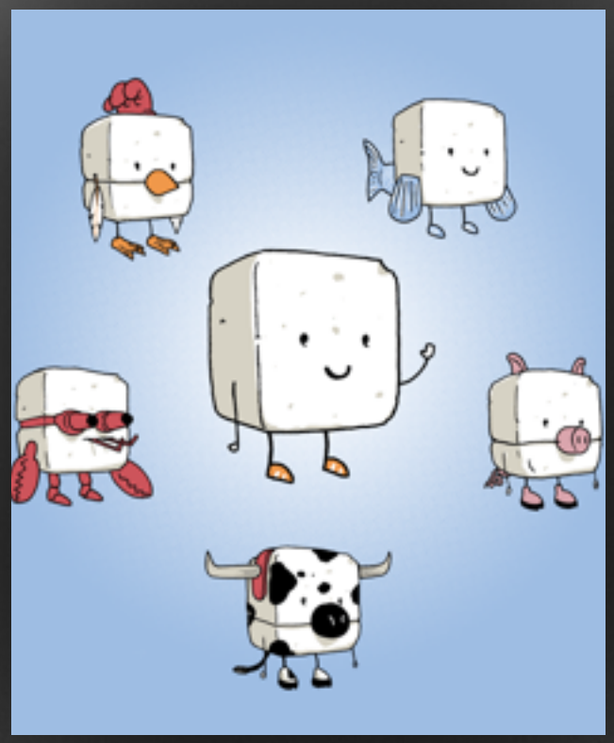
Reversing on the Edge

Jason Jones
Arbor ASERT

Jasiel Spelman
HPSR ZDI

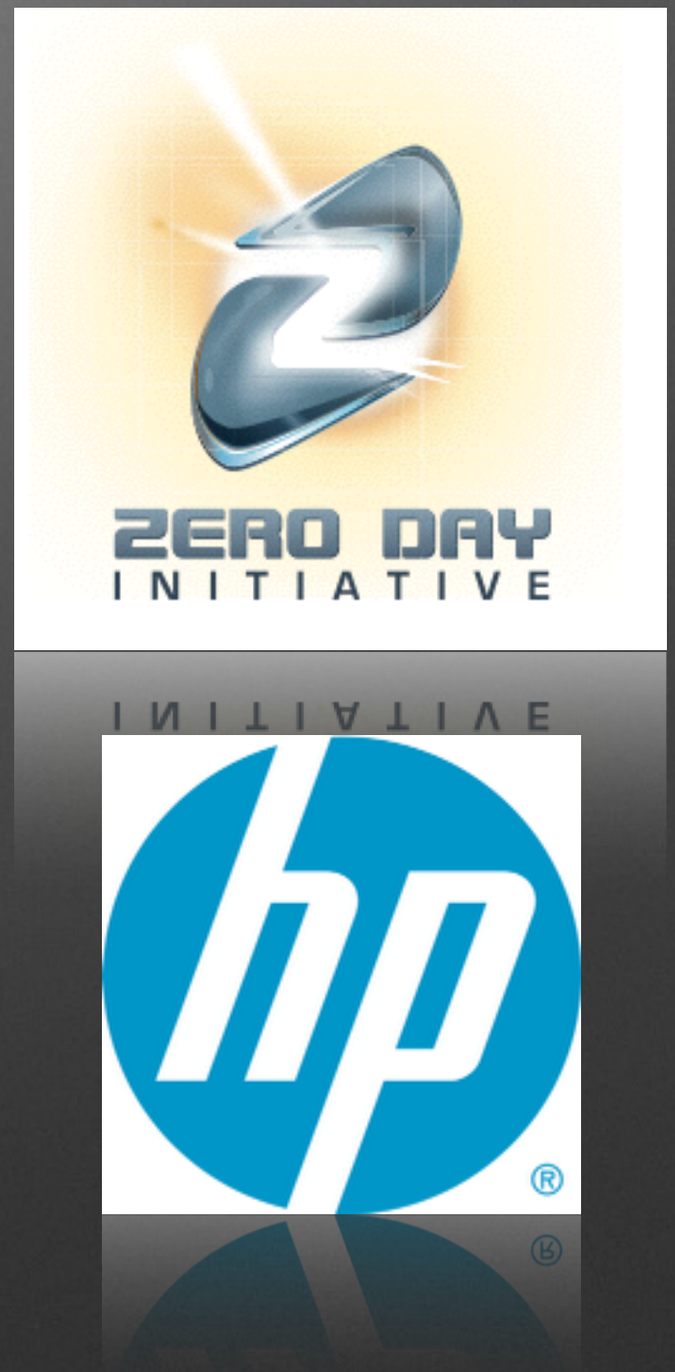
Jason Jones

- Sr Sec Research Analyst @ Arbor
  - ex-TippingPoint ASI
- Primarily reverse malware
- Interests / Research
  - DDoS
  - Botnet tracking
  - Malware Clustering
  - Bug hunting
  - RE Automation



Jasiel Spelman

- Security Researcher with HP's Security Research team
- Member of the Zero Day Initiative
- Interested in static analysis since taking Binary Literacy by Rolf Rolles



So… what are these GraphDBs you speak of?

