Scalable Monitoring
with
Apache Spark & friends

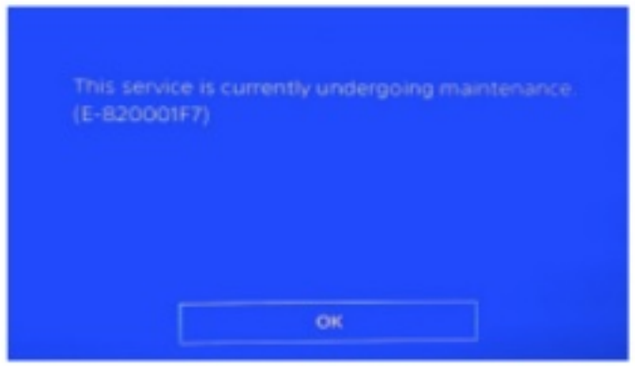
- Senior Software Engineer @ **Tinder**.
- Built monitoring pipeline at **Sony PlayStation**.
- An active contributor to **Grafana.**
- Speaker @ **GrafanaCon 2016.**

GitHub - **https://github.com/utkarshcmu**
Email – **utkarsh.cmu@gmail.com**

Monitoring

@

PlayStation Outage!



Problems at Playstation Network

Published: 10/09/2015 3:22 p.m. **By:** downdetector.com

Playstation Network is having issues since 3:22 PM EDT. Are you also affected? Leave a message in the comments.

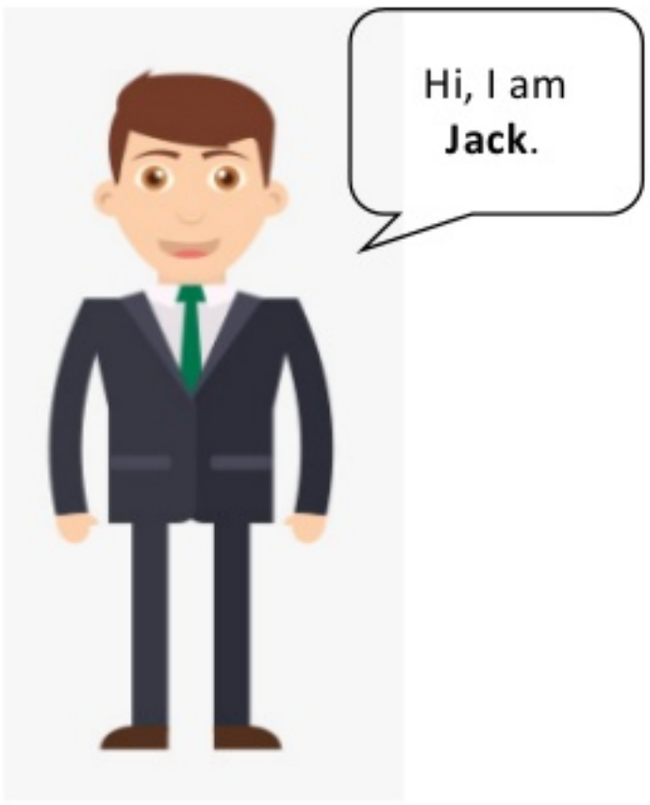
Most reported problems:

- Sign-in (67%)
- Game play (16%)
- Playstation Store (16%)

PLAYSTATION.Network

Oct. 9, 2015 Status overview

Sometime 3 years back...



POC on Monitoring

Requirements:

- 50,000 unique metrics from one source
- Data points every minute
- Roughly about **72 million data points per day**
- Data retention 60 days
- User friendly UI with possible customization

Monitoring Stack

Choosing the technology!

graphite

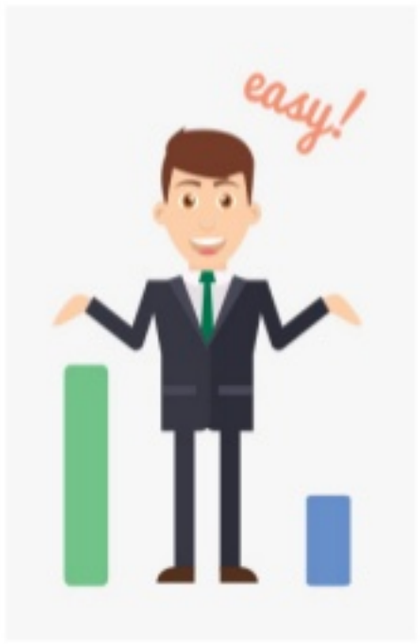
OPENTSDB

Grafana

InfluxDB

POC
Design & Architecture

POC Completed!



Mission accomplished!

1 metrics source
50,000 unique metrics
72 million data points per day

Metrics Onboarding

Team 1 Requirements:

- 100,000 unique metrics
- **About 200 million data points per day**

Team 2 Requirements:

- 400,000 unique metrics
- **About 600 million data points per day**

Team 3 Requirements:

- 500,000 unique metrics
- **About 2 billion data points per day**

Team 4 Requirements:

- 800,000 unique metrics
- **About 5 billion data points per day**

And more……..

POC
Design & Architecture

How to Scale?

Challenges:

- Multiple teams
- Millions of unique metrics
- Above 10 billion data points a day
- Process 3 million logs every minute and generate metrics
- Reprocessing of metrics and logs if needed
- **Provide real time monitoring for all of the above using GRAFANA!**

Should he continue with Graphite?
Should he ask to reduce metrics or datapoints?
How to dynamically scale Graphite?
Does Grafana support other datasources?
OpenTSDB / InfluxDB / KairosDB / Prometheus?
Support scaling Infrastructure to support variable load of metrics?

Strategy

Team 1 Requirements:

- 100,000 unique metrics
- **About 200 million data points per day**

Team 2 Requirements:

- 500,000 unique metrics
- **About 2 billion data points per day**

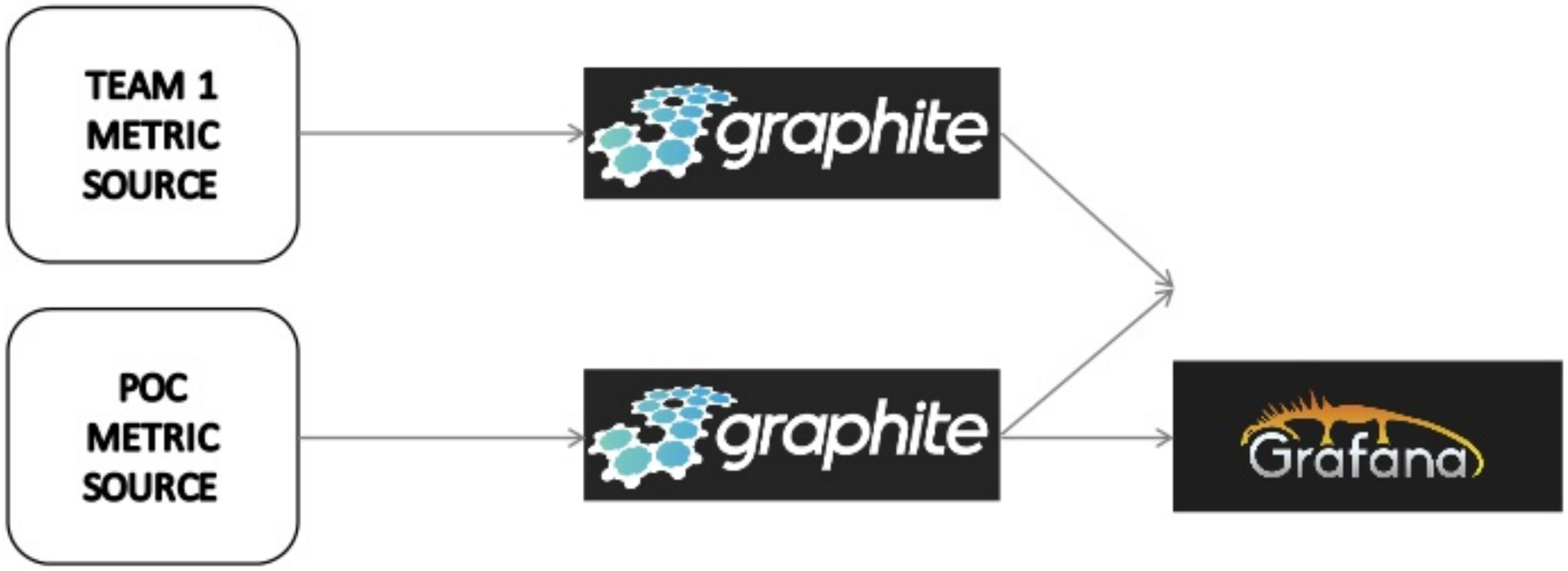
Team 3 Requirements:

- 3 million logs a minute
- **Generate metrics in real time**

And more……..

