Evolution of a Data Streaming Architecture: Enabling the Business to Turn Data into Insight

Joseph Paulchell

Directory, Principal Software Engineer, Capital One Digital Engineering

SATURN 2016

Introduction and Welcome

About me: Joseph Paulchell

- Principal Software Engineer and Solution Architect
- Software Engineering Certificate from the SEI
- Team lead for
  - Internally developed software framework Java, Spring, and Spring XD
  - Supports over 100 agile teams

About Capital One

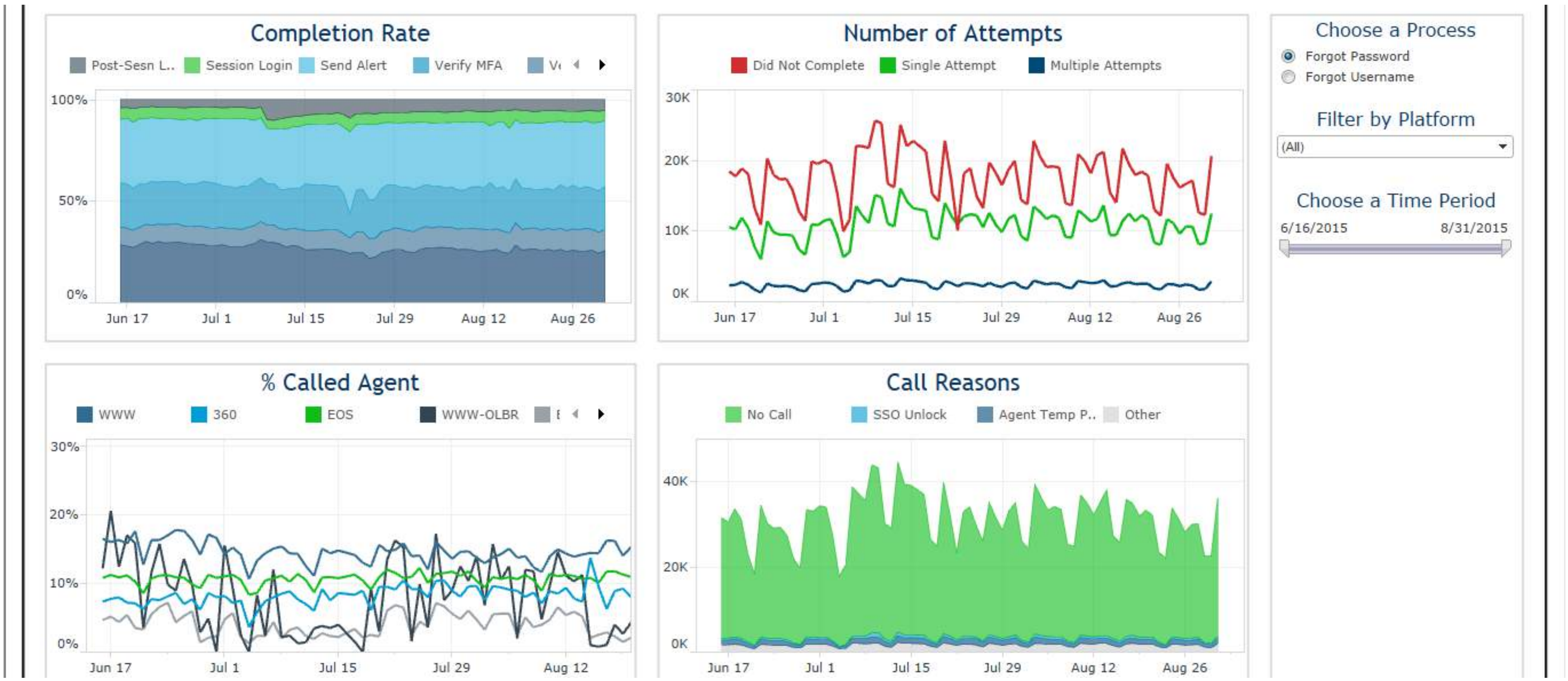
- A technology company AND a diversified financial services company
- A FORTUNE 500 Company - #124
- Over 65 million customer accounts
- Major operations in 15 U.S. cities, Canada, and U.K.

Big Data does not equal “Big Insight”… but it is a prerequisite

- **Companies collecting, storing, and leveraging huge amounts of digital data**
  - Facebook: over 500 TB collected per day[1]
  - Google: processes over 5.75 billion queries per day[2]
  - Amazon: estimated to have processed more than an exabyte of data (enough books to reach the moon)[3]
  - Capital One: growing to three petabytes of data in our data lake
- **Why?**
  - Potential to yield valuable insights: insights into behavior, preferences, and patterns of consumers that drive the trillion dollar eCommerce industry[4]
  - Insights into the behavior and quality of the systems they are using
- **Why so much?**
  - Predictive analytics – requires large amounts of data
  - Statistical model convergence –
  - Defect often occur with low probability or frequency – much data needed to observe and find root cause

1. https://gigaom.com/2012/08/22/facebook-is-collecting-your-data-500-terabytes-a-day/
2. http://aci.info/2014/07/12/the-data-explosion-in-2014-minute-by-minute-infographic/
3. http://cloudtweaks.com/2015/03/surprising-facts-and-stats-about-the-big-data-industry/
4. https://www.forrester.com/report/US+B2B+eCommerce+Forecast+2015+To+2020/-/E-RES115957