- Very much like it sounds
- Database designed to store vertices, edges, and properties attached to those edges
- Indexes can be created on properties
- Graph traversals go from one vertex and follow edges until a condition is met
- Leverage theorems / research in Graph Theory
  - Can implement many of these things in RDBMS
  - Lose ability to apply graph theory if you do that
- Primarily written in Java
  - It’s apparently the ‘big data’ language

GraphDB vs RDBMS

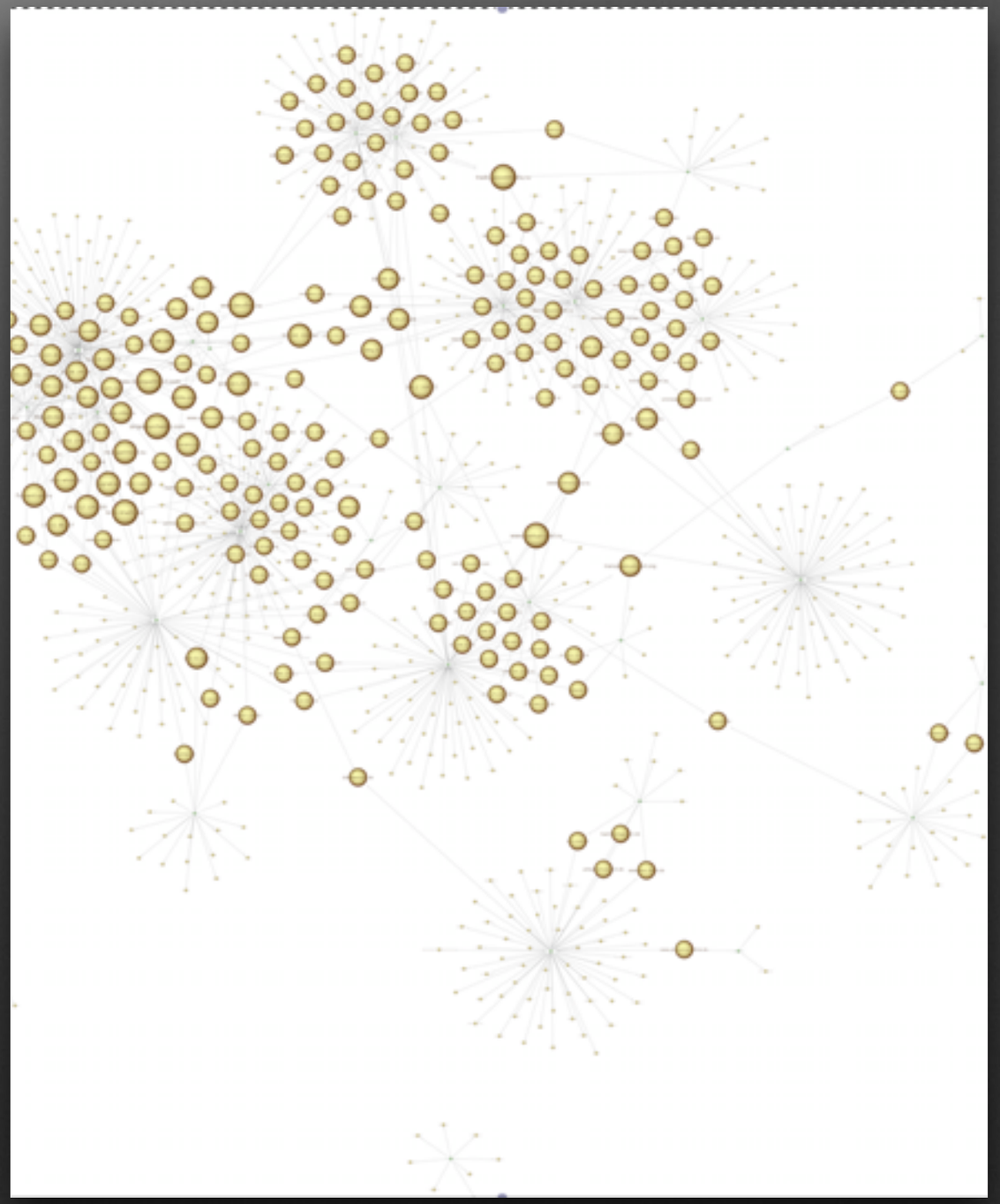
- RDBMS == Relational Database Management System
- Tried and true manner of storing data
- Individual data units as "rows" in a table
- Structured, tied to the schema for the table
- Relationships defined against a table
  - Table A is related to table B by column C

GraphDB vs RDBMS

- Graphs initially lost against RDBMS
  - Too space intensive
- Individual data units as "nodes" within the graph
- Loosely structured
- Relationships defined against the node
  - Node A is related to node B by property C

Maltego

- Created by Imperva
- Multi-platform desktop app
- Good for intel gathering / correlation
- Reversing? probably not
- Scale problems with many thousands of IP / host nodes



