Detecting Malicious Activities using Passive DNS Data

Sean Beck, Lanier Watkins

Problem

- Hackers will hide in DNS traffic
- Hackers use generated names for malware communication
  - Conficker
  - Kraken
  - Torpig
- Hard to stop without black list of some sort

Machine Learning

- Types
  - Supervised, Unsupervised, Semi-supervised

Data

- ~30 MB of DNS traffic from JHUISI
- CSV
- Fields
  - QR, OPCODE, QNAME, QTYPE, QCLS, RRNAME, RRTYPE, RCLS, TTL, RLEN, RDATA, AA, ID, QDCNT, ANCNT, NSCNT, ARCNT
- Larger data set to follow (~50 GB)

Tools

- Python!
  - Pandas
  - Scikit-learn
  - Jupyter
  - Matplotlib
  - Numpy
- Virus Total
- Machine Learning
  - Mean Shift
  - Affinity Propagation
  - K-means

Algorithms

- Clusters
  - Mean Shift and Affinity Propagation decide on own clusters
  - K-means requires you specify a desired number of clusters
- Mean shift
  - Guaranteed convergence
  - Not ideal for large data sets because of multiple nearest neighbor calculations
  - Discovers blobs by selecting centroid candidates
- Affinity propagation
  - O(TN^2), T = num iterations, N = num samples
  - Messages between pairs of samples, selects examplars
- K-means
  - Works off of sum-of-squares
  - Clusters described by mean of the cluster

Experiment

- Data exploration
- Data munging
- Submit names to VT
- Feature extraction
  - Shannon Entropy
- Learning algorithms
- Gather results

Virus Total

- Small data = manageable data
- Extracted all domain names
- Submitted all to VT for analysis
- Recorded scores for each name

Results

- 266 total bad domains (per VT)
- Mean shift performed best
  - Window size 300 (288 rows, 12 anchor seeds)
  - Run time of 1m 18s
  - 11934 “anomalous” domains
  - 263 of those are known malicious
- K-means with 2 clusters mimics Mean shift
  - Literally almost exactly the same
- Affinity Propagation not so great
  - Window size 200 (192 rows, 8 seeds)
  - 4m 38s
  - 16722 “anomalous”
  - 237 known malicious

Demo!

- A data science project from start to finish