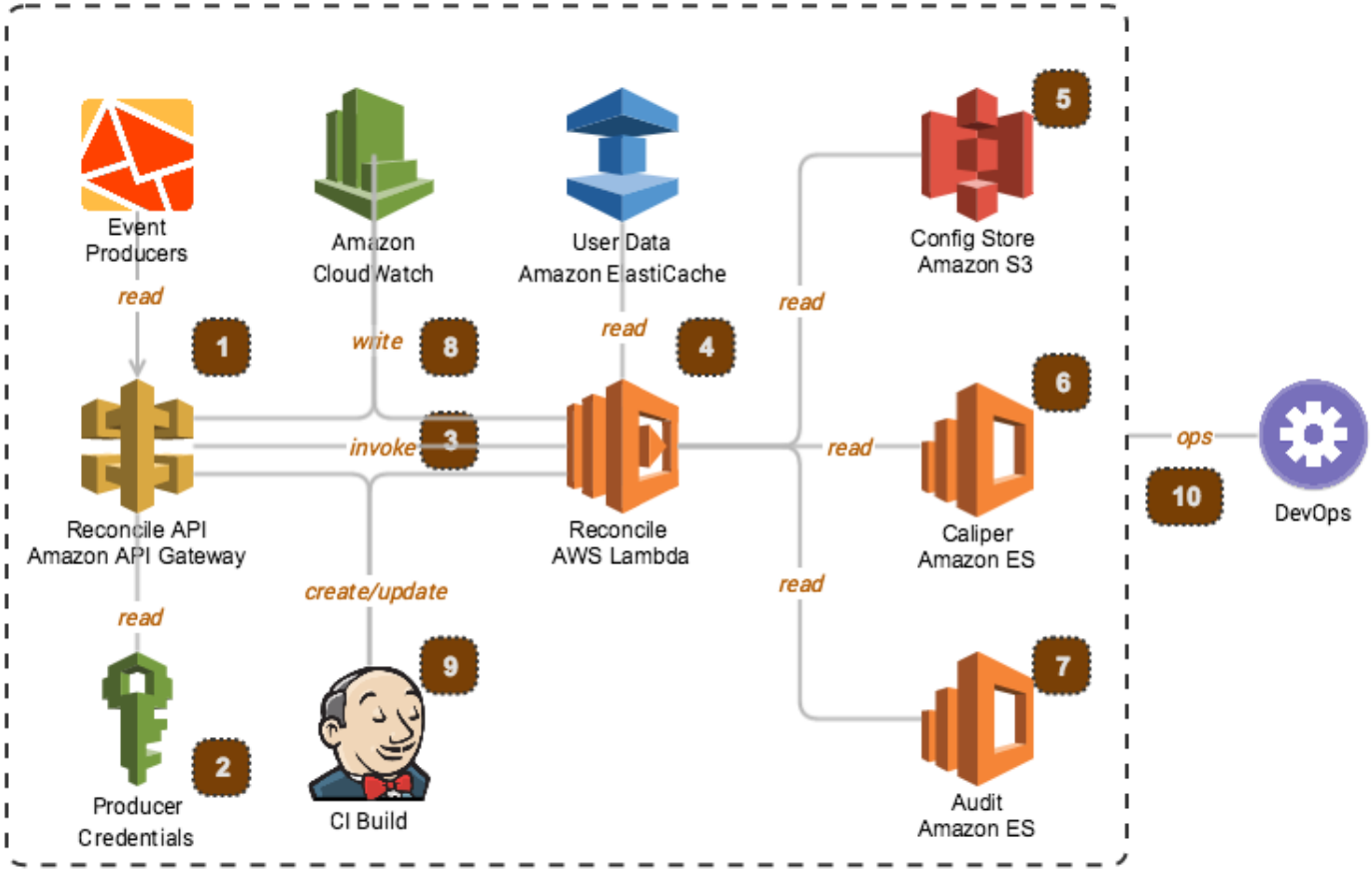
Output API


Event Consumers
Amazon CloudWatch
User Data
Amazon ElastiCache
Config Store
Amazon S3
read
write
read
read
1
7
4
5
Input API
API Gateway
invoke
3
Output
AWS Lambda
read
Caliper
Amazon ES
6
ops
9
DevOps
read
create/update
8
2
Producer
Credentials
CI Build