TitanGraph

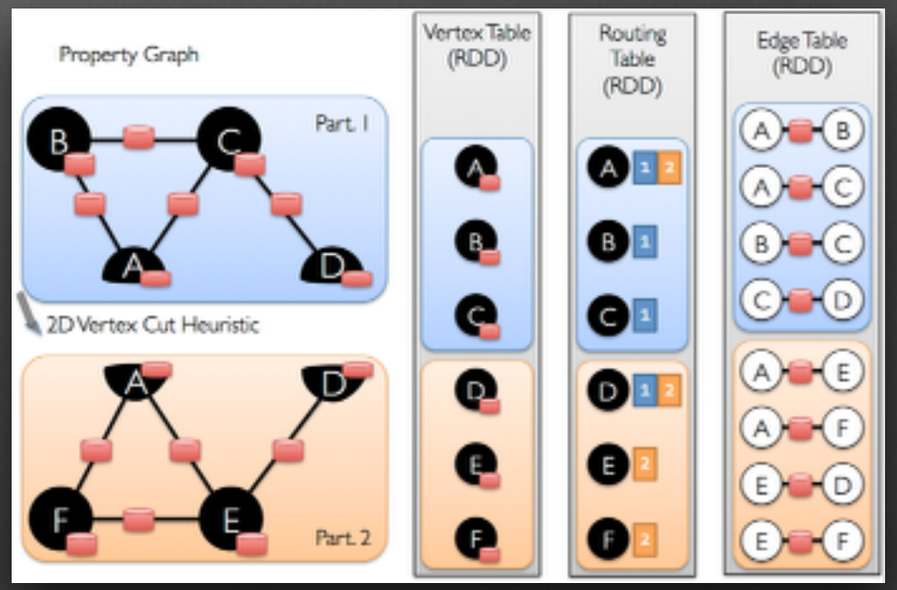
- Made by Aurelius
- Designed to handle large scale data
  - MSHTML/MSO Disassembly?
- Cassandra / HBase / etc DB backend support
- Gremlin Query Language
- Multi-language support via Rexster
  - RexPro / Bulbs for Python
  - Thunderdome also, but appears dead
- JJo’s favorite

Gremlin Query Language

- Simple query language to traverse query graph paths
- Developed by Titan devs, also supported in other GraphDBs
- Examples:
  - gremlin> hercules.out('battled').map
    - ==>{name=nemean, type=monster}
    - ==>{name=hydra, type=monster}
    - ==>{name=cerberus, type=monster}
  - gremlin> hercules.outE('battled').has('time',T.gt,1).inV.name
    - ==>hydra
    - ==>cerberus
  - gremlin> pluto.out('brother').as('god').out('lives').as('place').select{it.name}
    - ==>[god:jupiter, place:sky]
    - ==>[god:neptune, place:sea]

Spark GraphX

- Apache Spark is “fast and general-purpose cluster computing system”
  - Supports Java, Scala, Python
  - Alternative to Hadoop
  - The new “hotness” for data crunching
- GraphX is the Graph Processing portion of Spark



Spark GraphX Features

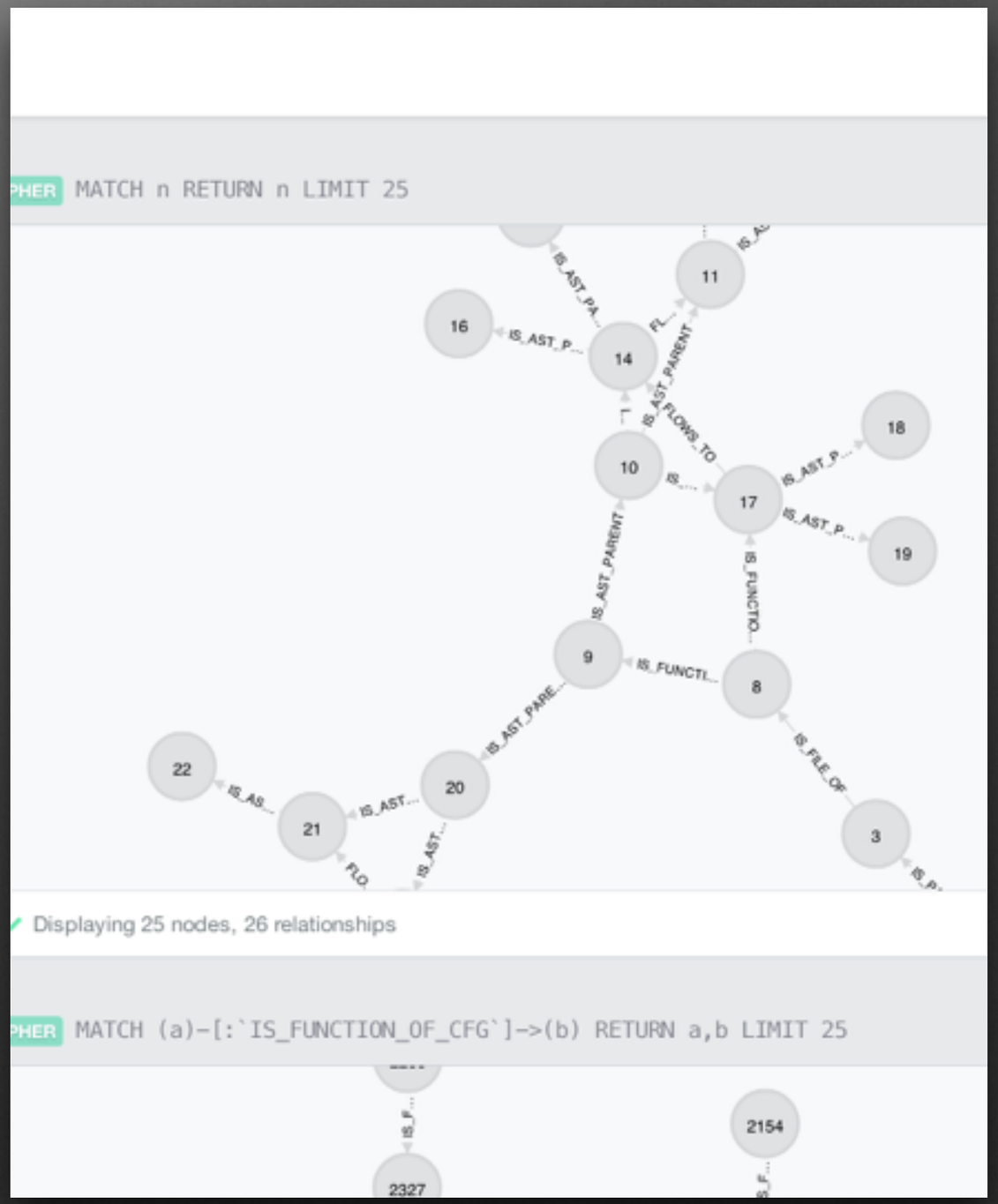
- Aims to merge “data parallel” and “graph parallel”
  - Their words, not mine
- Includes a number of graph algorithms by default
  - PageRank
  - Connected Components
  - Triangle Counting

