Detecting Malicious Domains and IPs by Fusing Threat Feeds and Passive DNS through Graph Inference

findervid.com/admin 195.154.34.135 stopwell.org/cp.php?m=login

havephun.org/frmcp1 rus99.net/se/logs 195.208.185.49

198.12.153.10 mmmoney1.com/panel/ sagradiana.net/lin/

Emily Heath, PhD

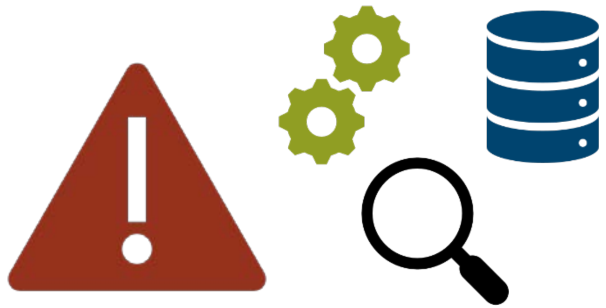
Eric Harley

The MITRE Corporation

MITRE

Approved for Public Release; Distribution Unlimited. Case Number 17-4450

Current Problems

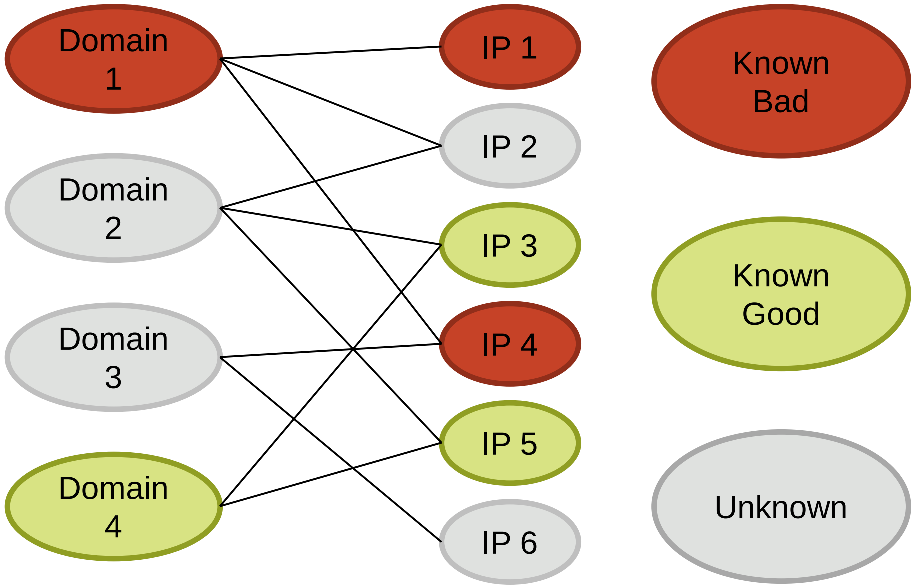


- **Analysts are inundated with threat feeds, indicators, network data, analytic results, etc.**
- **Besides handling the volume, there are other problems with using this information efficiently**
  - Timeliness
  - Coordinating and combining data
- **Intuition: threat actors re-use infrastructure and tend to get their infrastructure from similar places**
  - Analysts anticipate being able to pivot from one known malicious domain to more, or to malicious IPs

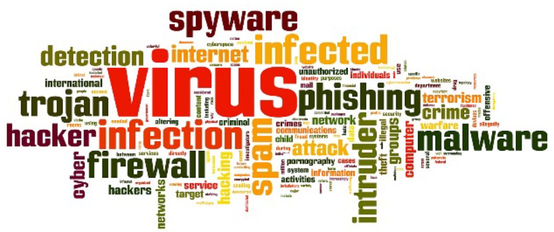
How can analysts exploit this intuition and move from flagging what is already known to be malicious to identifying new maliciousness?

Using the Intuition

- **Idea: the digital neighborhood of an unknown domain or IP can be used to estimate its likelihood of being malicious**
- **Homophily: birds of a feather flock together**
- **Unknown domains and IPs associated with known malicious domains and IPs are more likely to be malicious**
- **Known maliciousness could come from any information source**



