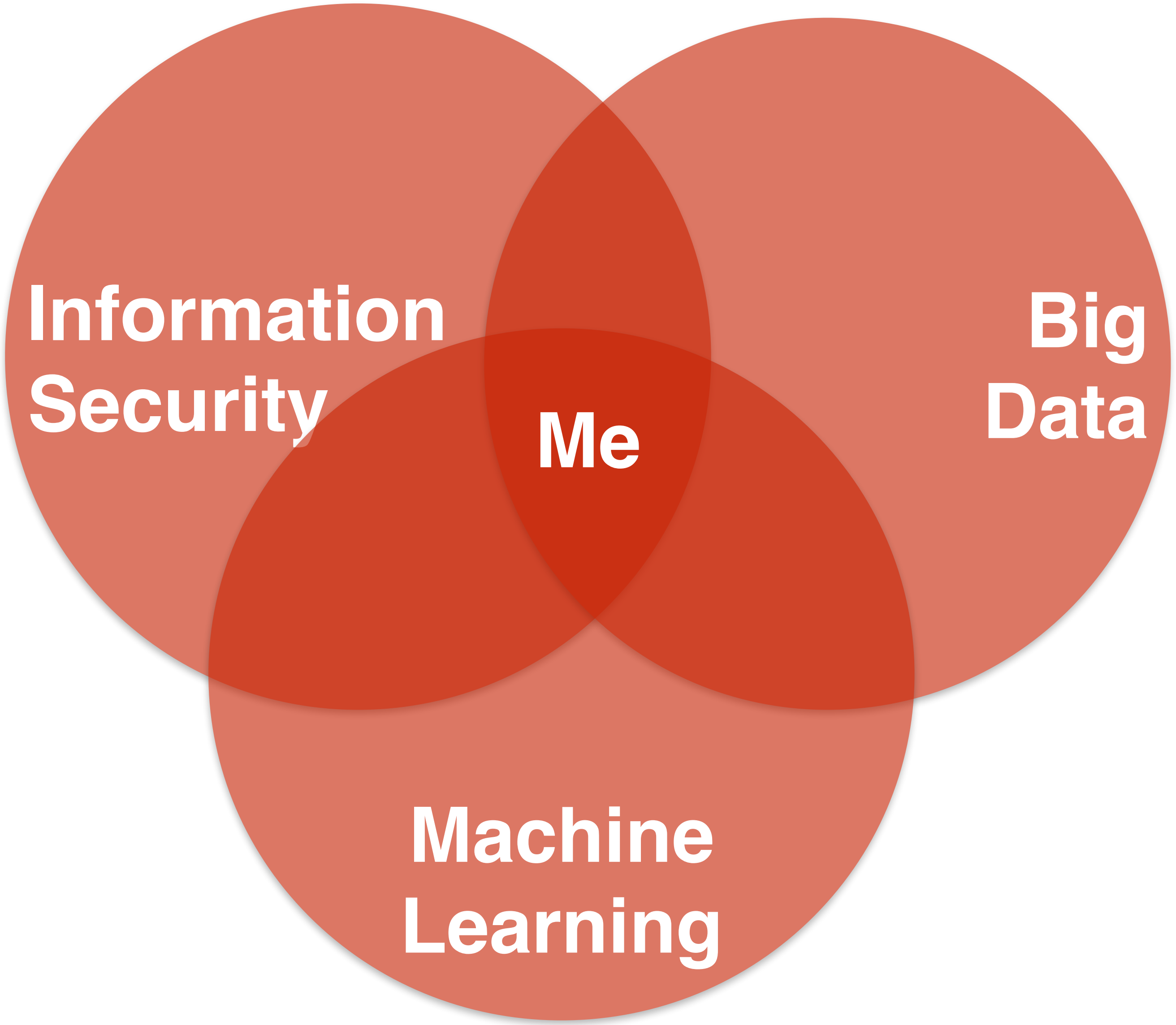
Python and Big data - An Introduction to Spark (PySpark)

Hitesh Dharmdasani

About me

- Security Researcher, Malware Reversing Engineer, Developer
- GIT > GMU > Berkeley > FireEye > On Stage
- Bootstrapping a few ideas
- Hiring!