Tinkerpop

- Blueprints - Common interface
- Gremlin - Query language
- Rexster - REST API
- Furnace - Graph algorithms
- Frames - Graph - Object mapping
- Pipes - Dataflow

Neo4J

- Pluggable architecture
- Cypher query language
  - Gremlin supported
- Very mature
- Single server node only



Cypher Query Language

- Very similar to SQL
- Get a count of all nodes

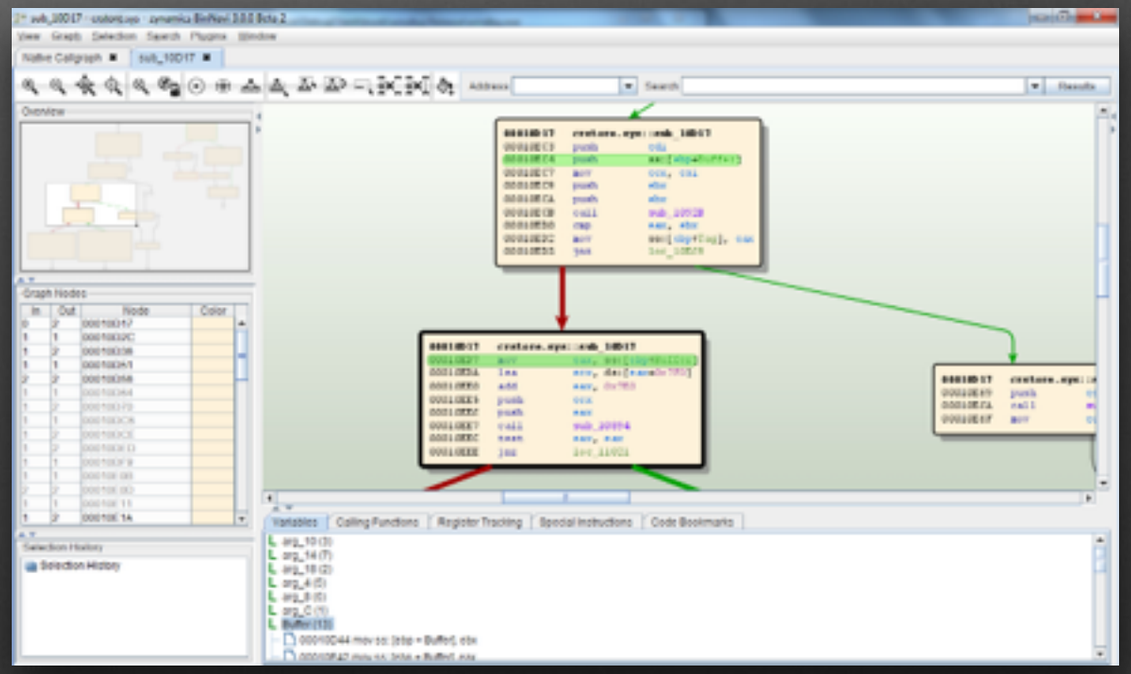
```
MATCH (n)
RETURN count(*);
```

- Get all nodes and relationships

```
MATCH (n)-[r]->(m)
RETURN n as from, r as `->`, m as to;
```