Divide & Conquer

Design & Architecture



POC works for:Team 1 requirements:

1 metrics source	1 metrics source
50,000 unique metrics	100,000 unique metrics
72 million data points per day	200 million data points per day

Team 1 Conquered!

This strategy works! Bring it on!

Strategy

Team 1 Requirements:

- 100,000 unique metrics
- **About 200 million data points per day**

Team 2 Requirements:

- 500,000 unique metrics
- **About 2 billion data points per day**

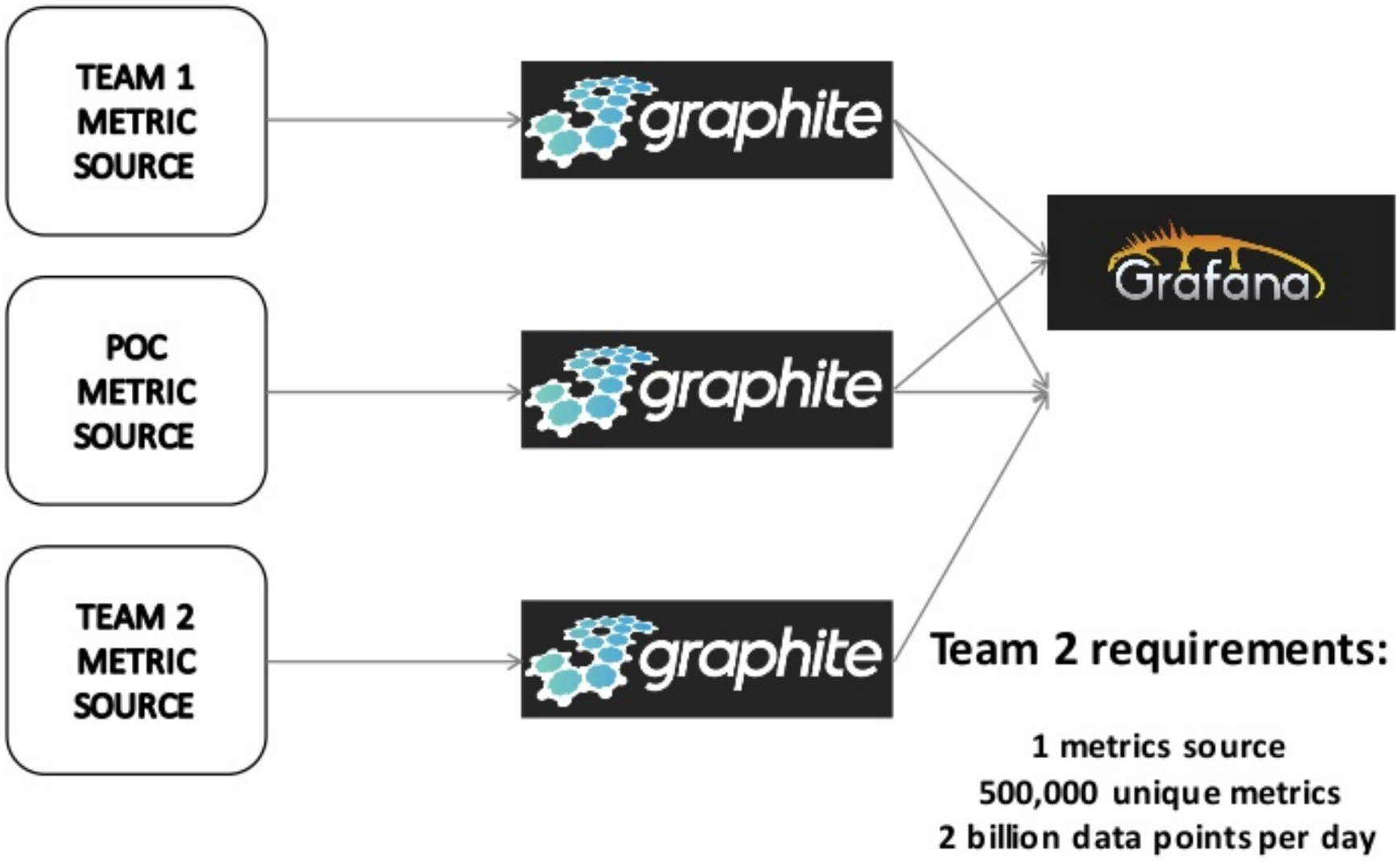
Team 3 Requirements:

- 3 million logs a minute
- **Generate metrics in real time**

And more……..

Divide & Conquer

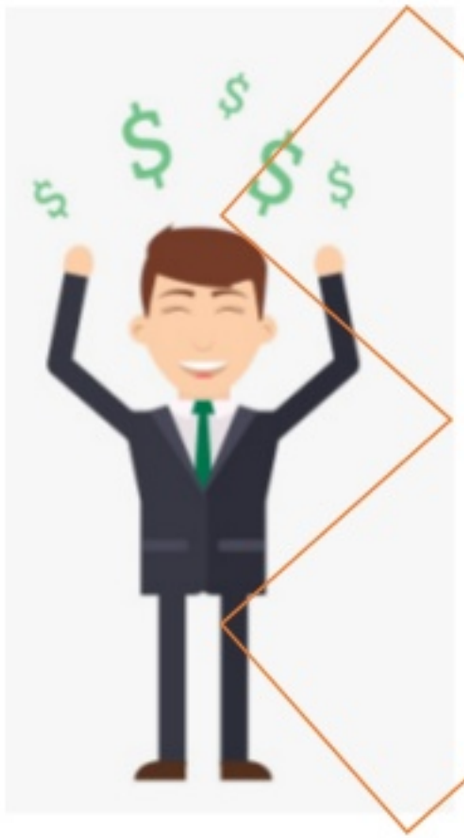
Design & Architecture



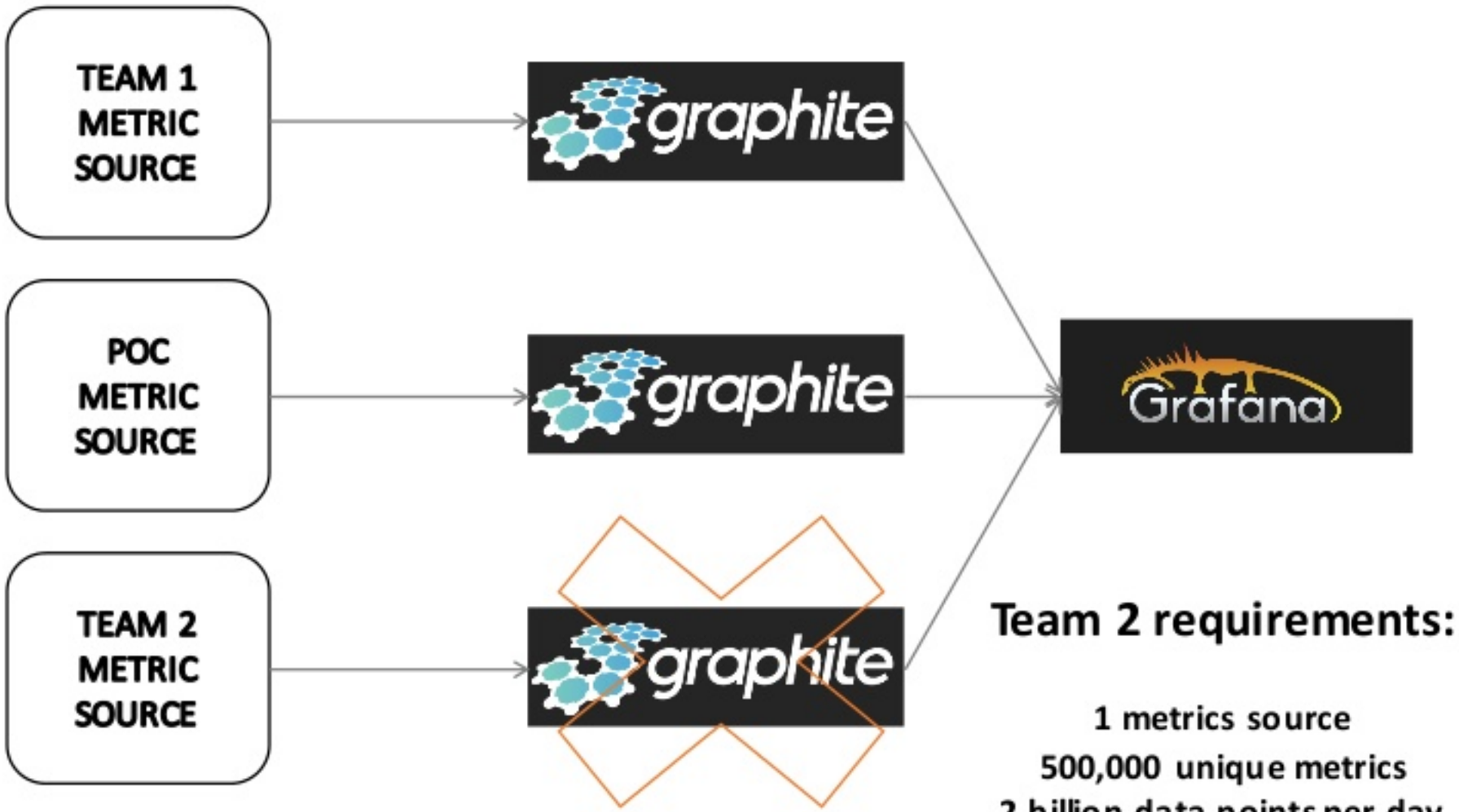
Team 2 requirements:

1 metrics source
500,000 unique metrics
2 billion data points per day

Team 2 Conquered!



Design & Architecture

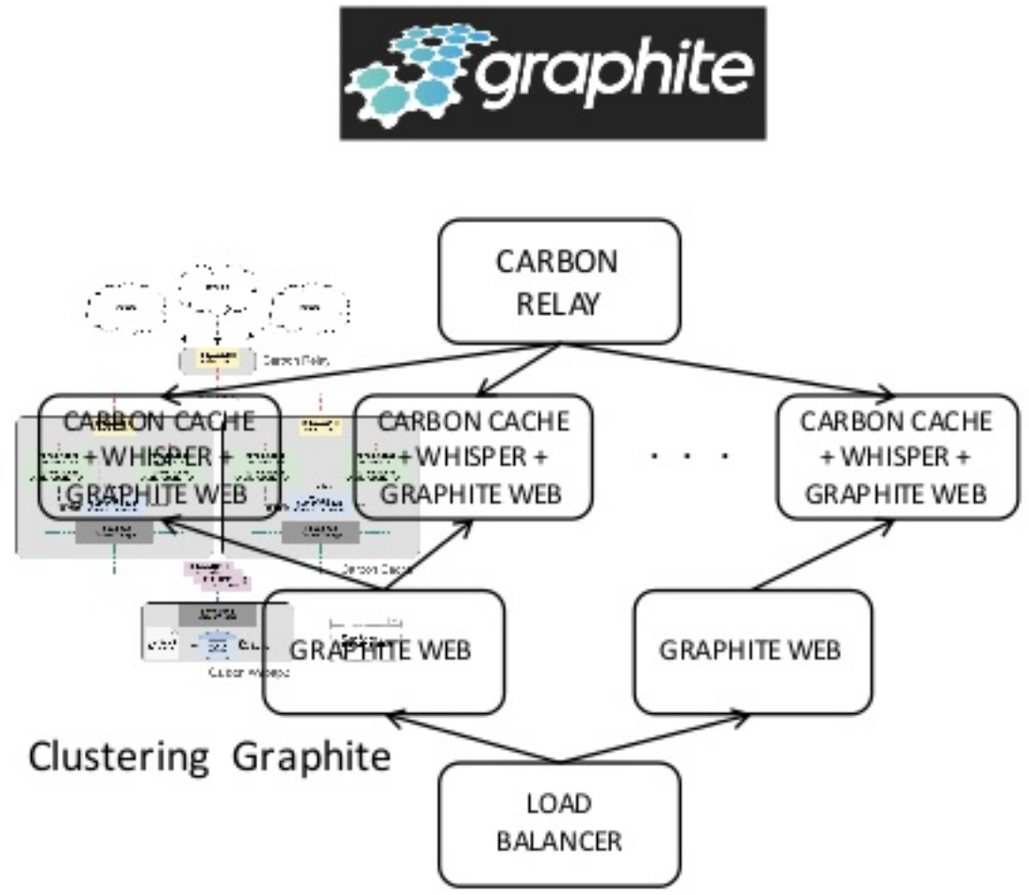


Team 2 requirements:

1 metrics source
500,000 unique metrics
2 billion data points per day

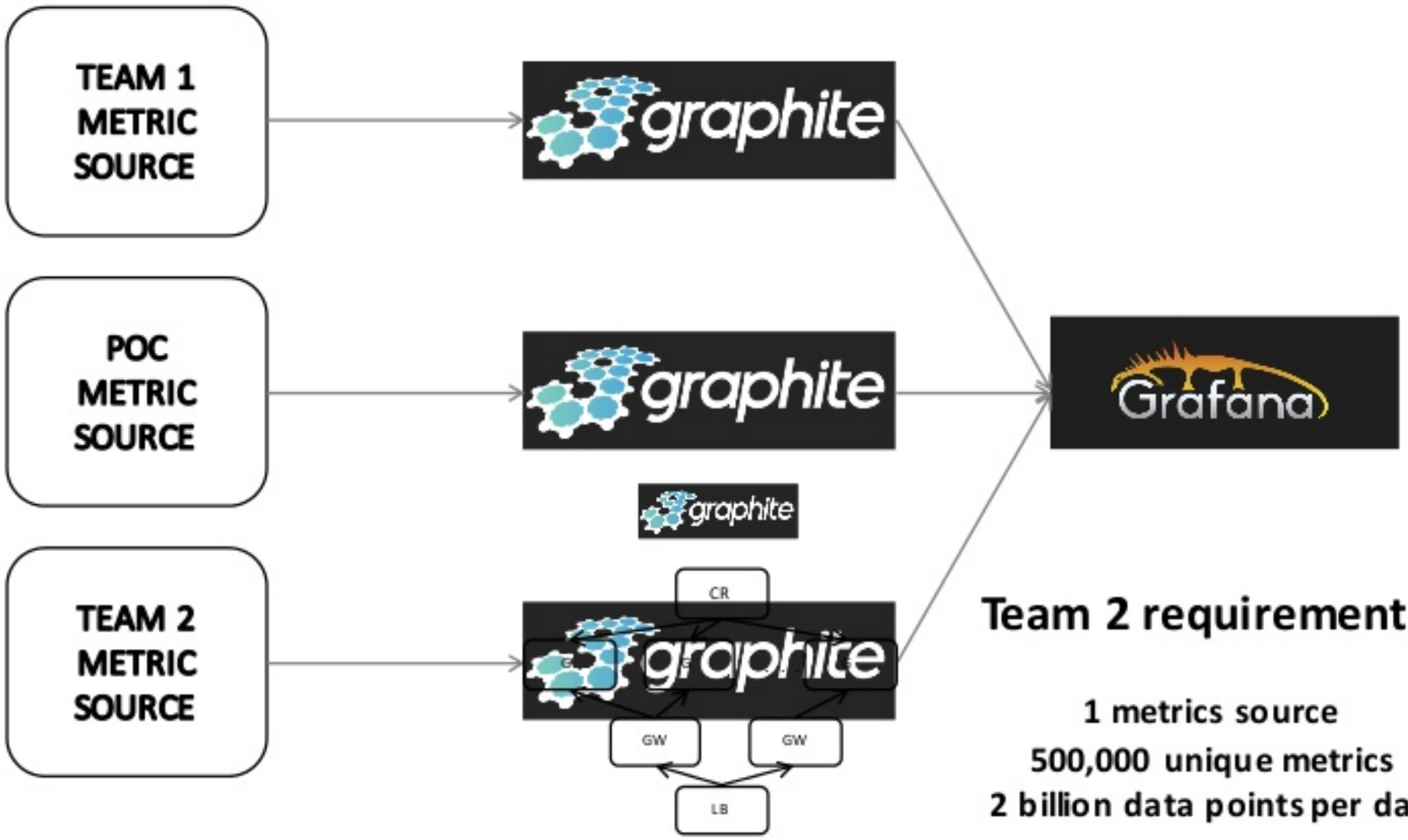

I approve!
NEVER
NEVER
GIVE UP

Scaling Graphite



Clustering Graphite

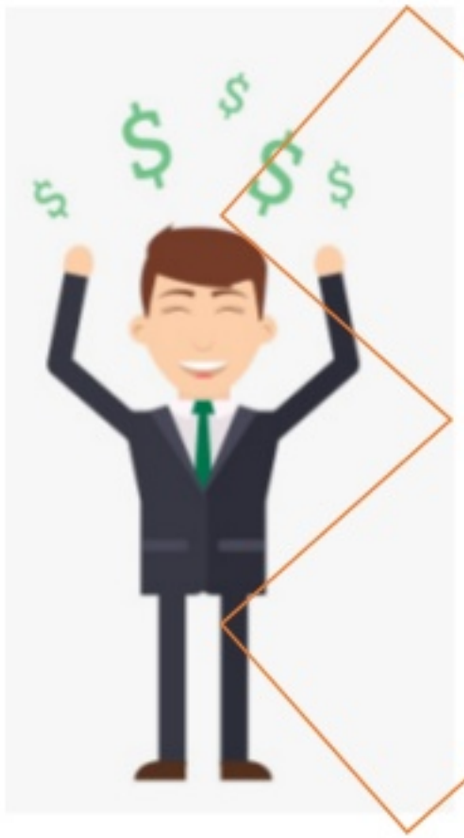
Design & Architecture



Team 2 requirements:

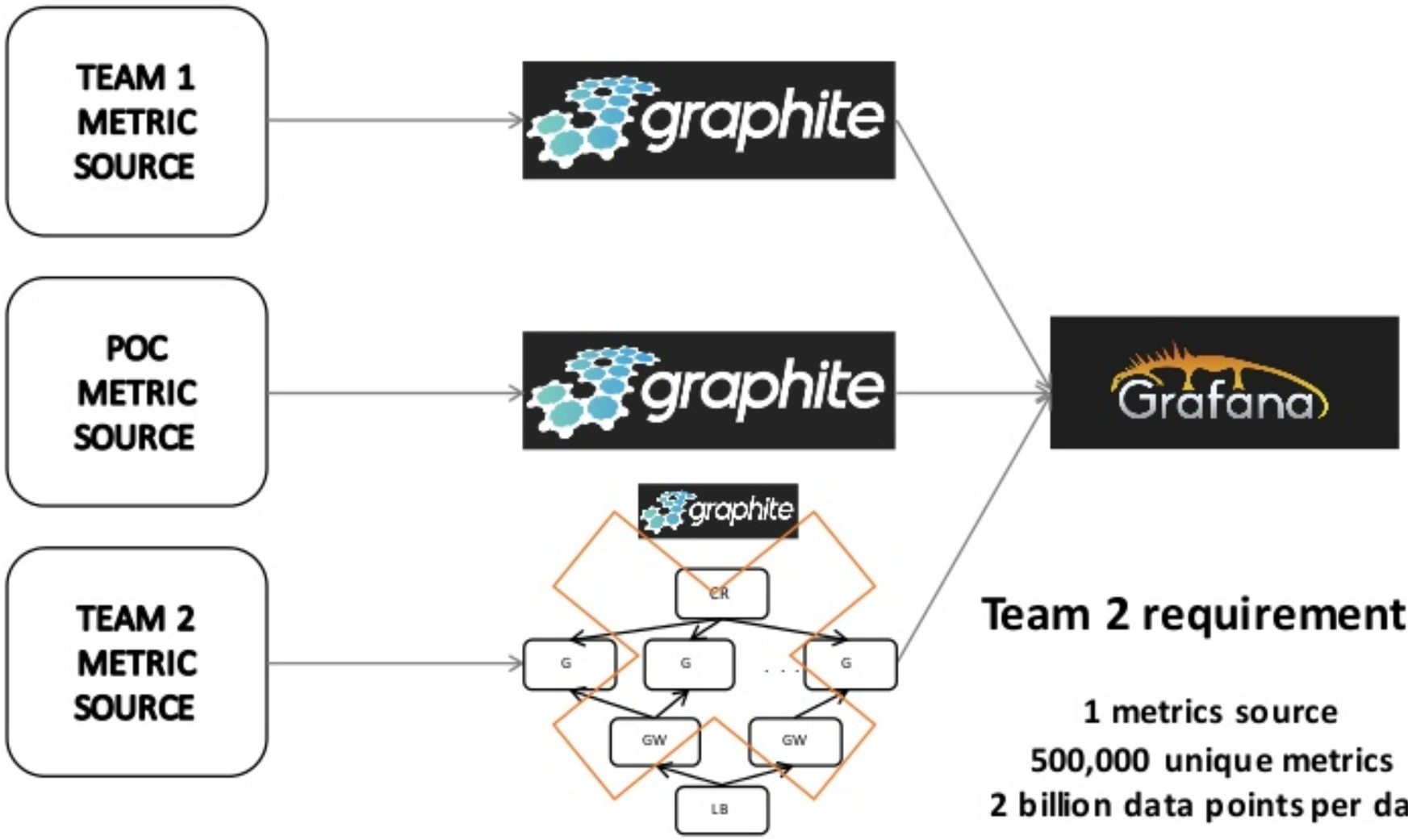
1 metrics source
500,000 unique metrics
2 billion data points per day

Team 2 Conquered!



