GOPS2016 Shenzhen

GOPS

2016 全球运维大会 深圳站

会议时间：3月25日-3月26日

会议地点：深圳 • 南山区 圣淘沙酒店(翡翠店)

主办单位：开放运维联盟 OOPSA Open OPS Alliance　高效运维社区 GreatOPS Community

指导单位：数据中心联盟 Data Center Alliance

协办单位：中国新一代IT产业推进联盟

GOPS 2016 Shenzhen

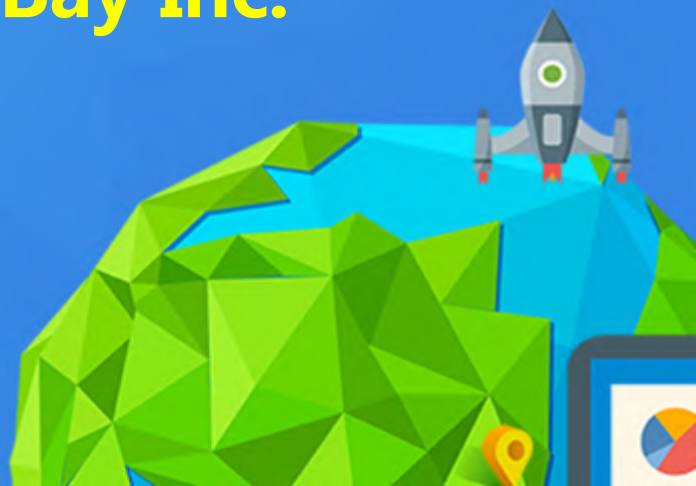
GOPS

2016 全球运维大会 深圳站

Apache Eagle：分布式实时监控预警框架

陈浩 eBay Inc.

ebay



关于我

Tech Lead, Sr. Software Engineer @ eBay Cloud Platform
hchen9@ebay.com

Co-creator, Committer and PMC @ Apache Eagle
hao@apache.org

Speaker @ Qcon / Hadoop Summit (SJC, SHA, BJ)
http://people.apache.org/~hao

Agenda

- **Introduction**
- Architecture
- Ecosystem
- Q & A

What is Apache Eagle

Apache Eagle' is a distributed real-time monitoring and alerting engine for hadoop from eBay

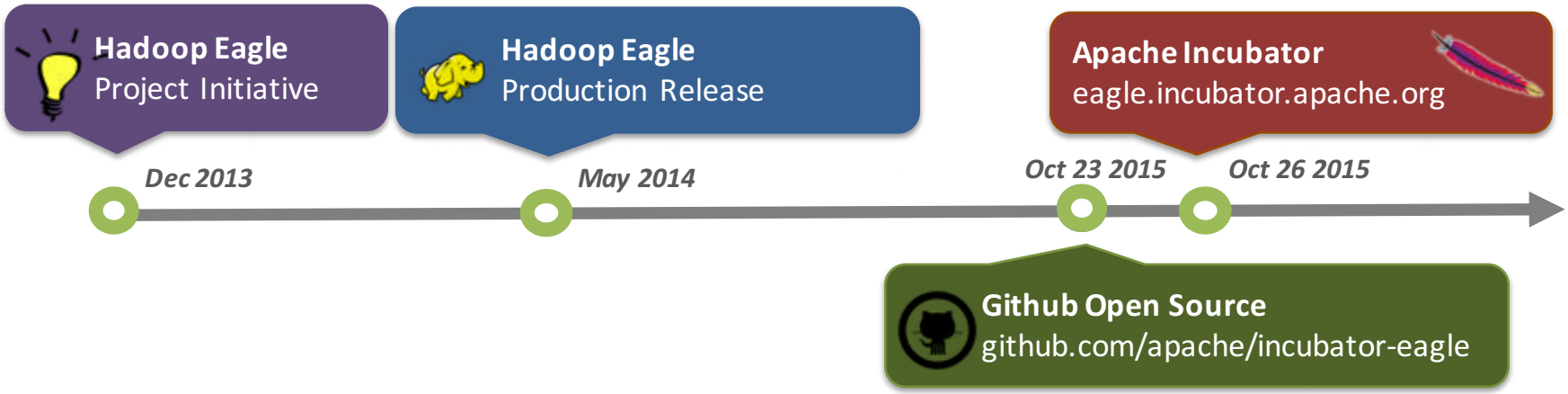
Open sourced as Apache Incubator Project on Oct 26th 2015

Secure Hadoop in Realtime *a data activity monitoring solution to instantly identify access to sensitive data, recognize attacks/ malicious activity and block access in real time.*

See http://eagle.incubator.apache.org or http://goeagle.io

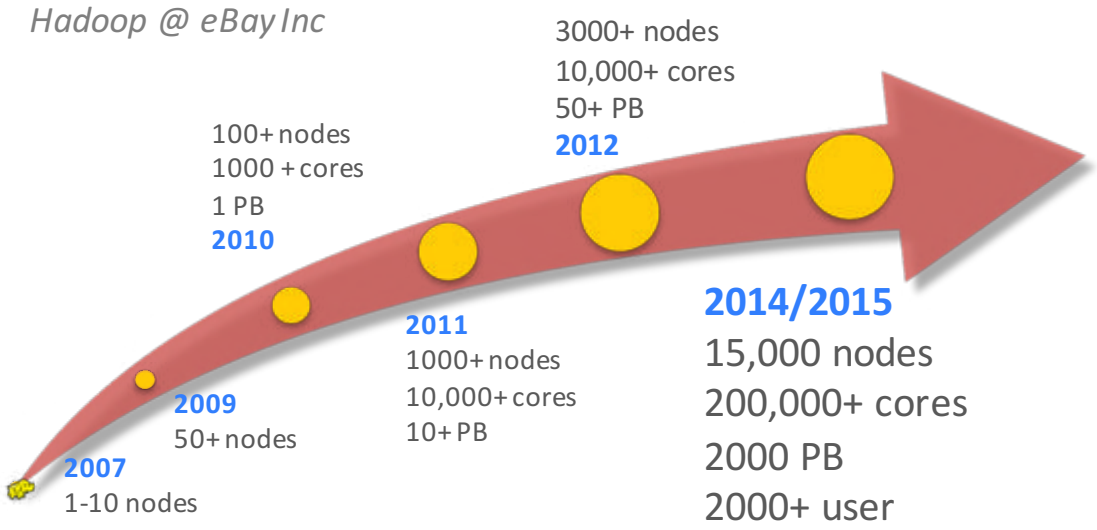
Apache Eagle History

Donated to Apache Software Foundation (ASF) from eBay at Oct 26th, 2015



Why build Eagle

Eagle was initialized by end of 2013 for hadoop ecosystem monitoring as any existing tool like zabbix, ganglia can not handle the huge volume of metrics/logs generated by hadoop system in eBay.