What we will talk about?

- What is Spark?
- How does spark do things
- PySpark and data processing primitives
- Example Demo - Playing with Network Logs
- Streaming and Machine Learning in Spark
- When to use Spark

http://bit.do/PyBelgaumSpark

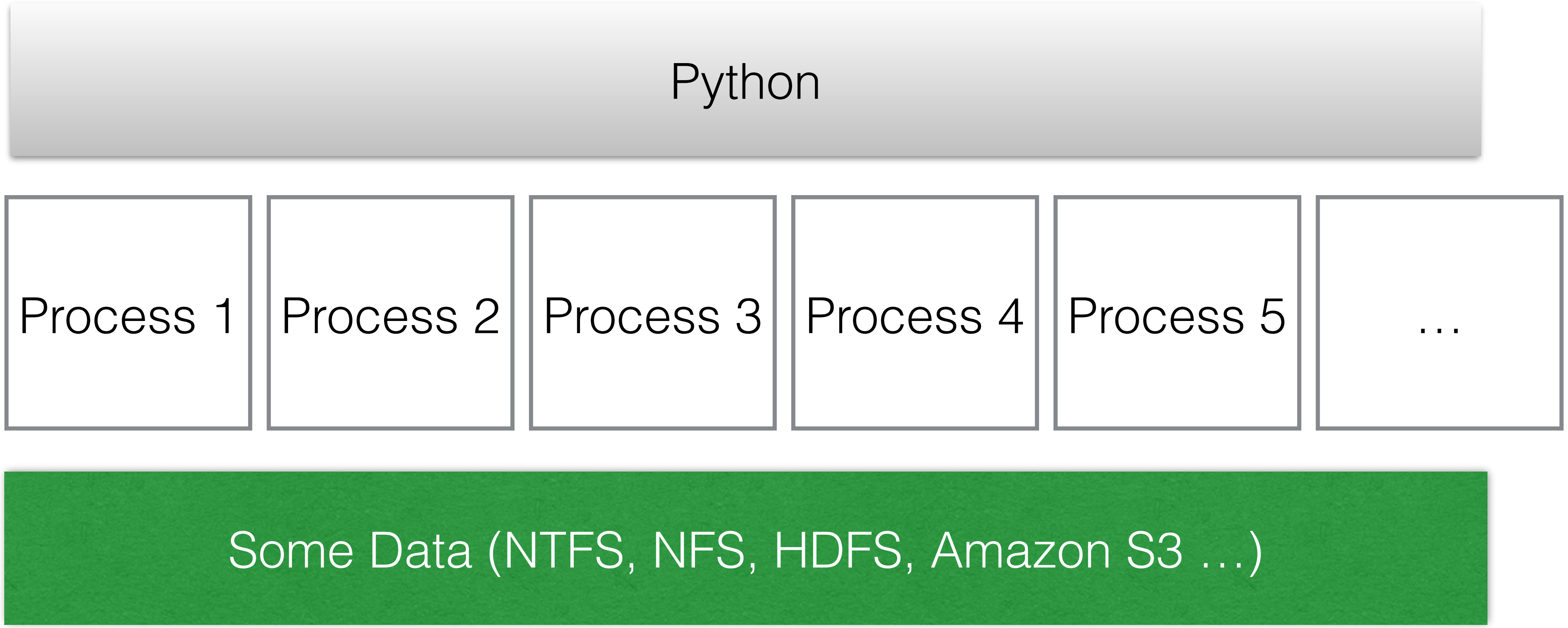
http://tinyurl.com/PyBelgaumSpark

What will we NOT talk about

- Writing production level jobs
- Fine Tuning Spark
- Integrating Spark with Kafka and the like
- Nooks and Crooks of Spark
- But glad to talk about it offline

The Common Scenario

You write 1 job. Then chunk,cut, slice and dice



Python

Process 1

Process 2

Process 3

Process 4

Process 5

…

Some Data (NTFS, NFS, HDFS, Amazon S3 …)

