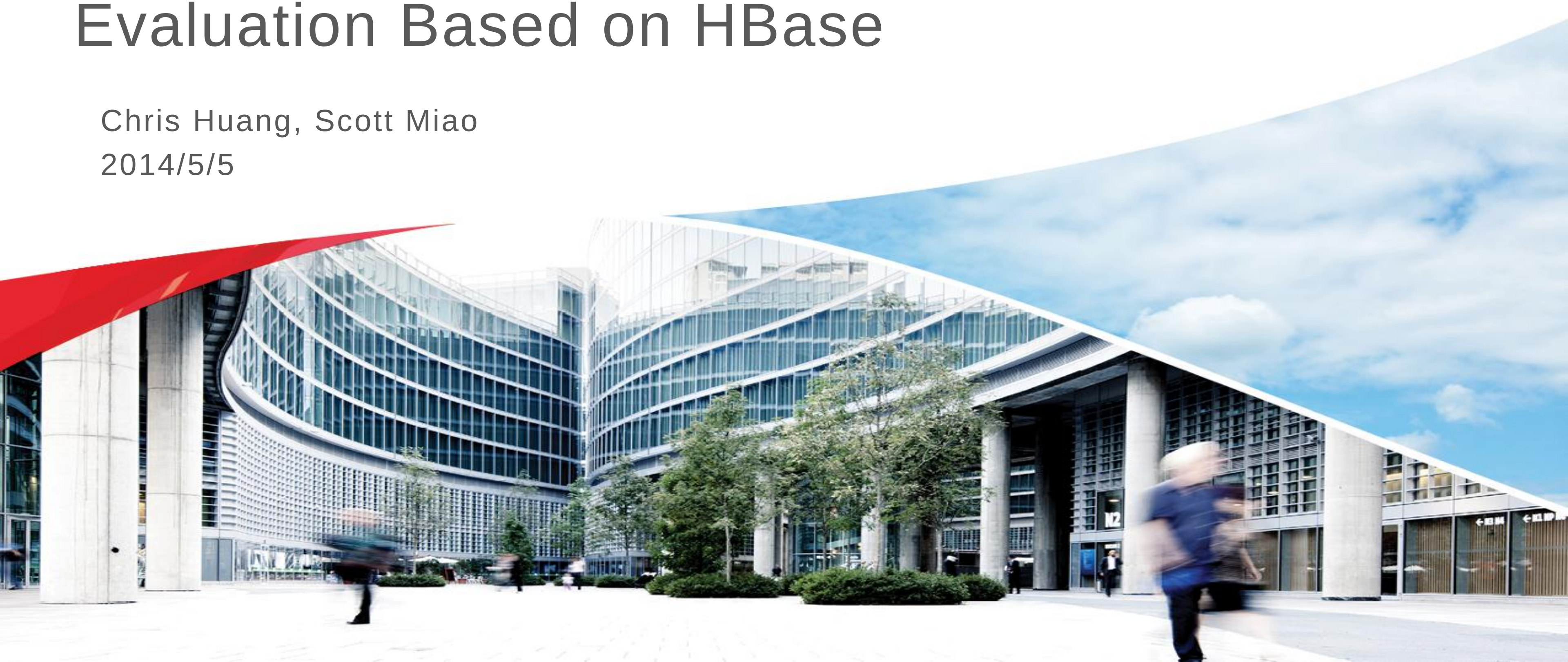
A Graph Service for Global Web Entities Traversal and Reputation Evaluation Based on HBase

Chris Huang, Scott Miao

2014/5/5



Who are we

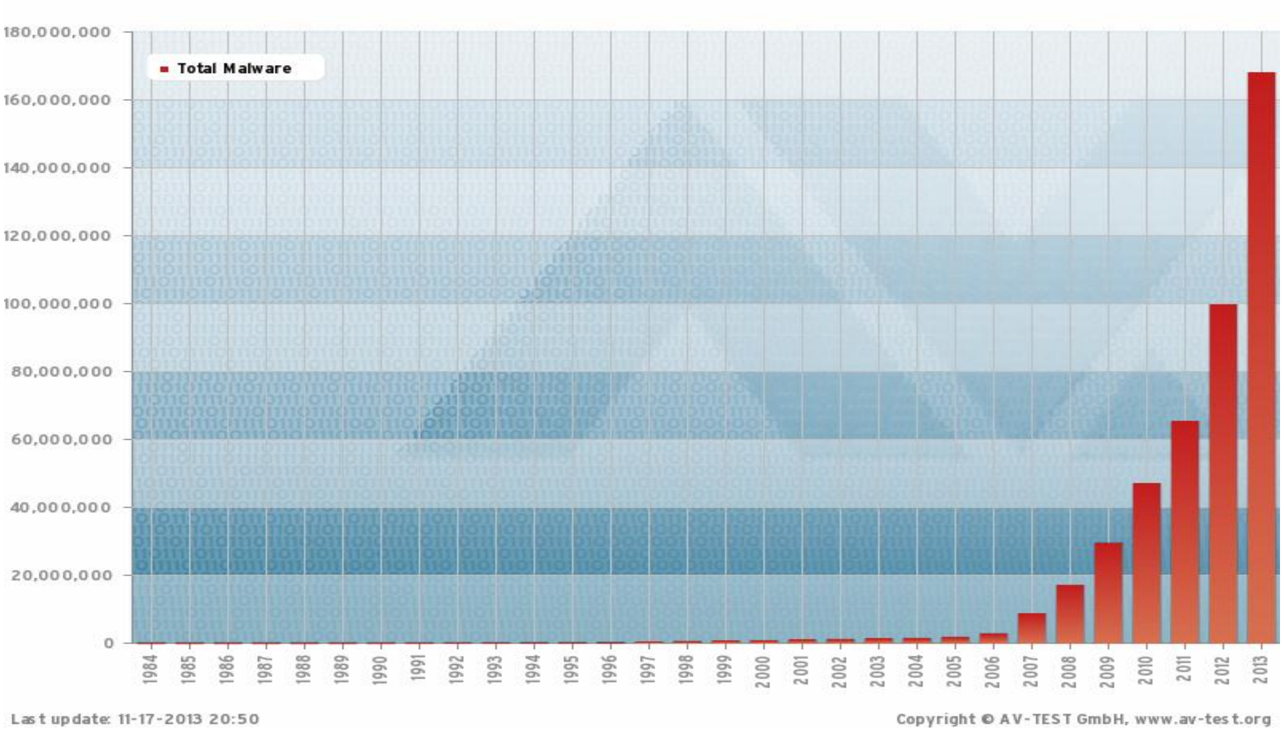
- **Chris Huang**
- RD Manager, SPN, Trend Micro
- Hadoop Architect
- Worked on hadoop ecosystem since 2009
- Contributor for Bigtop
- @chenhsiu48

- **Scott Miao**
- Developer, SPN, Trend Micro
- Worked on hadoop ecosystem since 2011
- Expertise in HDFS/MR/HBase
- Contributor for HBase/HDFS
- @takeshi.miao

Our blog ‘Dumbo in TW’: http://dumbointaiwan.blogspot.tw/

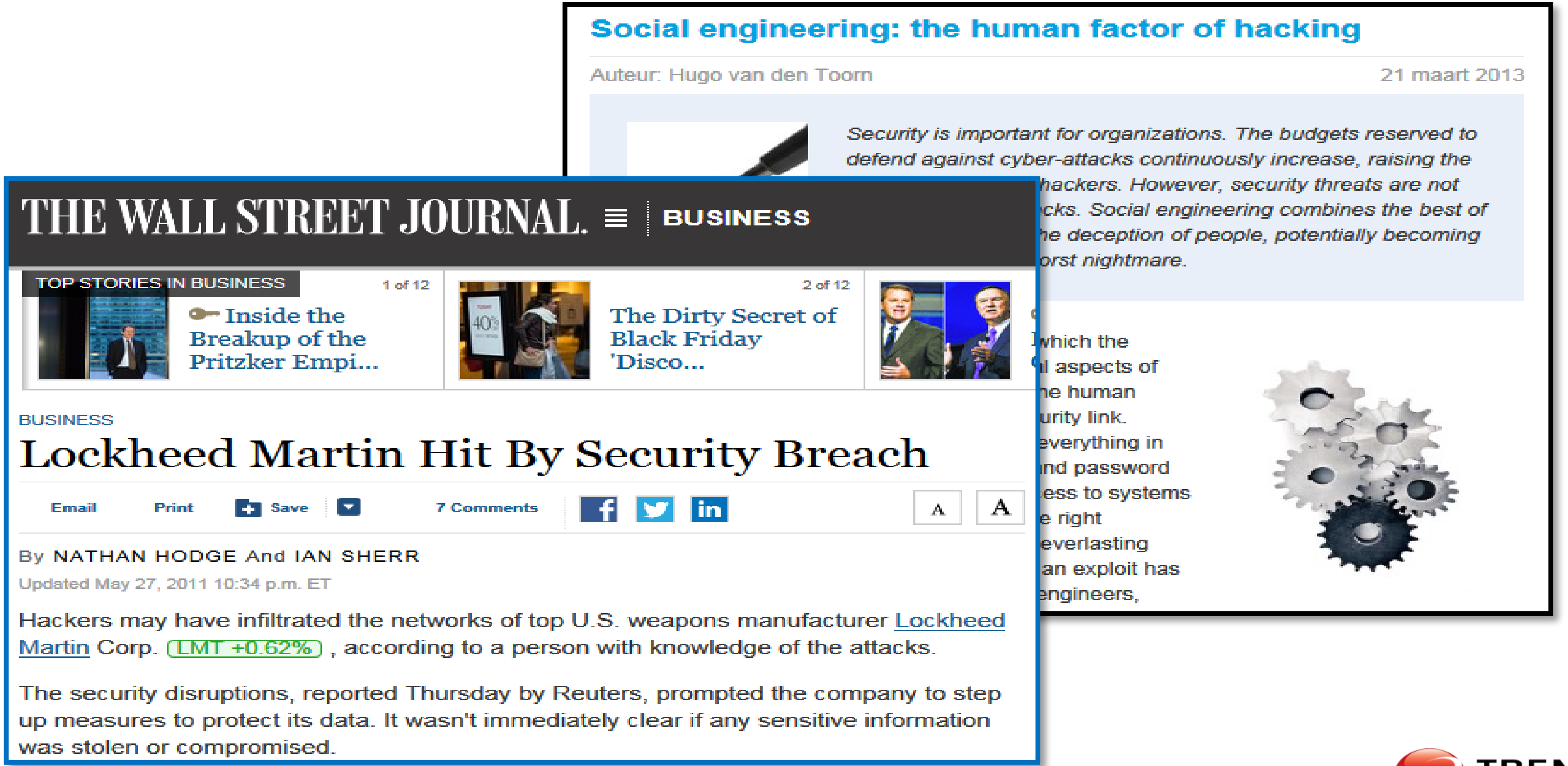
Challenges We Faced

New Unique Malware Discovered



http://www.av-test.org/en/statistics/malware/

Social Engineering vs. Cyber Attacks



Social engineering: the human factor of hacking

Auteur: Hugo van den Toorn

21 maart 2013

Security is important for organizations. The budgets reserved to defend against cyber-attacks continuously increase, raising the
hackers. However, security threats are not
cks. Social engineering combines the best of
he deception of people, potentially becoming
orst nightmare.

which the
l aspects of
he human
urity link.
everything in
nd password
ess to systems
e right
everlasting
an exploit has
engineers,

THE WALL STREET JOURNAL. ≡ BUSINESS

TOP STORIES IN BUSINESS
1 of 12
Inside the Breakup of the Pritzker Empi...
2 of 12
The Dirty Secret of Black Friday 'Disco...