Hadoop Data
- Security
- Activity

Hadoop Platform
- Heath
- Availability
- Performance

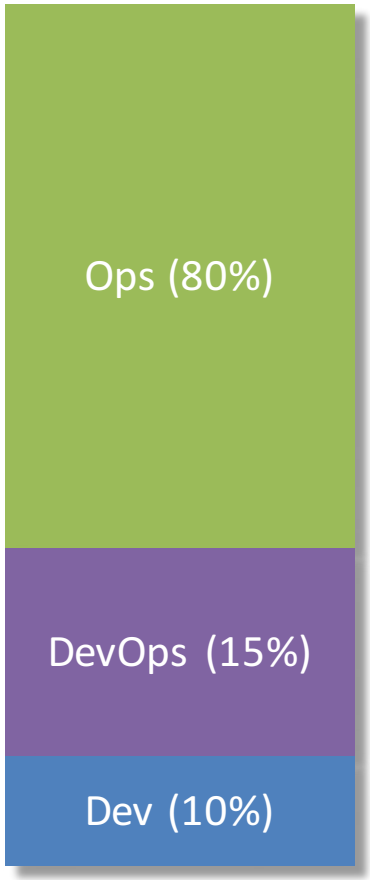
Challendges

Functional Requirements

- Flexible Threshold
- Slide Window
- Multiple-factors Correlation
- Real-time or historical data Join
- Complex monitoring cases
- …

Unfunctional Requirements

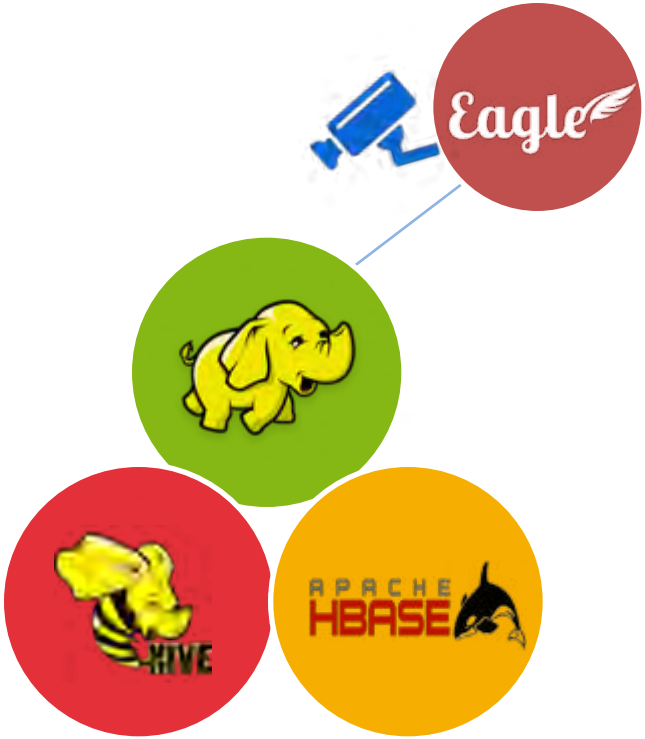
- Scalable: Distributed
- Real-time: Streaming
- Maintainability: Hot-deploy
- Intelligent: Machine Learning
- …



Agenda

- What’s Eagle
- **Architecture**
- Ecosystem
- Q & A

Eagle Architecture Overview



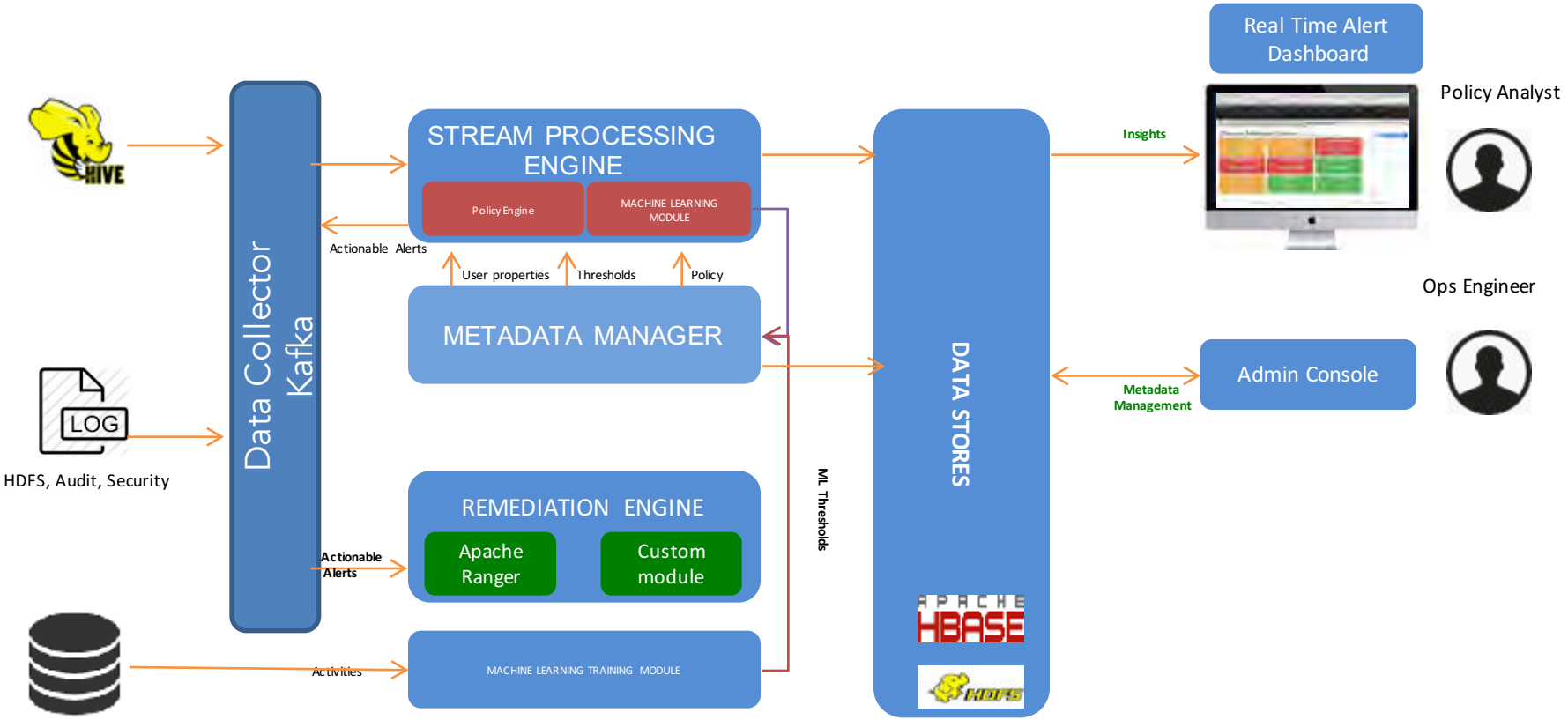
Scalable
Scales to monitor thousands of policies and billions of access events

Extensible
Eagle can be easily extended to monitor other data sources

Real-time
Generates alerts in real time and blocks users with malicious intent

Machine Learning
Create dynamic user profiles based on user behavior

Eagle Architecture Overview



Eagle Features Highlight

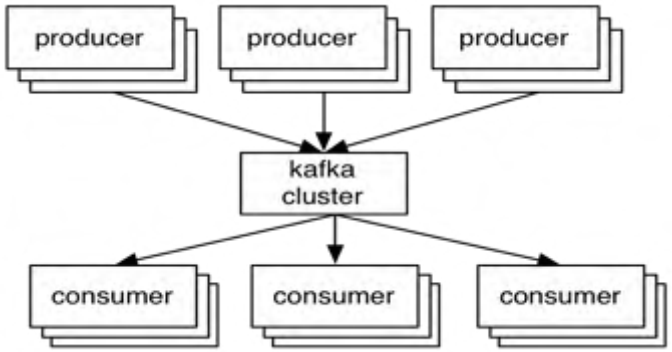
- Real-time Data Collection
- Distributed Policy Engine
- Stream Processing DSL
- Scalable Data Storage & Query
- Machine Learning Intergration
- Module Management
- Multiple Tenant Support

NOTE *{NAME}-{NUMBER} like HDFS-6914 means open source project ticket id contributed by us*

Eagle - Data Collection

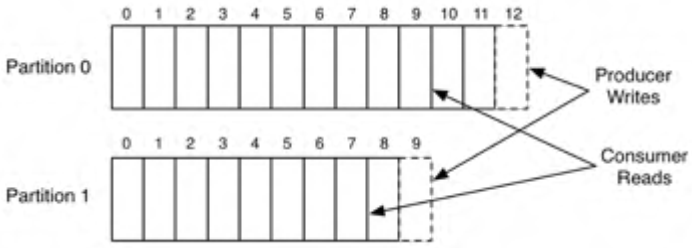
Decoupling with Message Bus

- **Apache Kafka:** high-throughput distributed messaging
- ***Partition:*** *balance between logic and throughput*