Compute where the data is

- Paradigm shift in computing
- Don't load all the data into one place and do operations
- State your operations and send code to the machine
- Sending code to machine >>> Getting data over network

MapReduce

```
public static MyFirstMapper {
  public void map {   . . .   }
  }

public static MyFirstReducer {
  public void reduce { . . . }
  }




public static MySecondMapper {
  public void map { . . . }
  }

public static MySecondReducer {
  public void reduce { . . . }
  }
```

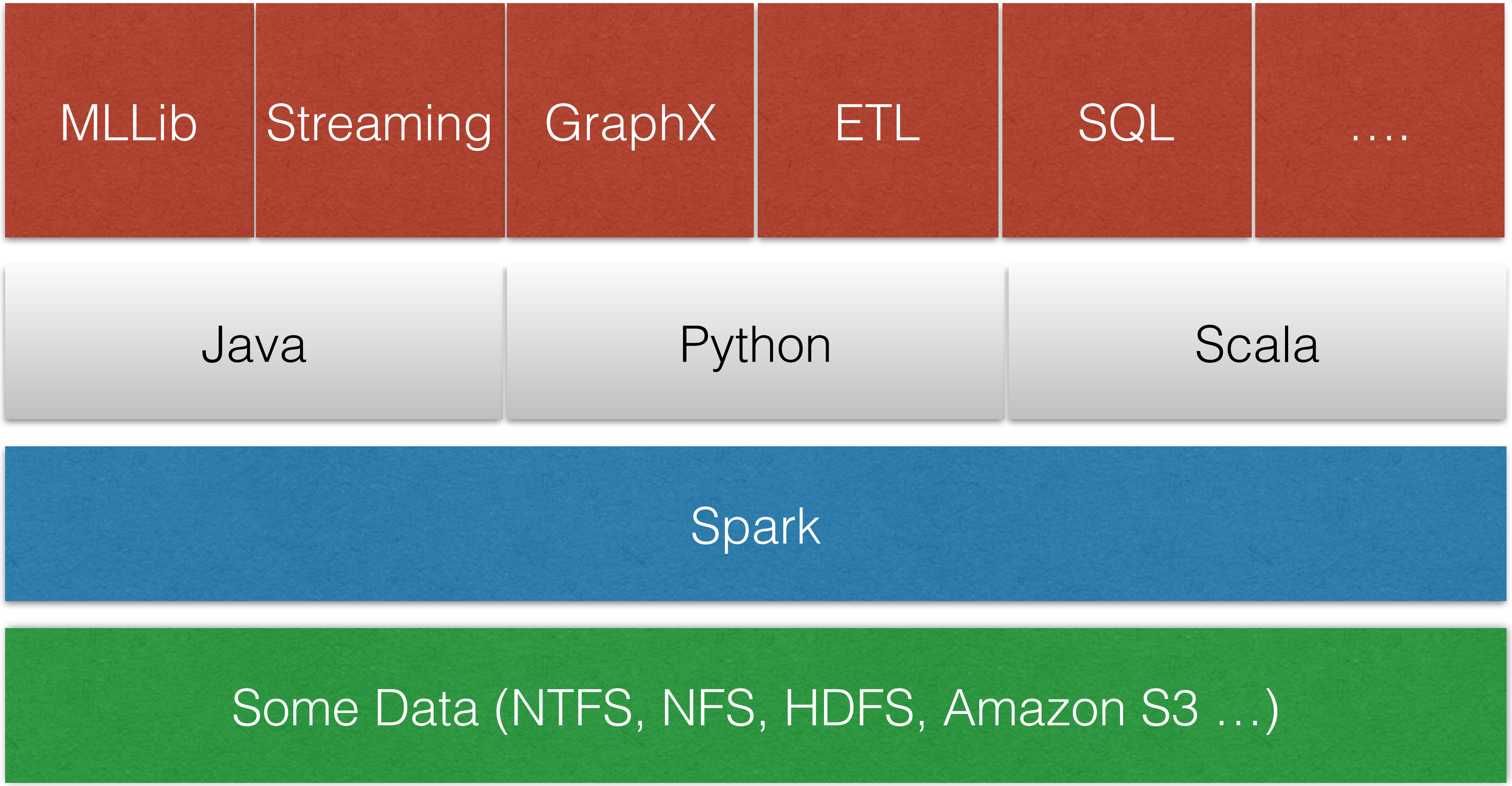
```
Job job = new Job(conf,
“First");
job.setMapperClass(MyFirstMapper
.class);
job.setReducerClass(MyFirstReduc
er.class);

/*Job 1 goes to Disk */

if(job.isSuccessful()) {

  Job job2 = new
  Job(conf,”Second”);
  job2.setMapperClass(MySecondMap
  per.class);
  job2.setReducerClass(MySecondRe
  ducer.class);
}
```