But……. Happiness lasted only for a month ☹

Design & Architecture

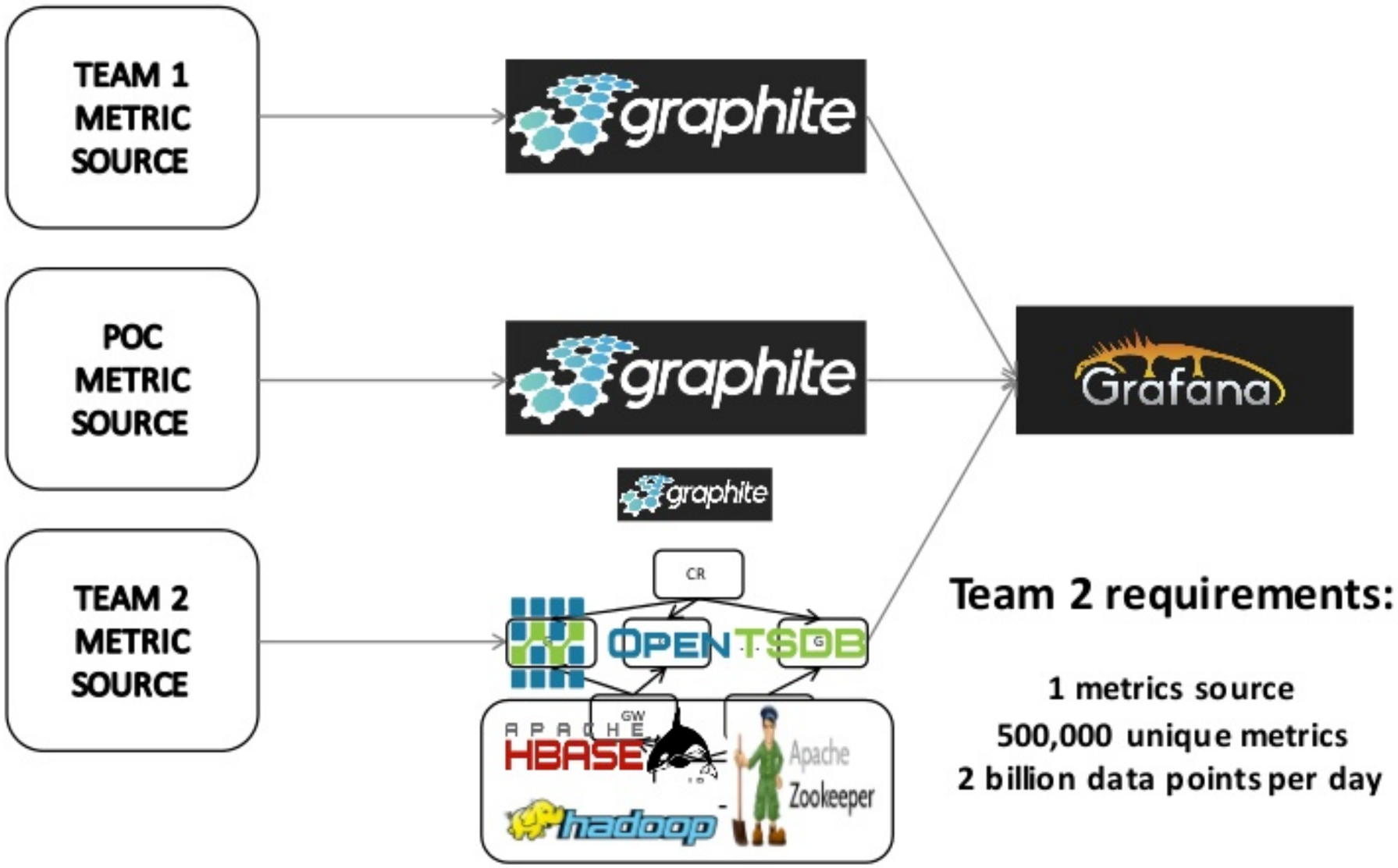


Team 2 requirements:

1 metrics source
500,000 unique metrics
2 billion data points per day

Scalable Alternatives To Graphite

Design & Architecture



Team 2 requirements:

1 metrics source
500,000 unique metrics
2 billion data points per day

Team 2 Conquered!

Finally!

Strategy

Team 1 Requirements:

- 100,000 unique metrics
- **About 200 million data points per day**

Team 2 Requirements:

- 500,000 unique metrics
- **About 2 billion data points per day**

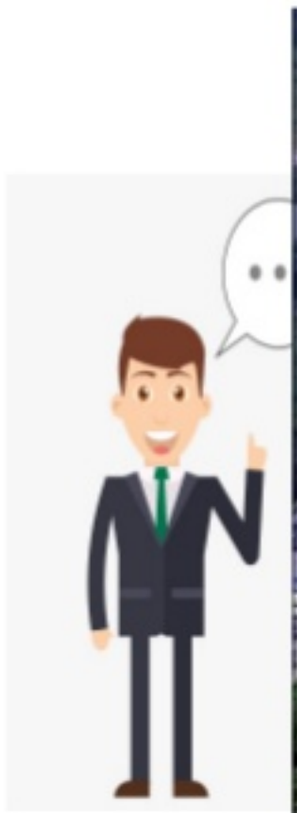
Team 3 Requirements:

- 3 million logs a minute
- **Generate metrics in real time**

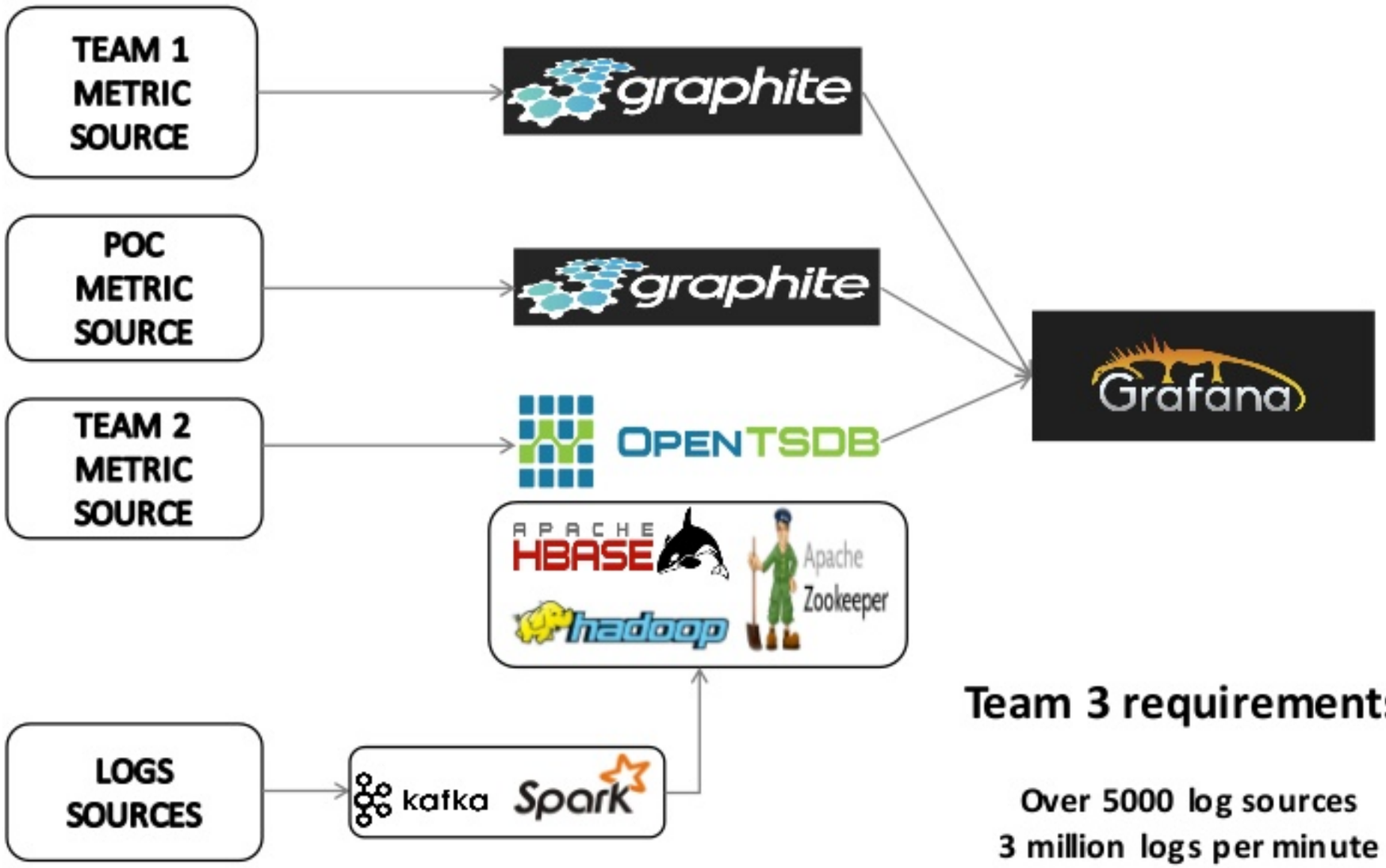
And more……..

Divide & Conquer

How to process logs at scale?