Cross-Platform Integration

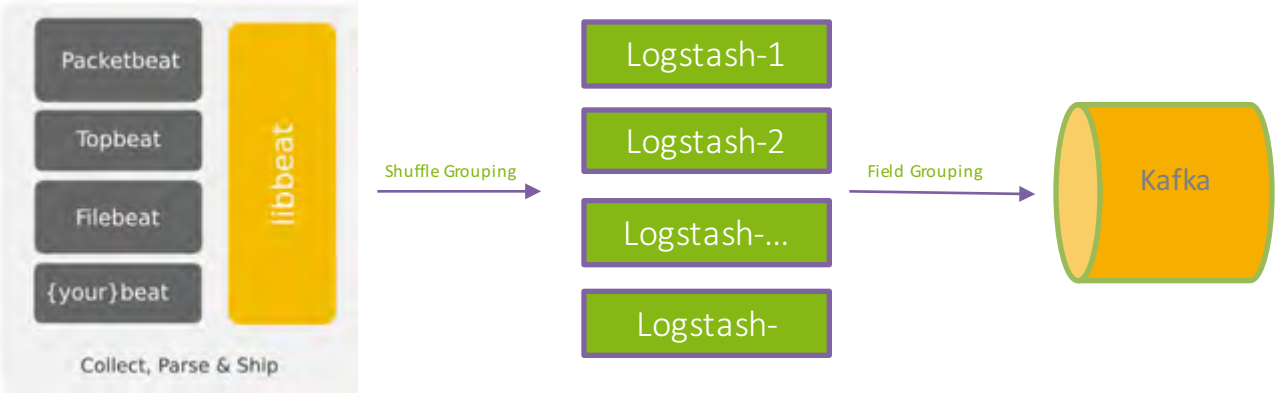
- **Community Kafka Client** (18+)
  - Python/Go/C/C++/JAVA ..
- **Enhanced Log4j-kafka**
  - KAFKA-2041: Extensible Partition Key
  - KAFKA-2077: Advanced Topic Selector



Eagle - Data Collection

Availability: Filebeat + Logstash

Resource consumption balance
Message throughput balance (LOGSTASH-179)



Ligh-weight collector (golang) with daemon

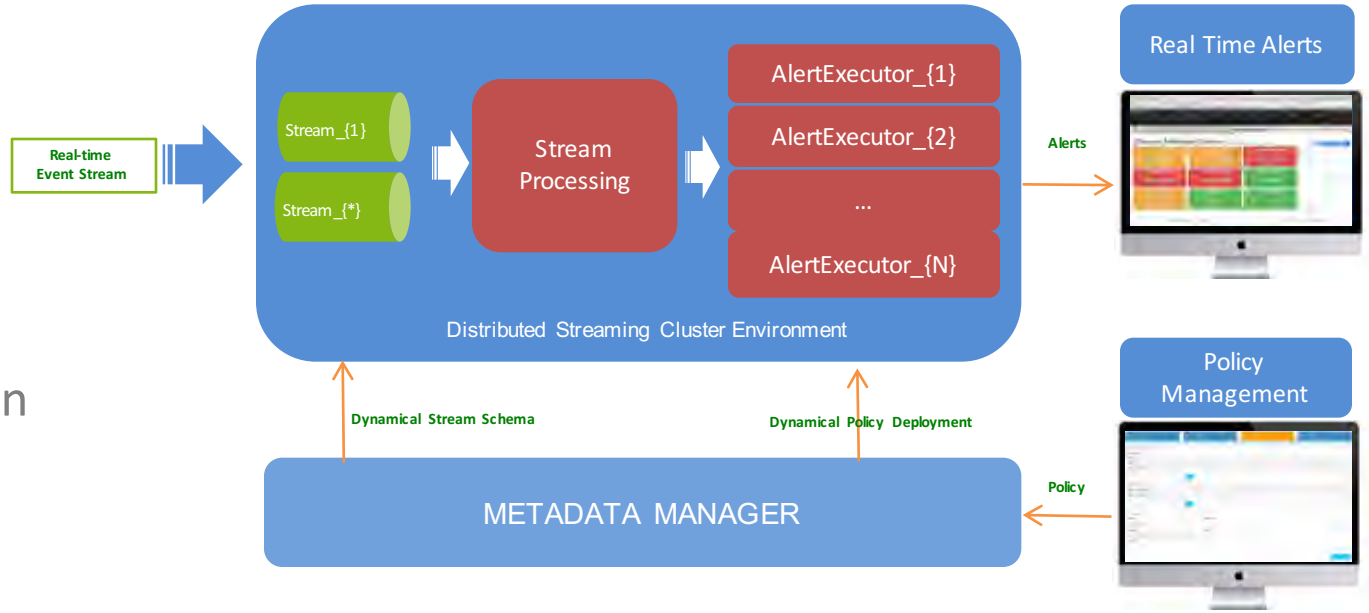
Logstash instances cluster

Distributed Message Bus

Eagle - Distributed Real-time Policy Engine

Highlights

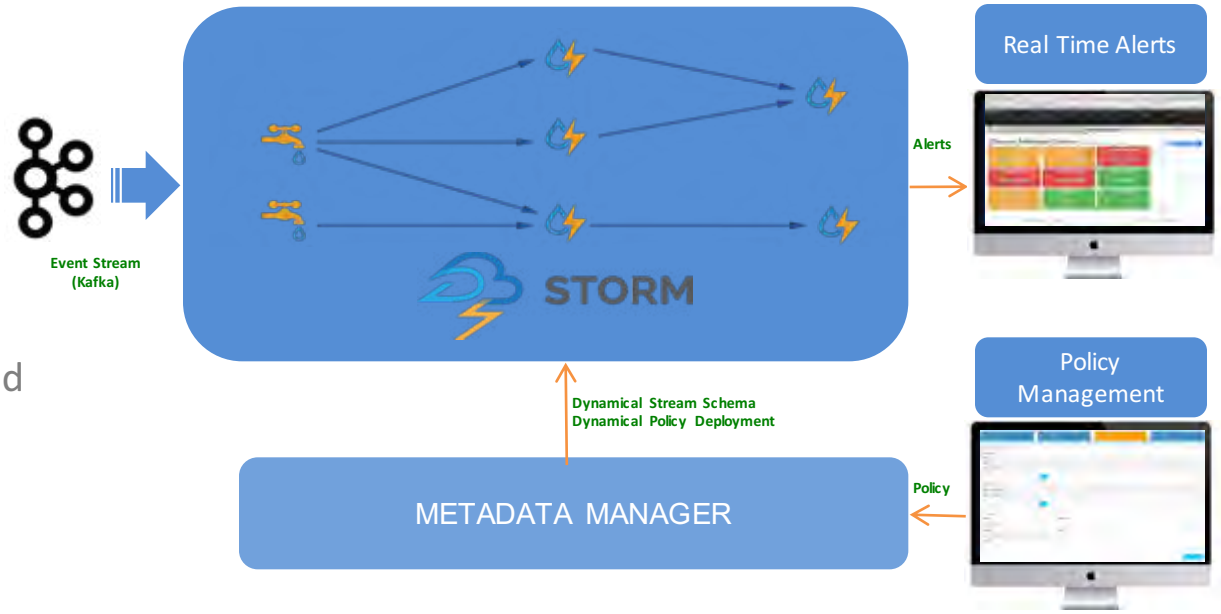
- Real-time
- Usability
- Scalability
- Extensibility
- Metadata-driven