BUSINESS

Lockheed Martin Hit By Security Breach

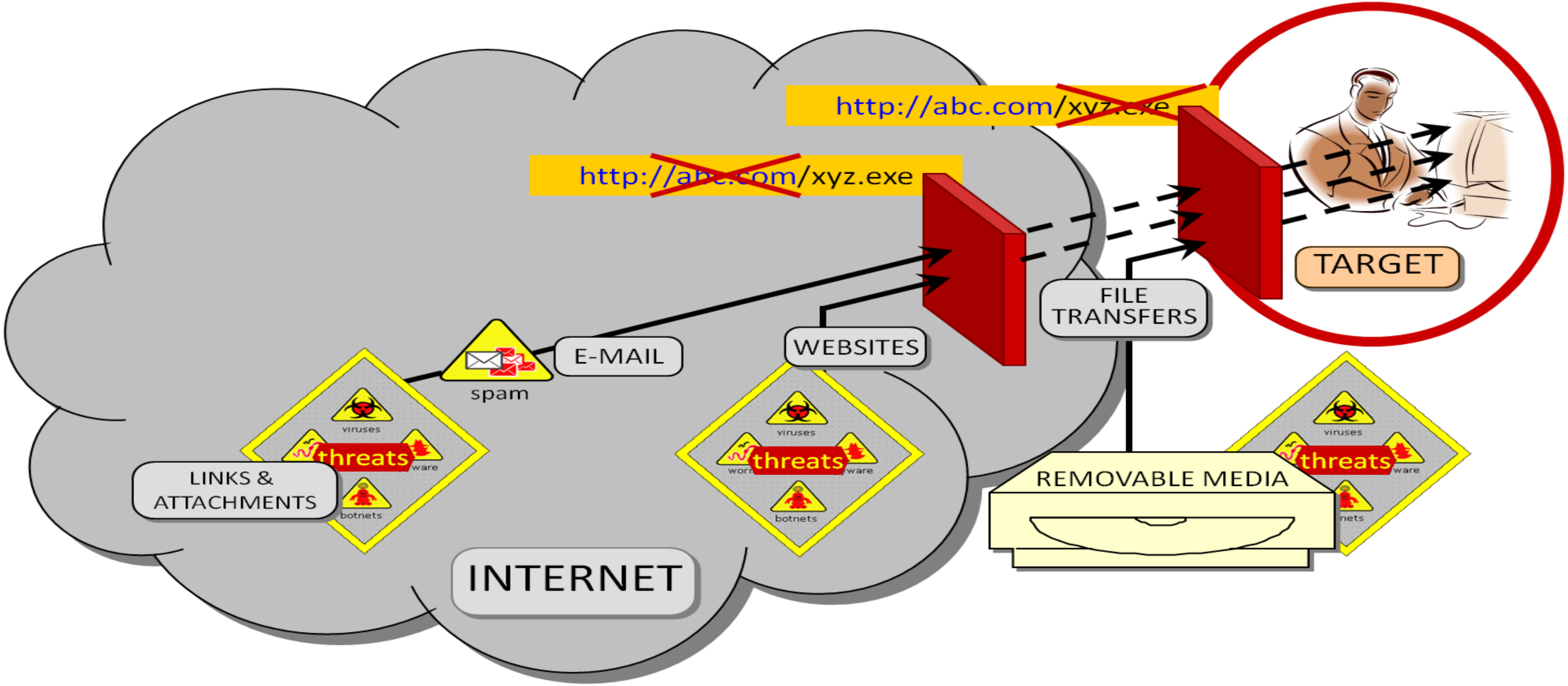
By NATHAN HODGE And IAN SHERR

Updated May 27, 2011 10:34 p.m. ET

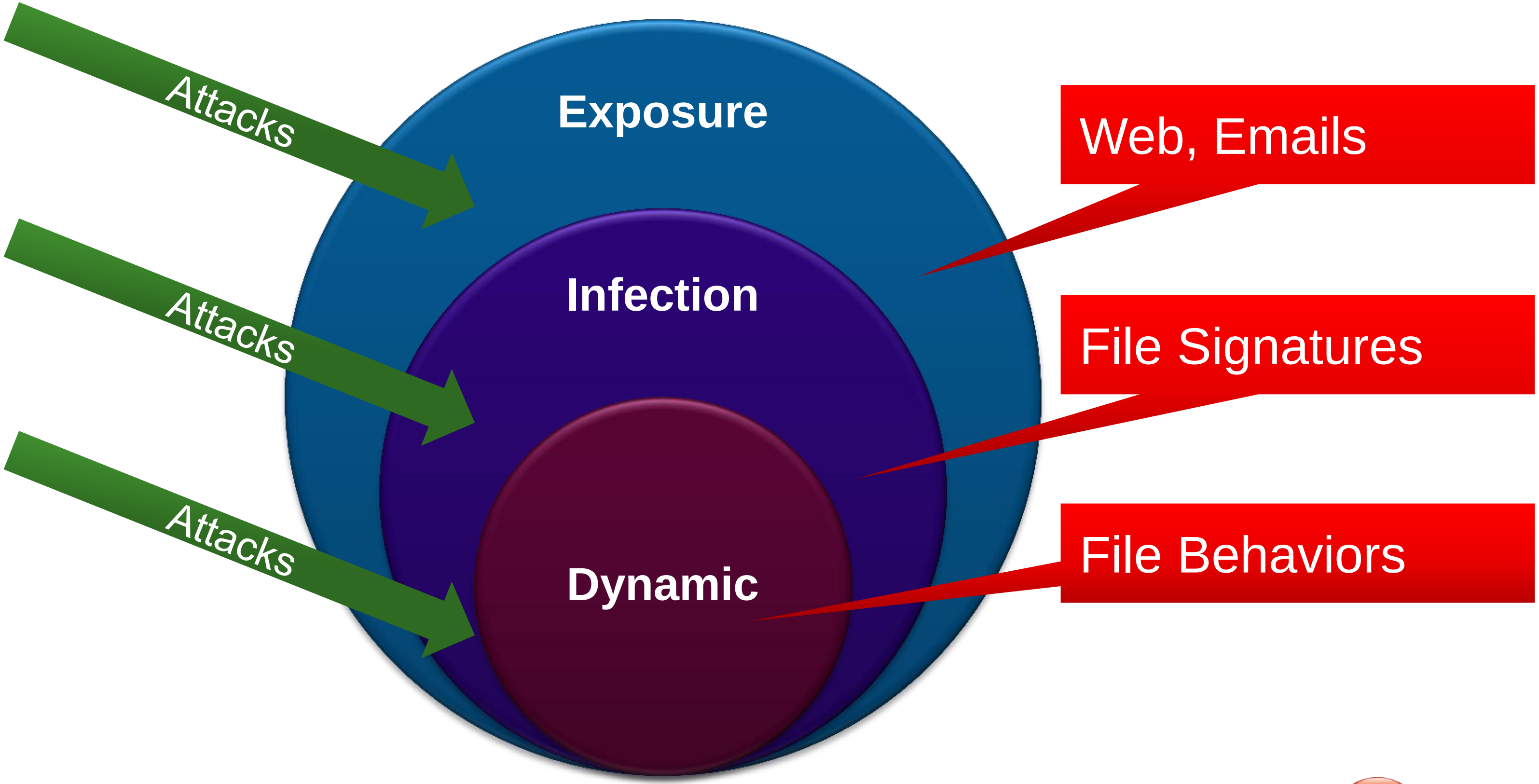
Hackers may have infiltrated the networks of top U.S. weapons manufacturer Lockheed Martin Corp. (LMT +0.62%) , according to a person with knowledge of the attacks.

The security disruptions, reported Thursday by Reuters, prompted the company to step up measures to protect its data. It wasn't immediately clear if any sensitive information was stolen or compromised.

Trend Micro Defense Strategy



Layer of Protection



Attacks

Attacks

Attacks

Exposure

Infection

Dynamic

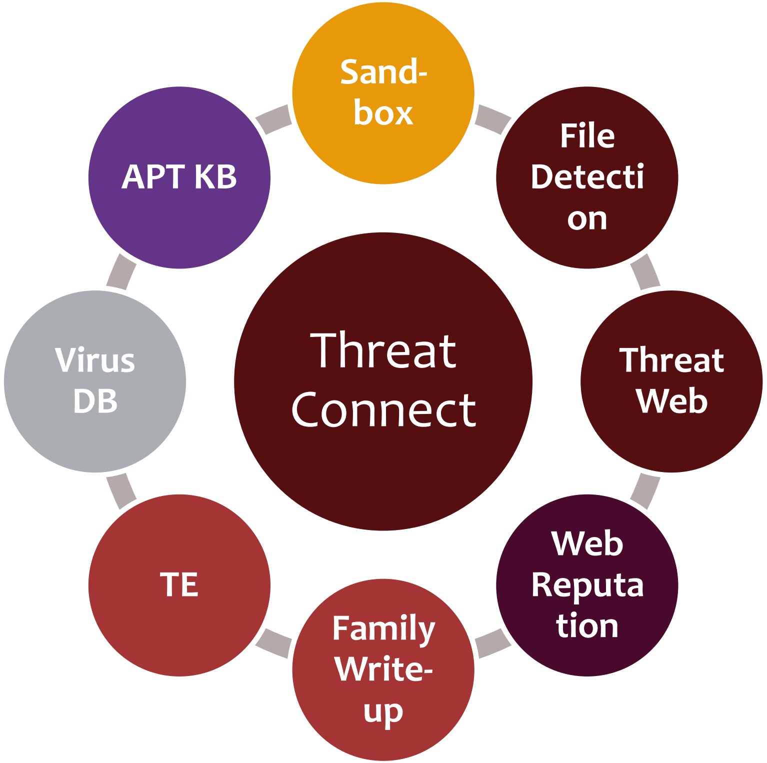
Web, Emails

File Signatures

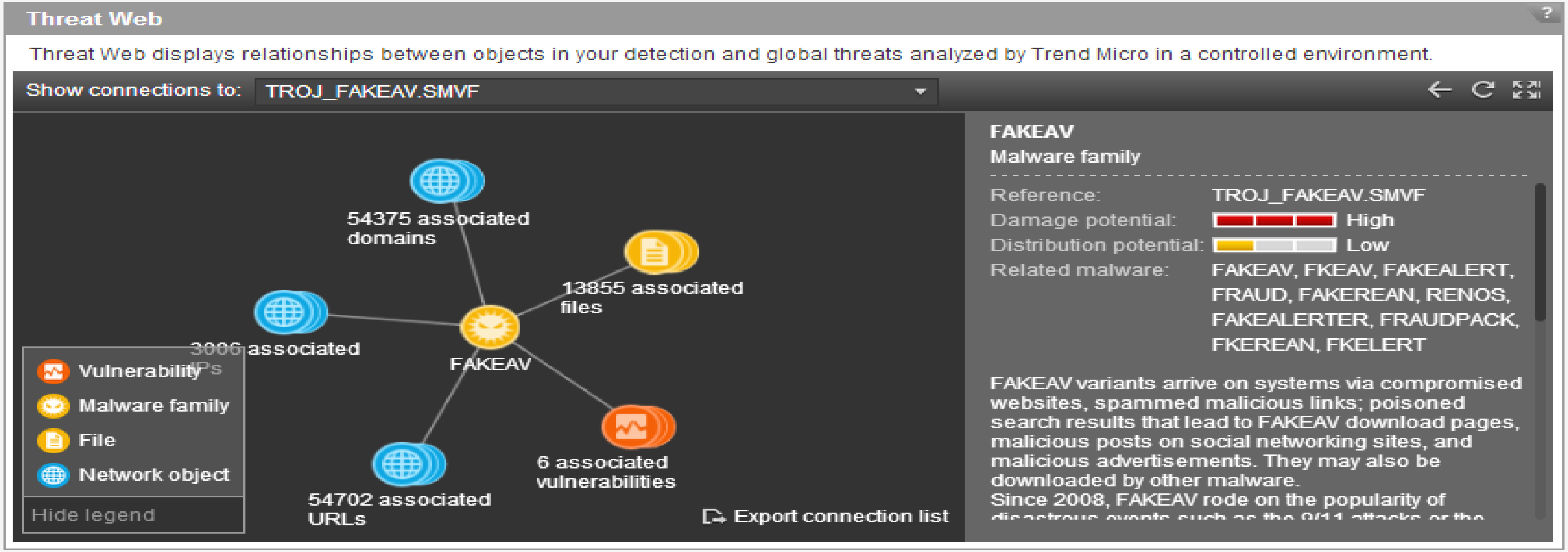
File Behaviors

See The Threat Entity Connectivity

Connectivity From Different Data Sources



ThreatWeb: Threat Entities as a Graph



Threat Web

Threat Web displays relationships between objects in your detection and global threats analyzed by Trend Micro in a controlled environment.