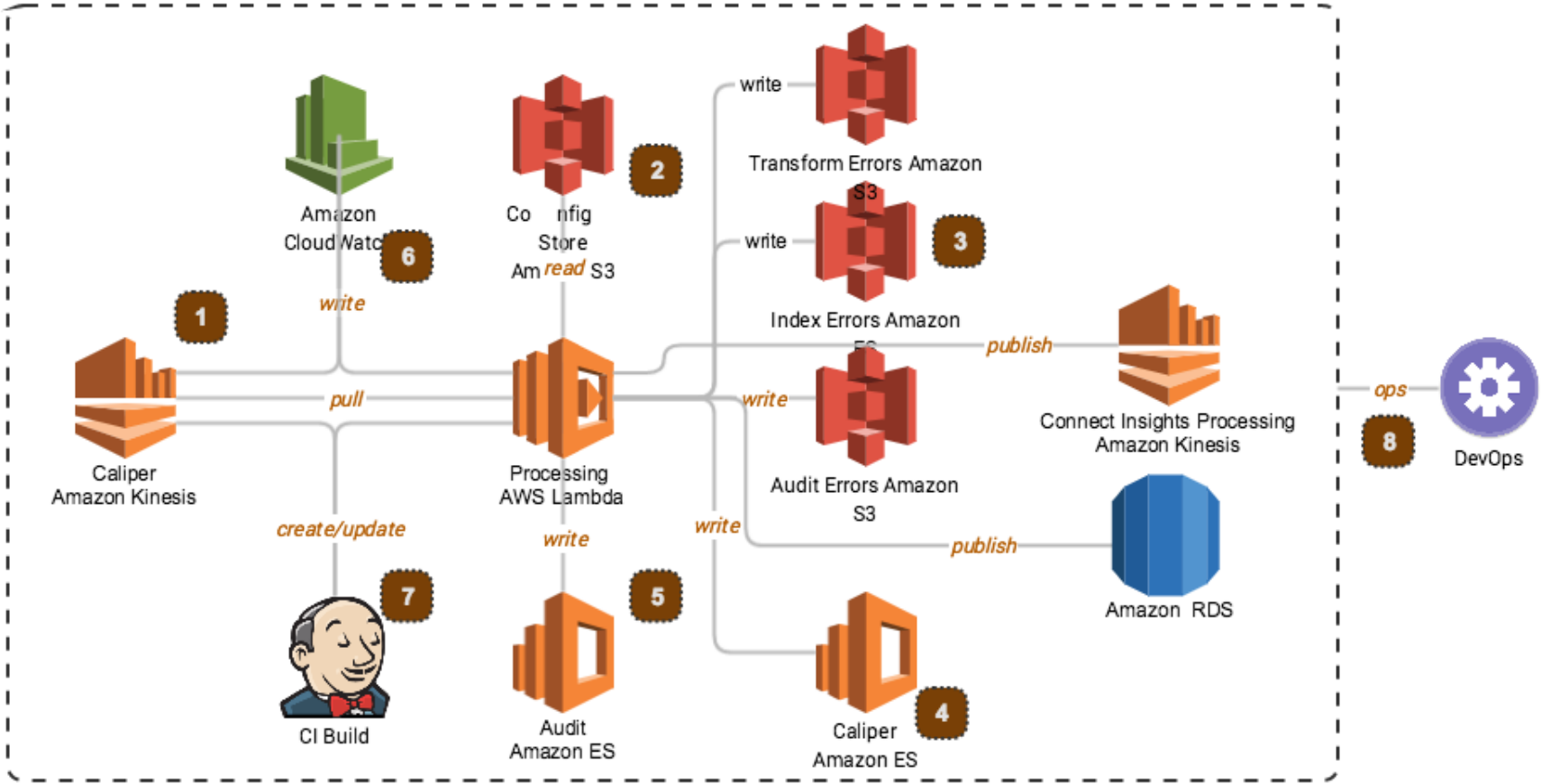
Output API


Event Consumers
read
1
Input API
API Gateway
read
2
Producer
Credentials
Amazon CloudWatch
write
7
invoke
3
create/update
8
CI Build
User Data
Amazon ElastiCache
read
4
Output
AWS Lambda
5
Config Store
Amazon S3
read
read
6
Caliper
Amazon ES
ops
9
DevOps

Reconcile API



Stream Processing



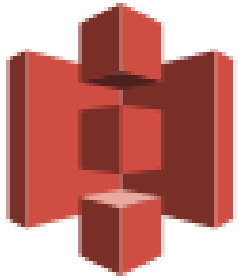
Architecture Tradeoffs (Amazon API Gateway)

- Compare with implementing code for HTTP/HTTPS
- Pro
  - API Gateway is highly integrated service out of the box
  - Automatically scales
  - Handles thousands of concurrent calls
  - Traffic management, authorization and access control, monitoring, and API version management
- Con
  - Does not currently support GZIP compression. Workaround is to set up CloudFront server and enable compression

Architecture Tradeoffs (AWS Lambda)

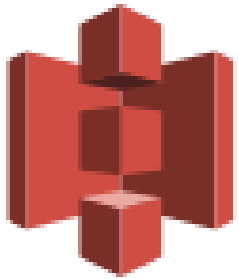
- Compare with provisioning and managing EC2 instances
- Pro:
  - No servers and instances to manage,
  - Built-in automatic scaling
  - Fixed cost model
  - Don’t need a team of 7 DevOps resources to manage
- Con:
  - Limited experience with Lambda
  - Limited to CloudWatch and 6 MB data
  - Debugging logs is time-consuming

Architectural Tradeoffs (Amazon Kinesis Streams)