This also looks ugly if you ask me!

What is Spark?

- Open Source Lighting Fast Cluster Computing
- Focus on Speed and Scale
- Developed at AMP Lab, UC Berkeley by Matei Zaharia
- Most active Apache Project in 2014 (Even more than Hadoop)
- **Recently beat MapReduce in sorting 100TB of data by being 3X faster and using 10X fewer machines**

What is Spark?



MLLib | Streaming | GraphX | ETL | SQL | ….

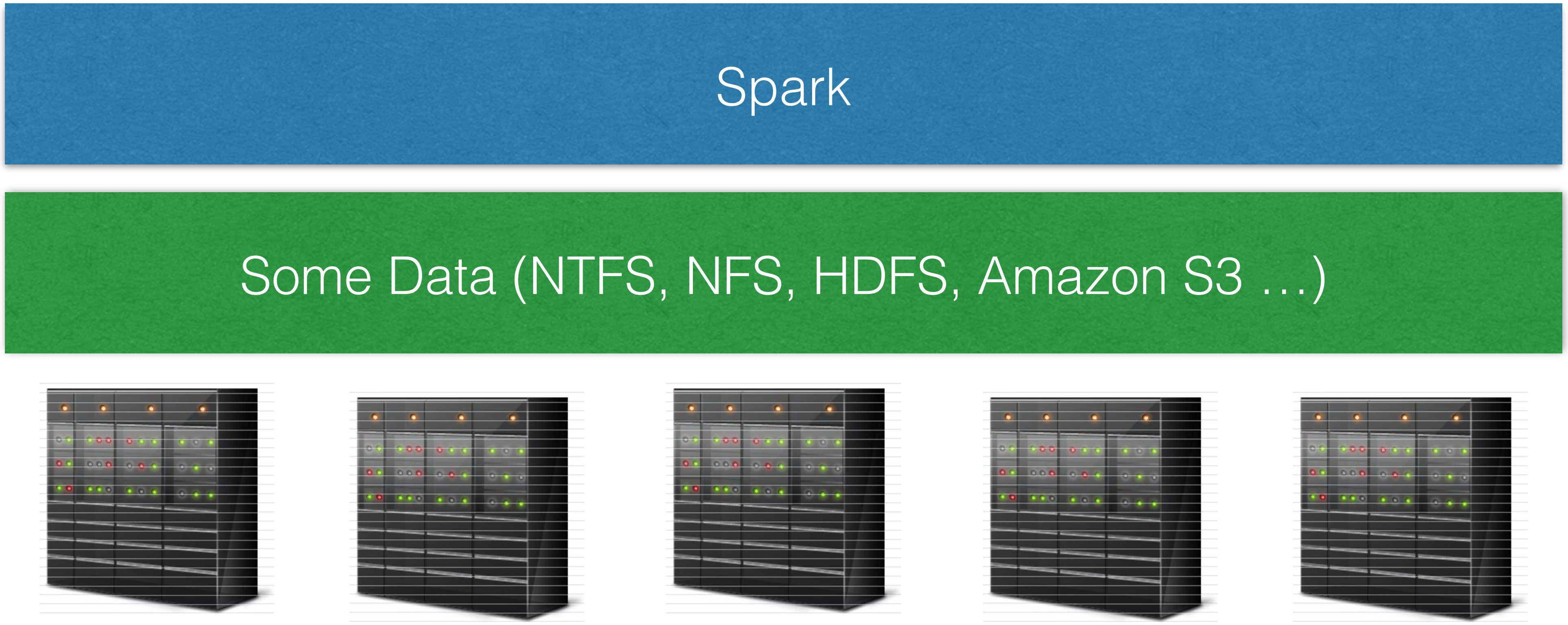
Java | Python | Scala

Spark

Some Data (NTFS, NFS, HDFS, Amazon S3 …)

What is Spark?