Design & Architecture



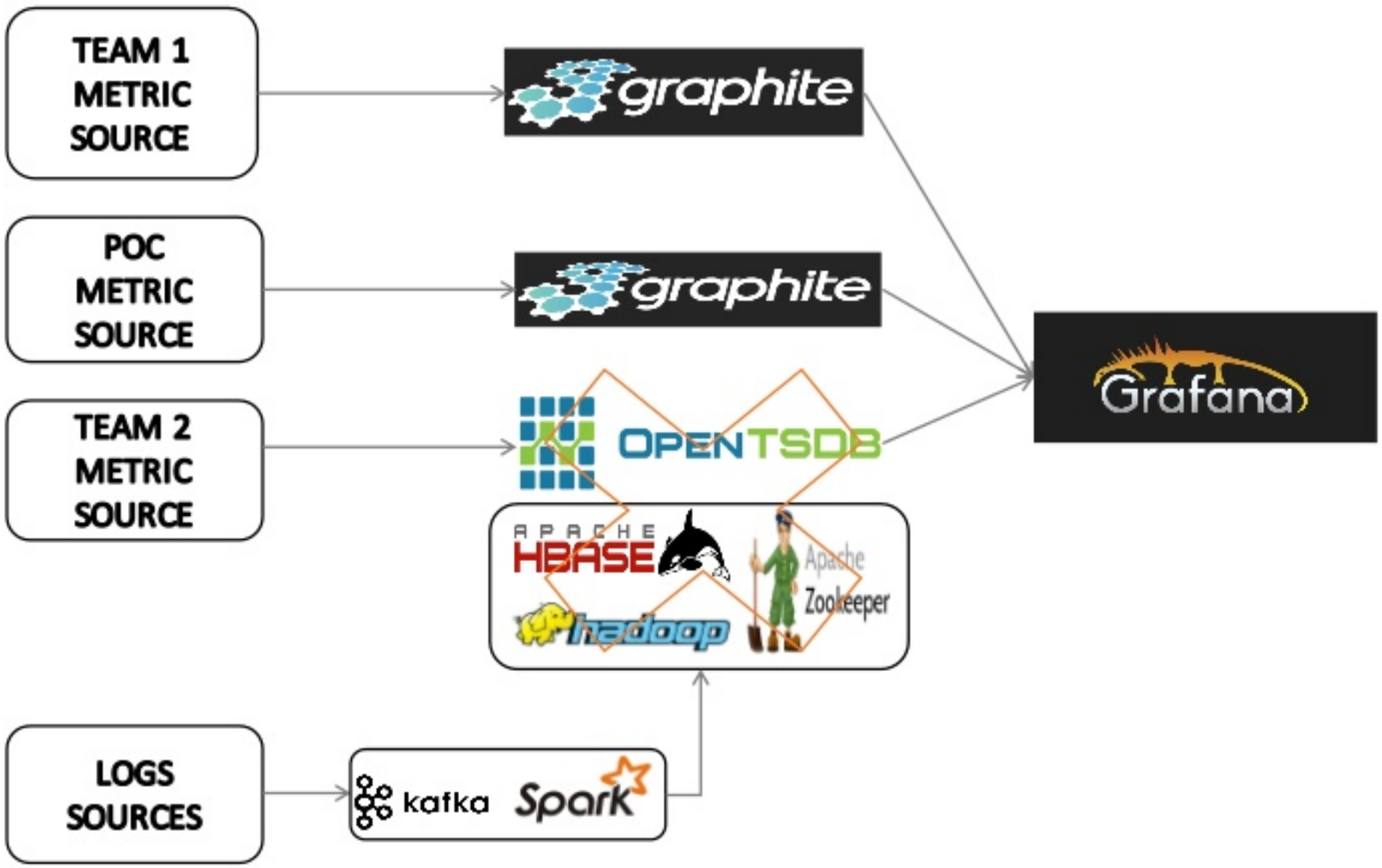
Team 3 requirements:

Over 5000 log sources
3 million logs per minute

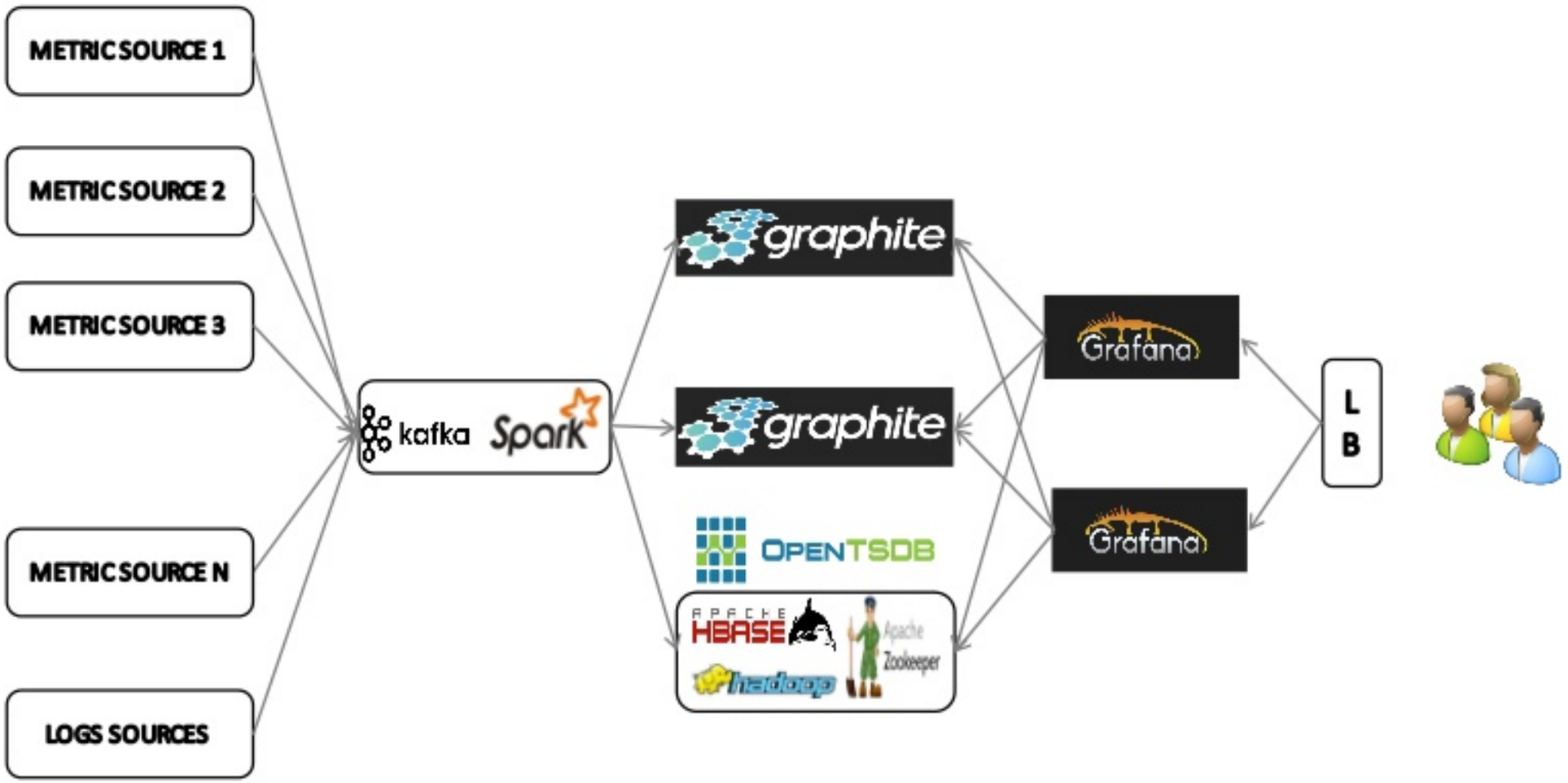
Team 3 Conquered!

But .... One day..

Design & Architecture



Design & Architecture



Metrics & Logs Sources

amazon web services CloudWatch

collectd

Apache Tomcat

jmxtrans

logstash

APACHE HTTP SERVER

beats

sensu

Nagios®

Too many sources!

All different formats!

More Spark Streaming jobs!

Writing a Spark Streaming job!

Spark Documentation

```
public final class JavaDirectKafkaWordCount {
  private static final Pattern SPACE = Pattern.compile(" ");

  public static void main(String[] args) throws Exception {
    if (args.length < 2) {
      System.err.println("Usage: JavaDirectKafkaWordCount <brokers> <topics>\n" +
          "  <brokers> is a list of one or more Kafka brokers\n" +
          "  <topics> is a list of one or more kafka topics to consume from\n\n");
      System.exit(1);
    }

    StreamingExamples.setStreamingLogLevels();

    String brokers = args[0];
    String topics = args[1];

    // Create context with a 2 seconds batch interval
    SparkConf sparkConf = new SparkConf().setAppName("JavaDirectKafkaWordCount");
    JavaStreamingContext jssc = new JavaStreamingContext(sparkConf, Durations.seconds

    Set<String> topicsSet = new HashSet<>(Arrays.asList(topics.split(",")));
    Map<String, String> kafkaParams = new HashMap<>();
    kafkaParams.put("metadata.broker.list", brokers);

    // Create direct kafka stream with brokers and topics
    JavaPairInputDStream<String, String> messages = KafkaUtils.createDirectStream(
        jssc,
        String.class,
        String.class,
        StringDecoder.class,
        StringDecoder.class,
        kafkaParams,
        topicsSet
    );

    // Get the lines, split them into words, count the words and print
    JavaDStream<String> lines = messages.map(Tuple2::_2);
    JavaDStream<String> words = lines.flatMap(x -> Arrays.asList(SPACE.split(x)).iter
    JavaPairDStream<String, Integer> wordCounts = words.mapToPair(s -> new Tuple2<>(s
        .reduceByKey((i1, i2) -> i1 + i2);
    wordCounts.print();

    // Start the computation
    jssc.start();
    jssc.awaitTermination();
  }
}
```