This Tableau report shows login attempts and what drives call center volumes



Choose a Process

[✓ Forgot Password] Forgot Username

Filter by Platform

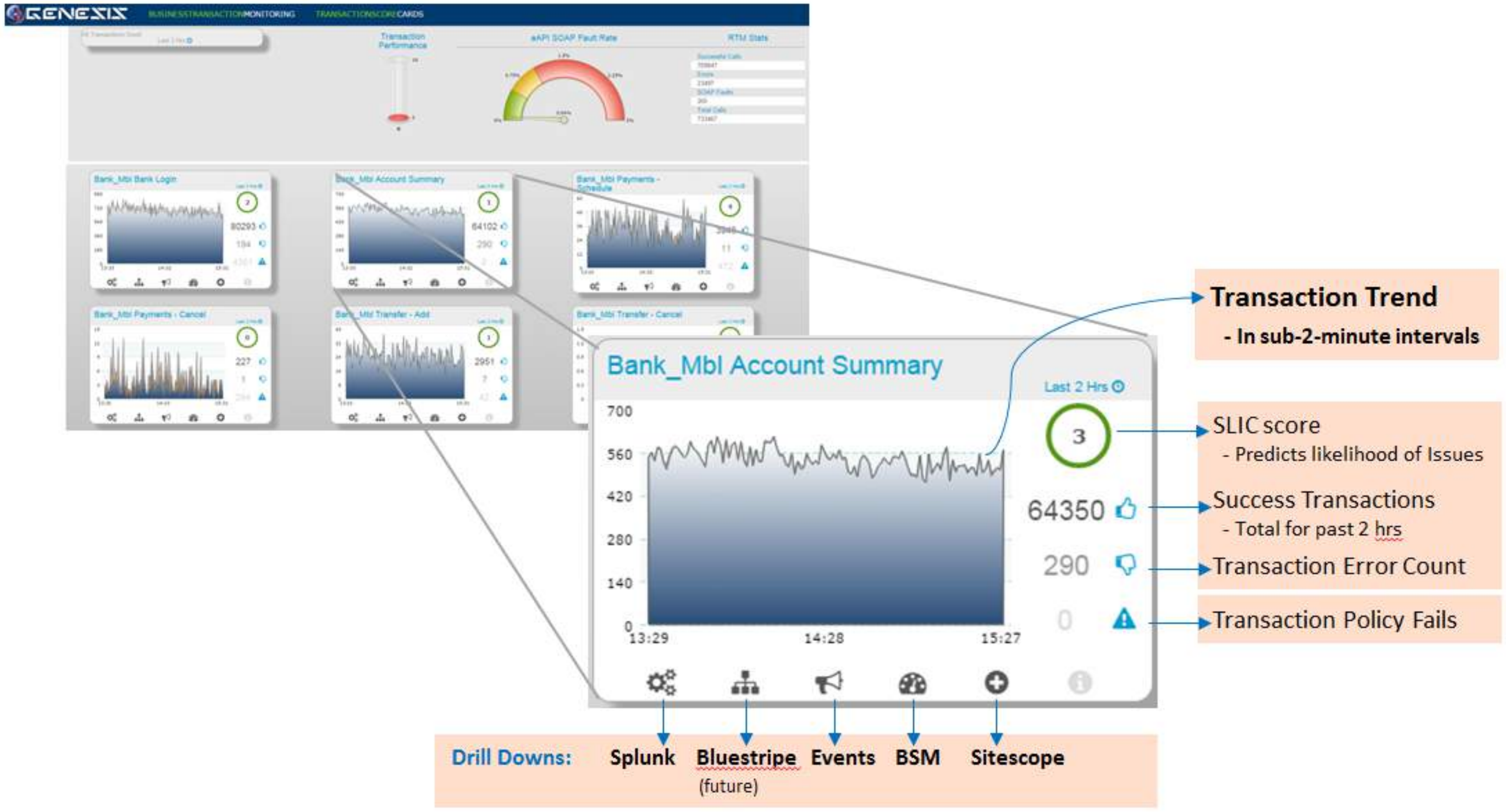
(All)

