Stratio Khermes
A distributed data generator

Apache Big Data North America
May 2017
INDEX

1. Who are we?
2. Why & What?
3. How?
4. Use cases
5. Roadmap
6. Demo
ALBERTO RODRÍGUEZ DE LEMA

After graduating I've been programming for more than 13 years. I've built high performance and scalable web applications for companies such as Prudential Financial, Springer Verlag Ltd and Stratio.

@ardlema
EMILIO AMBROSIO CORDERO

During the last 10 years, as Scala/Java developer, I’ve been involved in the design and development of solutions for different Spanish Banks innovative projects, using cutting edge technologies such as Kafka, Spark, Akka,...

@_eambrosioc
Stratio accompanies businesses on their journey through complete Digital Transformation. Through hard work and creativity, our ambition is to reinvent companies around their data so they can compete in a changed world.

Our solution is a transformational product that uses third generation Big Data technologies to execute the most comprehensive form of Digital Transformation, ensuring scalability, maximum flexibility and adaptation to new markets.

If you have any questions or would like a demonstration of our product, get in touch: contact@stratio.com
Why & What?

What is Stratio Khermes & why do we need it in Stratio?
Hermes is considered the god of transitions and boundaries. He is also portrayed as an emissary and messenger of the gods.
“What about the “K”?"
A data generator tool has been a must throughout the Stratio’s life:

- Produce data to run our well-known retail demo
- Ingest specific data for clients’ projects
- Measure the performance of specific components on a heavy-load environment

Some of our stakeholders questions about our platform:

“Have you guys stress this stuff?”
“How stable is your platform under heavy-load?”
So far we’ve been using Stratio fakenator which has the following limitations:

- Data generated was very coupled to a retail demo
- Every time we needed a change we need to modify the project code
- It is not ready to be executed in a distributed fashion in order to increase the load and the data to be ingested
Taking these facts into account we have created a distributed data generator called Khermes with the following main goals:

- Easy-to-generate user-defined dynamic data
- Broad catalogue of data generators
- Designed to run within a distributed environment
- Easily scalable (through Marathon)
- API to configure and modify its behaviour on-the-fly
- Friendly UI (roadmap)
How?

Architecture & Technical decisions
How

Data templates

The data generated will be defined by the user on-the-fly through templates

```scala
@import com.stratio.hermes.utils.Hermes
@import com.stratio.hermes.utils.generators.Positive
@import scala.util.Random
@import org.joda.time.DateTime

@(hermes: Hermes)
@defining(hermes.Music.playedSong, hermes.Geo.geolocation) { case (randomSong, randomGeo) =>
{
  "song": "@{randomSong.song}",
  "artist": "@{randomSong.artist}"",
  "genre": "@{randomSong.genre}"",
  "album": "@{randomSong.album}"",
  "playDuration": @{hermes.Number.number(3, Positive)},
  "rating": @{hermes.Number.rating(5)},
  "user": "@{hermes.Name.fullName}"",
  "userType": "@{Seq("free", "membership")}(Random.nextInt(2))",
  "country": "@{hermes.Geo.country}"",
  "city": "@{randomGeo.city}"",
  "latitude": @{randomGeo.latitude},
  "longitude": @{randomGeo.longitude},
  "startTime": "@${Random.nextInt(24)}:${Random.nextInt(60)}:${Random.nextInt(60)}:${Random.nextInt(1000)}""
}
}
```
Akka cluster

- Fault-tolerant & decentralized peer-to-peer cluster
- No single point of failure/bottleneck
- Automatic failure detection
“Kafka has great throughput, built-in partitioning, replication, and fault-tolerance which makes it a good solution for large scale message processing applications”.
How

Apache Kafka Connect

KAFKA CONNECT

Data Source → Kafka Connect → Kafka → Kafka Connect → Data Sink
To infinity, and beyond!

**Legend:**
- S1: Seed
- N1 ... N4: Nodes
- C1: Client

**Twirl Template**

```scala
@(khermes : khermes)
{
  "customerId": @(khermes.Number.number(3, Positive)),
  "customerName": "@(khermes.Name.fullName)",
  "latitude": @(khermes.Geo.geolocation.latitude),
  "longitude": @(khermes.Geo.geolocation.longitude),
  "productIds": 
      (!((1 to 5).map(x => 
        khermes.Number.number(1, Positive)).mkString(""))}
}
## Getting started

<table>
<thead>
<tr>
<th>Standalone</th>
<th>Docker</th>
<th>DC/OS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Clone the project</td>
<td>1. Clone the project</td>
<td>1. Add the app to your universe</td>
</tr>
<tr>
<td>2. mvn clean install</td>
<td>2. docker build ...</td>
<td>2. Deploy it using DCOS UI</td>
</tr>
<tr>
<td>3. Start zookeeper</td>
<td>3. docker-compose...</td>
<td></td>
</tr>
<tr>
<td>4. Start kafka</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5. Start khermes: java -jar ...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Interacting with the cluster

1. Create kafka-config
2. Create twirl-template
3. Create generator-config
4. Create avro-config (optional)
5. Start producing data!
Use cases
Use Cases

Demo data producer

SOURCES  INGESTION  BATCH PROCESSING  STORAGE  DATA VISUALIZATION

Stratio
KHermes

kafka

Spark

Postgres

stratio
viewer

stratio
intelligence

stratio
crossdata
Projects’ Day 0 data ingestion
Use Cases

Stress tool + monitoring

© Stratio 2016. Confidential, All Rights Reserved.
Current features & Roadmap
Features & Roadmap

Current status & features

- ✔ Akka cluster
- ✔ Websockets communication
- ✔ 5 data generators: numbers, names, geolocation, timestamp, music generation!
- ✔ Stratio EOS integration
- ✔ Web console and simple cluster status interface
Features & Roadmap

Features to come

- More & more data generators
- “Weighted” generators (WIP)
- Improve UI (Kafka & Connect status, throughput, config…)
- Get rid of zookeeper dependency to store configs
- Discover schemas and ingest data from datastores!!
Demo time!

Let’s see how Khermes works