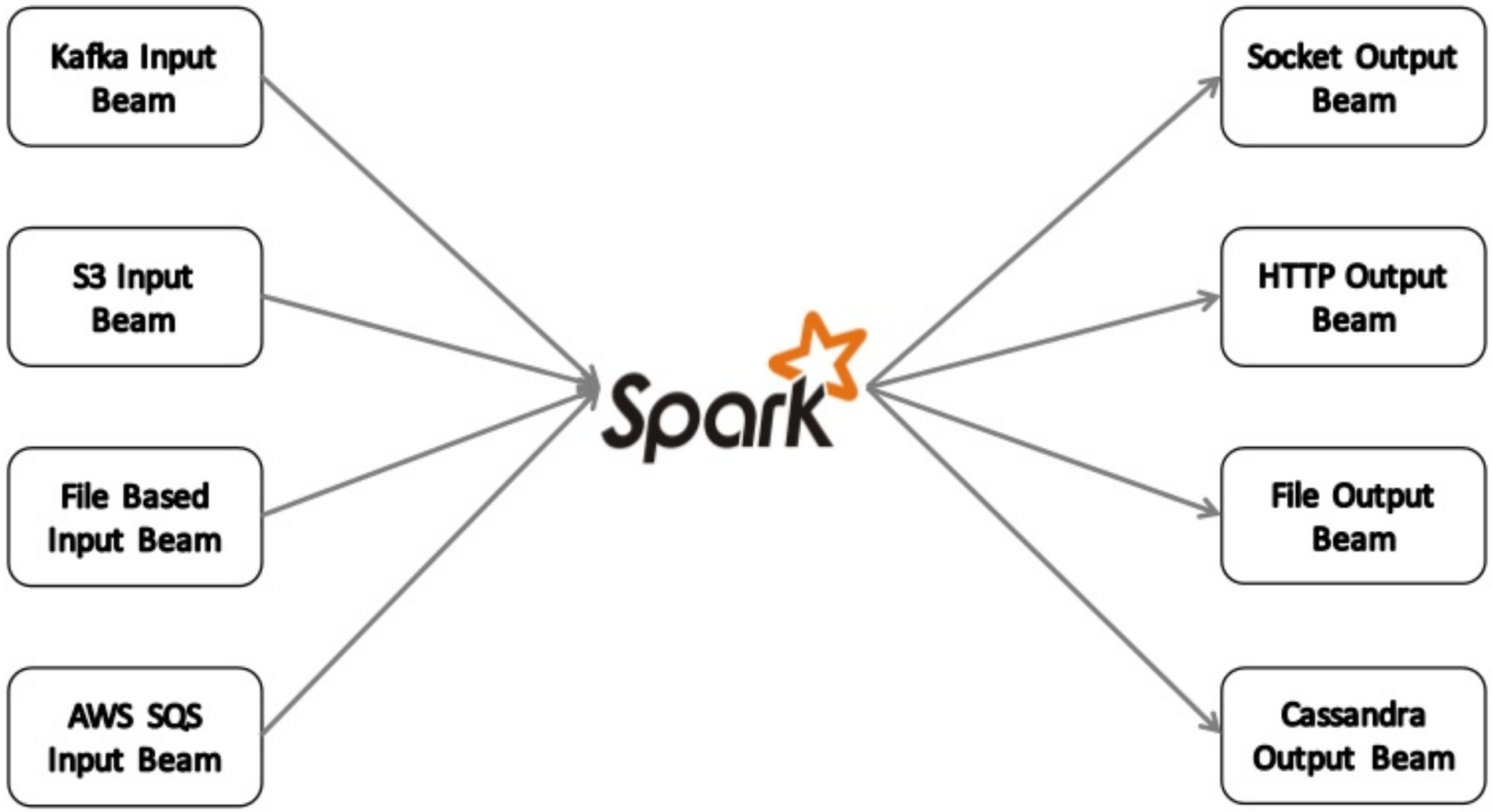
Lazer – Wrapper Library for Streaming

```
public class JavaDirectKafkaWordCount {

        private static final Pattern SPACE = Pattern.compile(" ");

        public static void main(String args[]) {
                Lazer lazer = new Lazer("exampleBeans/events.json");
                InputBean[] inputBeans = lazer.getInputBeans();
                JavaDStream<String> lines = inputBeans[0].read();
                JavaDStream<String> words = lines.flatMap(x -> Arrays.asList(SPACE.split(x)).iterator());
            JavaPairDStream<String, Integer> wordCount = words.mapToPair(s -> new Tuple2<>(s, 1)).reduceByKey((i1
            wordCount.print();
                lazer.startAndAwaitTermination();
        }
}
```

Lazer – Wrapper library for Streaming



Lazer – Wrapper library for Streaming

events.json

```
{
  "beam.settings": {
    "name": "KafkaLogCountExample",
    "duration": 10
  },
  "spark.settings": {
    "spark.streaming.backpressure.enabled": true,
    "spark.executor.memory": "4g"
  },
  "input.settings": [
    {
      "class": "io.github.utkarshcmu.input.KafkaInputBeam",
      "config": {
        "topics": [
          "topicA",
          "topicB"
        ],
        "params": {
          "metadata.broker.list": "kafka:9092",
          "group.id": "consumerGroupName",
          "auto.offset.reset": "lastest"
        }
      }
    }
  ],
  "output.settings": [
    {
      "class": "io.github.utkarshcmu.output.SocketOutputBeam",
      "config": {
        "host": "graphite",
        "port": 2003
      }
    }
  ]
}
```

```
public class JavaDirectKafkaWordCount {

        private static final Pattern SPACE = Pattern.compile(" ");

        public static void main(String args[]) {
                Lazer laser = new Lazer("exampleBeams/events.json");
                InputBeam[] inputBeams = laser.getInputBeams();
                JavaDStream<String> lines = inputBeams[0].read();
                JavaDStream<String> words = lines.flatMap(x -> Arrays.asList(SPACE.split(x)).iterator());
            JavaPairDStream<String, Integer> wordCount = words.mapToPair(s -> new Tuple2<>(s, 1)).reduceByKey((i1
            wordCount.print();
                laser.startAndAwaitTermination();
        }

}
```

Lazer – Open sourced now!

https://github.com/utkarshcmu/lazer

Some numbers

- More than 3 million unique metrics supported
  - creation and deletion happens all the time

- More than 11 billion data points written per day
  - across all TSDBs

- Processing about 40 billion events per day
  - logs and metrics events in near real time (within 30 seconds)

- More than 3000 requests per minute to Grafana dashboards
  - around 7000 requests in during outages

Lessons Learned

Strategy

Divide & Conquer

Look for alternatives!

graphite

OPENTSDB

Grafana

InfluxDB

Choose scalable components!

kafka Spark

OPENTSDB

APACHE HBASE

Apache Zookeeper

hadoop

Grafana

graphite

(Subject to effort and time)