Choose a Time Period

6/16/2015 8/31/2015

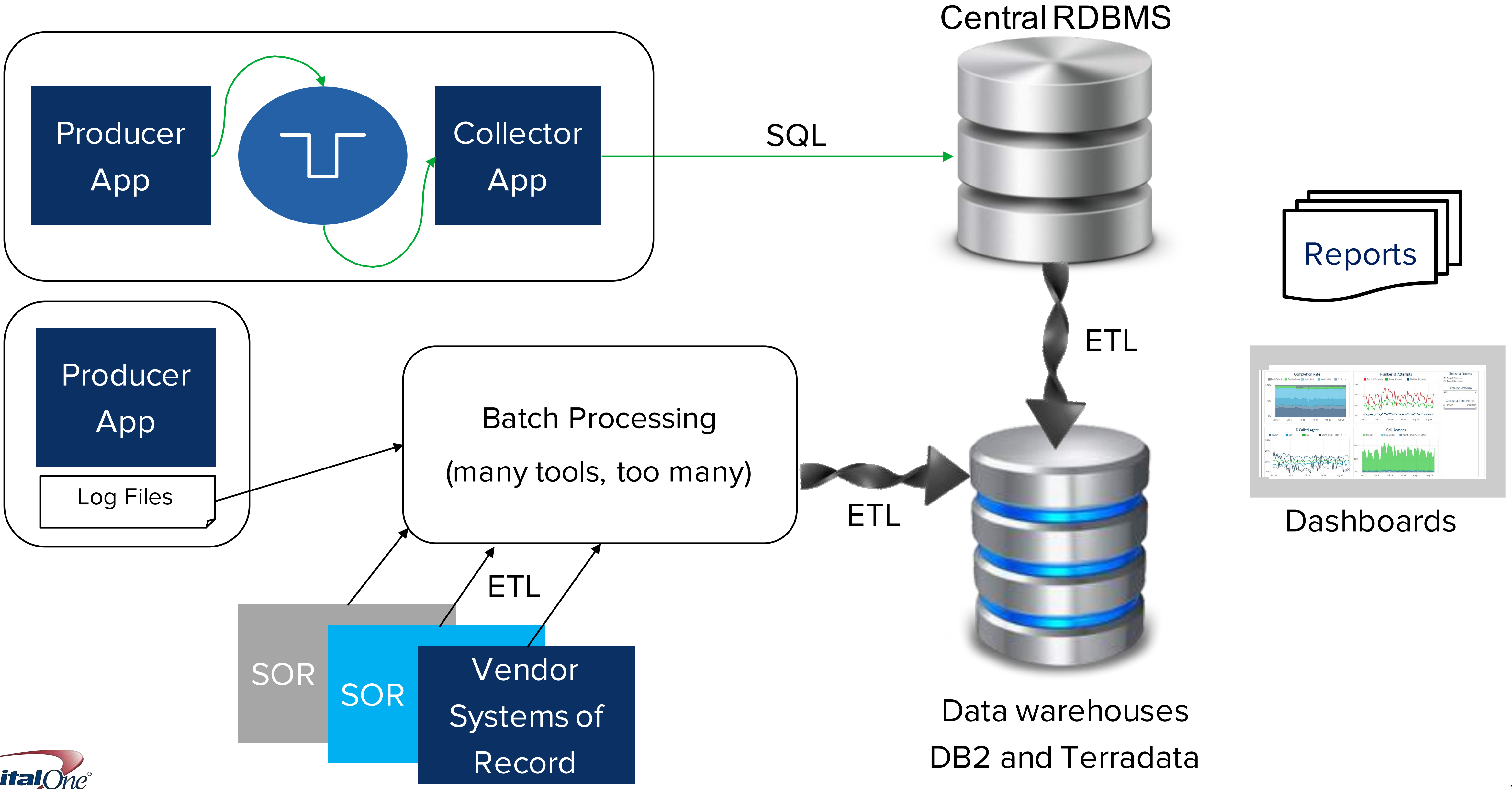
Capital One

This Splunk report provides visibility into critical transactions and the likelihood that trouble is brewing

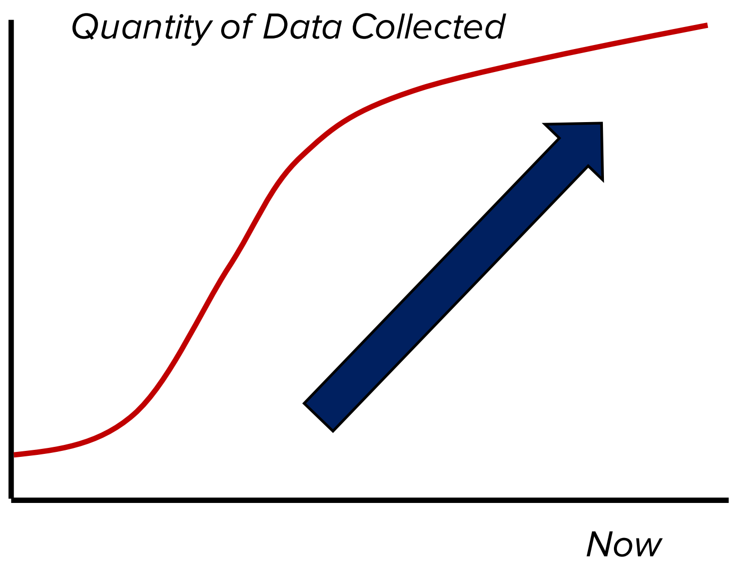


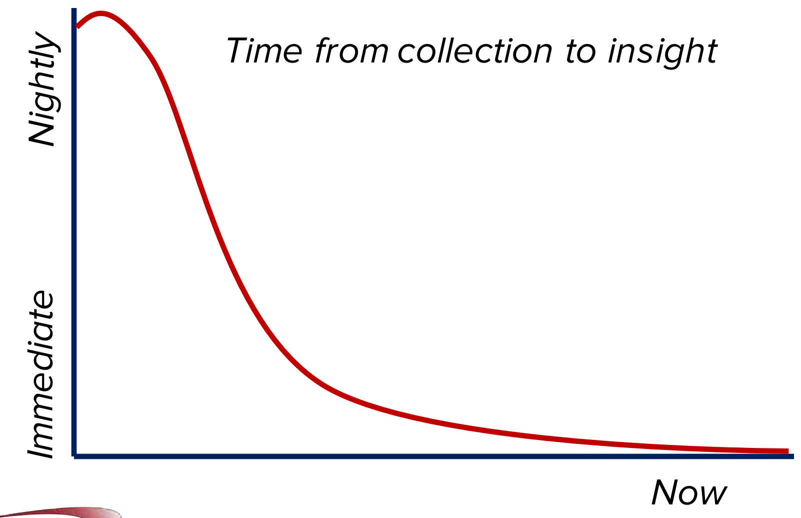
STARTING POINTS

Like many companies, in-place solutions relied heavily on messaging, warehouses, and batch ETL



Growth in data collection and the corresponding contraction in desired application to valuable use cases required re-architecting