- Compare to Kafka and Zookeeper
- Pro
  - Able to process high-volume, streaming data
  - 15 million records at peak load and growing
  - Maintained and don’t have to predict storage and volume
  - Managed service with cross Availability Zone replication
- Con
  - May lose records depending on max configuration setting (24 hours to 7 days)

Architectural Tradeoffs (Amazon S3)



- Compare with Rackspace CloudFiles and OpenStack Swift
- Pro
  - Secure, durable, highly-scalable cloud archive
  - Managed service
  - Easy to use, inexpensive, multiple means of security content, backup of content, high availability
- Con
  - SSL mismatch errors if you want to use own domain name as domain name is (bucketname).(region).amazonaws.com

Architectural Tradeoffs (Amazon ES)

- Compare with Lucene and Elasticsearch company
- Pro
  - Managed service that provides out of box integrations with Amazon Kinesis and S3
  - Use as data lake for learning events
- Con
  - Release behind Elasticsearch so may not have needed feature

Architectural Tradeoffs (Amazon DynamoDB)

- Compare to MongoDB
- Pro
  - Experience
  - Low cost
  - Fast and flexible NoSQL data store
  - Fully managed
- Con
  - Limit – 400 KB row size, 1 MB queries
  - Size is multiples of 4 KB for reads

Architectural Tradeoffs (RDS PostgreSQL)

