Backwaters: Security Streaming Platform

Comcast TPX Security Solutions Engineering (SSE)
The Team

Chris Maenner
Principal Security Developer

Ryan Van Antwerp
Principal Security Developer

Will Weber
Senior Security Developer
Agenda

• Apache Kafka Overview
• Intelligence Driven Security
• Methods of Receiving Logs
• Architecture of Backwaters: Security Streaming Platform
• How security utilizes Apache Kafka's API
Apache Kafka Overview

Apache Kafka is a distributed streaming platform which has three key capabilities:
- Publish and subscribe to streams of records, similar to a message queue
- Store streams of records in a fault-tolerant durable way
- Process streams of records as they occur

Kafka is generally used for two broad classes of applications:
- Building real-time streaming data pipelines that reliably get data between systems or applications
- Building real-time streaming applications that transform or react to the streams of data

Kafka includes four core data-centric APIs:
- Producer
- Consumer
- Streams
- Connector

Reference: https://kafka.apache.org/documentation/#gettingStarted
Intelligence Driven Security

LOG PRODUCERS
- NETWORK & AUTHENTICATION
- TOOLS & METADATA
- APPLICATION DATA

TRANSFORM LAYER
- BACKWATERS
- KAFKA

LEVEL 1
- THREAT CORRELATION
  - SIEM
    - SECURITY EVENTS CORRELATION AND ALERTING
  - REAL-TIME KNOWN THREAT CORRELATION

LEVEL 2
- ADVANCED THREAT DETECTION
  - TOOLS
    - CLOUD & DATA CENTER INFRASTRUCTURE
    - BEHAVIORAL ANALYTICS, THREAT SIMULATION & DECEPTION

LEVEL 3
- DATA SCIENCE
  - ADVANCED DETECTION
    - NETWORK, TOOLS, INFRASTRUCTURE
    - UNKNOWN THREAT DETECTION
Methods of Receiving Logs

**Comcast Data Centers**
- Kafka Producer
- Syslog Producer

**Amazon Web Services Cloud**
- EC2 Producer
- Lambda Producer

**Microsoft Azure Cloud**
- Azure Functions Producer
- Azure VM Producer
Syslog Ingest Path

Data Producers
- Linux Servers
- Other sources

Load Balancer(s)
- Availability Zones
  - East
  - Central
  - West

Message Bus
- Comcast Private Cloud
  - Elastic Beats
  - Primary
  - Secondary

Consumers
- Security Information and Event Management (SIEM)
- Data Science Tools
- Compliance Tools
- Log Indexing
- Kafka Compatible Tools
AWS Consumer Path

- Comcast Private Cloud
  - Winlogbeat
  - Logstash
- Kafka
- Comcast Managed
  - AWS Direct Connect
- AWS VPC
  - Amazon EC2
- AWS Cloud
  - Amazon S3
AWS Ingest Path

- **AWS Virtual Private Cloud (VPC)**
- **AWS GuardDuty**
- **AWS S3**
- **AWS Lambda**
- **AWS Direct Connect**
- **Comcast Managed**
- **Backwaters**
- **SIEM**
- **Data Science Tools**
- **Elastic Search Cluster**
Backwaters Multi-Tenant Data Framework
Apache Kafka’s API

- The **Producer** API allows an application to publish a stream of records to one or more Kafka topics.

- The **Consumer** API allows an application to subscribe to one or more topics and process the stream of records produced to them.

- The **Streams** API allows an application to act as a *stream processor*, consuming an input stream from one or more topics and producing an output stream to one or more output topics, effectively transforming the input streams to output streams.

- The **Connector** API allows building and running reusable producers or consumers that connect Kafka topics to existing applications or data systems. For example, a connector to a relational database might capture every change to a table.

- The **AdminClient** API allows managing and inspecting topics, brokers, and other Kafka objects.

Reference: [https://kafka.apache.org/documentation/#api](https://kafka.apache.org/documentation/#api)
Apache Kafka Producer/Consumer API

The Producer API:
• Write access to one or more topics
• Allows applications to send streams of data to topic(s)

The Consumer API:
• Read access to one or more topics
• Read streams of data from topic(s)

![Diagram of Apache Kafka Producer/Consumer API](image-url)
Apache Kafka Streams API

- High level abstraction language using Java’s API
- Unbounded, continuous real-time flow of records
  - *You don’t need to explicitly request new records, you just receive them*
- **Domain Specific Language (DSL)** is built on top of the Streams Processor API:
  - Built-in abstractions for streams and tables:
    - **Kstream**: append-only ledger (**INSERT** only)
    - **Ktable**: **UPSER**T changelog stream for one partition
    - **GlobalKTable**: **UPSER**T changelog stream for all partitions
  - Supports stateless and stateful transformations:
    - **Map**: unique keys to values
    - **Filter**: evaluate Boolean to retain or drop elements
    - **Aggregations** (e.g. count, reduce)
    - **Joins** (e.g. Inner, Left, Outer)
    - **Windowing** (e.g. *group records that have the same key*)

Reference: [https://kafka.apache.org/documentation/streams](https://kafka.apache.org/documentation/streams)
Kafka Streams API (Transform)

Kafka Streams app transform object(s) and write to new topic

Raw Topic

```
"2019-01-10 20:20:39"; \
"alice"; \
"Windows"; \
"Desktop"; \
"10.0.0.126"
```

Parsed Topic

```
{
  "timestamp": "2019-01-10 20:20:39",
  "username": "alice",
  "os": "Windows",
  "systemType": "Desktop",
  "ipAddress": "10.0.0.126"
}
```

Consumers

Comcast Cloud

Source Raw Data

Backwaters
Apache Kafka Connect API

• Connectors tool for scalably and reliably streaming data between Kafka and other systems
  • *Kafka Connect is intended to be run as a service*

• Kafka Connect currently supports two modes of execution:
  • **Standalone**: all work is performed in a single process (*Simplest*)
  • **Distributed**: handles automatic balancing of work, allows you to scale up (or down) dynamically

• **Core Concepts and APIs:**
  • Connectors come in two flavors (e.g. Pull or Push):
    • **SourceConnectors**: import data (e.g. JDBCSourceConnector would import relational database)
    • **SinkConnectors**: export data (e.g. HDFSSinkConnector export topic to an HDFS file)
  • Connectors are responsible for breaking jobs into a set of Tasks:
    • **SourceTask**: pull interface with two APIs, `commit` and `commitRecord`
    • **SinkTask**: push interface
  • REST API Layer:
    • View the status/configuration of connectors
    • Alter current behavior (e.g. change config or restart task)

Reference: [https://kafka.apache.org/documentation/#connect](https://kafka.apache.org/documentation/#connect)
Kafka Connect API (SourceConnectors)

Kafka Streams app transform object(s) and write to new topic

<table>
<thead>
<tr>
<th>timestamp</th>
<th>user</th>
<th>os</th>
<th>type</th>
<th>ipAddress</th>
</tr>
</thead>
<tbody>
<tr>
<td>2019-01-10 20:20:39</td>
<td>Alice</td>
<td>OSX</td>
<td>Desktop</td>
<td>10.0.0.126</td>
</tr>
</tbody>
</table>

Parsed Topic

```
{
   "timestamp": "2019-01-10 20:20:39",
   "user": "alice",
   "os": "Windows",
   "type": "Desktop",
   "ipAddress": "10.0.0.126"
}
```

Kafka Connect app performing JDBC connection to database

Backwaters

Consumers

MSSQL

JDBC

JAR

Comcast Cloud
Questions?