- Inherently distributed
  - Computation happens where the data resides



Spark

Some Data (NTFS, NFS, HDFS, Amazon S3 …)

What is different from MapReduce

- Uses main memory for caching
- Dataset is partitioned and stored in RAM/Disk for iterative queries
- Large speedups for iterative operations when in-memory caching is used

Spark Internals

The Init

- Creating a SparkContext
- It is Sparks' gateway to access the cluster
- In interactive mode. SparkContext is created as 'sc'

```
$ pyspark
...
...
SparkContext available as sc.

>>> sc
<pyspark.context.SparkContext at 0xdeadbeef>
```

Spark Internals

The Key Idea

Resilient Distributed Datasets

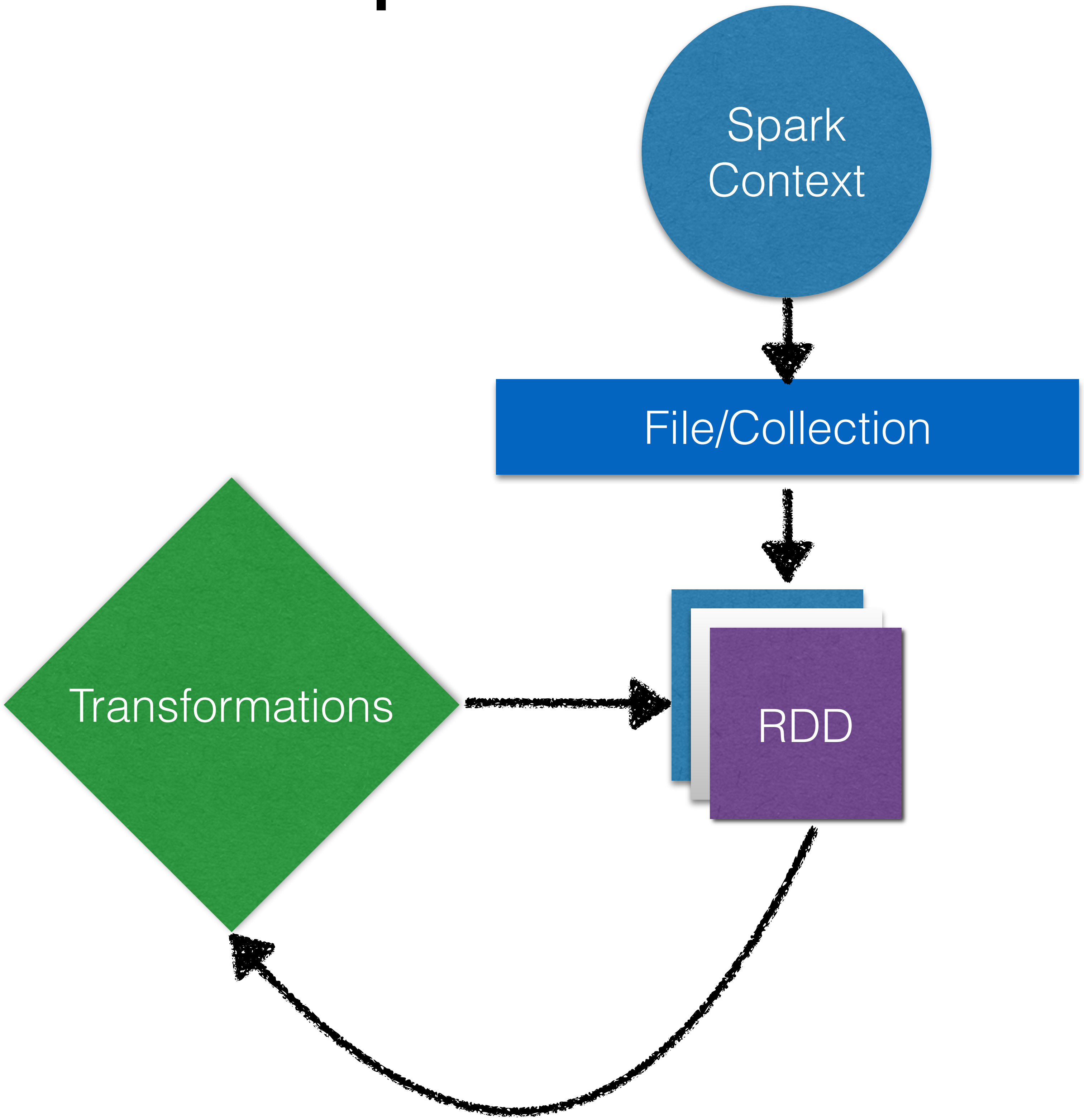
- Basic unit of abstraction of data
- Immutable
- Persistance

```
>>> data = [90, 14, 20, 86, 43, 55, 30, 94 ]
>>> distData = sc.parallelize(data)
ParallelCollectionRDD[13] at parallelize at
PythonRDD.scala:364
```