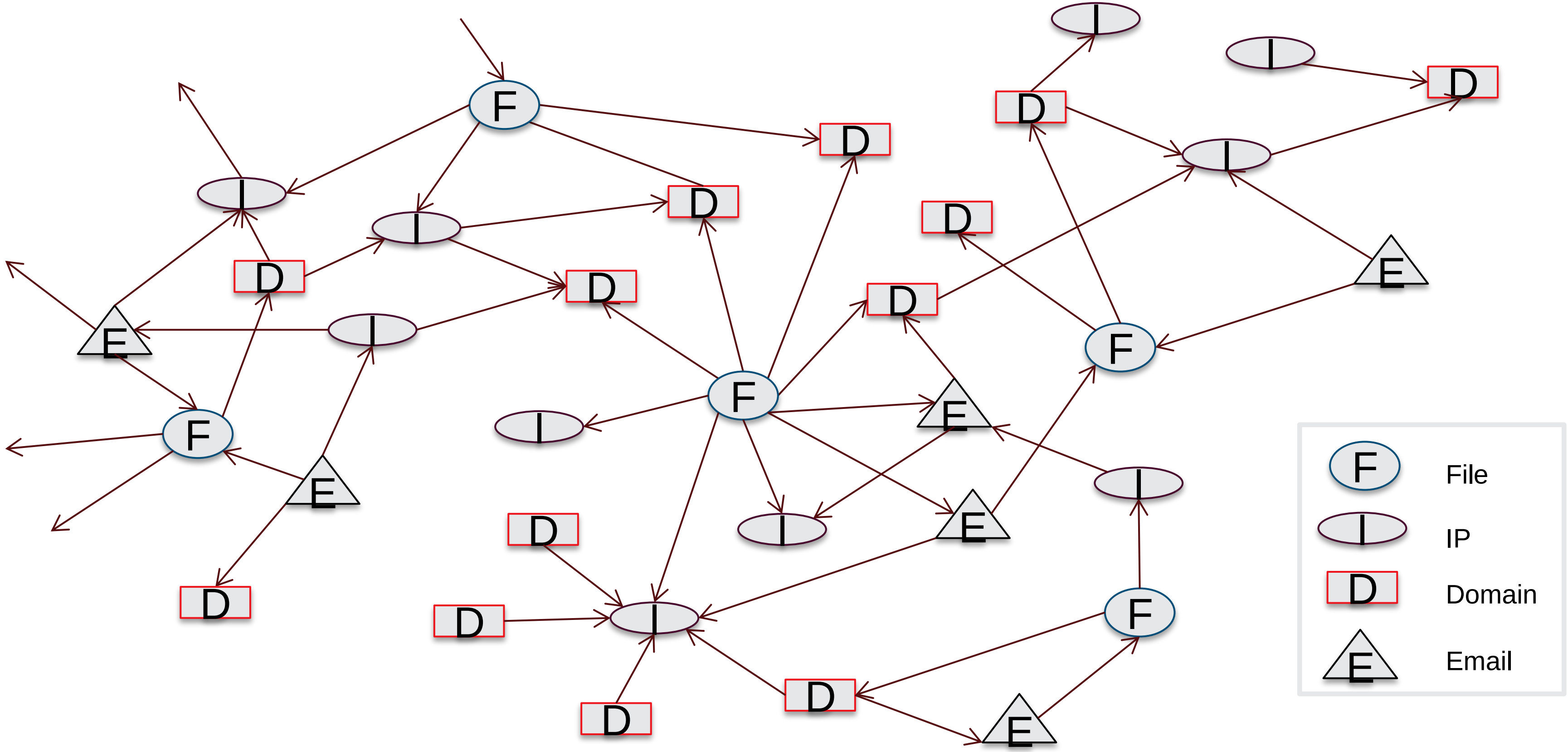
Show connections to: TROJ_FAKEAV.SMVF

FAKEAV
Malware family

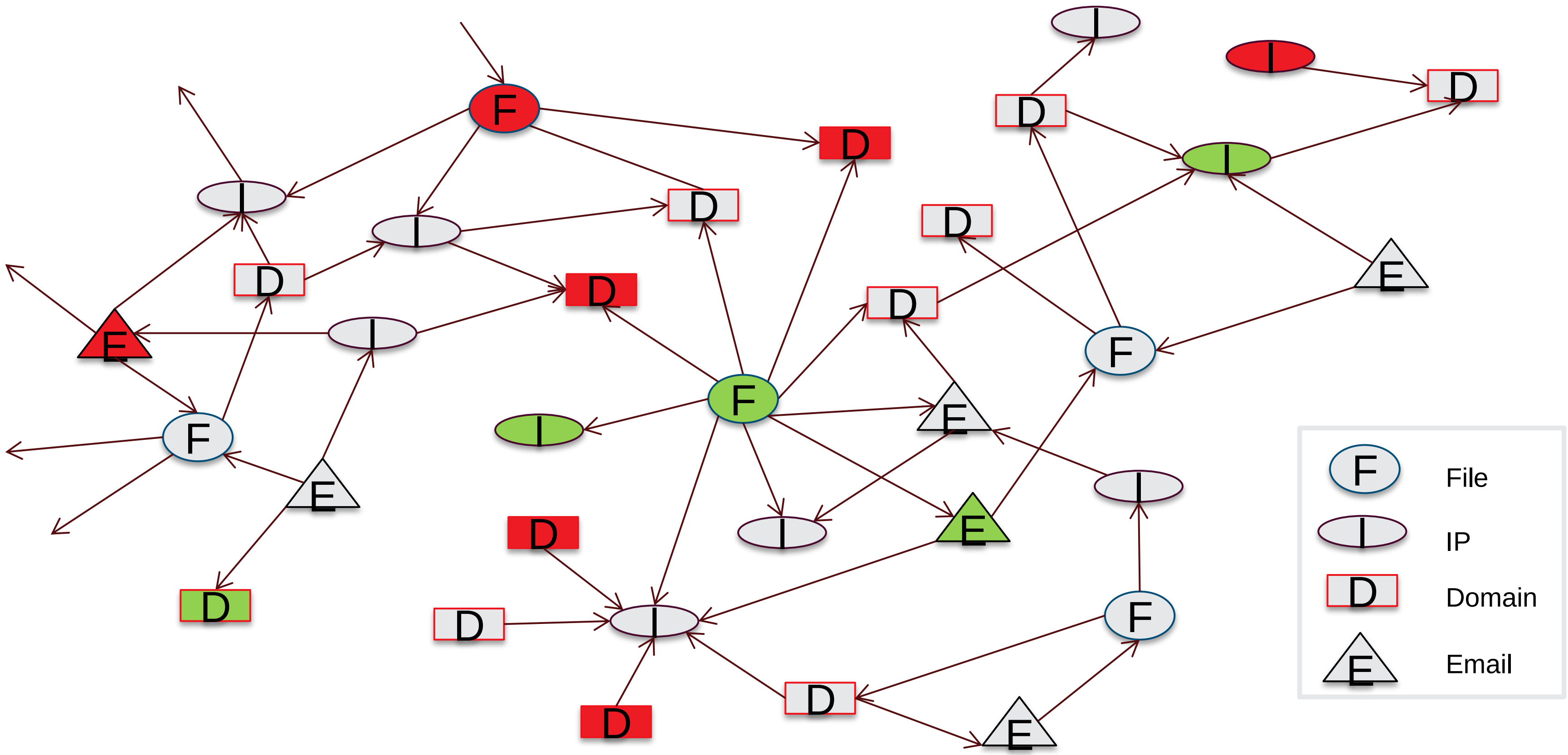
Reference: TROJ_FAKEAV.SMVF
Damage potential: High
Distribution potential: Low
Related malware: FAKEAV, FKEAV, FAKEALERT, FRAUD, FAKEREAN, RENOS, FAKEALERTER, FRAUDPACK, FKEREAN, FKELERT

FAKEAV variants arrive on systems via compromised websites, spammed malicious links; poisoned search results that lead to FAKEAV download pages, malicious posts on social networking sites, and malicious advertisements. They may also be downloaded by other malware.
Since 2008, FAKEAV rode on the popularity of disastrous events such as the 9/11 attacks or the

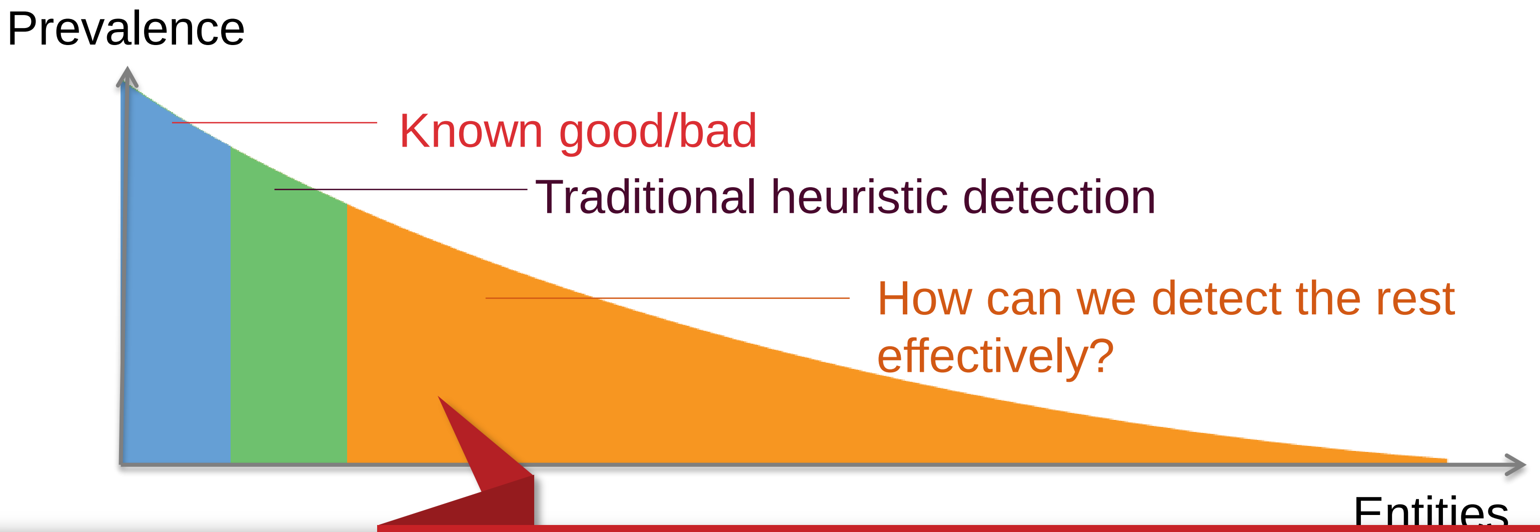
Threat Entities Relation Graph



Most Entity Reputations are Unknown



Security Solution Dilemma – ***Long Tail***



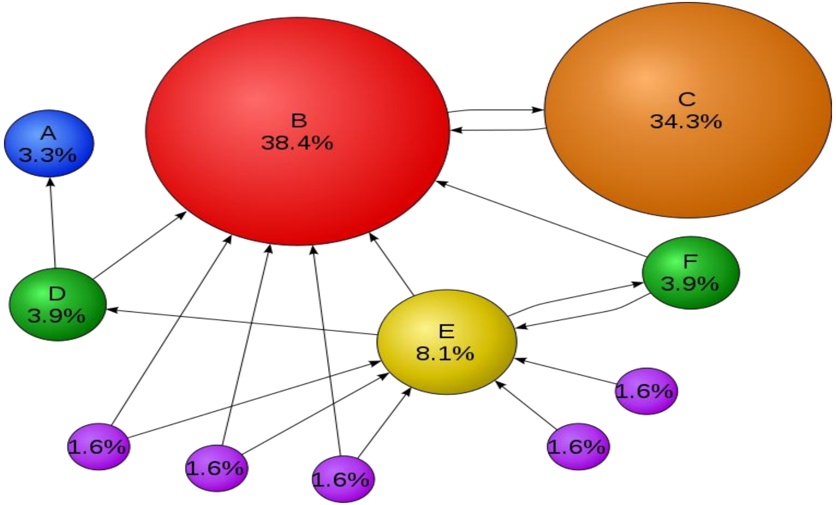
Known good/bad

Traditional heuristic detection

How can we detect the rest effectively?

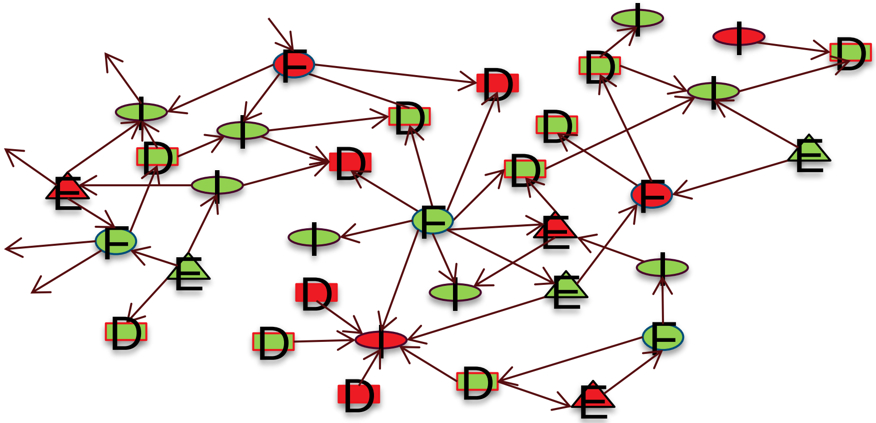
Big Data can help!