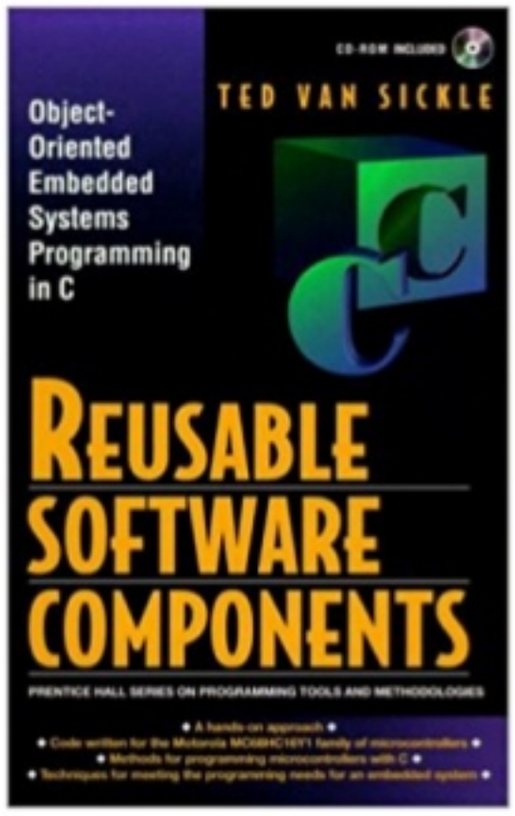
Automation


ANSIBLE
CHEF
puppet labs
SALTSTACK

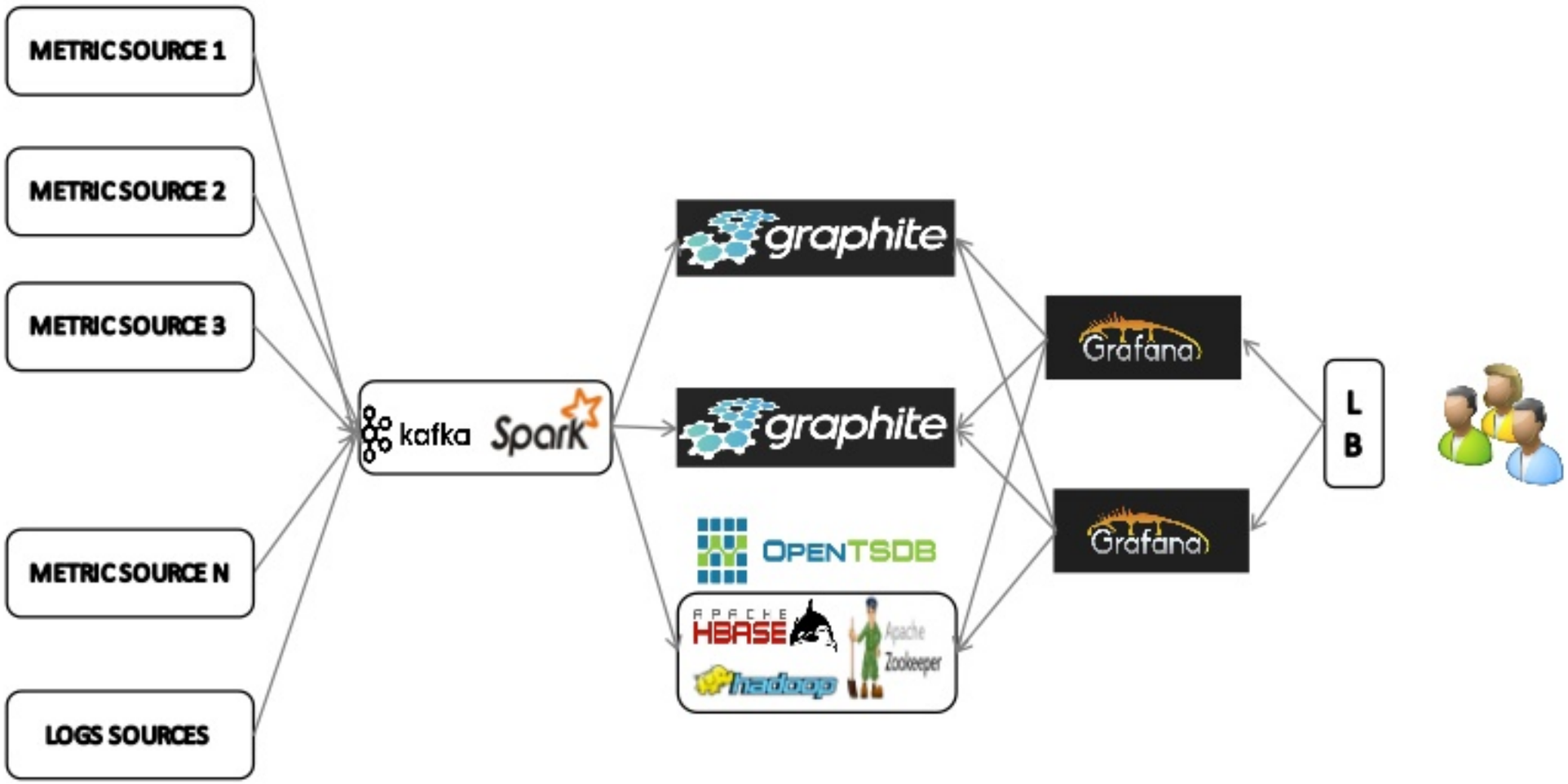

AUTOSCALING
AUTOSCALING EVERYWHERE

Re-use components

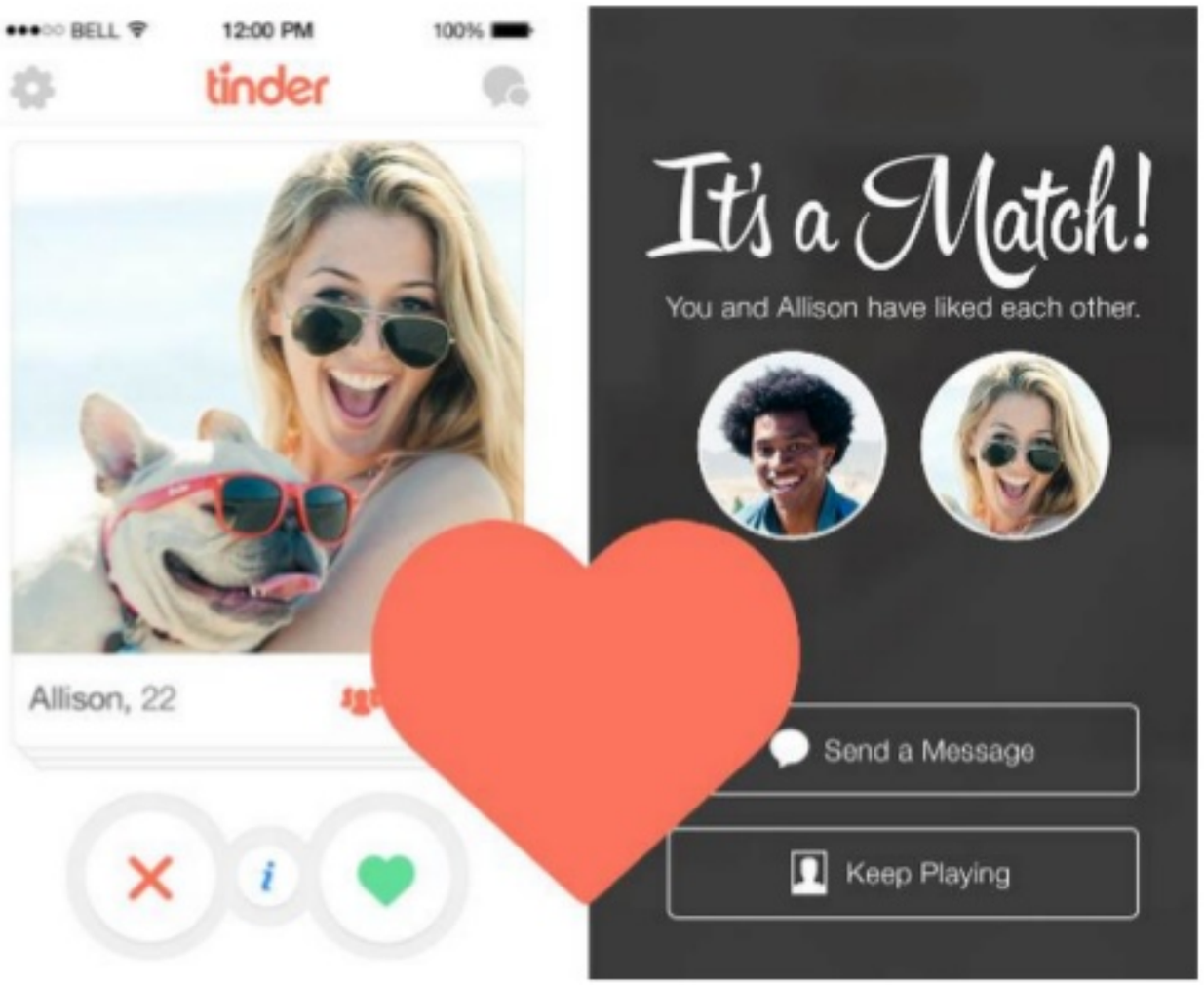
- Libraries
- SDKs
- Common artifacts



Scalable Monitoring Platform



Happy Users!


BELL
12:00 PM
100%
tinder
Allison, 22
It's a Match!
You and Allison have liked each other.
Send a Message
Keep Playing

We are hiring!

Email – **utkarsh.bhatnagar@gotinder.com**