Spark Internals

Operations on RDDs - Transformations & Actions

Spark Internals



Spark Context

File/Collection

Transformations

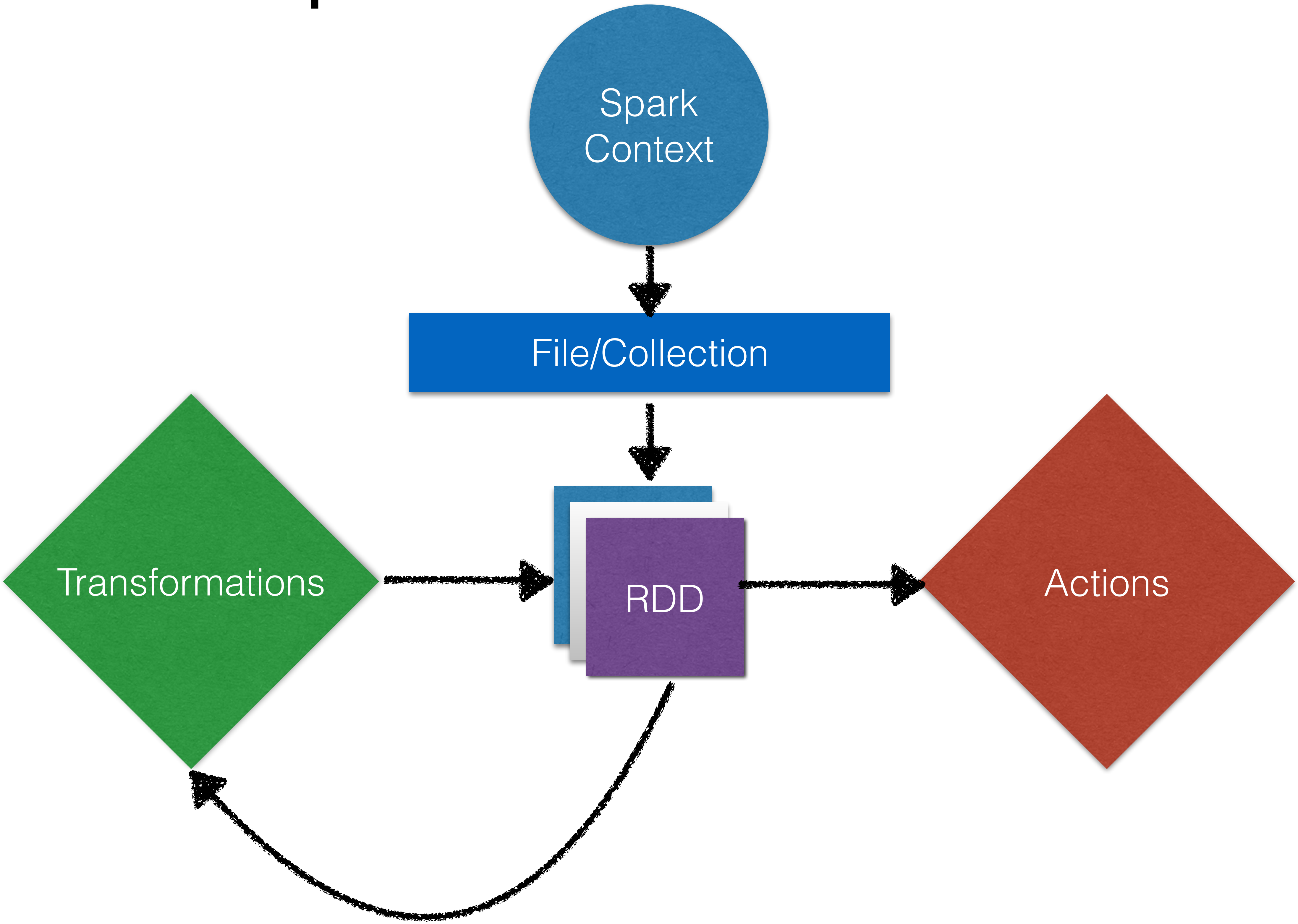
RDD

Spark Internals

Now what?

Lazy Evaluation

Spark Internals



Spark Internals

Transformation Operations on RDDs

Map

```
def mapFunc(x):
   return x+1

rdd_2 = rdd_1.map(mapFunc)
```

Filter

```
def filterFunc(x):
   if x % 2 == 0:
     return True
   else:
     return False

rdd_2 = rdd_1.filter(filterFunc)
```