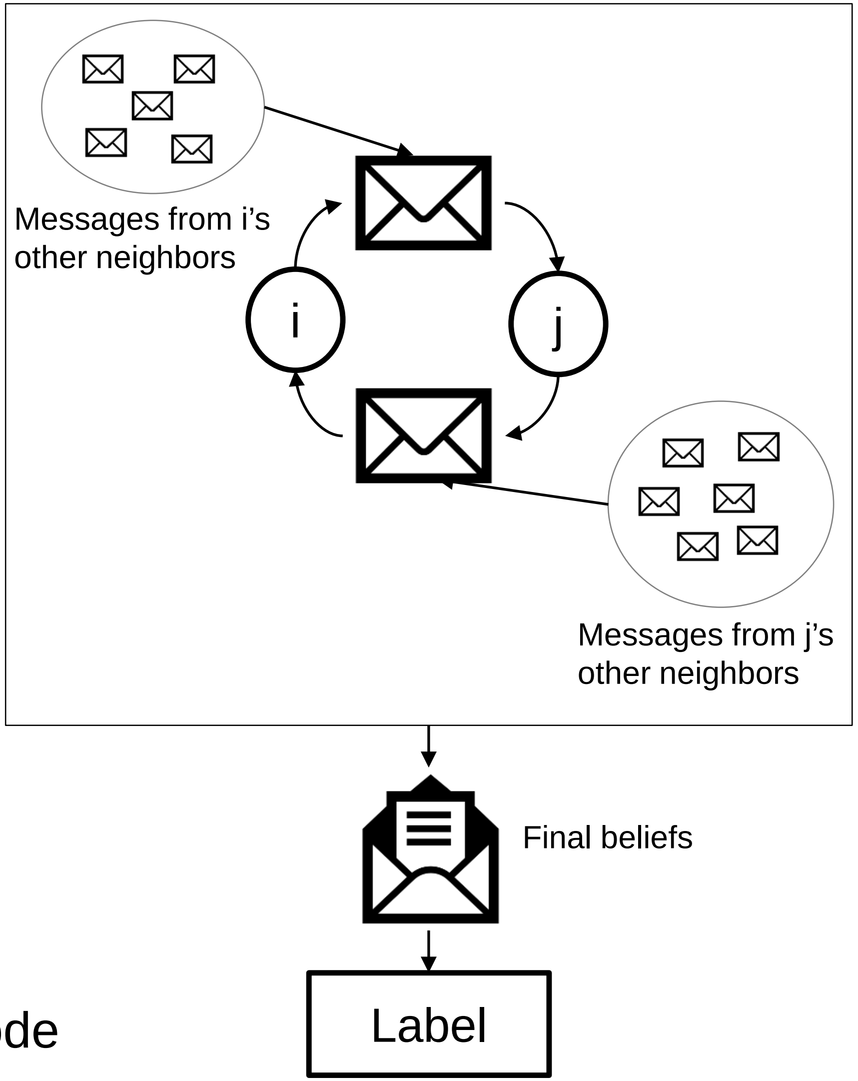
- **Information can be propagated throughout the network to uncover new maliciousness**
- **Formal method: graph inference**



MITRE

Belief Propagation Algorithm (BPA)

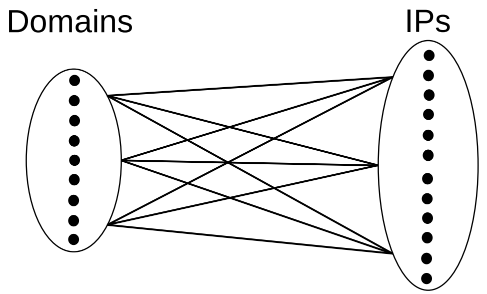
- **BPA: Graph inference method for estimating a node's marginal probability**
  - Prior knowledge for some nodes (known states)
  - Statistical dependencies between nodes (homophily or heterophily)
- **Nodes pass messages to neighbors each round**
  - Messages: vectors with an entry for each state
  - Entry contains sender's perception of the recipient's
  - likelihood of being in that state
  - Synchronous update schedule: messages in one iteration depend upon messages in previous iteration
- **After message passing, final belief values can be computed for each node**
  - Beliefs: vectors with final value for each state
  - With threshold, values can be used to assign a label to a node



MITRE

BPA for Malicious Domains and IPs

- **Build a bipartite graph of domains and IPs**
  - Include edge if domain resolves to IP
  - Use passive DNS data to construct

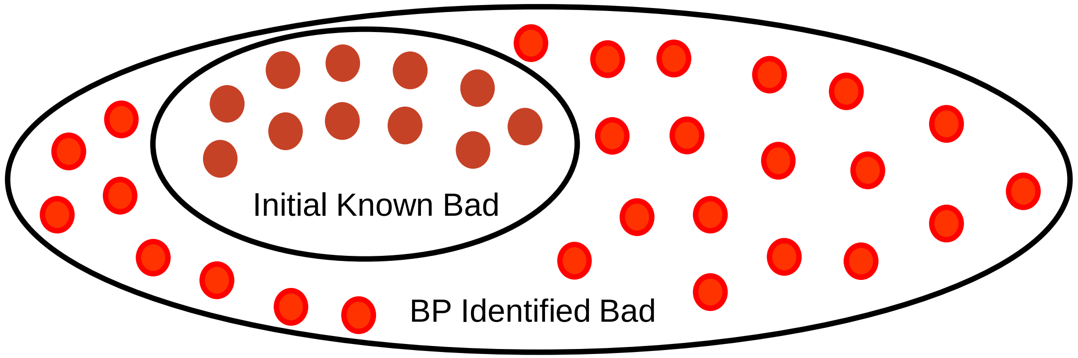


- **Modify key parameters of interest**
  - Seed size of known labels
  - Number of iterations
  - Strength of relationships between nodes
  - Threshold values for label decisions