Inspired by PageRank



- Too many un-visited pages!
- Users browse pages through links
- Let users' clicks (BIG DATA) tell us the rankings of those un-visited pages!

Revised PageRank Algorithm



- Too many un-rated threat entities!
- Malware activities interact with threat entitles
- Let malware’s behaviors (BIG DATA) tell us the **reputations** of those un-rated threat entities!

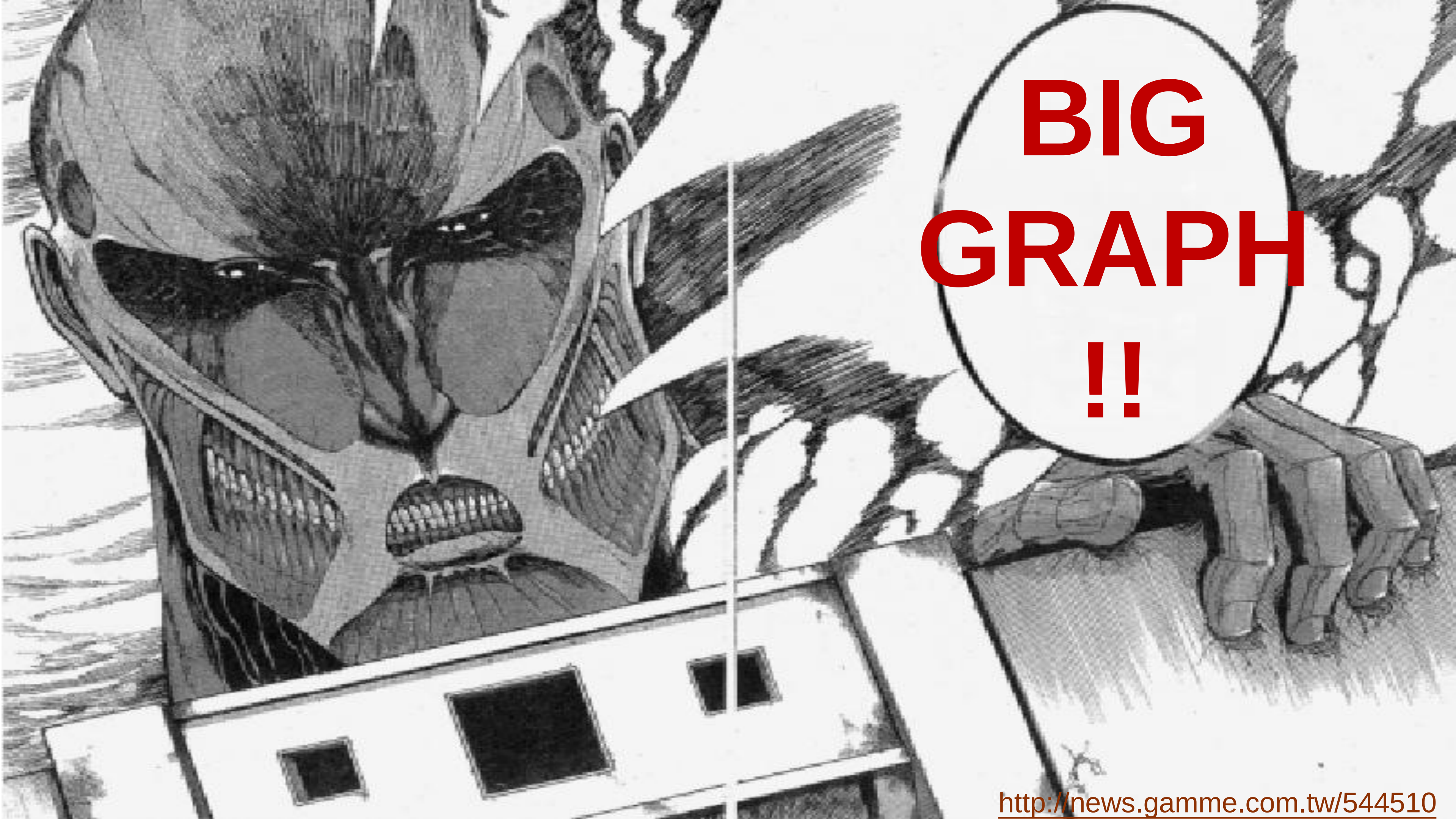
The Graph Problem

The Problems

- Store large size of Graph data
- Access large size of Graph data
- Process large size of Graph data

Data volume

- Dump ~450MB (150 bytes * 3,000,000 records) data into Graph per day
  - Extract from 3GB of data
- Keep it for 3 month
  - ~450MB * 90 = ~40,500MB = ~39GB
  - With Snappy compression
  - ~20 - 22GB
- Dataset
  - ~40,000,000 vertices and ~100,000,000 edges
- Data query volume about hundreds of thousands per day

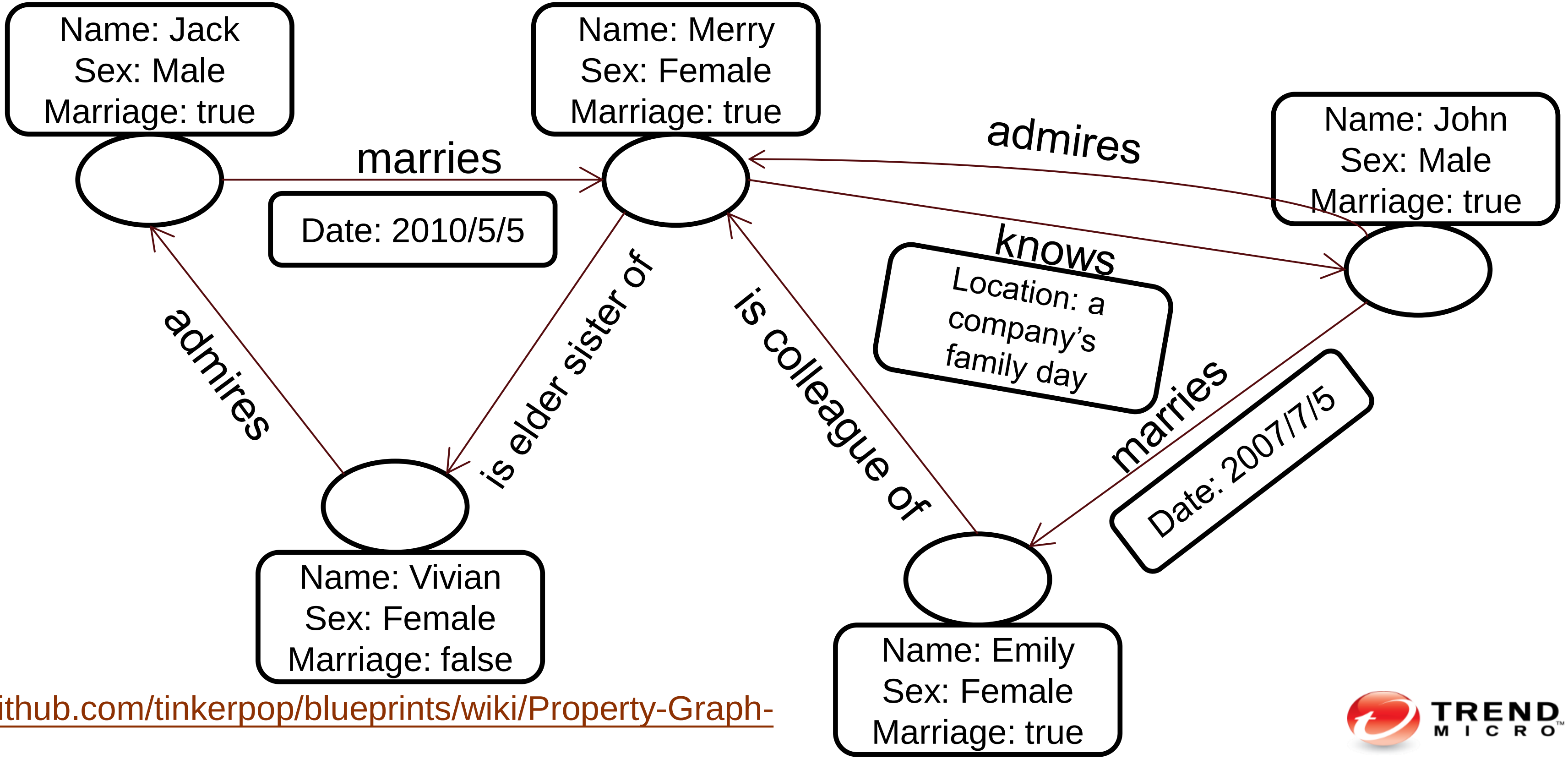


http://news.gamme.com.tw/544510

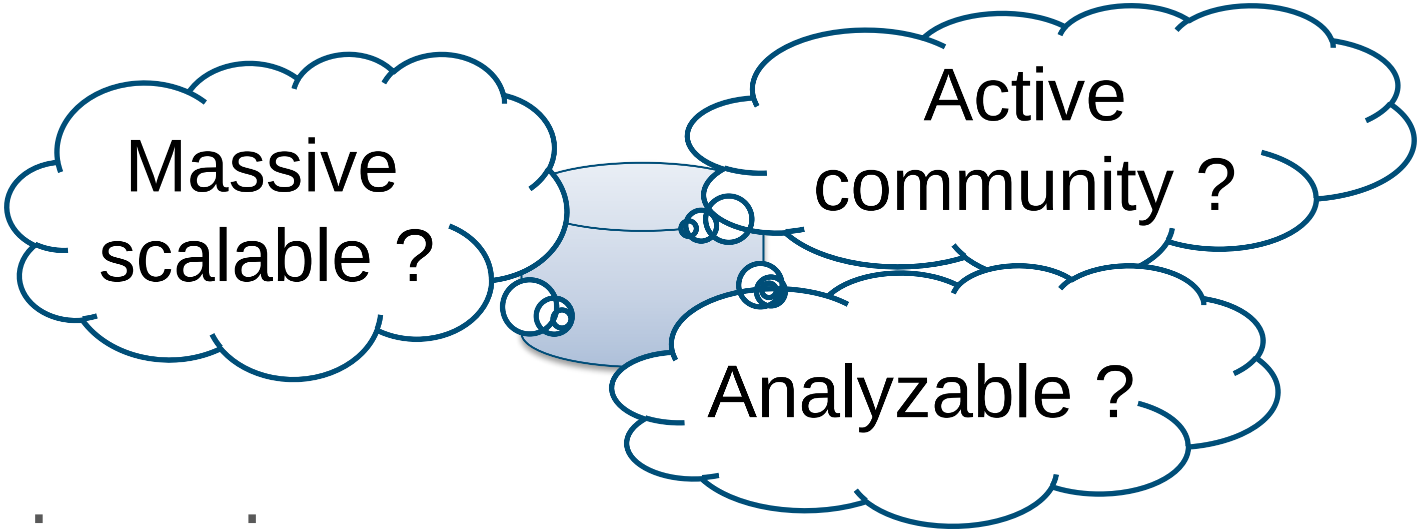
Store

Property Graph Model

From a soap opera…



https://github.com/tinkerpop/blueprints/wiki/Property-Graph-Model



Massive scalable ?

Active community ?

Analyzable ?