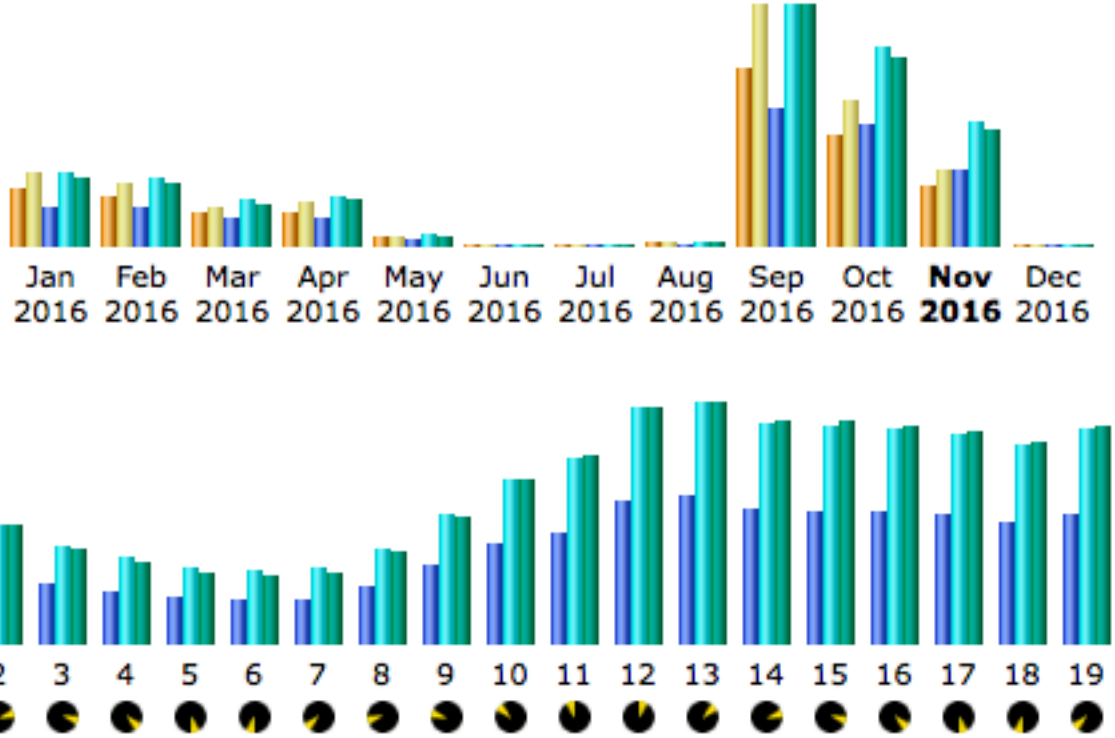
- Compare to RDS Aurora and Amazon Redshift
- Pro
  - Scales to 6 TB – within our immediate needs
  - More concurrent connections than Amazon Redshift
  - Has full analytical engine
- Con
  - Data volumes in future – address using archiving and other strategies to reduce volume

AWS Estimated Cost Savings of 1 Billion Events

AWS Service	Original Cost	Estimated Cost	Estimated Savings
Amazon API Gateway	$4,319	$4,319	$0
AWS Lambda	$0	$5,000	-$5,000
Amazon Elasticsearch Service	$11,000	$11,000	$0
Amazon Kinesis	$9,232	$9,232	$0
Amazon EC2	$410,000	$100,000	$310,000
Total	$434,551	$129,551	$305,000

- Amazon Lambda biggest cost saver
- Pay for what we use
- Auto-scales
- Additional capabilities (logging, monitoring)
- Fewer DevOps resources
- Gains in Agility

