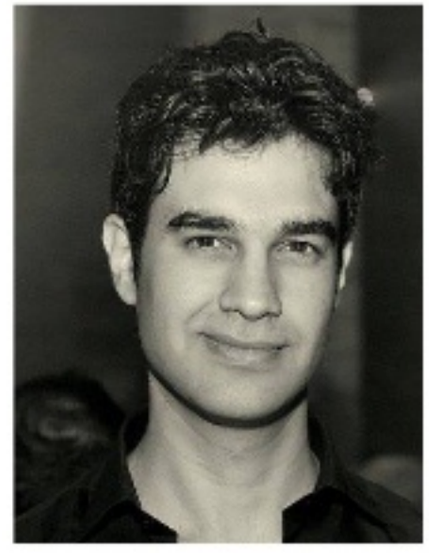
From Python Scikit-Learn to Scala Spark

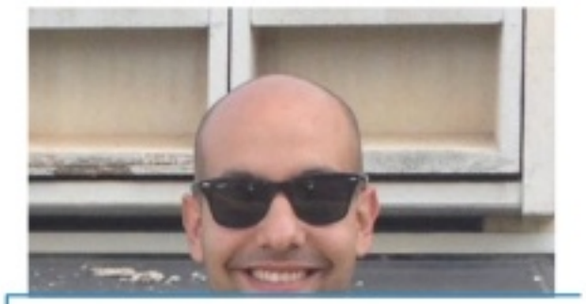
The Road to Uncovering Botnets

whoami

- Avi Aminov
  - ~2 years Security Researcher at Akamai
  - Physics PhD student



- Asaf Nadler
  - ~1.5 years Security Researcher at Akamai
  - CS PhD student



SPARK SUMMIT 2017

Enterprise Threat Protection

- Detect malware presence from outbound traffic
  - Behavioral pattern analysis
  - Domain blacklisting