Eagle - Distributed Real-time Policy Engine

Real-time

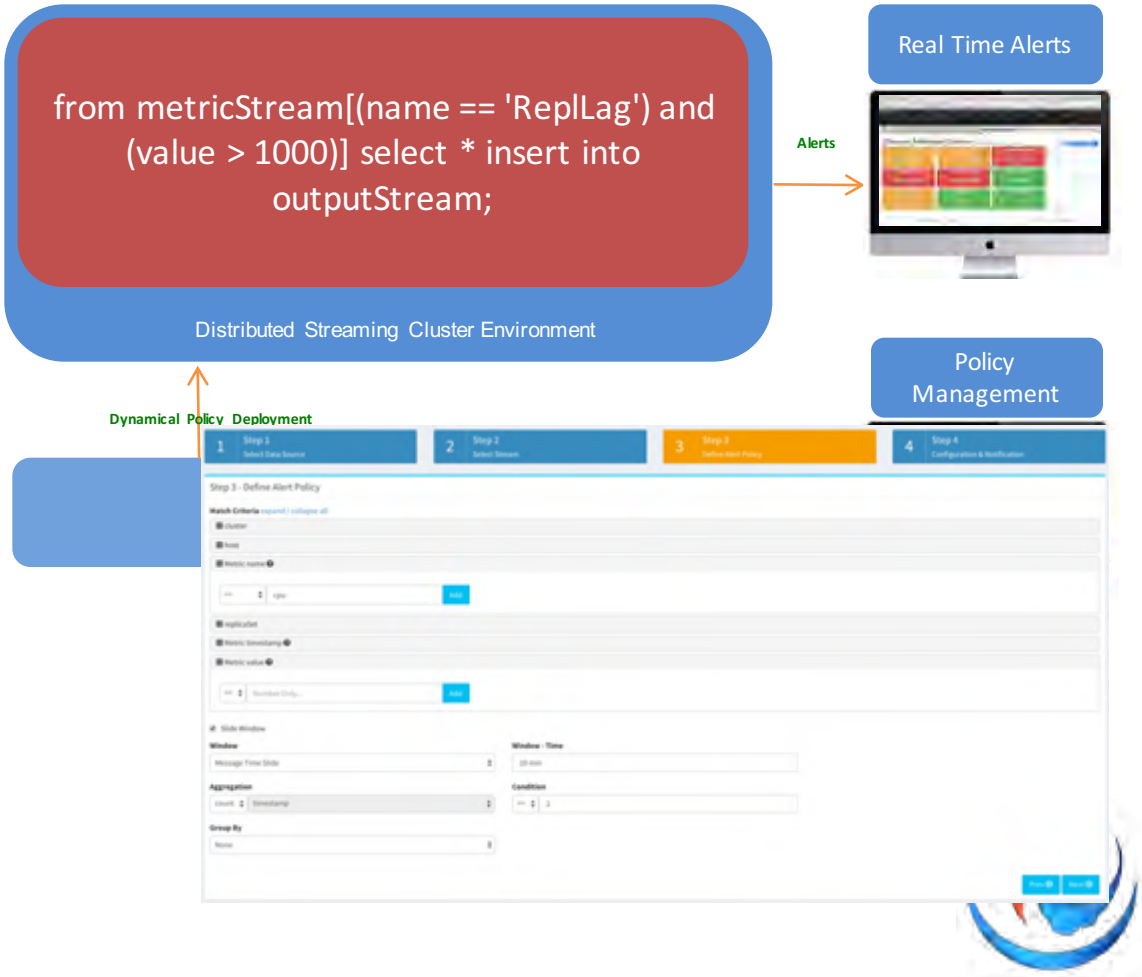
- Kafka-based Distributed Message Bus (Extensible)
- Storm-based Real-time Execution Environment (Extensible)
- Stream events are processed and alerts are evaluated during streaming



Eagle - Distributed Real-time Policy Engine

Usability

- Powerful SQL-Like CEP CQL for Policy Definition
- Dynamical Poilcy Metadata Lifecycle Management (Deployment/Update)
- Easy-to-use Policy management and Alert analytics UI



Eagle - Distributed Real-time Policy Engine

Full-function Streaming CEP CQL: Siddhi on Storm by default

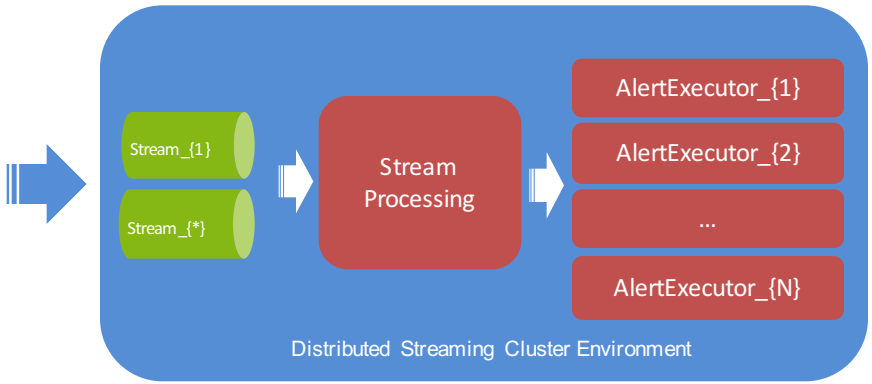
```
hdfsAuditLogEventStream[(src == '/tmp/private')]#window.externalTime(timestamp,10 min) select user, count(timestamp) as aggValue group by user having aggValue >= 5 insert into outputStream;
```

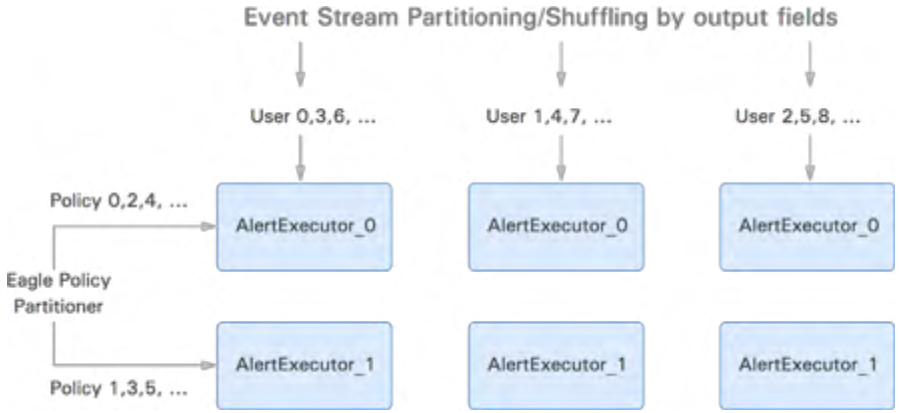
- **Filter**
- **Join**
- **Aggregation**: Avg, Sum , Min, Max, etc
- **Group by**
- **Having**
- **Stream handlers for window**: TimeWindow, Batch Window, Length Window
- **Conditions and Expressions**: and, or, not, ==,!=, >=, >, <=, <, and arithmetic operations
- **Pattern processing**
- **Sequence processing**
- **Event Tables**: intergrate historical data in realtime processing
- **SQL-Like Query**: Query, Stream Definition and Query Plan compilation

Eagle - Distributed Real-time Policy Engine

Scalability: dynamic policy partition by {event} * {policy}

- *N Users with 3 partitions, M policies with 2 partitions, then 3*2 physical tasks*
- *Physical partition + policy-level partition*

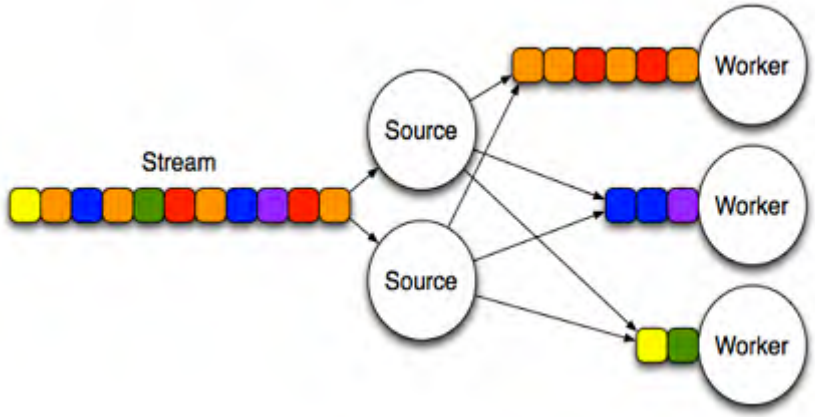




Eagle - Distributed Real-time Policy Engine

Distributed Streaming Partition Problem

https://en.wikipedia.org/wiki/Partition_problem



groupBy[GreedyStrategy]((_.key1,_.key2))