Connect Insights Usage Trends



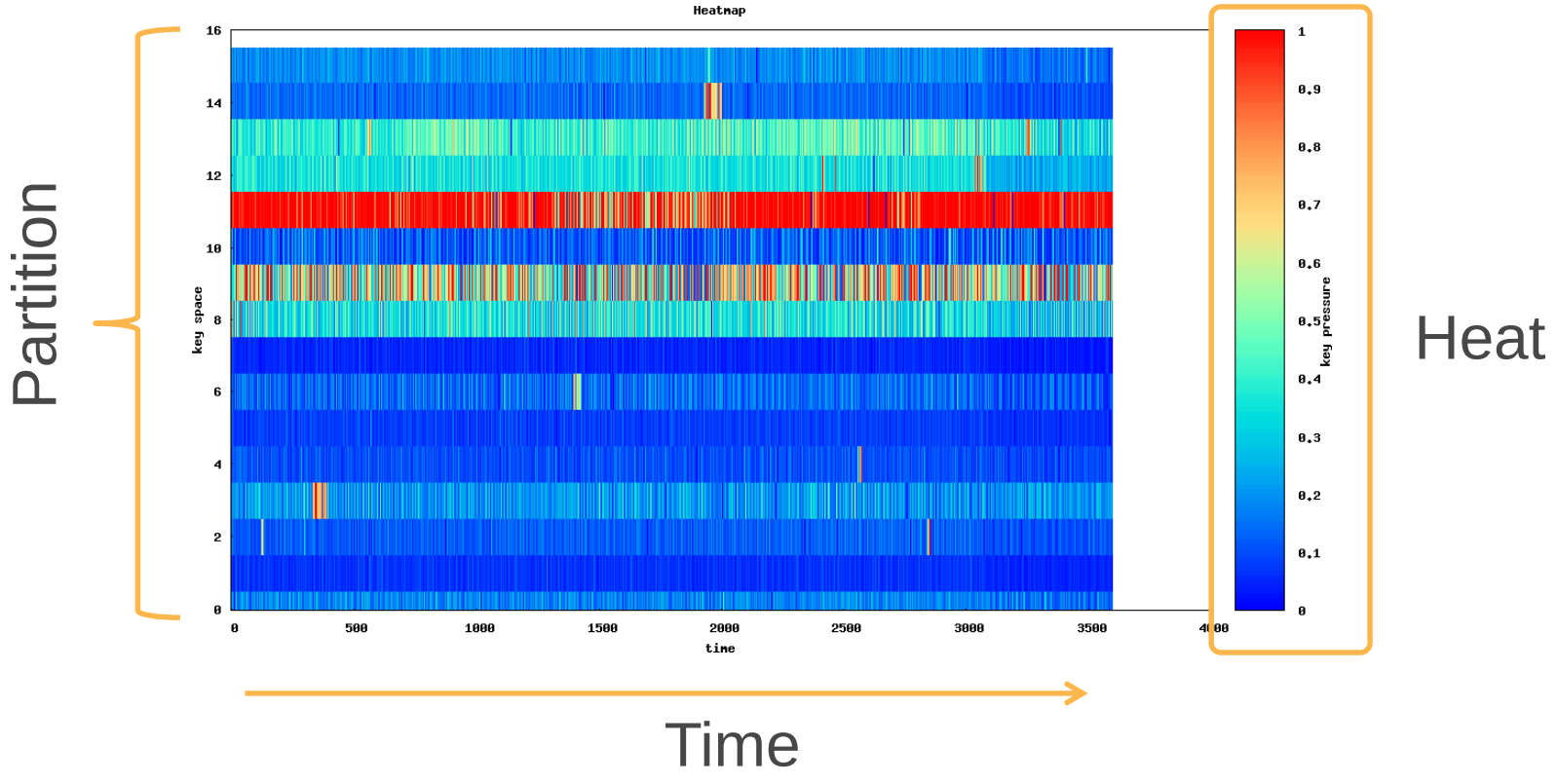
Challenges

- Lost events
- Elasticsearch performance
- Events to fail indexing in Elasticsearch
- Be aware of Elasticsearch limits
- Amazon Kinesis stream retention from 24 hours to 7 days
- DynamoDB hot spots