- Availability – End of June ’17

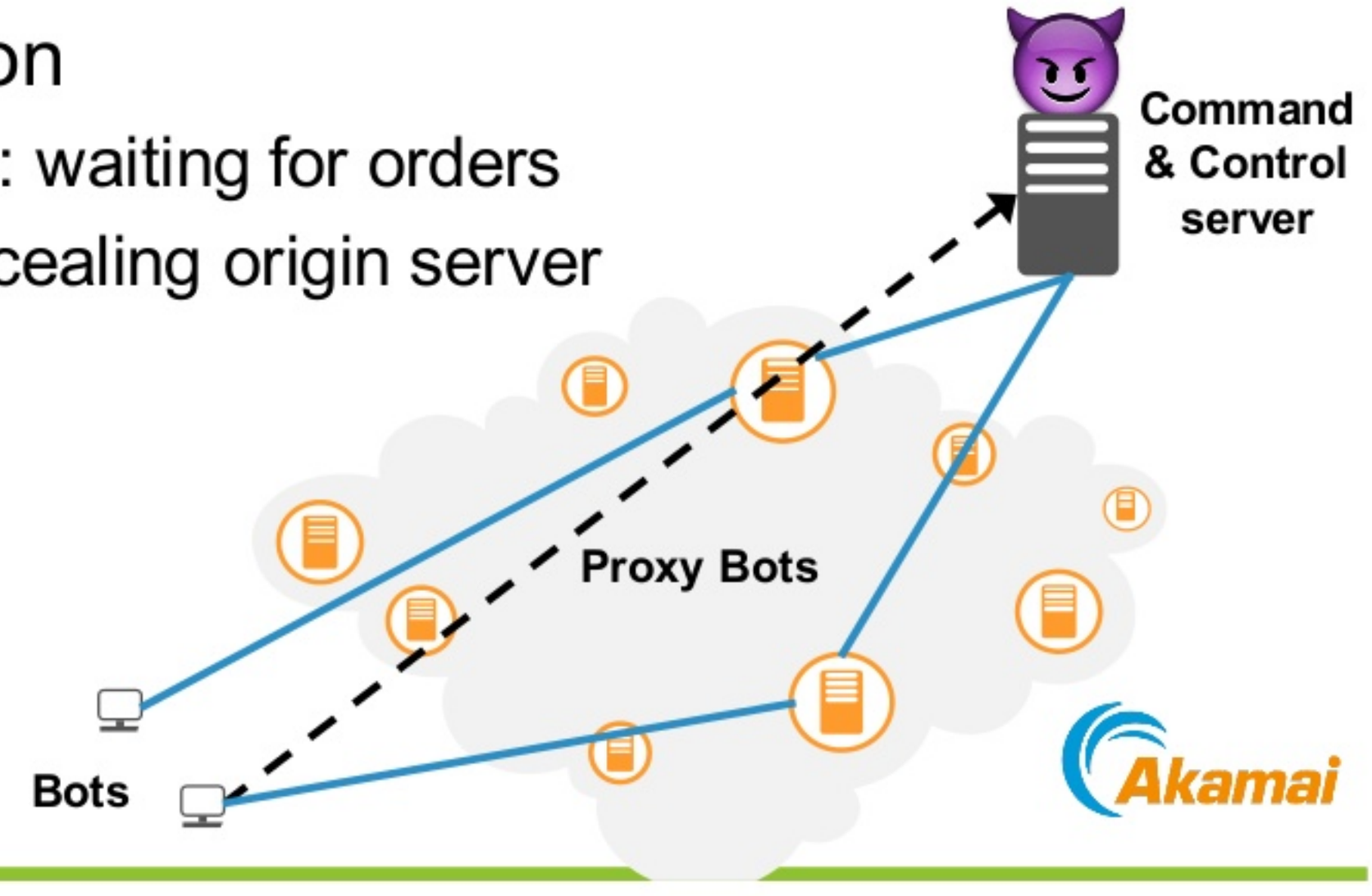
SPARK SUMMIT 2017

Data

- Akamai Data
  - 20-30% of internet traffic
  - Customer ISP/Enterprise logs – 20B DNS queries/day
- Third party data
  - e.g. Authoritative DNS log lines
- Open data sources
  - e.g. WHOIS information

Bot Networks – IP Fluxing

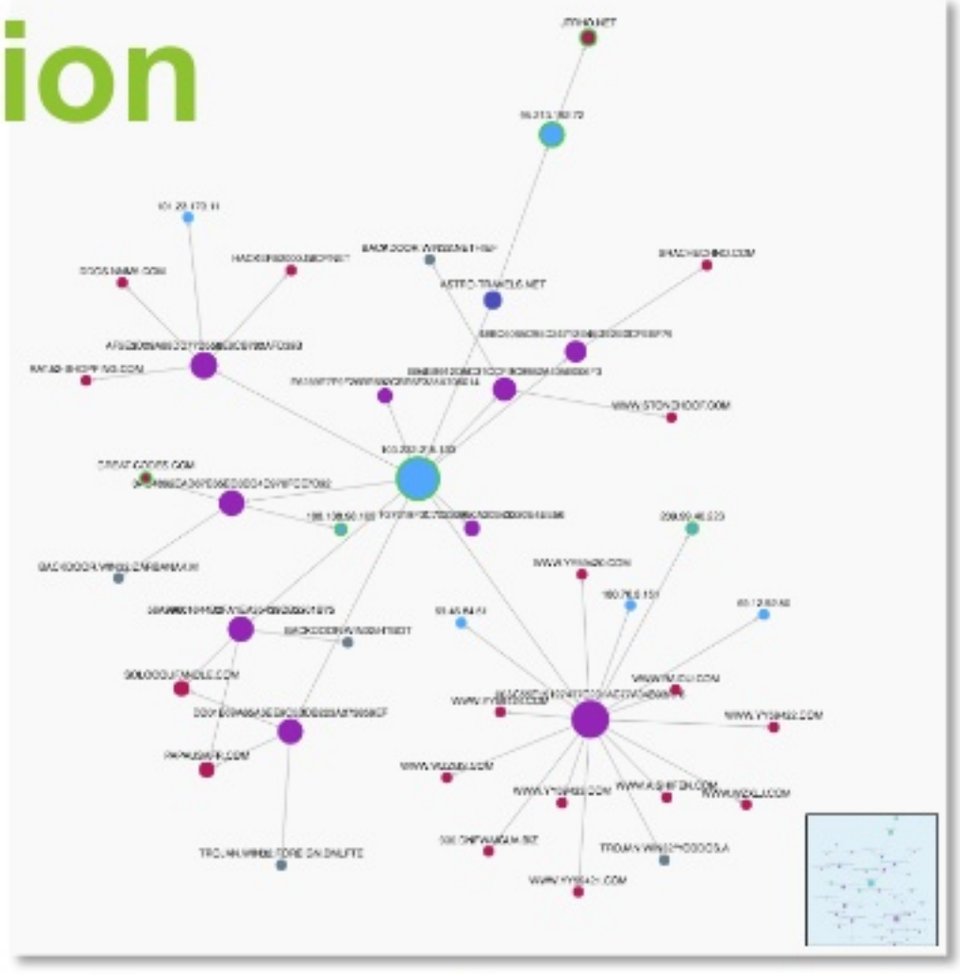
- Goal – Evasion
  - Regular bots: waiting for orders
  - Proxies: concealing origin server