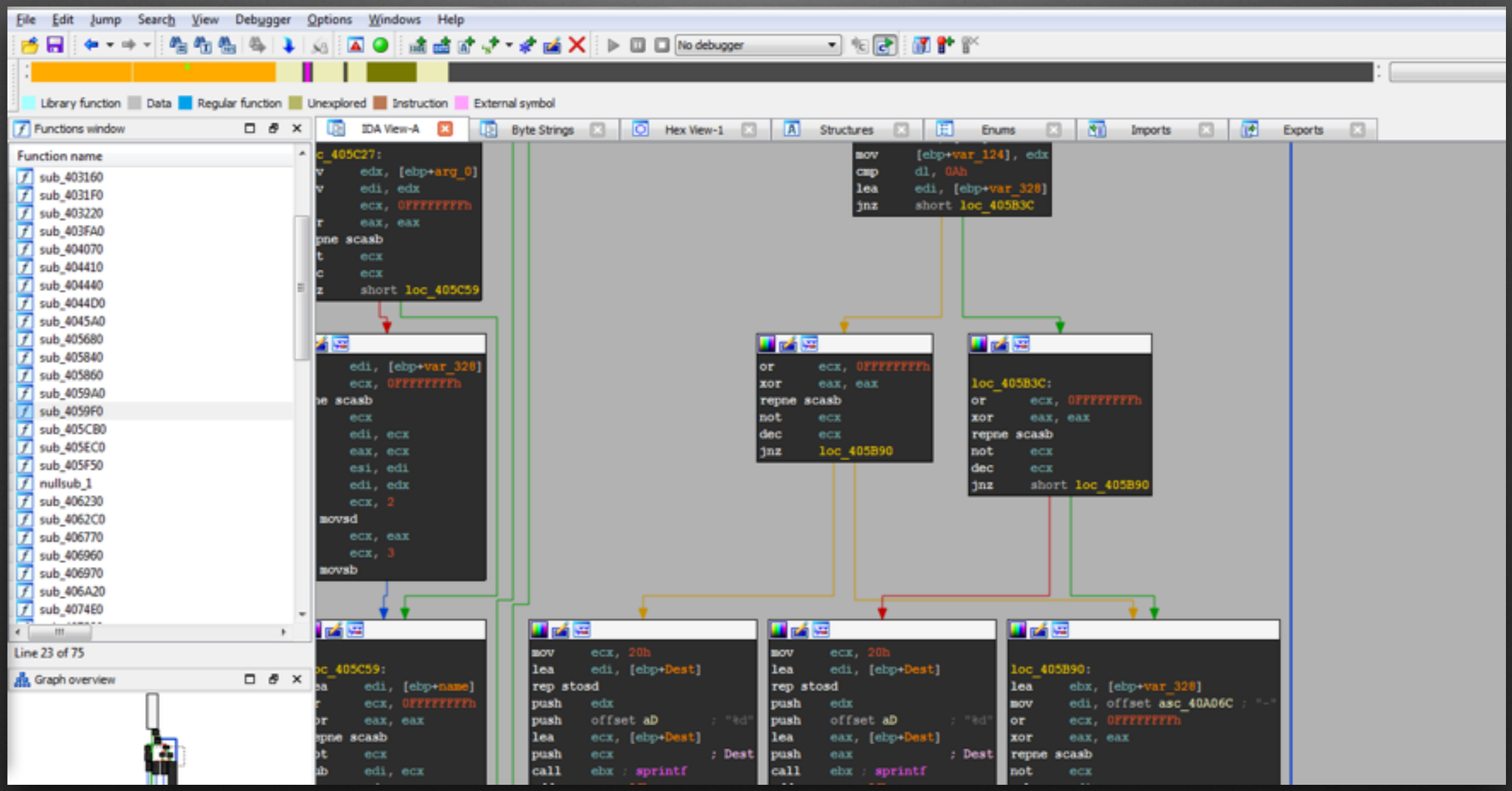
BinNavi

- Created by Zynamics, now owned by Google
- Uses RDBMS as backend
- Java Client
- Relies on IDA Pro



IDA Pro

- Everyone's favorite disassembler



How does this relate to reversing?