The winner is…

- We use HBase as a Graph Storage
  - Google BigTable and PageRank
  - HBaseCon2012
    - Storing and manipulating graphs in HBase

Use HBase to store Graph data (1/3)

- Tables
  - create 'vertex', {NAME => 'property', BLOOMFILTER => 'ROW', COMPRESSION => ‘SNAPPY', TTL => '7776000'}
  - create 'edge', {NAME => 'property', BLOOMFILTER => 'ROW', COMPRESSION => ‘SNAPPY', TTL => '7776000'}

Use HBase to store Graph data (2/3)

- Schema design
  - Table: vertex

    ‘<vertex-id>||*<entity-type>’, ‘property:<property-key>*@*<property-value-type>’, <property-value>*

  - Table: edge

    ‘<vertex1-row-key>-->*<label>*-->*<vertex2-row-key>’, ‘property:<property-key>*@*<property-value-type>’, <property-value>*

Use HBase to store Graph data (3/3)

- Sample
  - Table: vertex

‘myapps-ups.com||domain’, ‘property:ip@String’, ‘…’
‘myapps-ups.com||domain’, ‘property:asn@String’, ‘…’
…
‘track.muapps-ups.com/InvoiceA1423AC.JPG.exe||url’, ‘property:path@String’, ‘…’
‘track.muapps-ups.com/InvoiceA1423AC.JPG.exe||url’, ‘property:parameter@String’, ‘…’

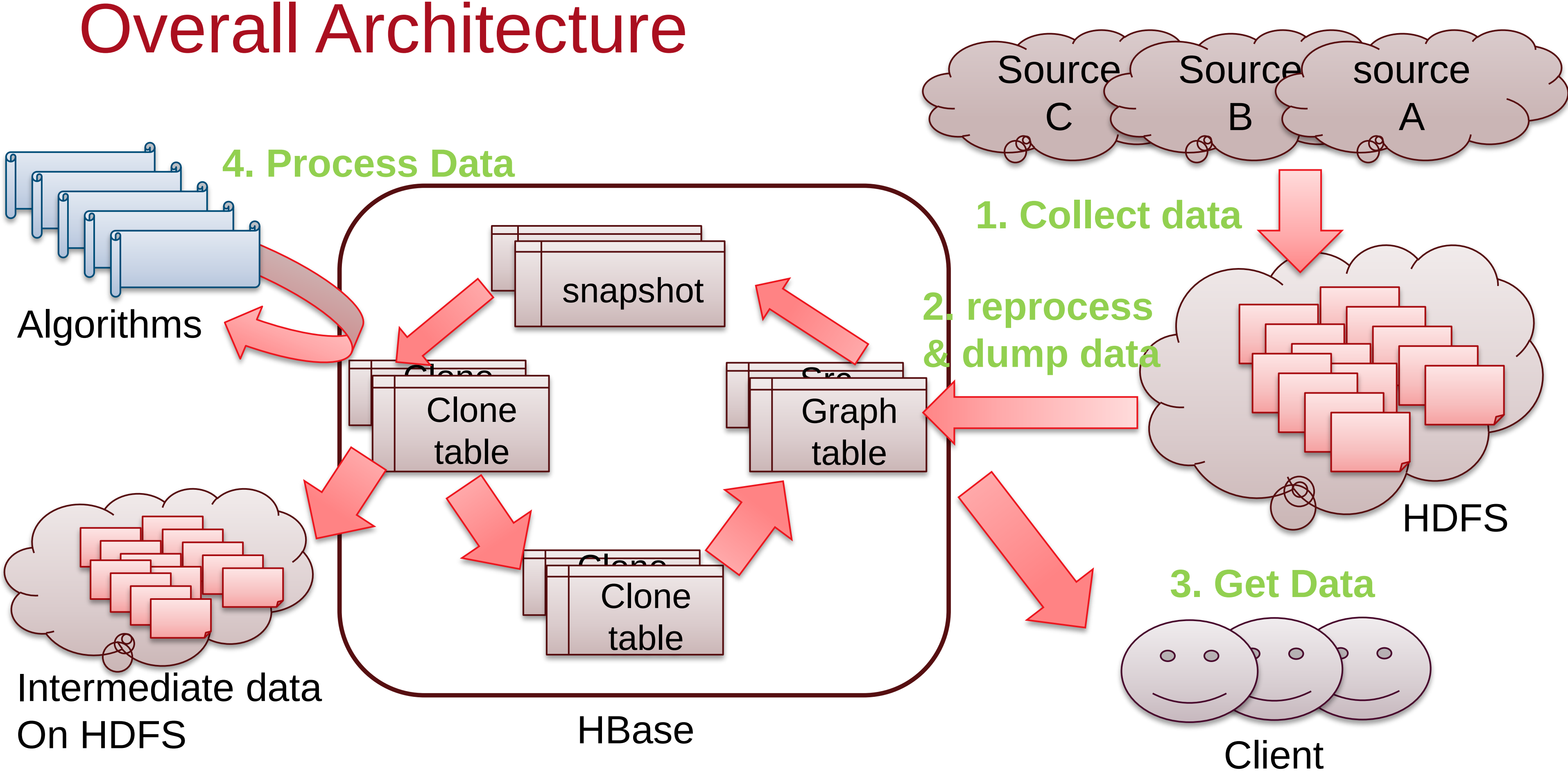
  - Table: edge

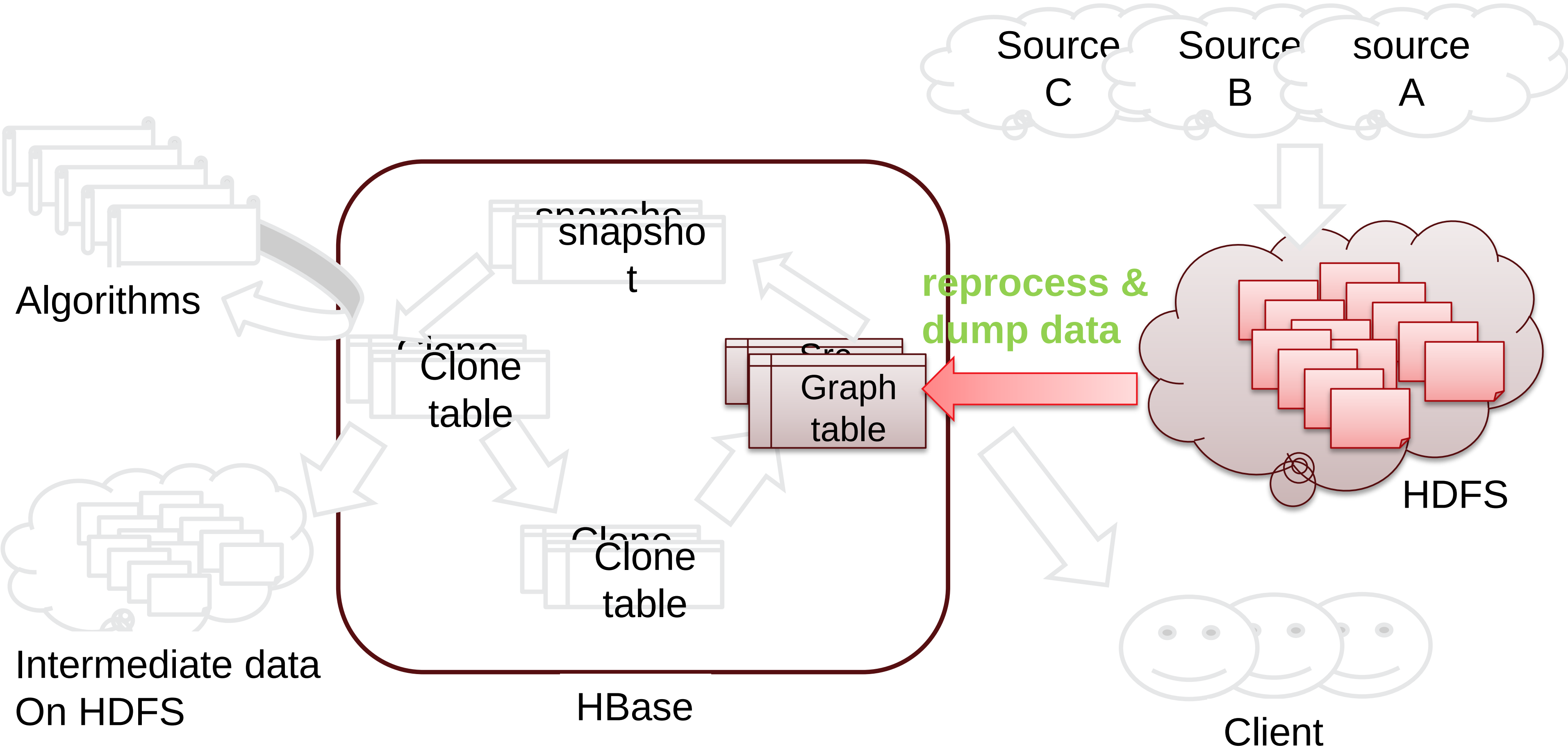
‘myapps-ups.com||domain-->host-->track.muapps-ups.com/InvoiceA1423AC.JPG.exe||url’, ‘property:property1’, ‘…’
‘myapps-ups.com||domain-->host-->track.muapps-ups.com/InvoiceA1423AC.JPG.exe||url’, ‘property:property2’, ‘…’

Keep your rowkey length short

- With long rowkey length
  - It does not impact your query performance
  - But it does impact your algorithm MR
    - OutOfMemoryException
- Use something like HASH function to keep your rowkey length short
  - Use the hash value as rowkey
  - Put the original value into a property

Overall Architecture





Source C

Source B

source A

Algorithms

snapsho
t

Clone table

Graph table

Clone table

reprocess & dump data

HDFS

Intermediate data
On HDFS

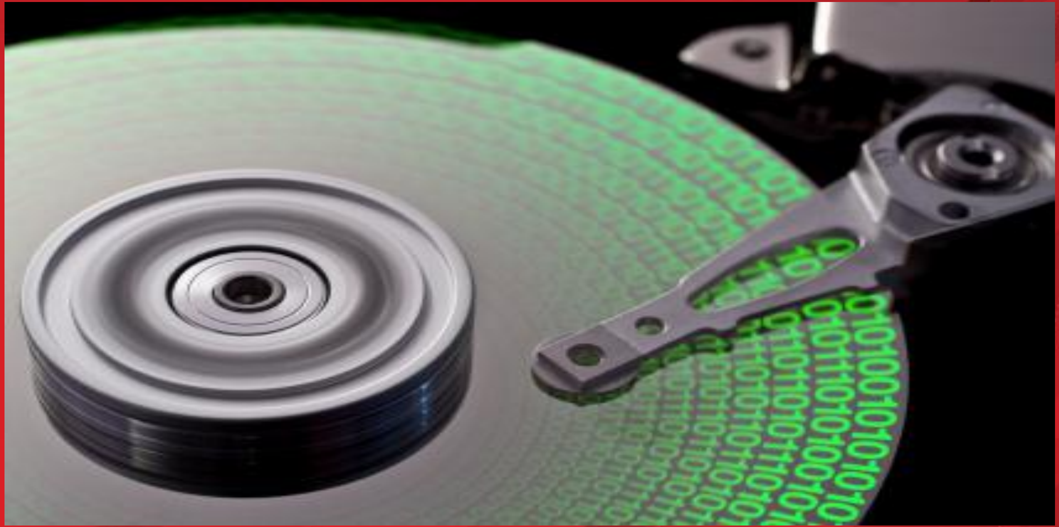
HBase

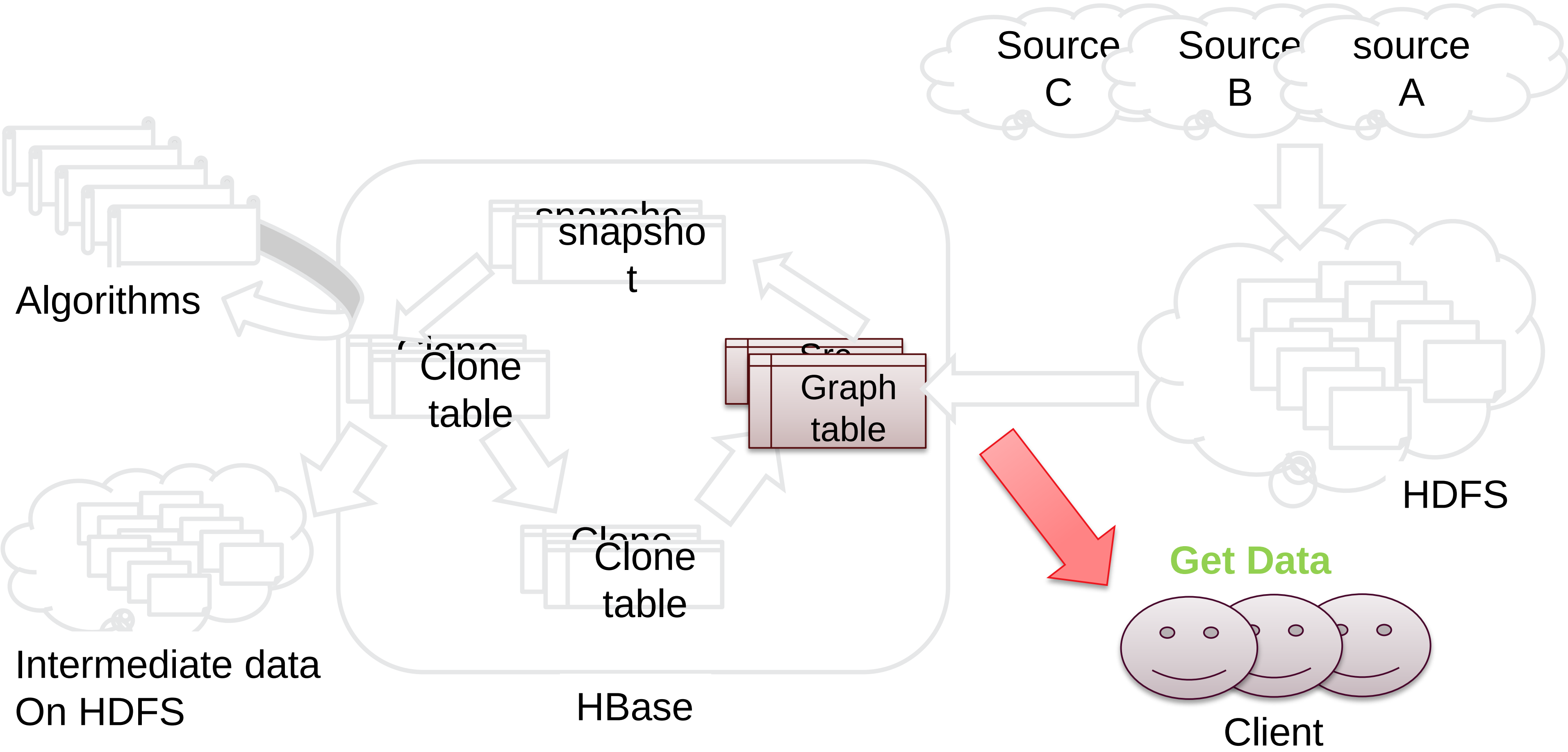
Client

Preprocess and Dump Data

- HBase schema design is simple and human-readable
- It is easy to write your dumping tool if needed
  - MR/Pig/Completebulkload
  - Can write cron-job to clean up the broken-edge data
  - TTL can also help to retire old data
- We already have a lot practices for these tasks