Realtime Partition Strategy

Key Distribution Statistics (Online/Offline)

Strategy

- Greedy (Online/Offline)
- PoTC
- PKG
- Hashing

Key Statistics Cache

Async

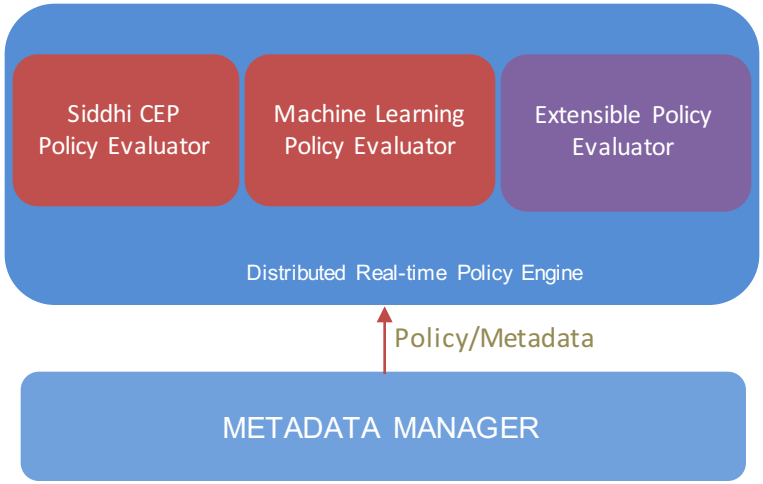
HBase

Eagle - Distributed Real-time Policy Engine

Extensibility

- Support WSO2 Siddhi CEP as first class
- Extensible policy engine implementation

```
public interface PolicyEvaluatorServiceProvider {
                public String getPolicyType(); // literal string to identify one type of policy
                public Class getPolicyEvaluator(); // get policy evaluator implementation
                public List getBindingModules(); // policy text with json format to object
mapping
}
```



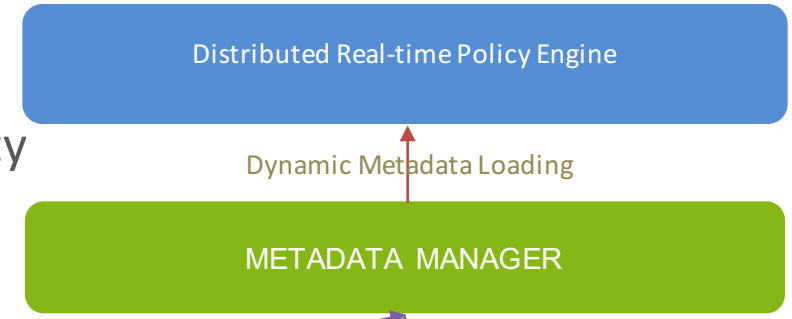
- Extensible policy lifecycle management

```
public interface PolicyEvaluator {
                public void evaluate(ValuesArray input) throws Exception; // evaluate input event
                public void onPolicyUpdate(AlertDefinitionAPIEntity newAlertDef);// policy update
                public void onPolicyDelete(); // invoked when policy is deleted
}
```

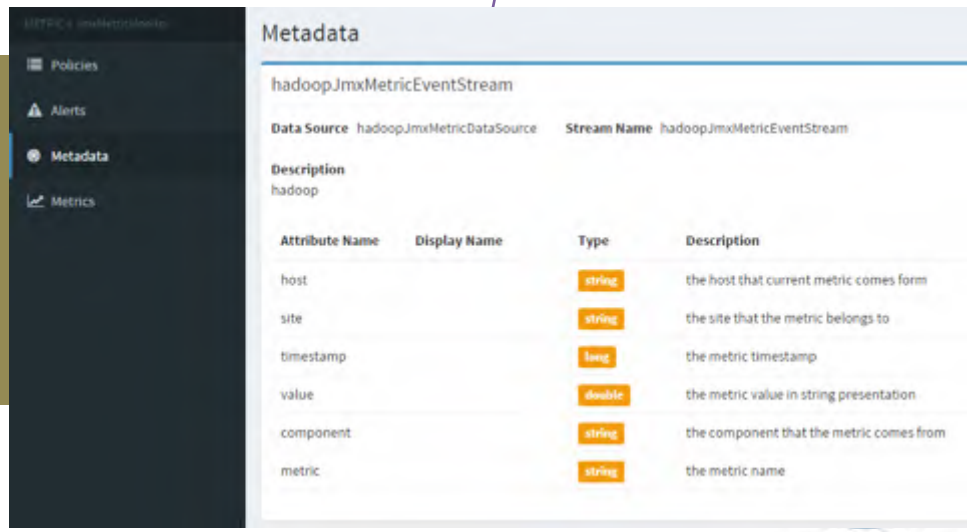
Eagle - Distributed Real-time Policy Engine

Metadata-Driven

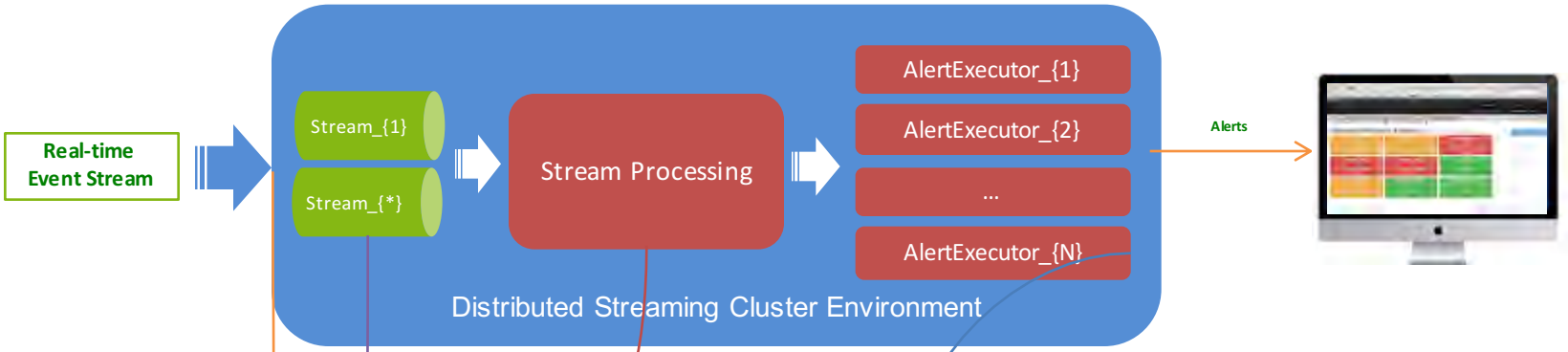
- Stream Schema: AlertStreamSchemaEntity
- Policy Definition: AlertDefinitionAPIEntity
- Central metadata management
- Dynamic metadata deployment




```
@Table("alertdef")
@ColumnFamily("f")
@Prefix("alertdef")
@Service(AlertConstants.ALERT_DEFINITION_SERVICE_ENDPOINT_NAME)
@JsonIgnoreProperties(ignoreUnknown = true)
@TimeSeries(false)
@Tags({"site", "dataSource", "alertExecutorId", "policyId", "policyType"})
@Indexes({
  @Index(name="Index_1_alertExecutorId",columns = { "alertExecutorID" }, unique = true),
})
public class AlertDefinitionAPIEntity extends TaggedLogAPIEntity{
  @Column("a")
  private String desc;
  @Column("b")
  private String policyDef;
  @Column("c")
  private String dedupeDef;
```