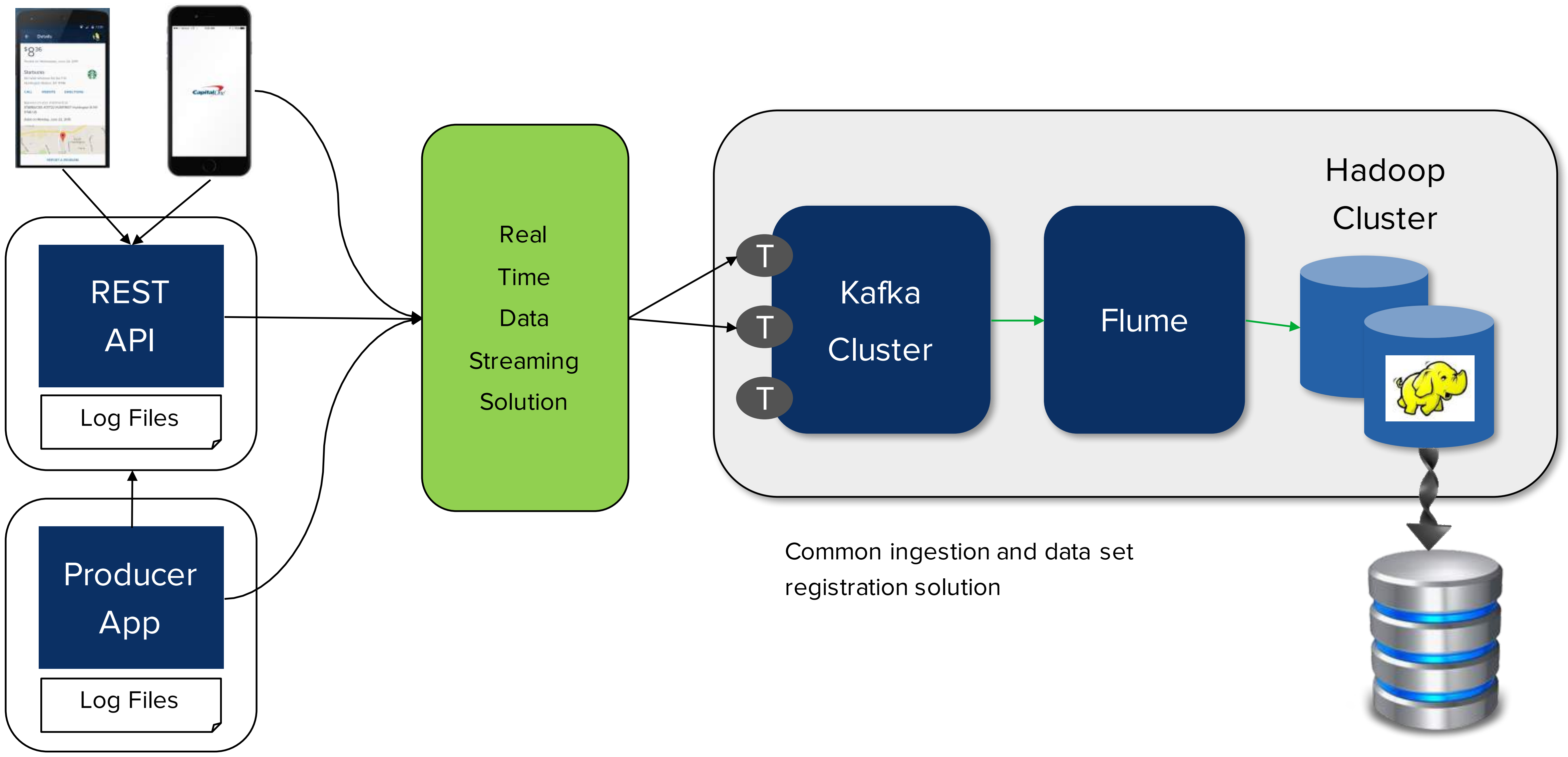
- **Easy to collect massive quantities of data via streaming**
- **Central DB growing quickly**
  - Gathering over 450GB per day in raw data
  - More DB instances (2 became 10), driving costs up
  - ETL processes missing SLA
- **Hard to get the right set of data**
  - As much of 80% of data meaningless and unused
  - Key data sometime missing
- **Operations and Statisticians demanding more data, more frequently**
  - Nightly became hourly → True desire was immediate
  - Still not fast enough → Don't just react… predict!
- **Key Use Cases:**
  - Real time Fraud detection
  - Reactive offer and message selection

Guiding principles in evolving the solution

- All roads should lead to Hadoop
- Open Source Software
- Centralize access to HDFS
- Embrace the Cloud
- Obsess over security
- Eliminate vendor RDBMS and reduce reliance on data warehouses
- Adopt Big Data tooling

SELECTING A DATA STREAMING TOOL

Goal: Stream user activity and data collection real time to the HDFS data lake, which becomes the lever to modernize analytics and deprecate most ETL



Common ingestion and data set registration solution

Selected Spring Extreme Data (XD) for the central data streaming platform

Technological Alignment

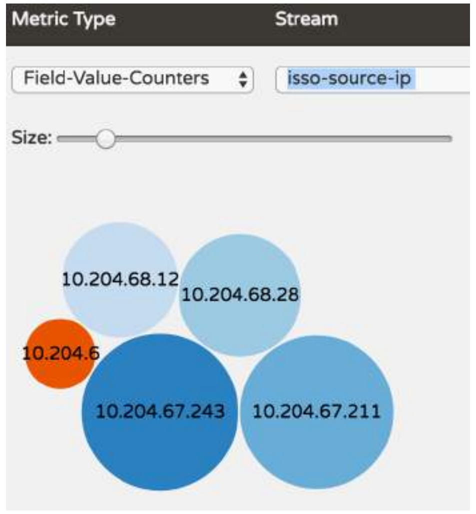
- ✓ Open Source
- ✓ Aligned to internal skill sets
  - ✓ (Java, Spring)
- ✓ Provided modules for common sources, processes, and sinks
  - ✓ Rabbit MQ, Active MQ, Kafka
  - ✓ Hadoop
  - ✓ HTTP
  - ✓ MongoDB
  - ✓ RDBMS / JDBC
  - ✓ Filters, Splitters, Aggregator, etc.