Akamai

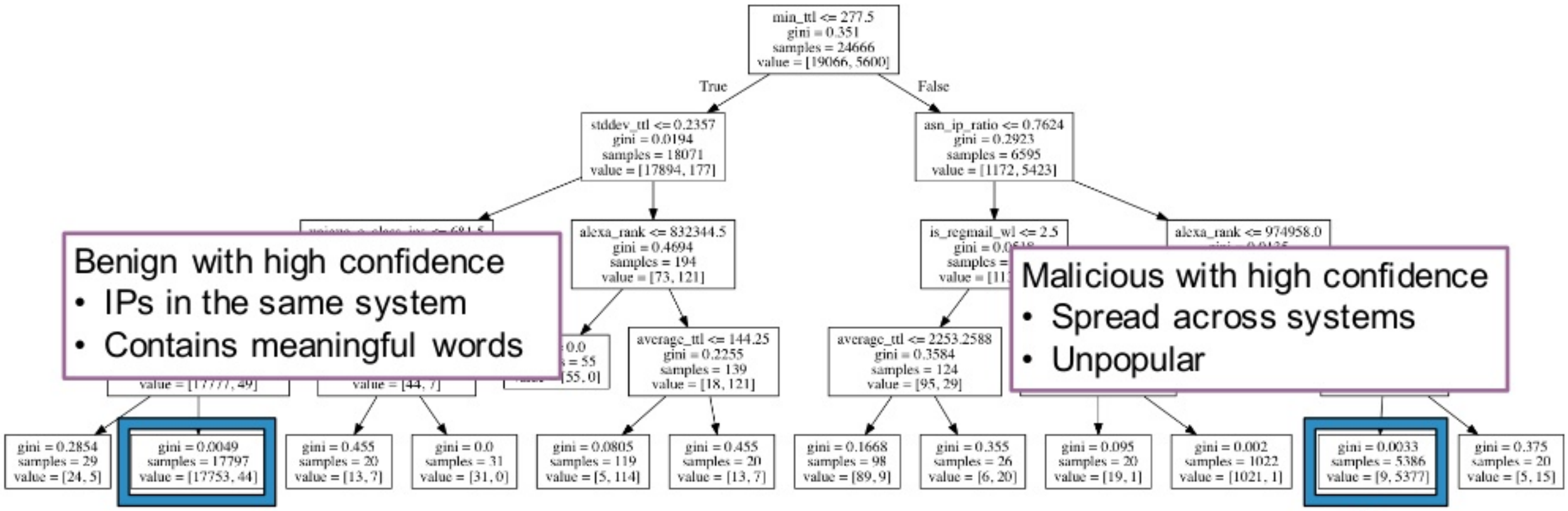
Bot Networks Detection

- Detect **illegitimate** IP fluxing
- Features
  - IP dispersity (Geo, systems)
  - TTL features
  - Lexical