Eagle - Fluent Stream Processing DSL




```
val env = ExecutionEnvironment.getStorm()
env.fromKafka(KafkaConfig)
    .flatMap(AuditLogTransformer)
    .groupBy(_.user)
    .flatMap(UserProfileAggregator);
   .alert.persistAndEmail
env.execute()
```

Eagle - Fluent Stream Processing DSL

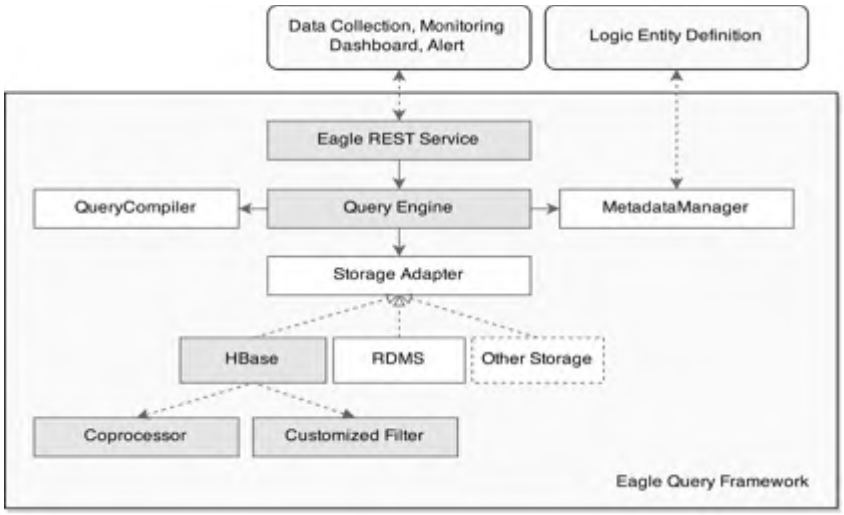

```
trait StreamProducer{
  filter
  flatMap
  map{1,2,3,4}
  groupBy
  streamUnion  // stream join is hard, not implemented for storm
  alertWithConsumer
}
```

```
StormExecutionEnvironment env =
ExecutionEnvironmentFactory.getStorm(config);
env.newSource(new
KafkaSourcedSpoutProvider().getSpout(config)).renameOutputFields(1)
      .flatMap(new AuditLogTransformer())
      .groupBy(0)
      .flatMap(new UserProfileAggregatorExecutor());
.alertWithConsumer("userActivity","userProfileExecutor")
env.execute();
```

- Physical execution platform independent
- Easily assemble data transformation, filtering, join and alerting DAG in fluent way
- DAG rewrite and optimization
  - StreamUnionExpansion
  - StreamGroupbyExpansion
  - StreamNameExpansion
  - StreamAlertExpansion
  - StreamParallelismConfigExpansion

Eagle - Scalable Data Storage and Query

- Entity Metadata on large-scale NoSQL storage like HBase
- Full-function SQL-Like REST Query
- Optimized rowkey design for time-series monitoring data
- HBase Coprocessor
- Secondary Index




```
@Table("alertdef")
@ColumnFamily("f")
@Prefix("alertdef")
@Service(AlertConstants.ALERT_DEFINITION_SERVICE_ENDPOINT_NAME)
@JsonIgnoreProperties(ignoreUnknown = true)
@TimeSeries(false)
@Tags({"site", "dataSource", "alertExecutorId", "policyId", "policyType"})
@Indexes({
  @Index(name="Index_1_alertExecutorId",columns = {"alertExecutorID"}, unique = true),
})
public class AlertDefinitionAPIEntity extends TaggedLogAPIEntity{
  @Column("a")
  private String desc;
  @Column("b")
  private String policyDef;
  @Column("c")
  private String dedupeDef;
```

```
query=
AlertDefinitionService[@dataSource="hiveQueryLog"]{@policyDef}
```

Eagle – Uniform HBase Rowkey Design

Uniform rowkey design

Rowkey ::= Prefix | Partition Keys | timestamp | tagName | tagValue | …

- Metric

  Rowkey ::= Metric Name | Partition Keys | timestamp | tagName | tagValue | …

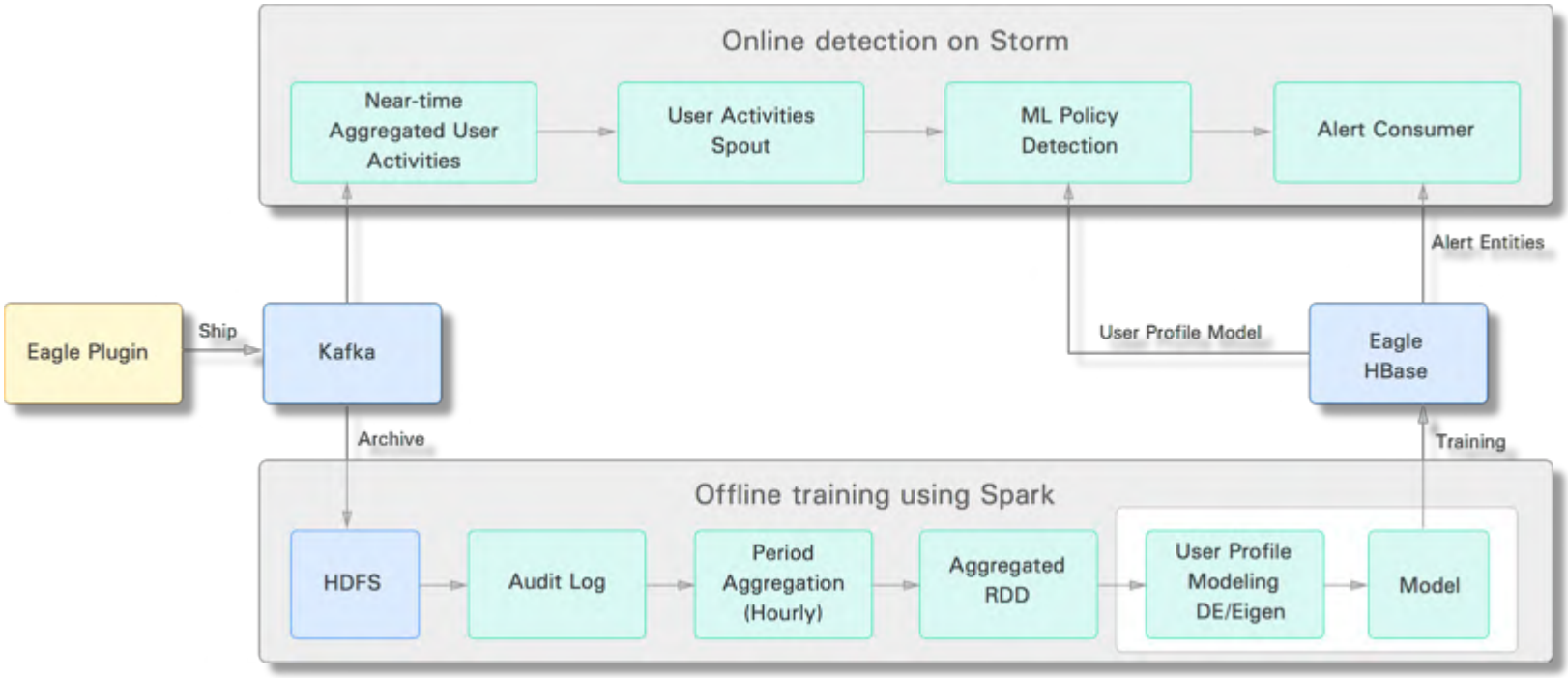
- Entity

  Rowkey ::= Default Prefix | Partition Keys | timestamp | tagName | tagValue | …

- Log

  Rowkey ::= Log Type | Partition Keys | timestamp | tagName | tagValue | …
  Rowvalue ::= Log Content