Challenges

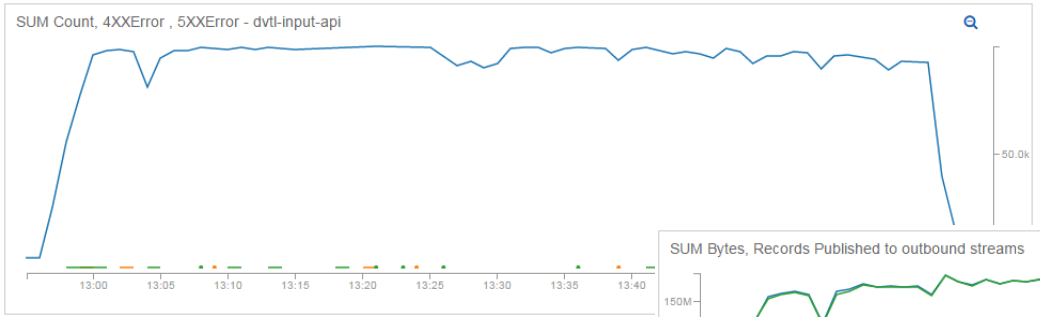
- Lost events
- Elasticsearch performance
- Events to fail indexing in Elasticsearch
- Be aware of Elasticsearch limits
- Amazon Kinesis stream retention from 24 hours to 7 days
- DynamoDB hot spots

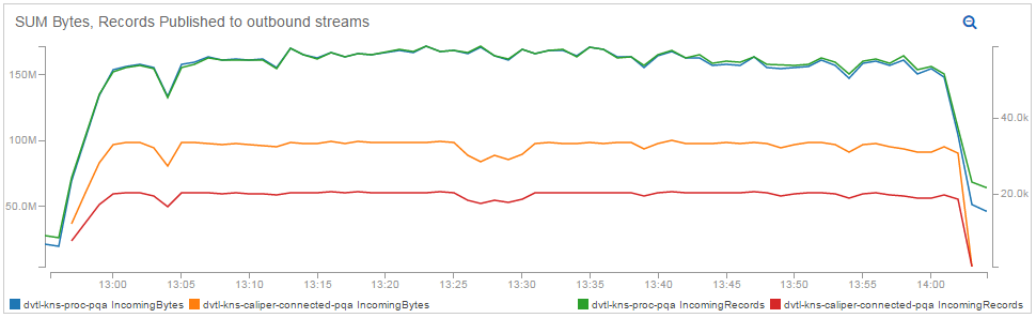
Example Heat Map



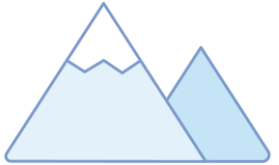
How We Built Confidence

- Built confidence thru robust testing strategy – performance, failover, functional, and business acceptance testing





How We Built Confidence (cont.)



- Caliper events sent to input API, validate persistence to Elasticsearch and S3
- Tools used for testing
- Reconcile API – playback
- Monitoring of components
- Service Level Agreements established

Lessons Learned

- General
  - Serverless framework
  - CloudWatch and Sumologic
  - Automated tests – about 80% coverage
  - Custom dashboards
- Amazon API Gateway
  - SigV4

Lessons Learned (cont.)

- AWS Lambda
  - Cold start
  - Great integration
  - Sensitivity to EC2 and Auto Scaling issues and outages
  - Need better debug tools
- Amazon Kinesis Streams
  - Scaling up and down shards
  - No purge functionality

Lessons Learned (cont.)

- Amazon Elasticsearch Service
  - Scripts
  - Queue capacity limits
  - Performance tuning and monitoring
- AWS Enterprise Support

Summary of Actionable Takeaways

- Think about production scale
- Estimated costs including DevOps, engineers, architects
- Take time and resources to design the right architecture – ilities – resiliency, redundancy, security, disaster recovery, reliability, maintainability
- Amazon Enterprise Support

Summary of Actionable Takeaways

- Think about production scale
- Estimated costs including DevOps, engineers, architects
- Take time and resources to design the right architecture – ilities – resiliency, redundancy, security, disaster recovery, reliability, maintainability
- Amazon Enterprise Support

Thank you!

Remember to complete your evaluations!