- **Seed the graph with some known labels**
  - Two states: malicious and benign

Malicious Labels	Benign Labels
-Threat Feeds -Analytic Outputs	-Alexa Domains -Umbrella List

- **Test as a semi-supervised learning problem**
  - Measure percentage improvement in baseline true positive rate (TPR)



Using Real Data

- **Data set constructed from following:**
  - Censys data set for passive DNS
  - Threat feeds for malicious labels
    - hpHosts EMD by Malwarebytes
    - Malware Domain Blocklist
    - CyberCrime Tracker
  - Alexa & Umbrella lists for benign labels

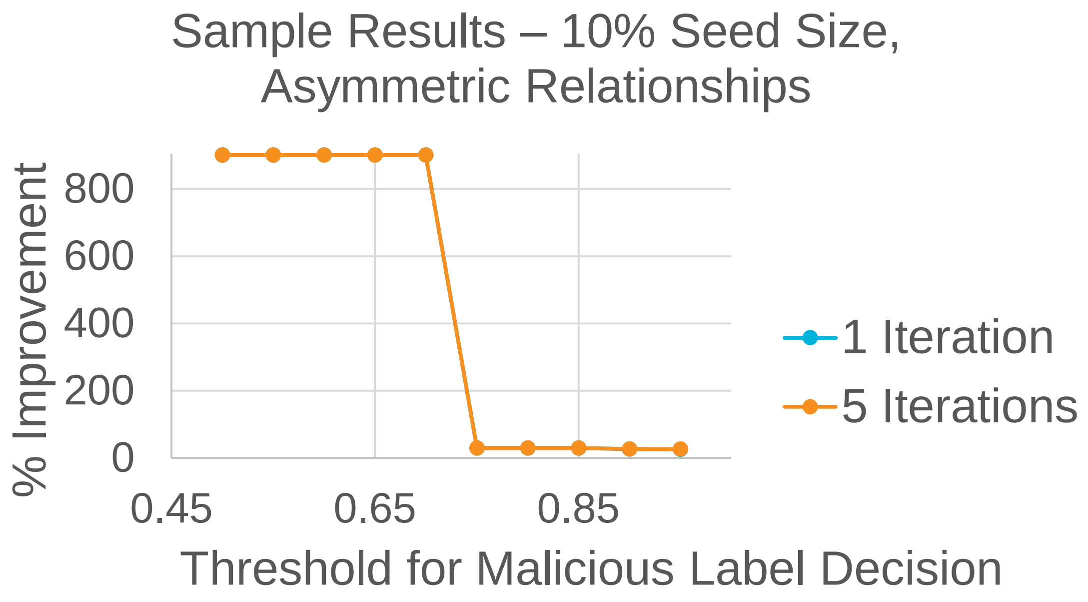
Network Statistics	
No. of Edges	2,120,375
No. of Nodes	152,904
Max node degree	22,432
Average node degree	27.89
Median node degree	2
Min node degree	1

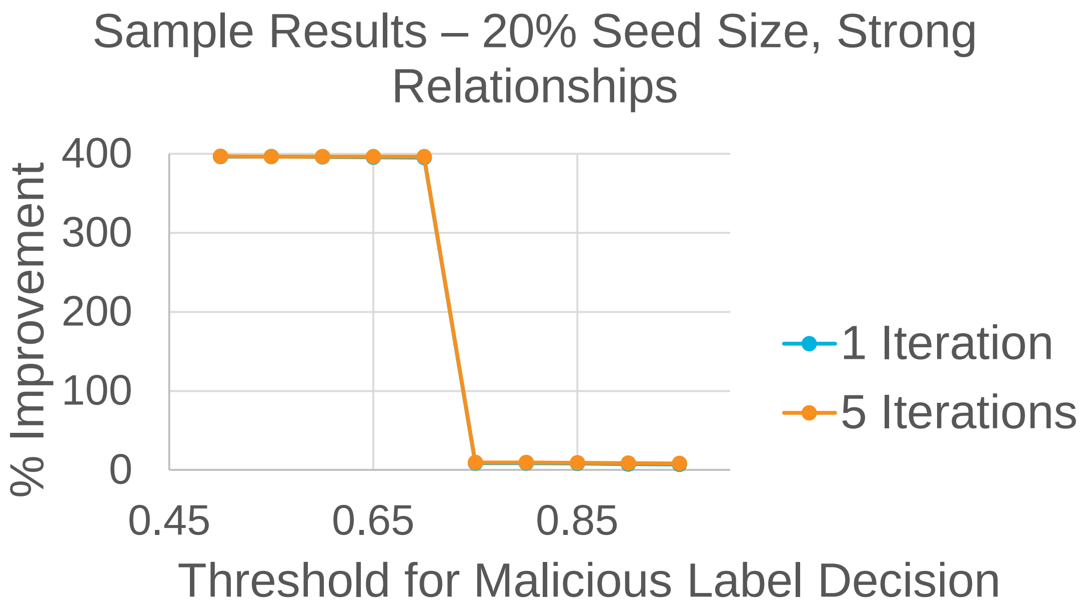
- **Surprise issue: underflow in message/belief computations**
  - Source: nodes with high numbers of neighbors
- **Resolution: Two implementations**
  - Decimal package approach
  - Log-space transformation approach
  - Optional feature to “shuffle” order of neighbors