Eagle - Machine Learning Intergration

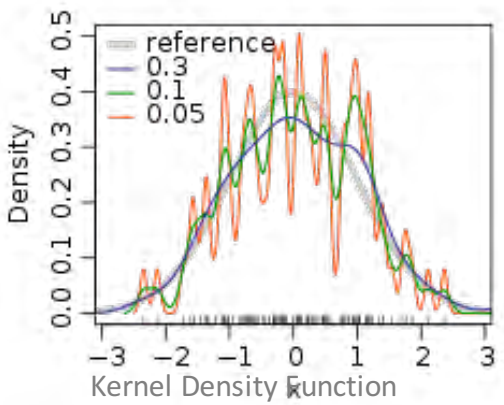


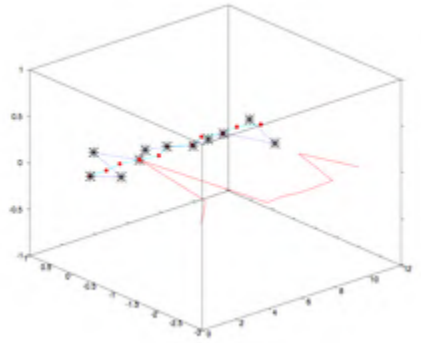
Eagle – User/System Activity Profiling

User Activity Profiling

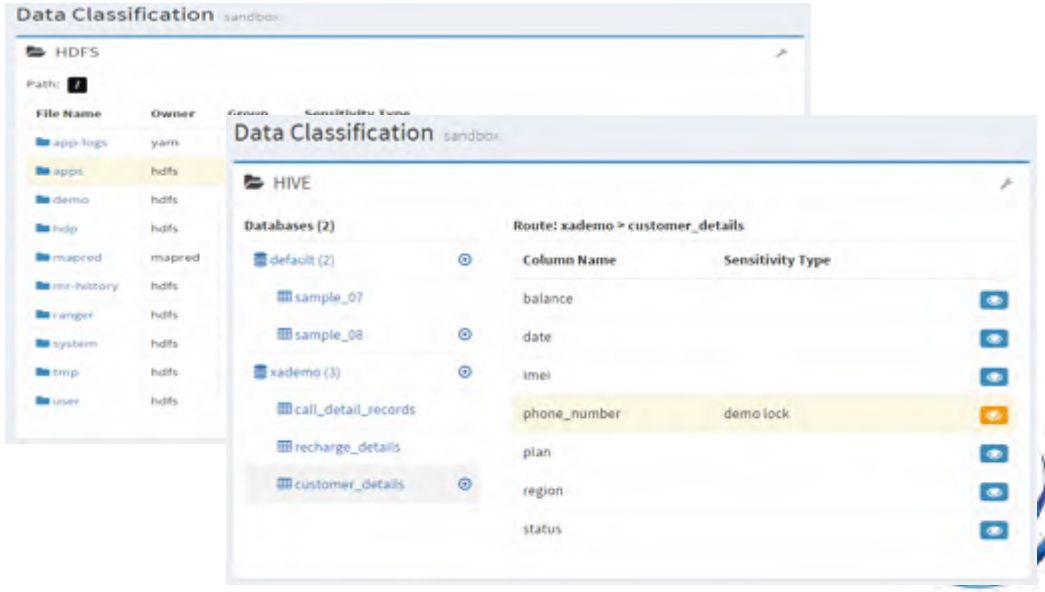
Offline: Determine bandwidth from training dataset the kernel density function parameters (KDE)

Online: If a test data point lies outside the trained bandwidth, it is anomaly (Policy)





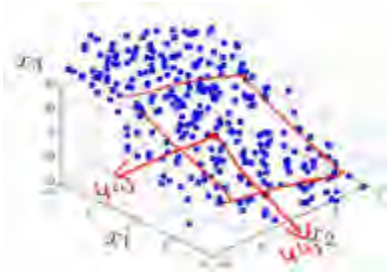
PCs(Principle Components) in EVD (Eigenvalue Value Decomposition)



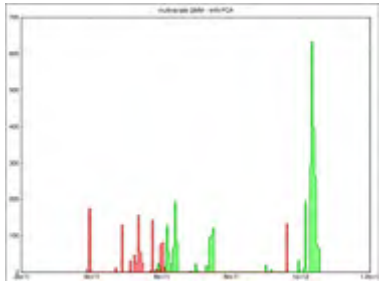
Eagle - Anomaly Metric Predictive Detection

Anomaly Metric Predictive Detection

Offline: *Analyzing and combining 500+ metrics together for causal anomaly detections (IG -> PCA -> GMM -> MCC)*
Online: Predictively alert for anomaly metrics



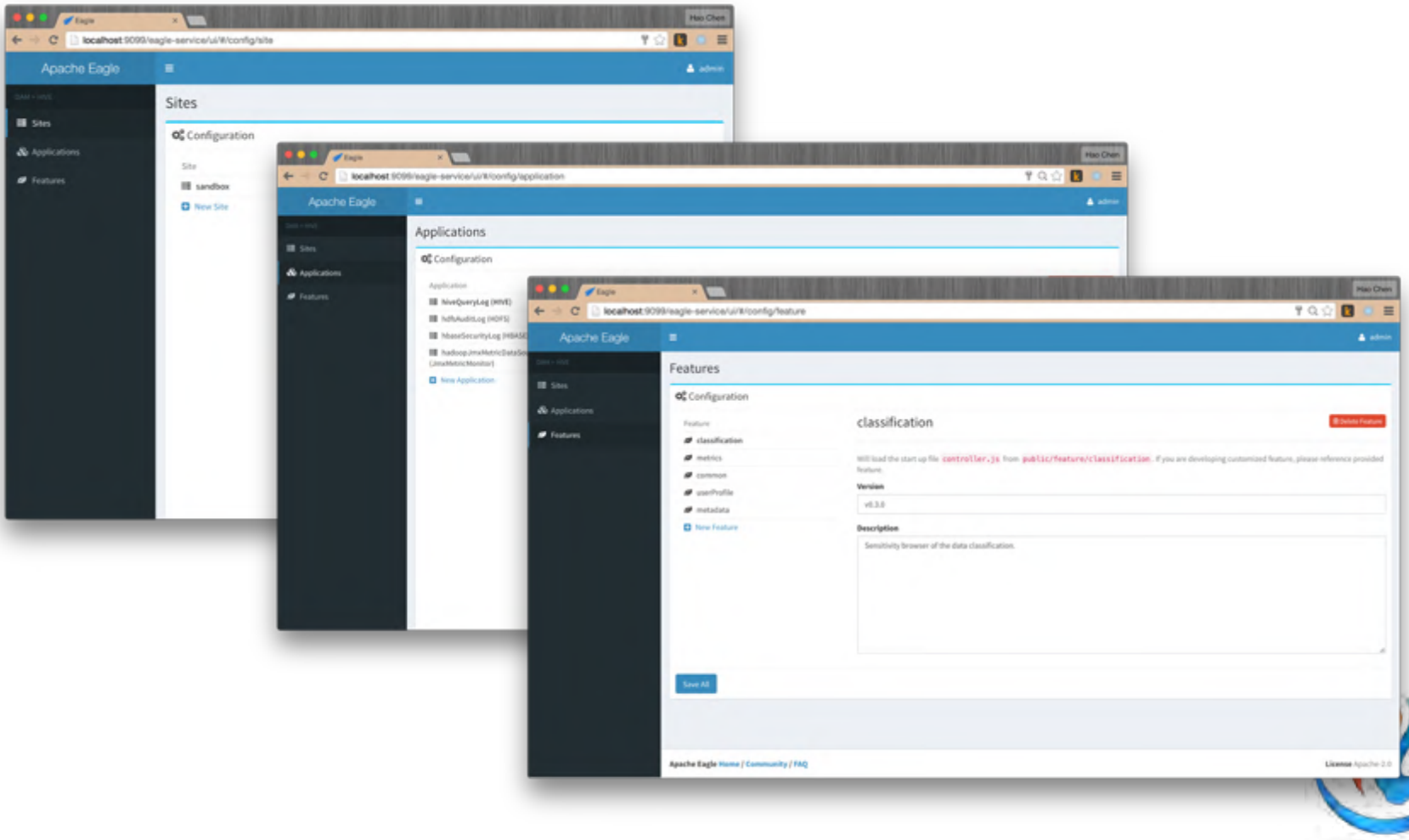
PCA (Principal Component Analysis)