Domain	Description	#Systems	#Countries
astro-travels.net	PoS CNC Host	157	11

Decision Tree Model



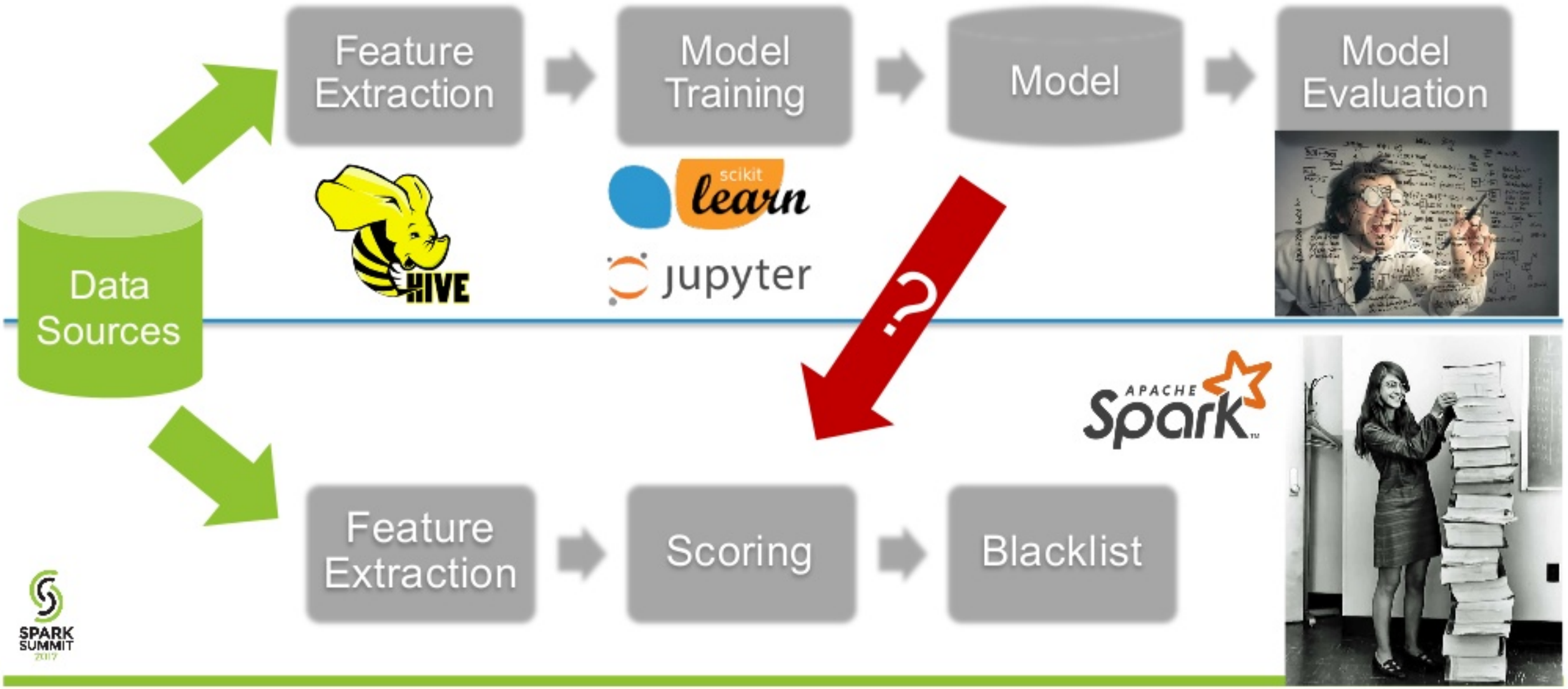
Benign with high confidence
- IPs in the same system
- Contains meaningful words

Malicious with high confidence
- Spread across systems
- Unpopular

SPARK SUMMIT 2017

Akamai

Challenge – Going to Production



What have we done so far?

- Flow
  - Researcher describes an algorithm (document + Hive query)
  - Dev rewrites the code in MapReduce (now Scala/Spark)
- Problems
  - Not applicable to ML pipelines
  - Prone to mistakes
  - Longer development cycle

Can We Do Better? Option #1

- Research side – Pipeline in Scala/Spark
- Dev side – Implement the algorithms
- Pros
  - Greater flexibility
  - Research scale
- Cons
  - Learning curve
  - Lose sklearn/R benefits

Can We Do Better? Option #2

- Research side – Train locally and export model
- Dev side – Transform data using imported model
- Pros
  - Quick implementation
  - Unified procedure
- Cons
  - No support for all models

Export scheme

- Predictive Model Markup Language
- General scheme for ML pipelines
  - Data transformations
  - Scoring models
- XML format – Readable
- Supported by major data science / ML frameworks using jPMML (R, sklearn)

PMML Simple Boilerplate

Python (Research side)

```python
from sklearn_pandas import DataFrameMapper
default_mapper = DataFrameMapper(
    [(i, None) for i in features + ["label"]]
)

from sklearn2pmml import sklearn2pmml
sklearn2pmml(
    estimator=clf,
    mapper=default_mapper,
    pmml="outputs/FFSNDecisionTreeClassifier.pmml"
)
```