Access





Source
C

Source
B

source
A

Algorithms

snapsho
t

Clone
table

Graph
table

HDFS

Clone
table

Get Data

Intermediate data
On HDFS

HBase

Client

Get Data (1/2)

- A Graph API
- A better semantic for manipulating Graph data
  - As a wrapper for HBase Client API
  - Rather than use HBase Client API directly
- A malware exploring sample

```
Vertex vertex = this.graph.getVertex(“malware");
Vertex subVertex = null;
Iterable<Edge> edges =
               vertex.getEdges(Direction.OUT, “connect", “infect", “trigger");
for(Edge edge : edges) {
  subVertex = edge.getVertex(Direction.OUT);
  ...
}
```

Get Data (2/2)

- We implement blueprints API
  - It provides interfaces as spec. for users to impl.
    - 824 stars, 173 forks on github
  - We can get more benefits from it
    - plug-and-play different Blueprints-enabled graph backends
      - Traversal language, RESTful server, dataflow, etc
      - http://www.tinkerpop.com/
  - Currently basic query methods are implemented

Clients

- Real time Client
  - Client systems
    - they need associated Graph data for a specific entity via RESTful API
  - Usually retrieve two levels of graph data
  - Quick responsiveness supported by HBase
    - With rowkey random access and appropriate schema design
    - *HTable.get(),Scan.setStartRow(), Scan.setStopRow()*
- Batch client
  - Threat experts
  - Pick one entity and how many levels interested in, generate a graph file format used by tools
    - To visualize and navigate what whether users interested in
- Graph Exploring Tools
  - Threat experts
  - Find out sub-graphs by given criteria
    - E.g. How many levels or associated vertices

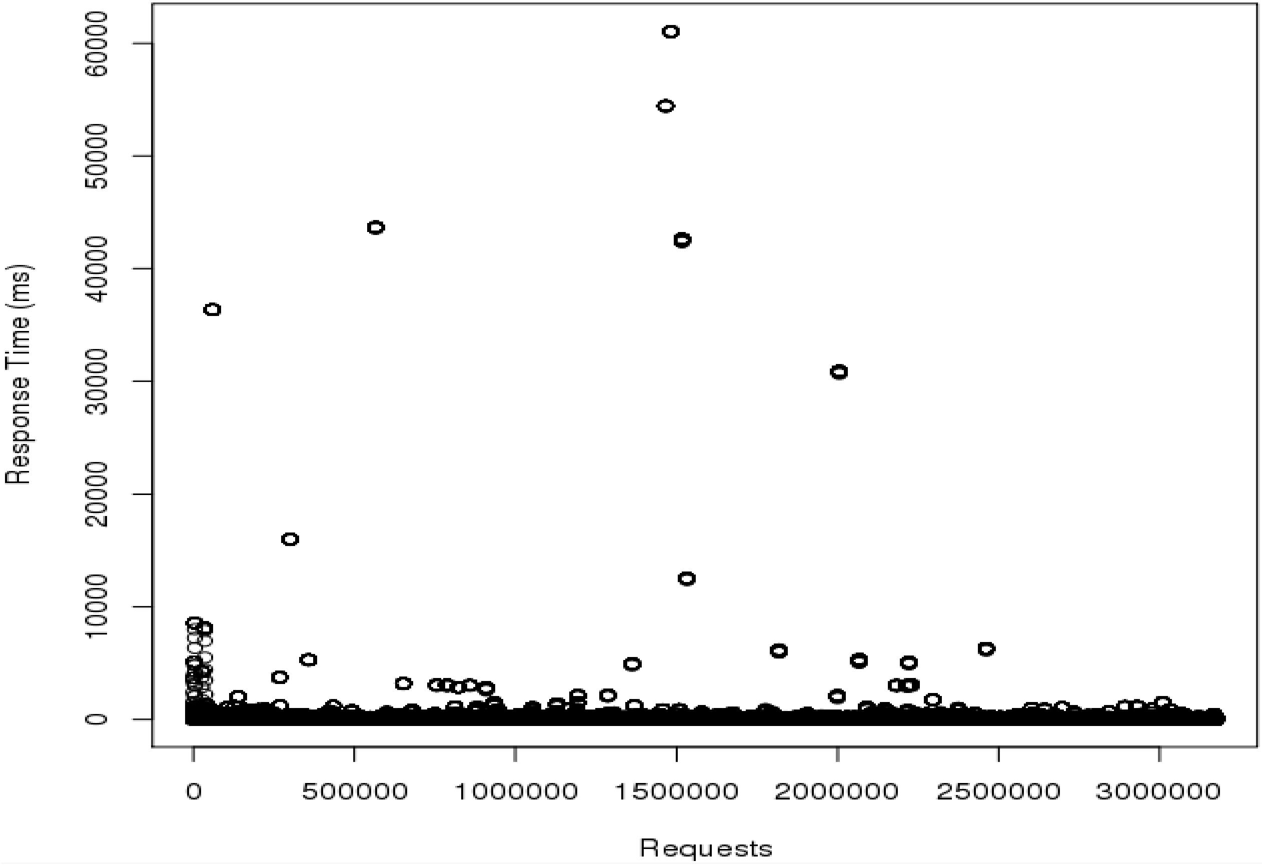
Malware Exploring Performance (1/3)

- one request
  - Use Malware exploring sample again

```
Vertex vertex = this.graph.getVertex(“malware");
Vertex subVertex = null;
Iterable<Edge> edges =
                    vertex.getEdges(Direction.OUT, “connect", “infect", “trigger");
for(Edge edge : edges) {
  subVertex = edge.getVertex(Direction.OUT);
  ...
}
```

  - 1 vertex with 2 levels associated instances (2 ~ 9 vertices)
- Dataset
  - 42,133,610 vertices and 108,355,774 edges
- Total requests
  - 31,764 requests * 100 clients = 3,176,400

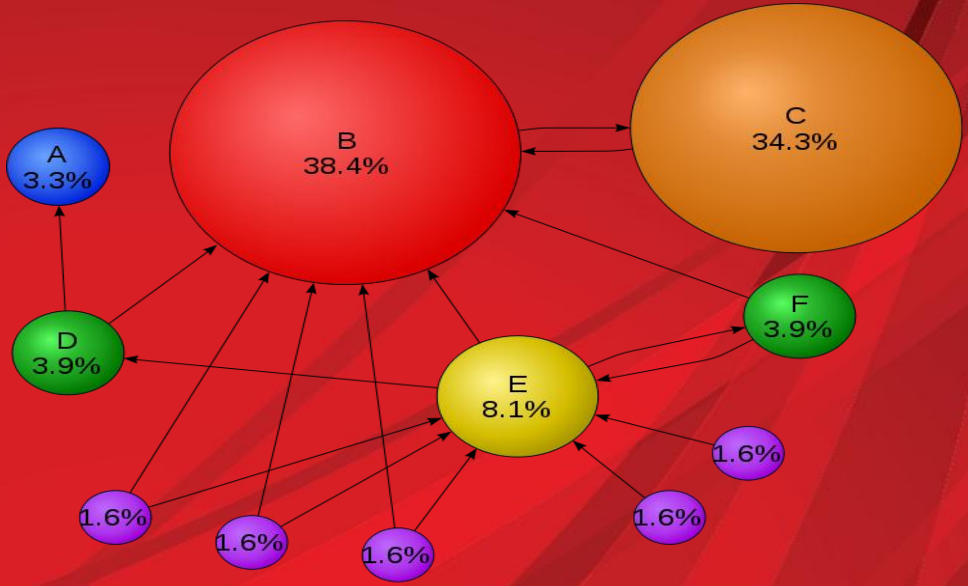
Malware Exploring Performance (2/3)



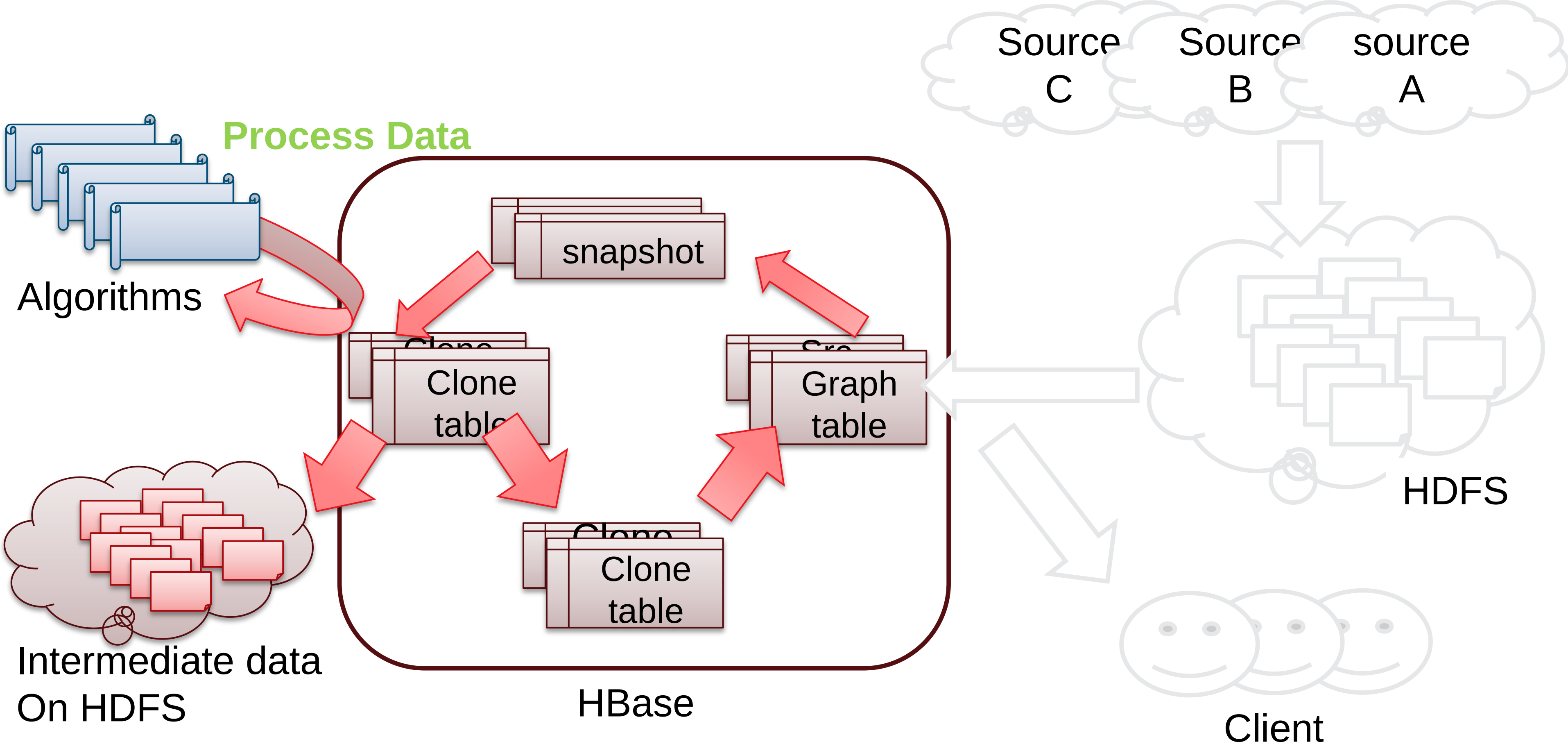
Malware Exploring Performance (3/3)