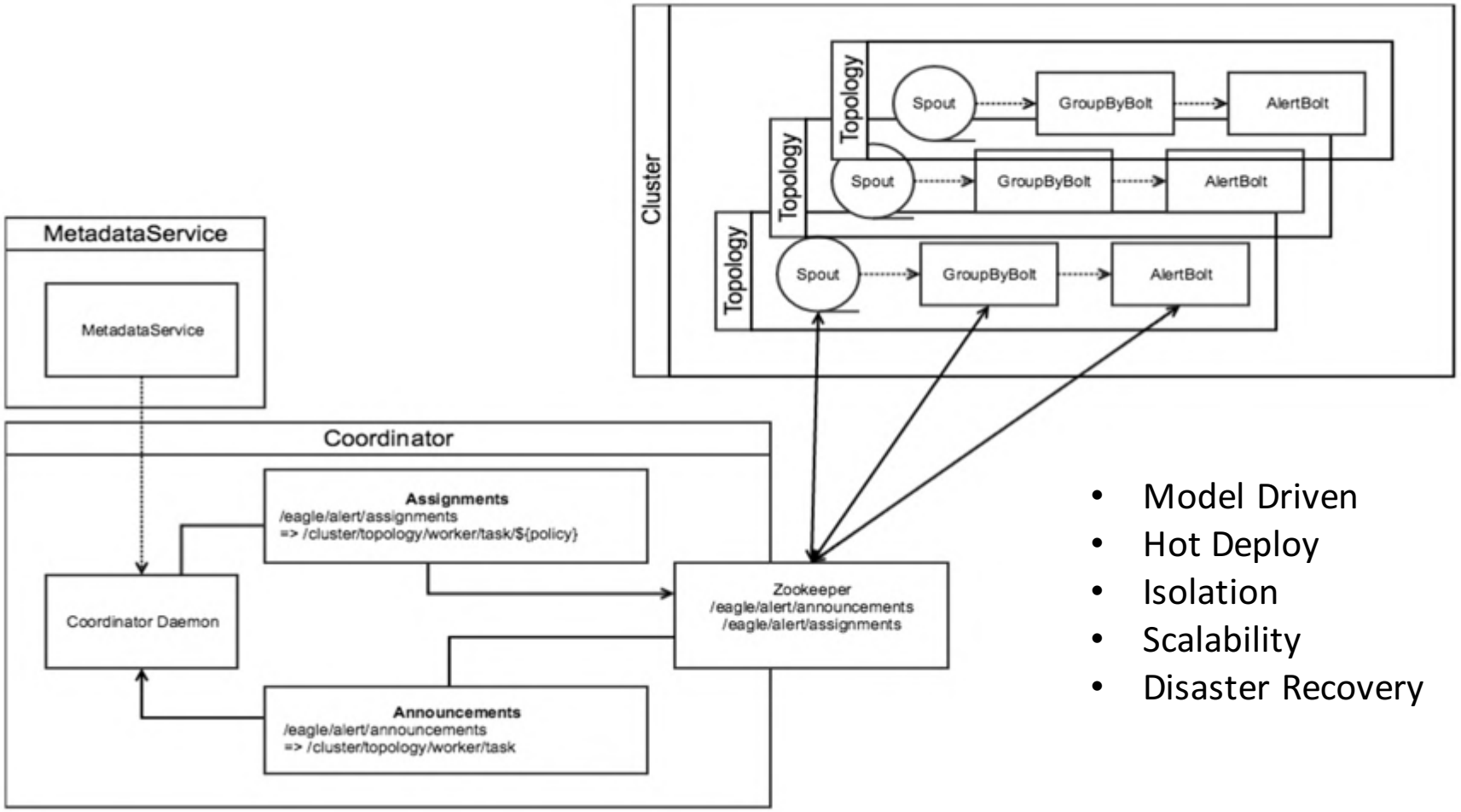
Normal (Green) and Abnormal (Red)
Data and Probability Distribution and Threshold Selection

Anomaly Metric Predictive Detection Case Study

Eagle - Module Management



Eagle – Multi Tenant Support



- Model Driven
- Hot Deploy
- Isolation
- Scalability
- Disaster Recovery

Agenda

- Introduction
- Architecture
- **Ecosystem**
- Q & A

Eagle Ecosystem

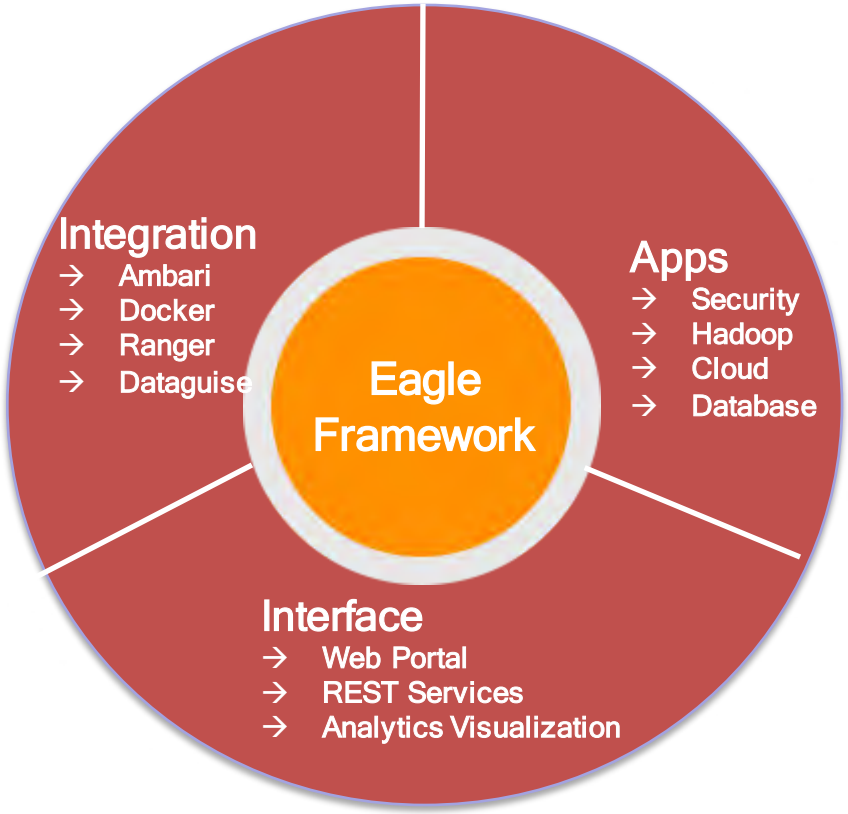
Eagle Framework
Distributed real-time framework for efficiently developing highly scalable monitoring applications

Eagle Apps
Securiy / Hadoop / Cloud / Database

Eagle Interface
REST Service / Management UI / Customizable Analytics Visualization

Eagle Integration
Ambari / Docker / Ranger / Dataguise

Open Source
Community-driven and Cross-community cooperation



Eagle Ecosystem – Open Source

If you want to go fast, go alone.
If you want to go far, go together.

-- African Proverb

Q&A

http://eagle.incubator.apache.org


apache/incubator-eagle


@TheApacheEagle


@ApacheEagle

谢谢