MITRE

General Results

- **Worst results: BPA gives the same TPR (0% improvement)**
  – Dependent on threshold
- **Best results: 400% and 900% improvement**
  – Moderate thresholds, strong or asymmetric relationship strengths





Algorithmic Comparisons

- **Decimal Implementation**
  - Decimal: Python package for representing numbers exactly
  - Resolves underflow by performing computations exactly
- **Log-Space Implementation**
  - Use logs of values and log identities to perform computations
  - Resolves underflow by performing computations with numbers far from 0
- **Shuffling option implemented for both methods**

Run Time Comparisons				
	Decimal (No Shuff.)	Decimal (Shuff.)	Log-Space (No Shuff.)	Log-Space(Shuff)
Time (Messages)	~11 h 1 iteration ~22 h 2 iterations	~12.5 h 1 iteration ~25 h 1 iterations	~1.5 h 1 iteration ~3 h 2 iterations	~3.5 h 1 iteration ~6.5 h 2 iterations
Time (Beliefs)	~26 sec	~30 sec	~15 sec	~20 sec
*Statistical testing confirmed all approaches agreed in terms of actual results (message and belief values)				

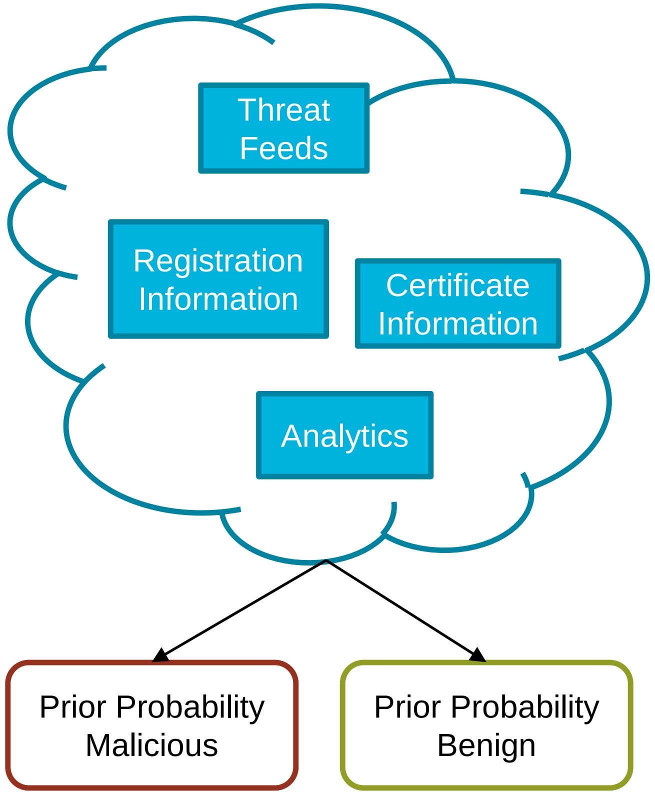
MITRE

Challenges & Areas for Future Work

- **Continuing to improve speed**

- **Building in more prior knowledge**
- **Expanding inferences to registrars, BGP ASNs**

- **Updating pDNS data, “known” labels**
- **Infrastructure that is both malicious and benign**



MITRE

Conclusions

- BPA shows a lot of potential for identifying previously unknown malicious domains and IPs quickly and accurately

- Simplicity of algorithm allows for multiple sources of information to be effectively fused

- Computational considerations resulting from messy real data can be handled efficiently in different ways

- Various open areas allow analysts the opportunity to tune the approach to their environment

MITRE

Thank you! Questions?