Met Architectural Attributes

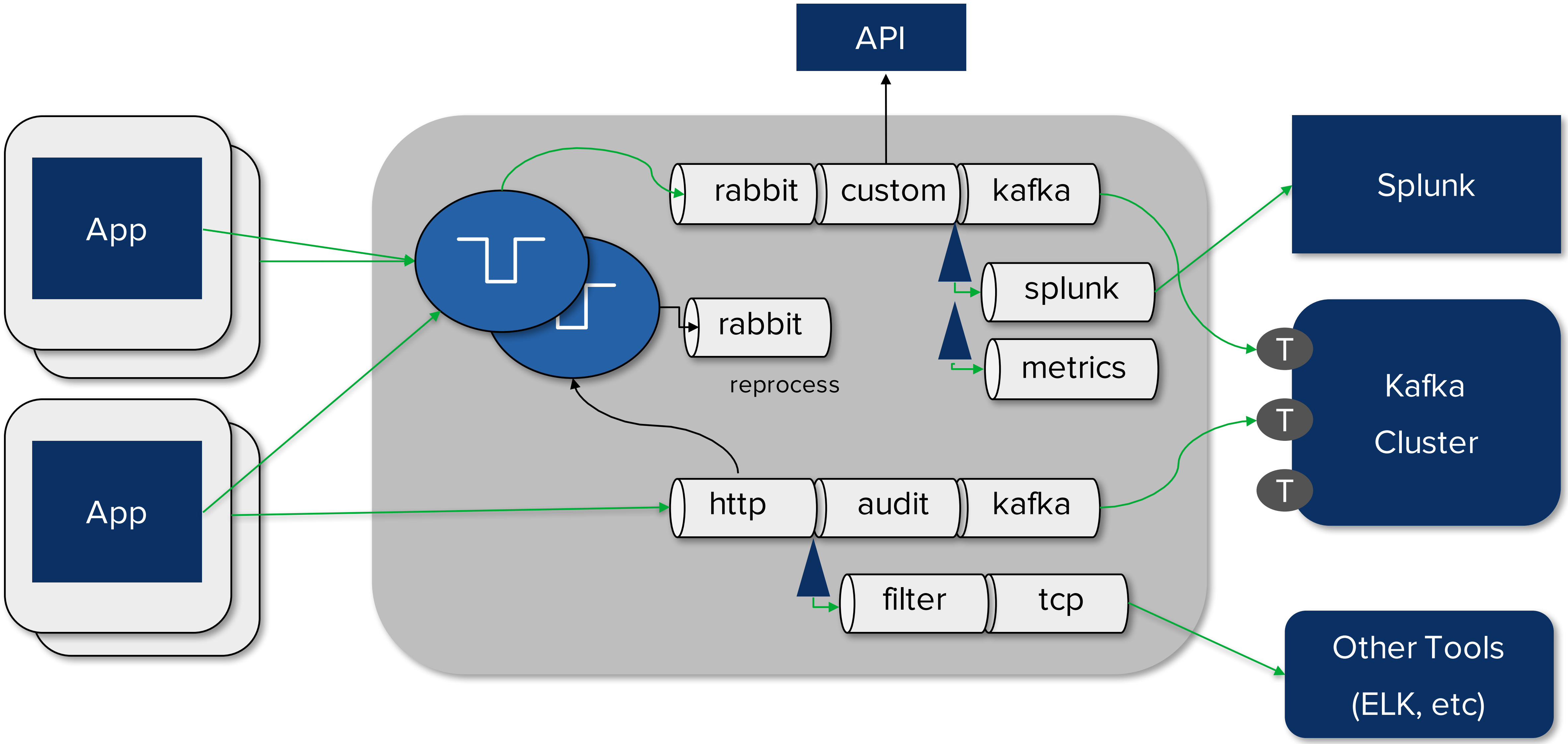
- ✓ Performance (we saw over 2800 tps)
- ✓ Scalable
  - ✓ Vertically and Horizontally
- ✓ Modular
- ✓ Extensible
- ✓ Sharable
- ✓ Deployment options
- ✓ Interoperate between batch and real time

Spring XD provides many useful constructs

- **Taps**
  - Takes input from some point in another stream
  - Creates duplicate flows of data to alternate sinks in parallel
  - Parallel processing
  - Support experiments (even in production)
- **Filters**
- **Composite modules**
  - Transform data
  - Circumvent data transport (Rabbit MQ or Kafka) to use in memory channels
- **Stream analytics**
  - Run through models using PMML
  - Counters, Gauges, Metrics


Metric Type: Field-Value-Counters
Stream: isso-source-ip

Our real time streaming solution collects data from multiple sources, augments it, transforms it, and writes it to multiple sinks while capturing analytics in flight



Key decisions

- **Multiple applications write to shared input queues**
  - JSON preferred format
- **Rabbit MQ used to both inbound data and data bus between modules**
  - Decouple each processing step
  - Cluster Rabbit with multiple slaves, durable queues, and DLQ
  - Use queues to decouple primary stream from taps to Splunk and EL
- **Primary streams to HDFS data lake**
  - Write to Kafka
  - Flume process periodically registers data set and ingested data file, converting to AVRO or Parquet as needed
  - HDFS stores data
    - Hive used for structured query
    - Map Reduce
    - ETL to legacy data warehouses until migrated to modern tooling
- **Use deployment properties to deploy modules across the cluster**
- **Create re-process streams on dead letter queues, then write to files bound for AWS S3 storage**