Spark Internals

Transformation Operations on RDDs

- `map`
- `filter`
- `flatMap`
- `mapPartitions`
- `mapPartitionsWithIndex`
- `sample`
- `union`
- `intersection`
- `distinct`
- `groupByKey`

Spark Internals

```
>>> increment_rdd = distData.map(mapFunc)
>>> increment_rdd.collect()
[91, 15, 21, 87, 44, 56, 31, 95]
>>>

>>> increment_rdd.filter(filterFunc).collect()
[44, 56]
```

OR

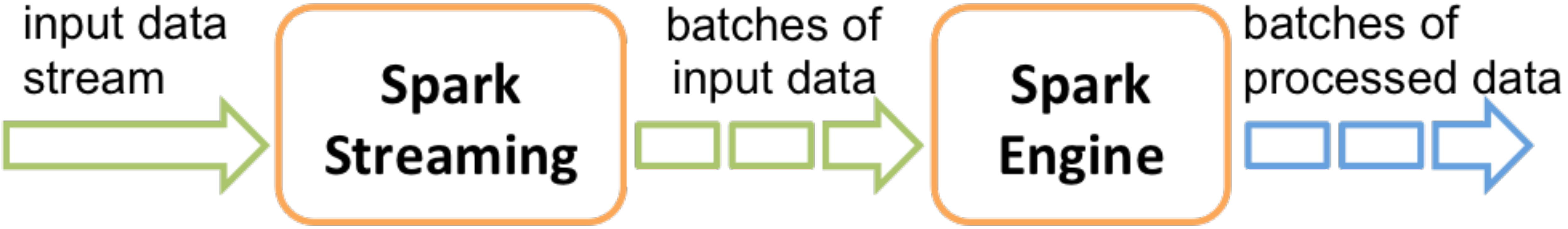
```
>>> distData.map(mapFunc).filter(filterFunc).collect()
[44, 56]
```