Scala (Dev side)

```scala
import org.jpmml.spark._
val data = ???
val pmmlFile = new File("FFSNDecisionTreeClassifier.pmml")
val evaluator = EvaluatorUtil.createEvaluator(pmmlFile)
val pmmlTransformerBuilder = new TransformerBuilder(evaluator)
  .withLabelCol("ffsn")
  .withProbabilityCol("ffsn_prob")
  .exploded(true)

val pmmlTransformer = pmmlTransformerBuilder.build
val output = pmmlTransformer.transform(data)
```

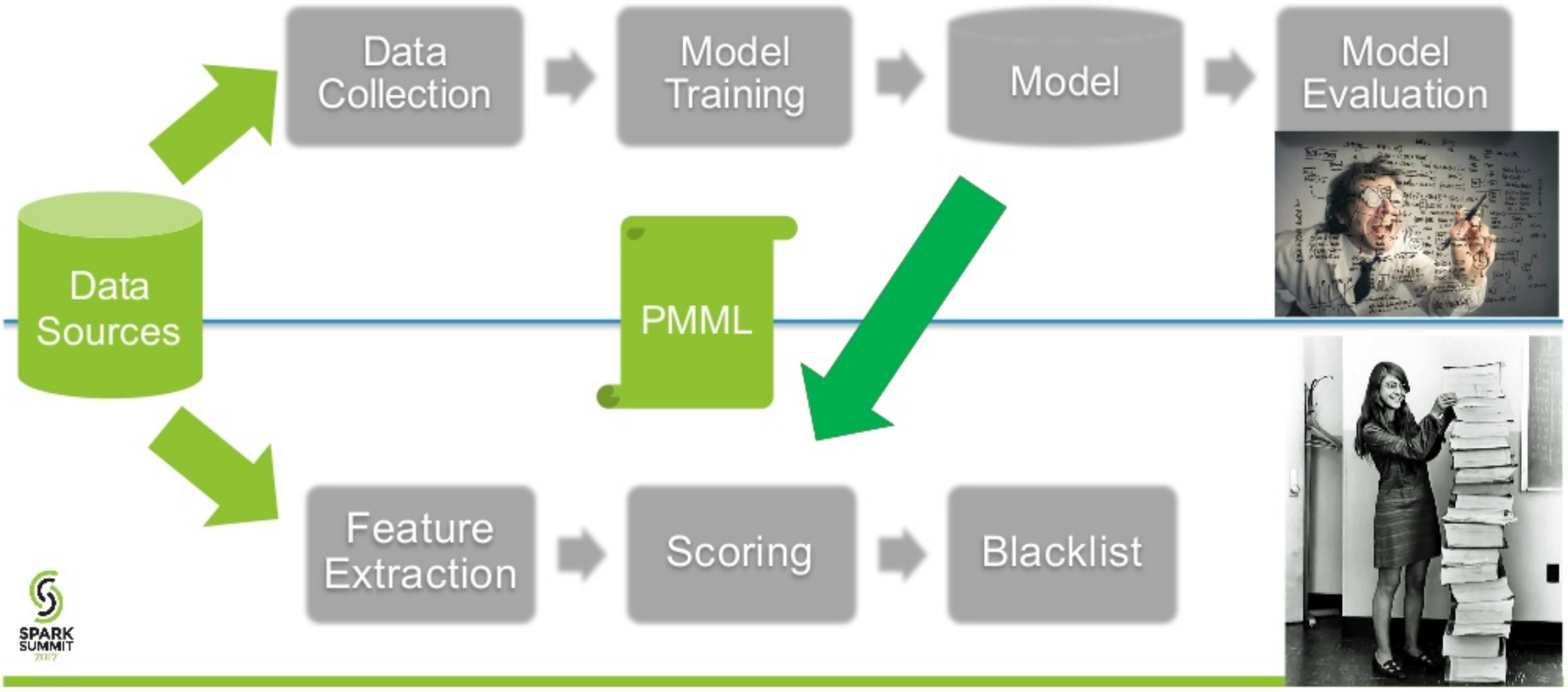
SPARK SUMMIT 2017

Credit: **jpmml** lib http://openscoring.io/ , https://github.com/jpmml/
Maintained by **Villu Ruusmann**

Lessons Learned

- Work process adjusted to the task
  - Training locally? Export the model
  - Training on larger scales? Better to use Spark
- Use jpmml for model export
- When applicable, reduce workload in production
  - Example – only look at domains with many IPs

Challenge solved



Thank you!

@AviBachsh