The stream definition DSL makes creating complex processing streams with multiple data paths and metrics simple. Deployment properties support non-functionals.

```
xd:> stream create --name "activity-logging" --definition "rabbit --queues=activity.in.queue | kafka --
     topic=activity_logs --brokerList='localhost:9395,localhost:9396,localhost:9397,localhost:9398'”

xd:> stream create --name "activity-splunk-tap" --definition "tap:stream:activity-logging > rabbit --
     exchange=activity.splunk.exchange --routingKey='\"activity.splunk.routing-key\"'”

xd:> stream create --name "activity-splunk1" --definition "rabbit --queues=activity.splunk.queue |
     tcp --host=digitalhfw1ae.cloud.capitalone.com --port=9500 --encoder=CRLF”

xd:> stream create --name "activity-fvc-dispcode" --definition "tap:stream:activity-logging > field-
     value-counter --fieldName=EventStreamData.dispositionCode --inputType=application/json”

// xd supports many properties to control how streams are deployed
// module counts and targeting to specific Spring Boot containers are valuable
// Provide scale, isolation, and resiliency
xd:> stream deploy --name=“http-api-logging” --properties “module.http.count=3”
```

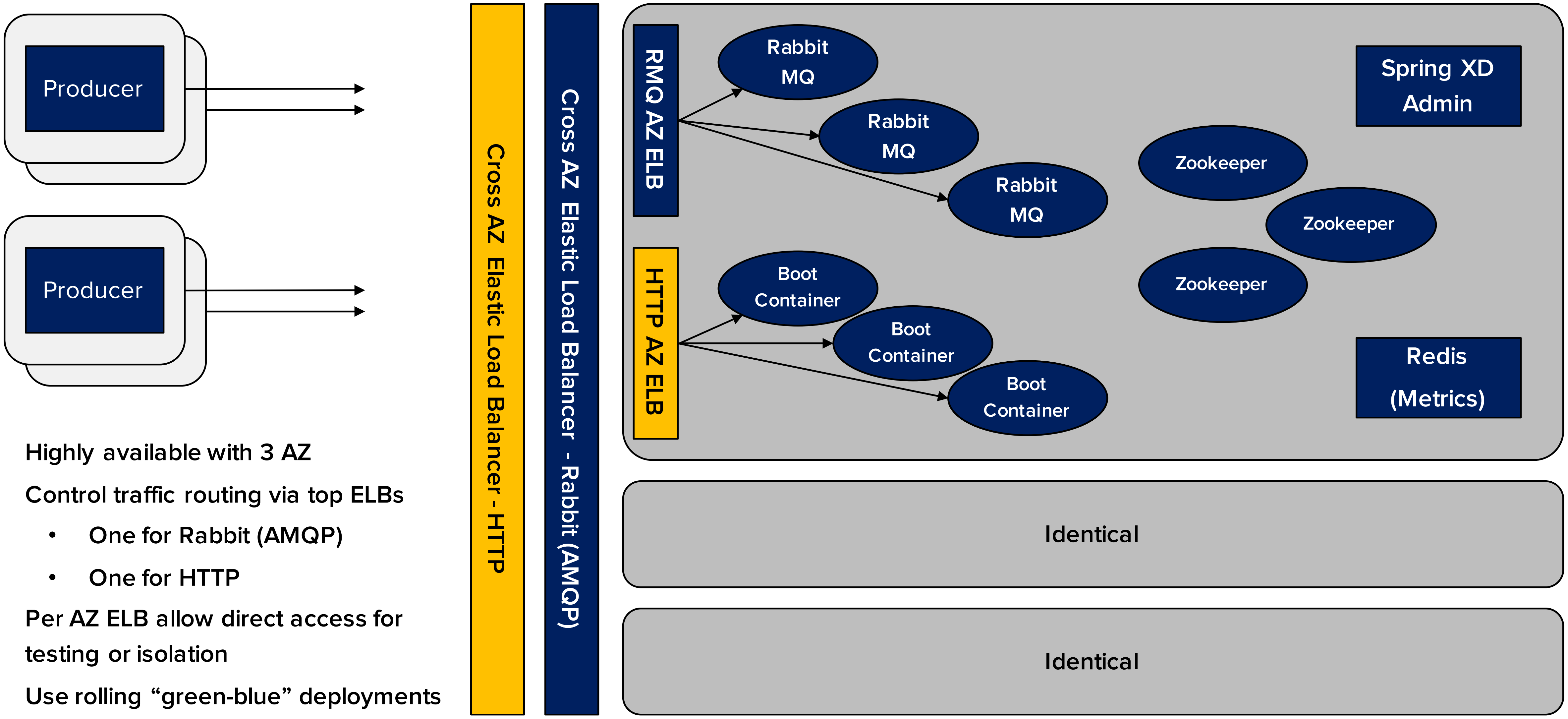
DATA STREAMING IN THE AWS CLOUD

IMPLICATIONS WHEN TECHNOLOGY NOT BUILT FOR CLOUD

Capital One's wide scale adoption of public cloud required:

- **Automation of environment provisioning essential**
- **Spring XD config uses ip addresses --> precluded use of AWS Auto-scaling Groups**
  - Provision fixed size clusters
  - Automation of provision
- **Use multiple parallel clusters with load balancing and CNAME's**
  - Enable blue-green deployments and rolling updates
- **Resiliency and Availability**
  - ELB with CNAMEs
  - Multi AZ
- **Shared environment required LDAP integration and**
  - Use Rabbit Policies to restrict admin access
  - Simple authorization on XD console precludes team self-service unless you're more trusting than us
  - Separate environments by Line of Business (each own AWS VPC)
- **Monitoring via JMX, AWS Cloudwatch, and Zabbix**

Designed for resiliency, availability, and isolation using parallel load-balanced clusters in three AWS Availability Zones