Spark Internals

Fault Tolerance and Lineage

Moving to the Terminal

Spark Streaming



Kafka
Flume
HDFS
Twitter
ZeroMQ

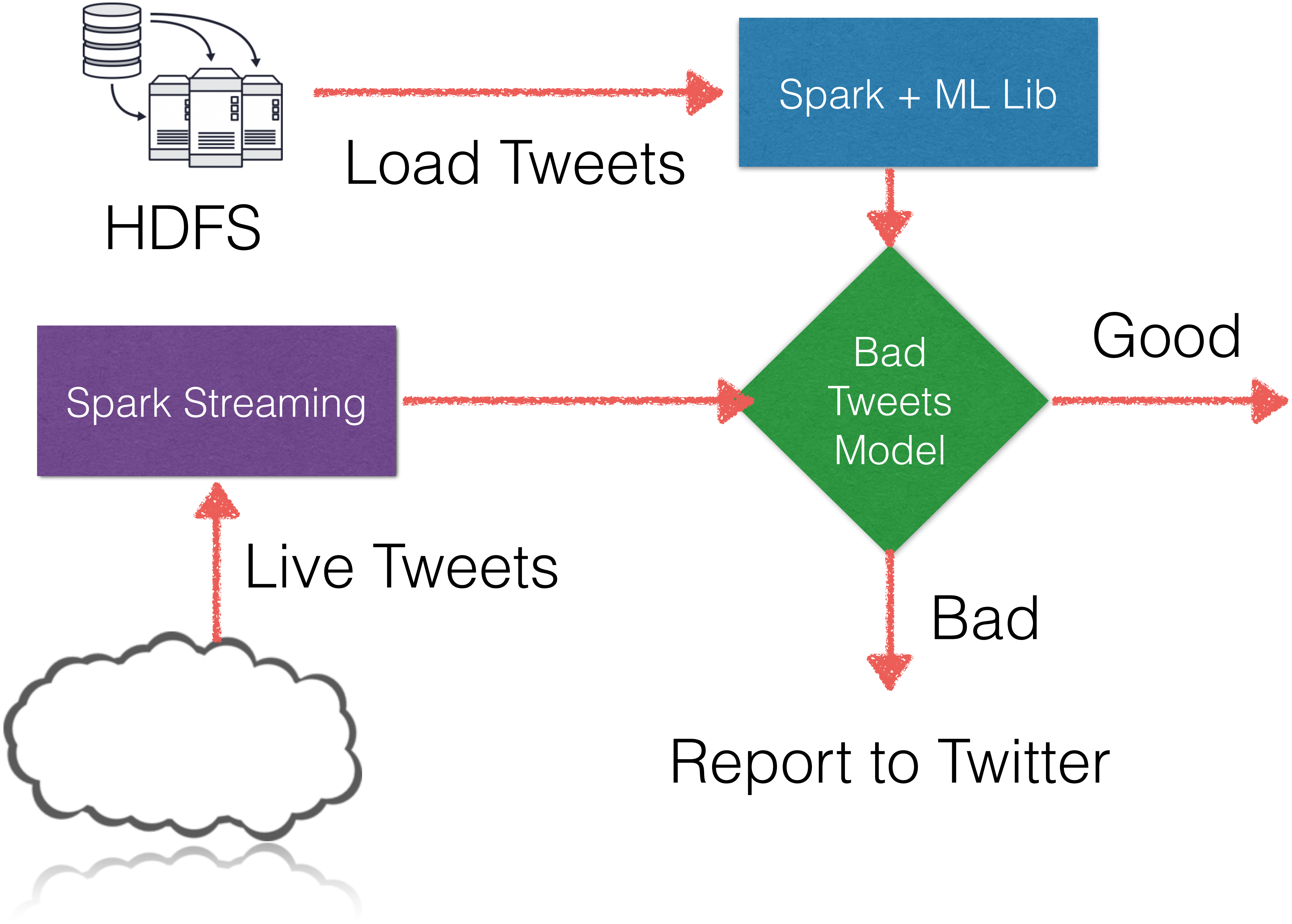
RDD

HDFS
Cassandra
NFS
TextFile

ML Lib

- Machine Learning Primitives in Spark
- Provides training and classification at scale
- Exploits Sparks’ ability for iterative computation (Linear Regression, Random Forest)
- Currently the most active area of work within Spark

How can I use all this?



HDFS

Load Tweets

Spark + ML Lib

Spark Streaming

Live Tweets

Bad Tweets Model

Good

Bad

Report to Twitter

To Spark or not to Spark

- Iterative computations
- “Don't fix something that is not broken”
- Lesser learning barrier

- Large one-time compute
- Single Map Reduce Operation