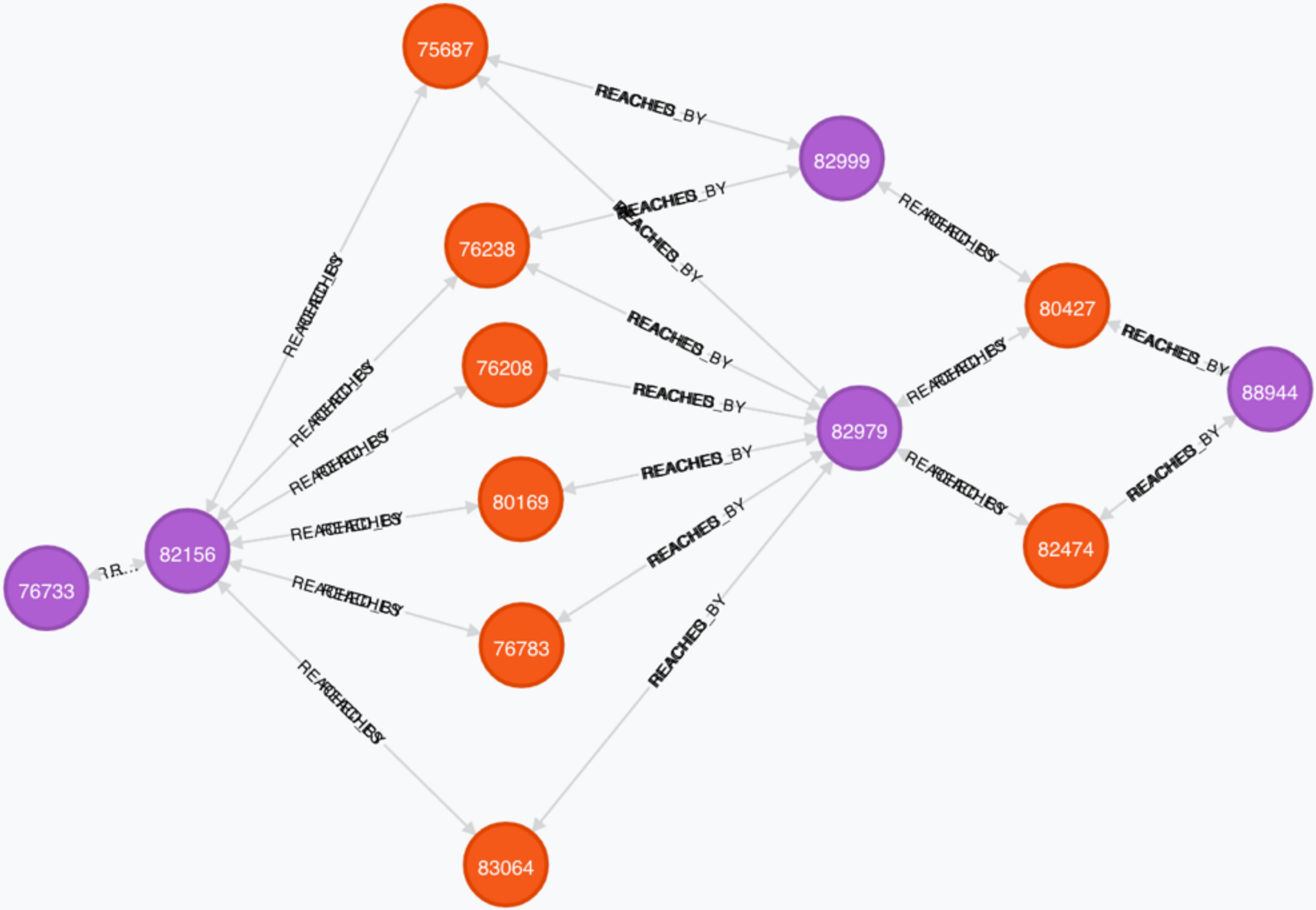
- IDA Pro was the last for a reason
- Binaries have a natural graph structure
  - Basic blocks as vertices
  - CALLs/JMPs as edges
    - Attach properties to the edge for conditionals
- Nice datastore to query from IDA or other apps

Path finding/traversals

- Exactly what GraphDBs excel at
- Loads basic blocks from IDA into Neo4j
- IDA has this functionality, but it is quite limited
- Code will be available at https://github.com/wanderingglitch

Path finding (cont.)

```
MATCH (begin:function {name:"srcfunc"}),
      (end:function {name:"destfunc"})