- Highly available with 3 AZ
- Control traffic routing via top ELBs
  - One for Rabbit (AMQP)
  - One for HTTP
- Per AZ ELB allow direct access for testing or isolation
- Use rolling “green-blue” deployments

Capital One

We had to design for and mitigate a number of challenges with cloud deployment and the use of automated provisioning scripts

Spring XD configuration files used static ip addresses for Rabbit, Zookeeper, and Redis nodes

- Allocate fixed sized environments; no ASG
- Proactive capacity expansion
- Use Chef search to find ip as nodes check in to Chef server

Real Time monitoring in the Cloud

- Provision Cloudwatch metrics via standard AWS AMI
- Enable metrics on ELB
- Install Zabbix agent to monitor servers and instances
- Turn on JMX and export via real time stream to Splunk

Deploying modules for Optimal Load Balancing

- XD deploays each module in a stream to a single container
- To load balance / provide resiliency used module count
- Allows entire RMQ and Container to be leveraged
- Make best use of ELB

Capital One

SECURITY CONSIDERATIONS

Kafka security gaps required interim solution

- **Selected Rabbit MQ for data transport over Kafka due to lack of TLS and authorizations on topics**

- **Require secure transport of data**
- **Selected stunnel**
  - Establishes an encrypted secure channel between a client and server for applications that do not natively support TLS or SSL
  - Installed stunnel on each container running a kafka sink
  - Installed stunnel on kafka side

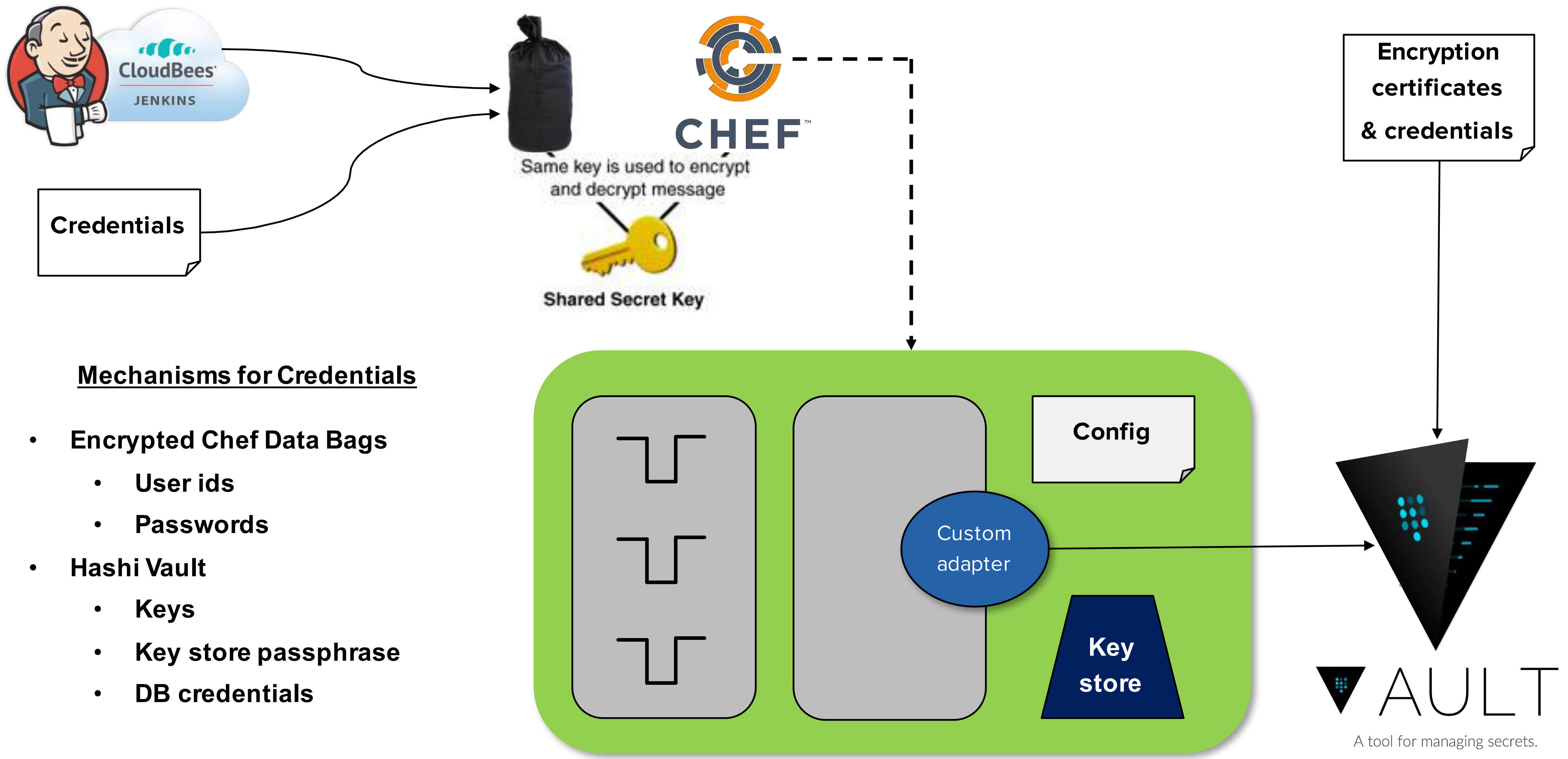
- **Provide self signed certificates**
  - Using certification level 4 to ignore CA chain in validation
- **Provide config file that maps ports to brokers**

```
fips = no
output = /etc/stunnel/stunnel.log

[broker1]
accept = 127.0.0.1:9195
connect = 10.20.30.5:9193
client = yes
verify = 4
cert = /root/stunnel/kafka-cli-stunnel.pem

[broker2]
accept = 127.0.0.1:9196
...
```