- Some statistics
  - Mean: 51.61 ms
  - Standard Deviation: 653.57 ms
  - Empirical rule: 68%, 95%, 99.7%
    - 99.7% of requests below 2.1 seconds
- But response time variances still happen
  - Use Cache layer between client and HBase
  - Warm-up after new data come in

Process



Process Data



Source C

Source B

source A

HDFS

Client

Algorithms

snapshot

Clone table

Graph table

Clone table

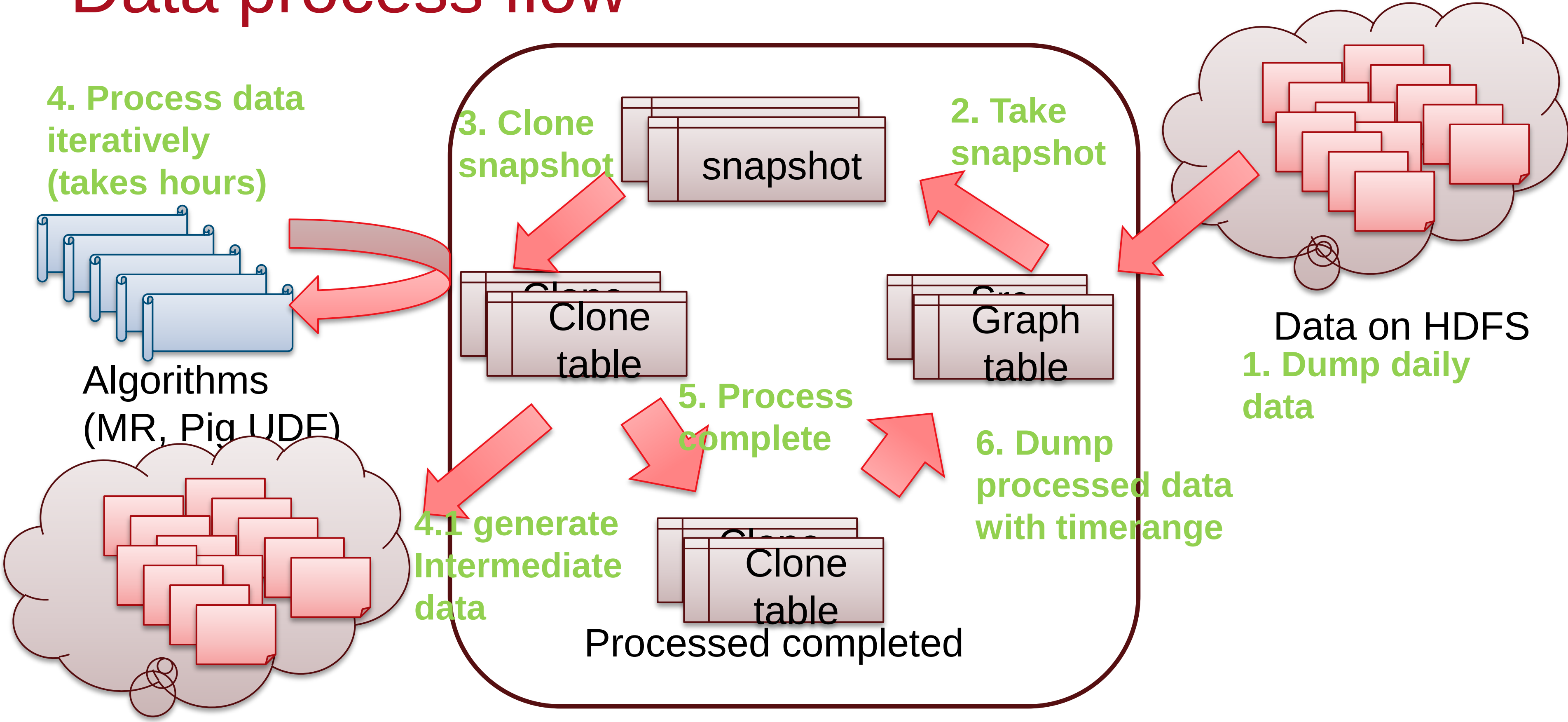
Intermediate data
On HDFS

HBase

- Human-readable HBase schema design
  - Write your own MR
  - Write your own Pig/UDFs

- So we can write the algorithms to further process our graph data
  - To predict unknown reputation by known threats
  - E.g. a revised PageRank algorithm

Data process flow



Data on HDFS

1. Dump daily data

2. Take snapshot

snapshot

3. Clone snapshot

Clone table

4. Process data iteratively (takes hours)

Algorithms (MR, Pig UDF)

4.1 generate Intermediate data

Intermediate data on HDFS

5. Process complete

Clone table

Processed completed

6. Dump processed data with timerange

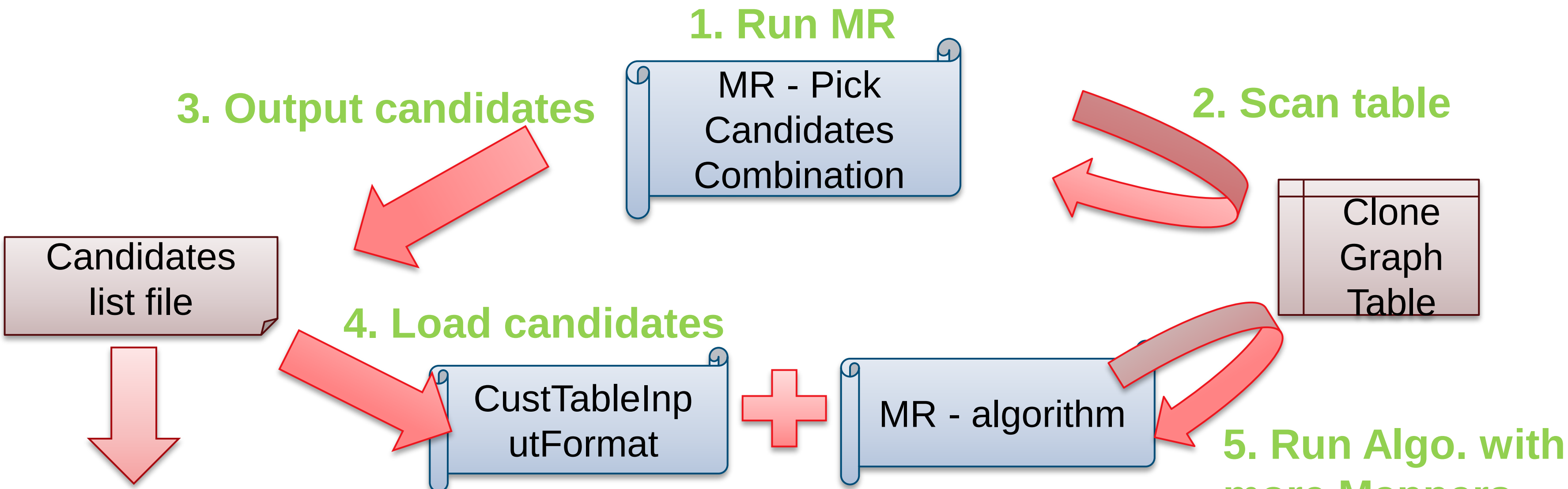
Graph table

HBase

A customized TableInputFormat (1/2)

- One Mapper for one region by default
  - Each Mapper process too much data
    - OutOfMemoryException
    - Too long to process
  - Use small split region size ?
    - Will overload your HBase cluster !!
- Before: about ~40 Mappers
- After: about ~500 Mappers

A customized TableInputFormat (2/2)



<encodedRegionName>\t<startKey>\t<endKey>

…

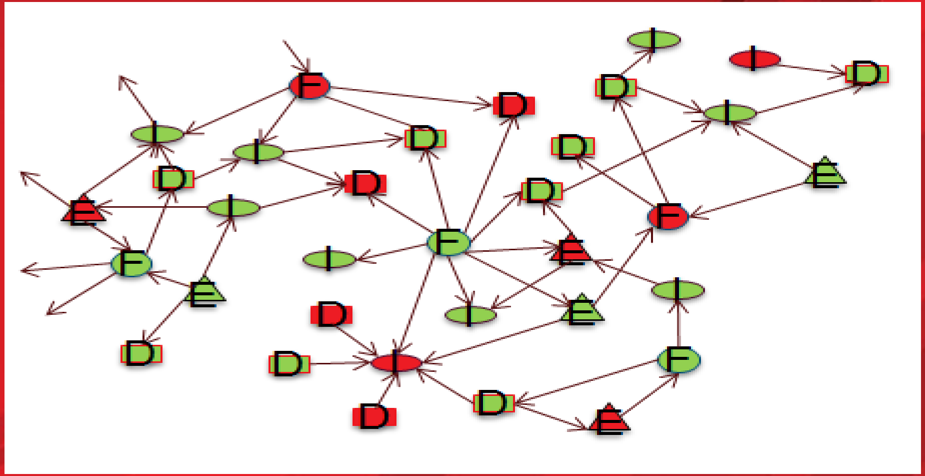
d3d1749f3486e850b263c7ecb2424dd3\tstartKey_1\tendKey_1
d3d1749f3486e850b263c7ecb2424dd3\tstartKey_2\tendKey_2
d3d1749f3486e850b263c7ecb2424dd3\tstartKey_3\tendKey_3
Cd91c08d656a19bdb180e0b7f8896575\tstartKey_4\tendKey_4
Cd91c08d656a19bdb180e0b7f8896575\tstartKey_5\tendKey_5

…

HGraph

- A project is open and put on github
  - https://github.com/trendmicro/HGraph
- A partial impl. released from our internal project
  - Follow HBase schema design
  - Read data via Blueprints API
  - Process data with our pagerank default impl.
- Download or ‘*git clone*’ it
  - Use ‘*mvn clean package*’
  - Run on unix-like OS
    - Use windows may encounter some errors

PageRank Result



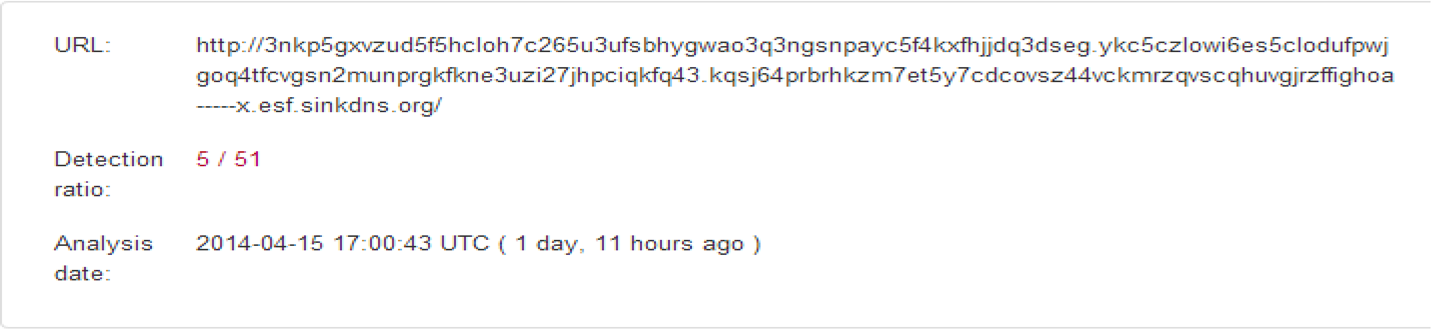
Experiment Result

- Testing Dataset
  - 42,133,610 vertices and 108,355,774 edges
  - 1 vertex usually associates 2 ~ 9 vertices
  - 4.13% of the vertices are **known bad**
  - 0.09% of the vertices are **known good**
  - The rests are unknown
- Result
  - Runs 34hrs for running 23 iterations.
  - **1,291** unknown vertices are ranked out
  - Top 200 has **99%** accuracy (explain later)