MATCH paths = (begin)-[:*0..10]-(end)
RETURN paths;
```


75687
82999
76238
80427
76208
88944
82979
80169
82474
82156
76733
76783
83064
REACHES_BY

Path finding (cont.)

- Overly simplistic example
- Can easily apply more constraints
  - Requires having a more intelligent importer

Taint Tracing

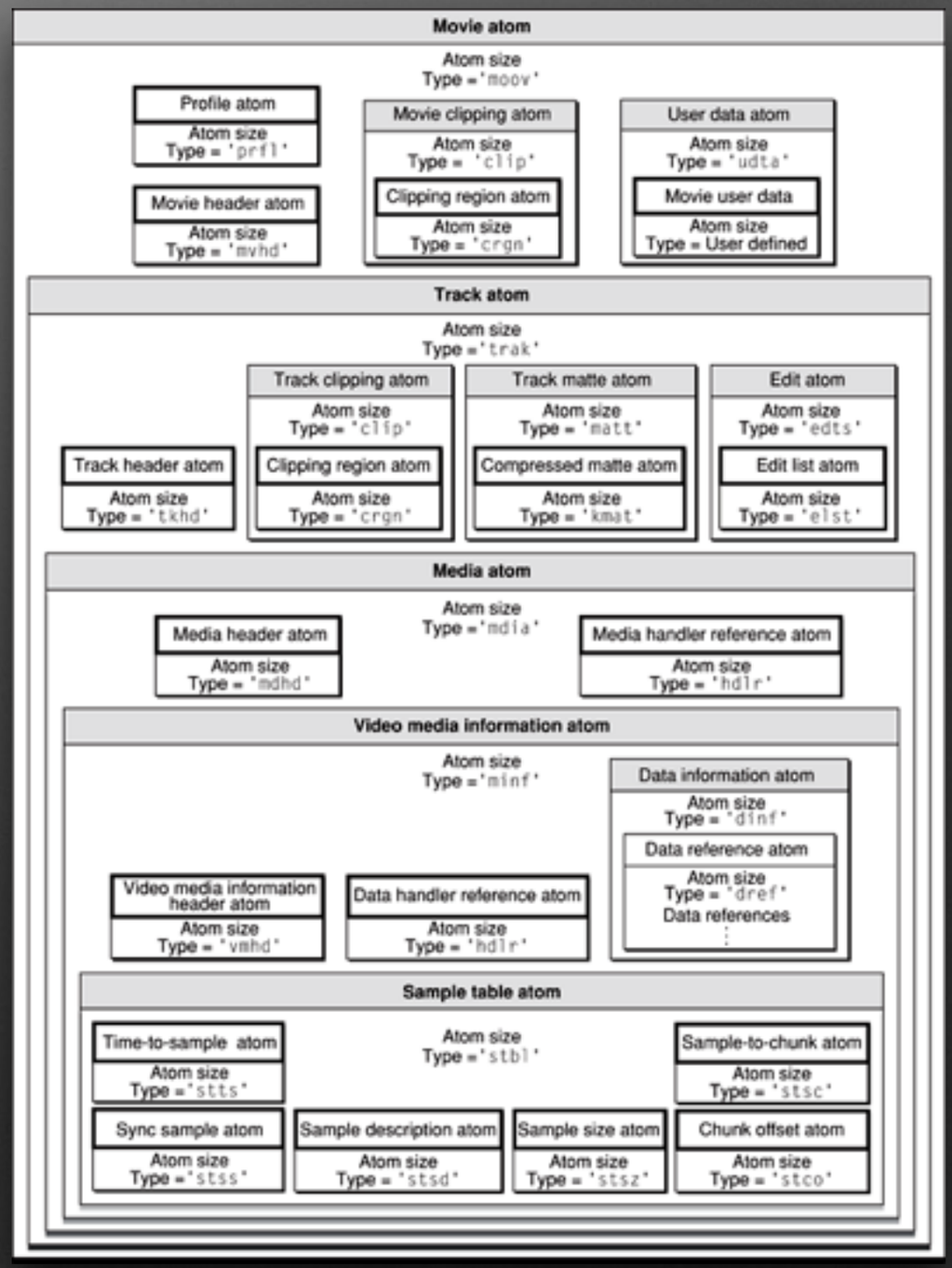
- Idea courtesy of Stephen Ridley (s7ephen) via twitter conversation
  - Also helped spawn the idea for this talk
- Use capstone or similar to disassemble for loading into graphdb
  - I can do the capstone part…
- Apply taint tracing to the constructed graph