We implemented solutions to restrict access and protect credentials

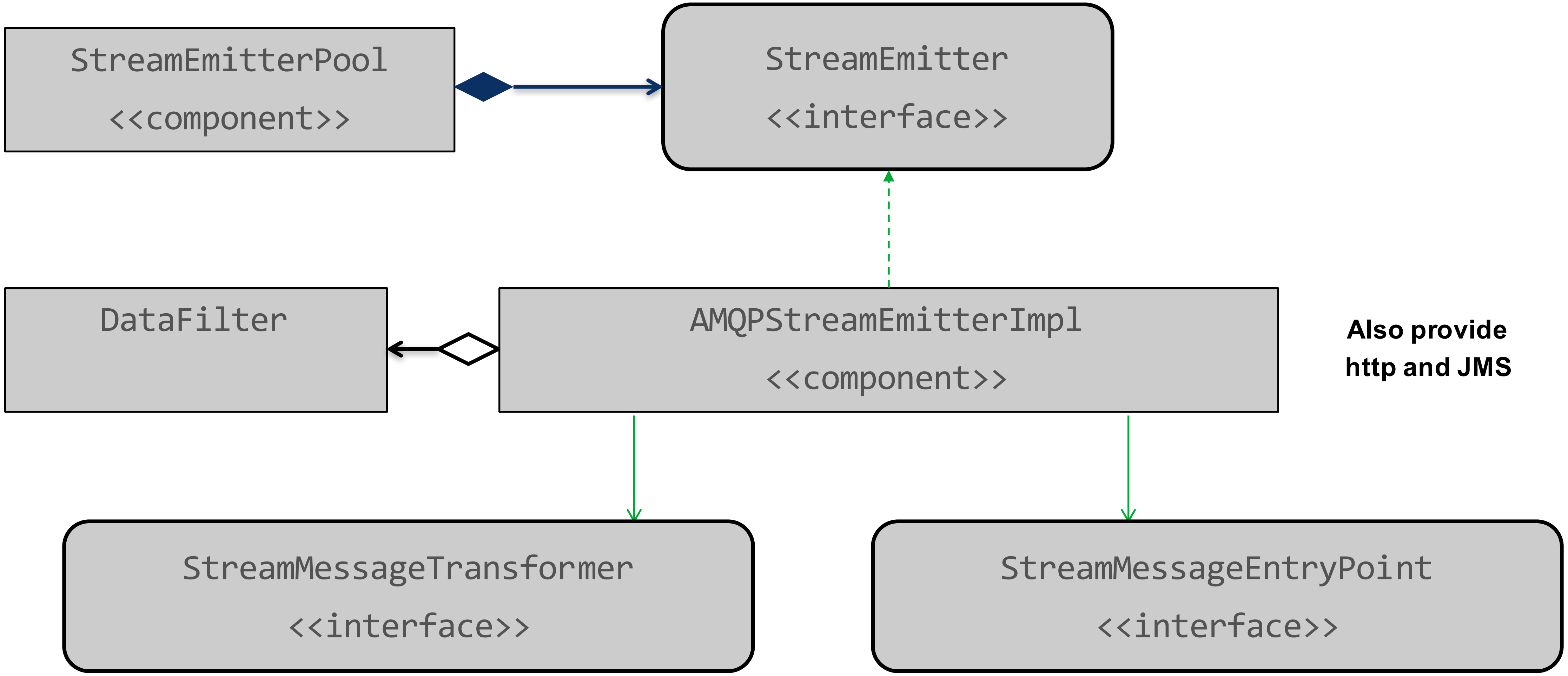

CloudBees
JENKINS
Credentials
CHEF
Same key is used to encrypt and decrypt message
Shared Secret Key
Encryption certificates & credentials
Config
Custom adapter
Key store
VAULT
A tool for managing secrets.

Mechanisms for Credentials

- **Encrypted Chef Data Bags**
  - **User ids**
  - **Passwords**
- **Hashi Vault**
  - **Keys**
  - **Key store passphrase**
  - **DB credentials**

CLIENT LIB

Configured emitters transform and write to XD



Also provide http and JMS

Providing a Java library made implementation easy while allowing for client flexibility

```
@Component
public class SomeBankingObject {
    @NonPublicInformation                                    // applies encryption
    private CustomerIdentity customerId;

    @Audited                                                 // uses configured StreamEmitter
    private AccountId accountId;

    @EventStream(type = ACTIVITY)                            // uses configured StreamEmitter
    public void  executeTransaction(SomeModel sm, SomeOtherModel som)  {
         …; }
    // Can get verbose if more StreamEmitter for different output
    @EventStream( eventName = “someBankTransaction”,
       standardEventStream = { EventStream.EventStreamType.LOG },
       customEventStream   = { @EventStream.CustomEventStream
         (type = EventStream.EventStreamType.AUDIT, emitters = { “auditEmitter”} ) } )
    public void  executeTransaction(SomeModel sm, SomeOtherModel som)  {
         …; }
}
```

For more information about Spring XD or Capital One…

- **To see more details on our Spring XD journey, see my presentation: http://www.slideshare.net/SpringCentral/supercharging-operations-analysis-using-spring-xd-to-support-analytics-and-cep?qid=0c5852a9-0cab-404c-8d98-463a3235f7d1&v=qf1&b&from_search=1**
- **Check out Capital One's Engineering blogs at http://www.capitalone.io**
  - Look for my upcoming post "Turning Data into Insight"
- **To learn more about Technology career opportunities: https://jobs.capitalone.com/search-jobs/engineering**

THANK YOU FOR ATTENDING