Suspicious DGA Discovered

- 3nkp***cq-----x.esf.sinkdns.org
  - 196 domains from Domain Generated Algorithms

virustotal

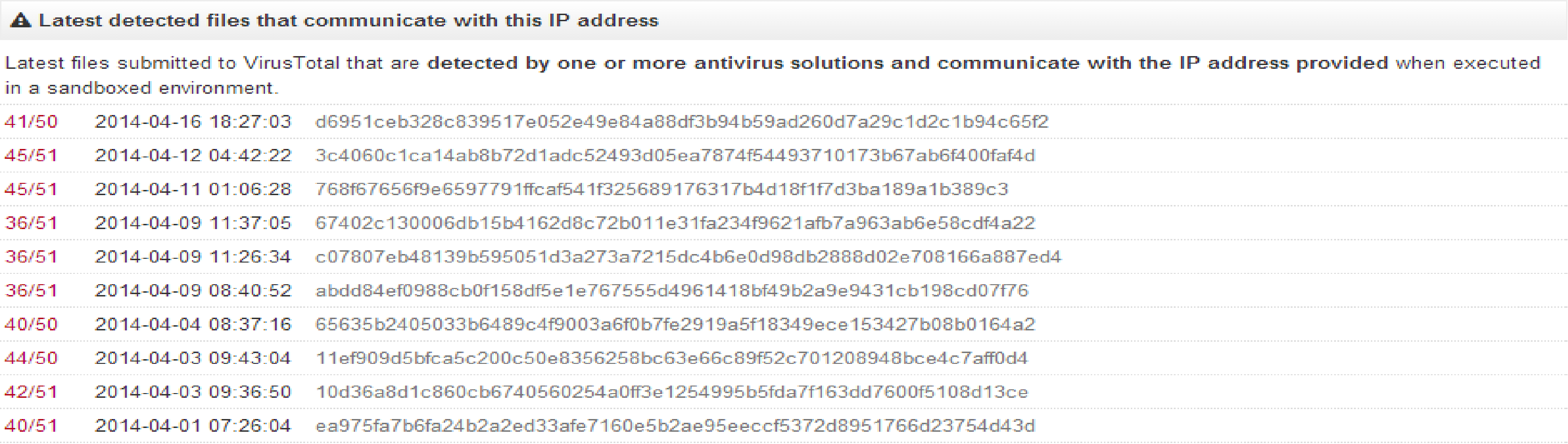


URL:	http://3nkp5gxvzud5f5hcloh7c265u3ufsbhygwao3q3ngsnpayc5f4kxfhjjdq3dseg.ykc5czlowi6es5clodufpwjgoq4tfcvgsn2munprgkfkne3uzi27jhpciqkfq43.kqsj64prbrhkzm7et5y7cdcovsz44vckmrzqvscqhuvgjrzffighoa-----x.esf.sinkdns.org/
Detection ratio:	5 / 51
Analysis date:	2014-04-15 17:00:43 UTC (1 day, 11 hours ago)

https://www.virustotal.com/en/url/871004bd9a0fe27e61b0519ceb8457528ea00da0e7ffdc44d98e759ab3e3caa1/analysis/

Untested But Highly Malware Related IP

- 67.*.*.132
  - Categorized as “Computers / Internet”, not tested



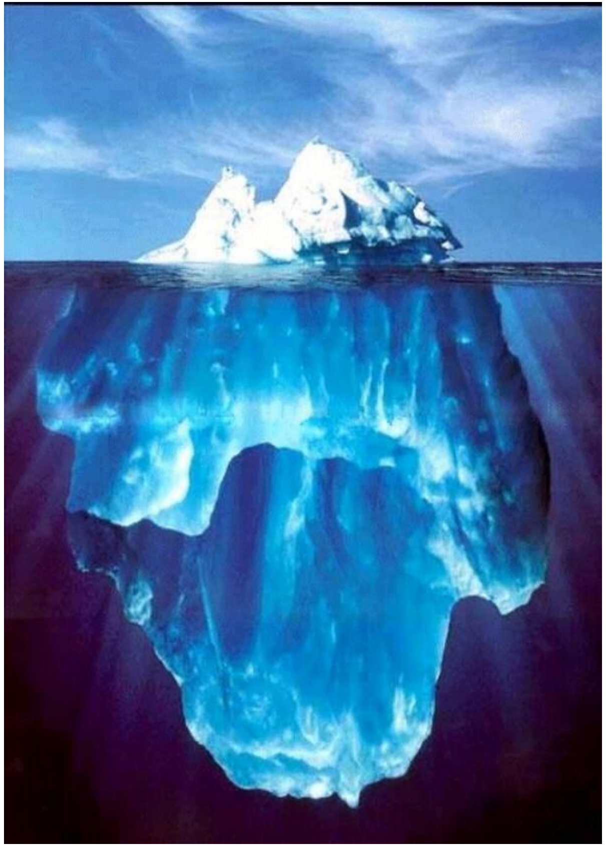
⚠ **Latest detected files that communicate with this IP address**

Latest files submitted to VirusTotal that are **detected by one or more antivirus solutions and communicate with the IP address provided** when executed in a sandboxed environment.

41/50	2014-04-16 18:27:03	d6951ceb328c839517e052e49e84a88df3b94b59ad260d7a29c1d2c1b94c65f2
45/51	2014-04-12 04:42:22	3c4060c1ca14ab8b72d1adc52493d05ea7874f54493710173b67ab6f400faf4d
45/51	2014-04-11 01:06:28	768f67656f9e6597791ffcaf541f325689176317b4d18f1f7d3ba189a1b389c3
36/51	2014-04-09 11:37:05	67402c130006db15b4162d8c72b011e31fa234f9621afb7a963ab6e58cdf4a22
36/51	2014-04-09 11:26:34	c07807eb48139b595051d3a273a7215dc4b6e0d98db2888d02e708166a887ed4
36/51	2014-04-09 08:40:52	abdd84ef0988cb0f158df5e1e767555d4961418bf49b2a9e9431cb198cd07f76
40/50	2014-04-04 08:37:16	65635b2405033b6489c4f9003a6f0b7fe2919a5f18349ece153427b08b0164a2
44/50	2014-04-03 09:43:04	11ef909d5bfca5c200c50e8356258bc63e66c89f52c701208948bce4c7aff0d4
42/51	2014-04-03 09:36:50	10d36a8d1c860cb6740560254a0ff3e1254995b5fda7f163dd7600f5108d13ce
40/51	2014-04-01 07:26:04	ea975fa7b6fa24b2a2ed33afe7160e5b2ae95eeccf5372d8951766d23754d43d

https://www.virustotal.com/en/ip-address/67.*.*.132/information/

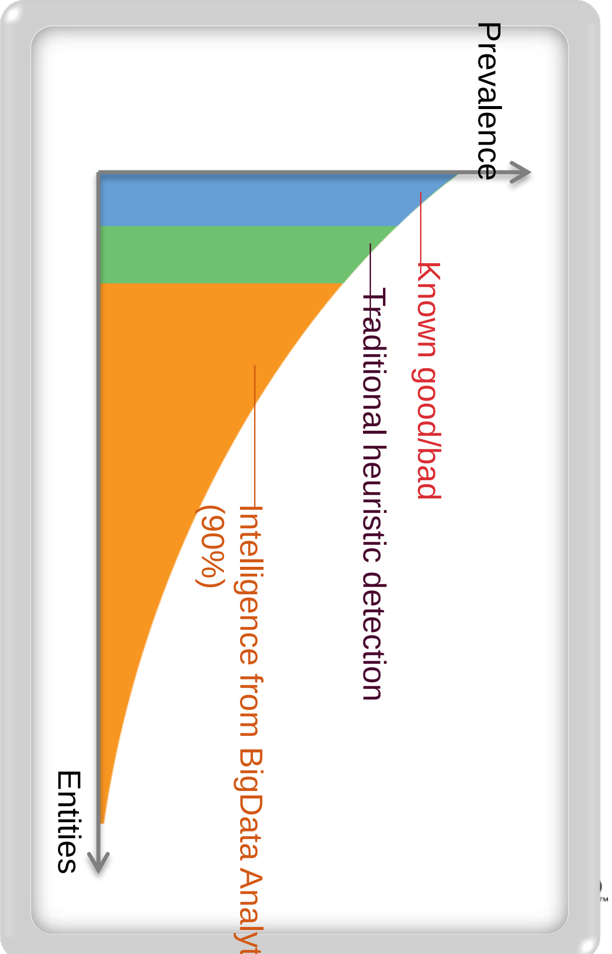
Discover What We Don’t Know



Security in Old Days

Cannot Protect What You Cannot See

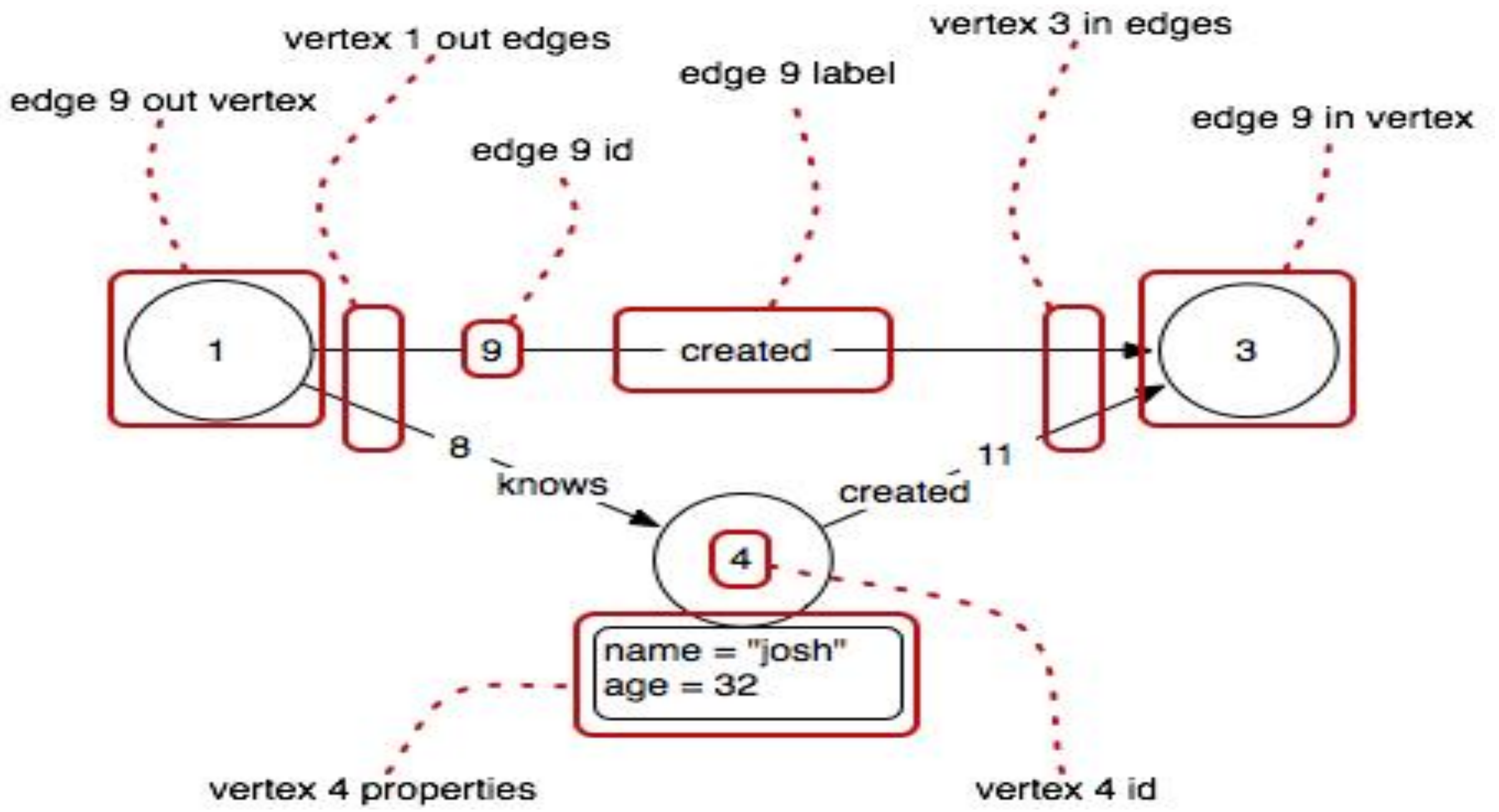
Next Generation Security
Unleash the Power of Data



Q&A

Backups

Property Graph Model (2/2)



Property Graph Model Definition

- A property graph has these elements
  - a set of vertices
    - each vertex has a unique identifier.
    - each vertex has a set of outgoing edges.
    - each vertex has a set of incoming edges.
    - each vertex has a collection of properties defined by a map from key to value.
  - a set of edges
    - each edge has a unique identifier.
    - each edge has an outgoing tail vertex.
    - each edge has an incoming head vertex.
    - each edge has a label that denotes the type of relationship between its two vertices.
    - each edge has a collection of properties defined by a map from key to value.

About regions

- Keep reasonable amount of regions for each regionserver

<hbase.regionserver.global.memstore.upperLimit> / <hbase.hregion.memstore.flush.size> = <active-regions-per-rs>

*e.g. (10G * 0.4) / 128MB = 32 active regions*

HBase Sizing Notes by Lars George

- Notice your splitted regions from one table
  - Dump data daily, cause regions splitting
  - Make sure your regions scattered evenly on each regionserver