Code identification

- Similar idea to BinDiff
- Can crunch a basic graph isomorphism routine to identify similar subroutines
- One recognizable function encountered in reversing malware is RC4
  - 2 loops in a row that iterate 256 times each
  - Final loop that iterates for len(str)

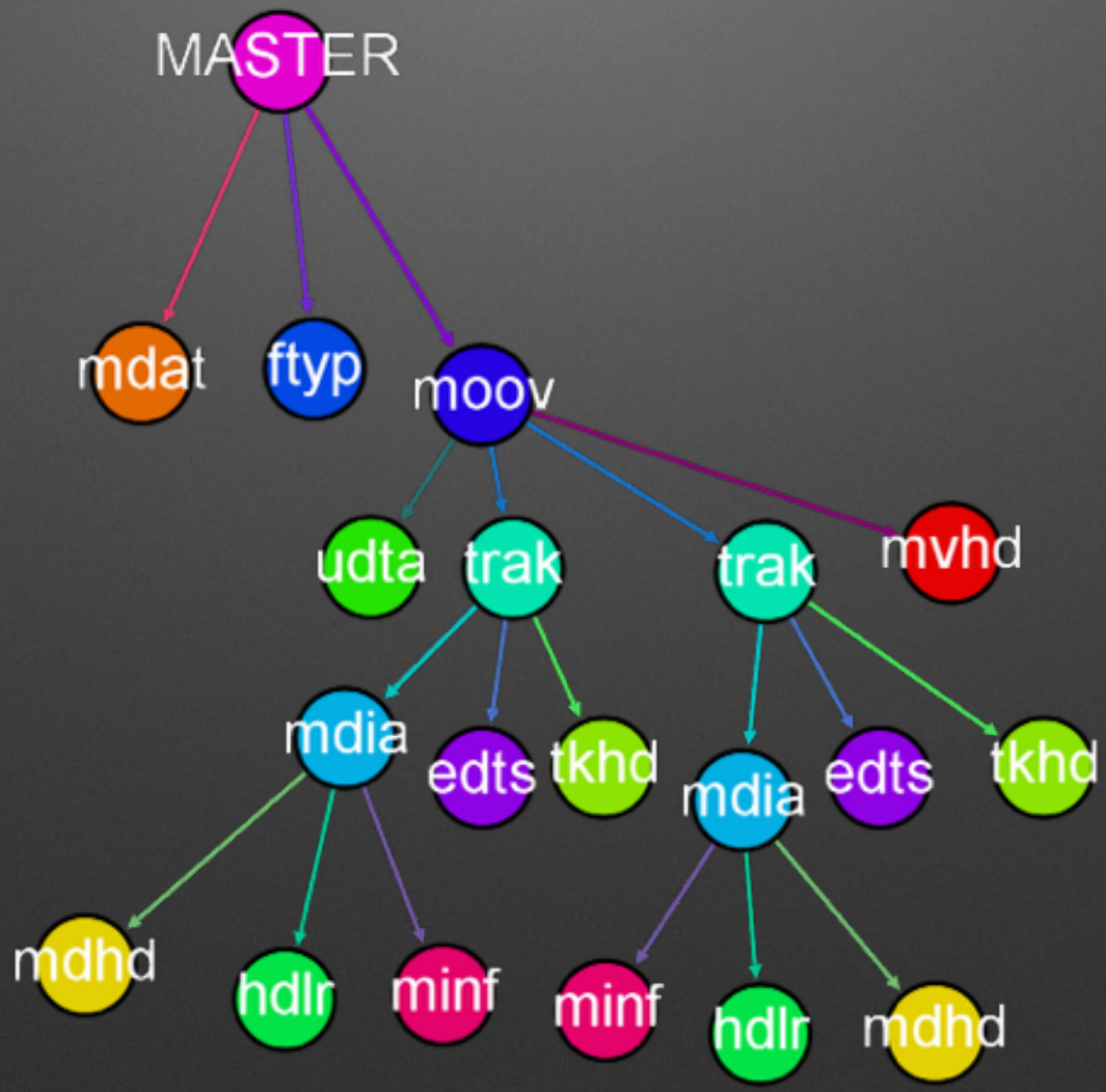
Mutational Fuzzing

- Some file formats are graph-like
- Some are not but could be faked for purpose of fuzzing
- Create a structure, process legitimate files
- Use that corpus as the baseline to fuzz against
- Who wants to do PDF for us?



FileFormat PoC - MP4

- Titan doesn’t have built-in visualization
- Gephi used to generate graph from exported GraphML



Collaboration / Sharing

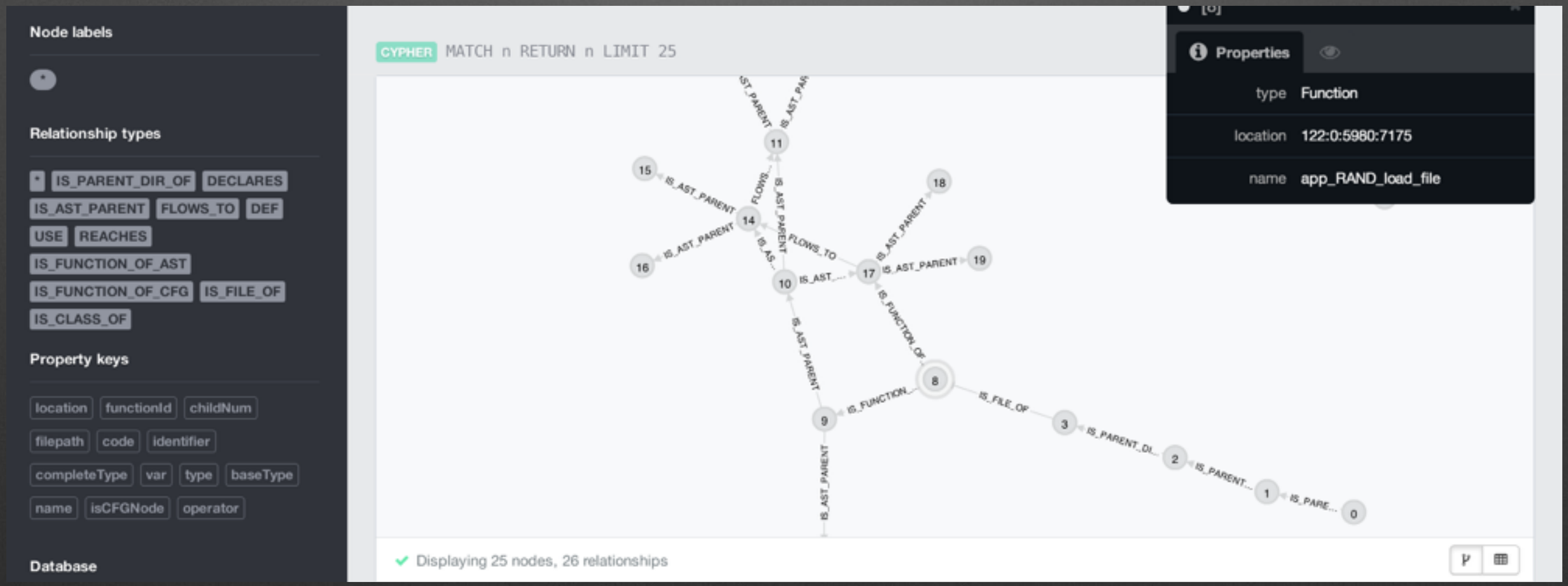
- Seems to still be an unsolved problem, though many have tried
- Use IDA-loading code to store all relevant IDB information into the graph
- Use code comparison / identification routines to identify “unknowns”
- Load in comments, names, structs, enums, etc. into local IDA from graph
- Useful when
  - reversing new versions of things people have already reversed
  - identifying shared code
  - new legit software ships w/o symbols

Joern

- Created by Fabian Yamaguchi (@fabsx00)
- Source code analysis tool
- Parses C/C++ into an AST
  - Uses Neo4j

Joern

- Taint arguments to functions
- Variable uses/definitions



What's next?

- Jasiel
  - Smarter import code
- Jason
  - More file format parsers
  - Graph comparison

Wrap-Up

- Can simplify some common operations
- Barrier to entry is low
- Still very resource intensive
  - and Java intensive

Questions?

References

- http://thinkaurelius.github.io/titan/
- http://thinkaurelius.com/blog/
- http://www.neo4j.org/
- http://www.orientechnologies.com/orientdb/
- https://spark.apache.org/docs/1.0.0/graphx-programming-guide.html
- http://mlsec.org/joern/
- Modern Graph Theory http://www.springer.com/new+%26+forthcoming+titles+(default)/book/978-0-387-98488-9
- http://www.tinkerpop.com/docs/current/