Summary Report on Detecting Malicious Domains via Graph Inference - Anomaly Detection Homework 6

Vishal A. Bhalla[1] and Nithish Raghunandanan[1]

***Abstract*— This report summarizes the approach, implementation and results within the enterprise security domain, for detecting malicious domains. The spread of malicious domains is the root cause of malware infections inside an enterprise and the approach discussed in the paper [1] scales well on data collected at the global enterprise level. We have highlighted the methodology, results and the contribution of this paper to the set of malware detection techniques.**

I. INTUITION

Analysis of large security related data sets to extract actionable security information, and hence to improve enterprise security, is a relatively unexplored area [2]. Scaling is a challenge with existing machine learning techniques that require accurately labeled training sets along with a large number of features. These techniques are resource intensive and cause additional delay in detection.

Alternatively to achieve scalability, the authors model the malicious domain detection problem as a graph inference problem by adapting a faster approximate estimation algorithm i.e. Belief Propagation (BP) [3] because it scales well to large graphs, and also takes advantage of the structure of malware communication. The incorporated approach in this paper uses event logs and minimal data from existing blacklists and whitelists. For scalability and identifying new malicious domains not present in the blacklists, it is then applied over to event logs collected at a global enterprise over 7 months.

II. ADOPTED METHODOLOGIES

A. Graph Inference Approach

They start by using an enterprise's host-domain graph, adding a node for each host in the enterprise and each domain accessed by the hosts and an edge connecting the two. Each node has a state, e.g., malicious or benign. A domain can be malicious or benign, based on its presence in a domain black list or a domain white list respectively. Similarly, a malware infected host is known to be malicious otherwise it is benign. The nodes with a known state act as ground truth while the rest are unknown nodes. The graph can be constructed from multiple enterprise event log data sets, e.g., HTTP proxy logs and DNS request logs.

[1]Master of Science students in the Department of Informatics, Technische Universitat Munchen, Germany. Email: vishal.bhalla@tum.de and nithish.raghunandanan@tum.de

1) Belief Propagation: Marginal probability estimation in graphs is known to be NP-complete. BP is an efficient technique to solve inference problems on graphical models. Given an undirected graph, G = (V, E), where V is a set of n nodes and E is a set of edges, we model every node i $\in$ V as a random variable, x_i, that can be in one of a finite set, S, of states. A graphical model defines a joint probability distribution, $P(x_1, x_2, \cdots, x_n)$, over G's nodes. The inference process computes the marginal probability distribution, $P(x_i)$, for each random variable, x_i.

BP estimates a nodes marginal probability from prior knowledge about the graphs nodes and their statistical dependencies and can approximate it of all nodes in time linear in the number of edges, which is at most $O(n^2)$. BP achieves this computational efficiency by organizing global marginal probability computation in terms of smaller local computations at each node via iterative message passing among neighboring nodes [1]. Malware communication, e.g., bots communicating with Control & Command servers, provides a structural advantage in using BP as messages can propagate over multiple hops using a synchronous update order for its simplicity. BP always converges and the beliefs represent accurate marginal probabilities. But if a graph has loops, then belief propagation on the graph may not converge or may converge to inaccurate marginal probabilities [4].

Consider a node, i, and its neighbors, N(i). In each iteration of the algorithm, i passes a message vector, m_{ij} , to each of its neighbors, j $\in$ N(i).

$$b_i(x_i) = C\Phi(x_i) \prod_{x_i \in N(i)} m_{ki}(x_i) \tag{1}$$

C is a normalization constant to ensure that i's beliefs add up to 1, i.e., $\sum_{x_i \in S} b_i(x_i) = 1$

and where, each component, $m_{ij}(x_j)$, of the message vector is proportional to i 's perception of j's likelihood of being in the state x_j . i's outgoing message vector to its neighbor j depends on i's incoming message vectors from its other neighbors and is computed as follows.

$$m_{ij}(x_j) = \sum_{x_i \in S} \Phi(x_i)\Psi_{ij}(x_i, x_J) \prod_{x_i \in N(i)/j} m_{ki}(x_i) \tag{2}$$

III. IMPLEMENTATION & INTERPRETATION OF RESULTS

A. Implementation

The proxy logs over a 7 month period from 98 proxy servers in a global enterprise's worldwide locations was

used. Each entry in the log represented an HTTP request and contains the requesting host's IP address, the domain requested, a time stamp, an HTTP header, and the request status. The corresponding graph for the request was constructed after avoiding duplication of domains by using the second level domain names. Collapsing domain nodes in this manner increases the number of paths in the graph, making paths between nodes more likely and hence information propagation between nodes more likely.

For the BP Parameters, the priors were assigned to graph nodes according to ground truth data. For example, a prior, P (malicious) = 0.99, was assigned to the nodes present in the blacklist. Also an edge potential matrix was present to reflect the statistical dependencies among neighboring nodes. They assumed a homophilic relationship, i.e., two neighboring nodes are more likely to be of the same state than different states.The relationship is based on our intuition that hosts that visit benign sites are likely to be benign and hosts that visit malicious sites are likely to be infected.

B. Confusion Matrix

The number of True Positives (TP), False Positives (FP), False Negatives (FN) and True Negatives (TN) constitute the confusion matrix. A benign node labeled as malicious by the approach is a false positive (FP) and a correctly identified malicious node is a true positive (TP).A benign node labeled as benign is a true negative (TN) while a malicious node identified as benign is a false negative (FN).

C. K-Fold Cross Validation

They used K-fold cross validation to compute malicious domain detection performance, i.e., divide the ground truth data into K folds, mark one fold as test data and the remaining K-1 folds as training data. This is after seeding the host-domain graph with the training data, resetting the priors of the nodes in the test data to unknown priors, running belief propagation, and then computing beliefs following the procedure described earlier.

The results were represented as as Receiver Operating Characteristics (ROC) plots, i.e., plots showing false positive rates and true positive rates. An ROC plot is obtained by thresholding a nodes malicious belief value. For example, given a threshold, t, if a node, n's, malicious belief, $b_n(malicious) > t$, then they predict n as malicious; else, n is benign. They then use n's ground truth state and predicted state to label n as false positive, true positive, false negative, or true negative.

The areas under the ROC curves (AUC) for K-fold cross validation for K = 2, 3, and 10 were 98.53%, 98.72%, and 98.80% respectively as shown in figure 1 [1]. These results reassure the underlying intuition that more training data (when K = 10, they use 9/10 th of the ground truth data as training data) leads to a better detection.

IV. CONCLUSION

The approach can achieve high detection rates with low false positive rates using minimal ground truth information.

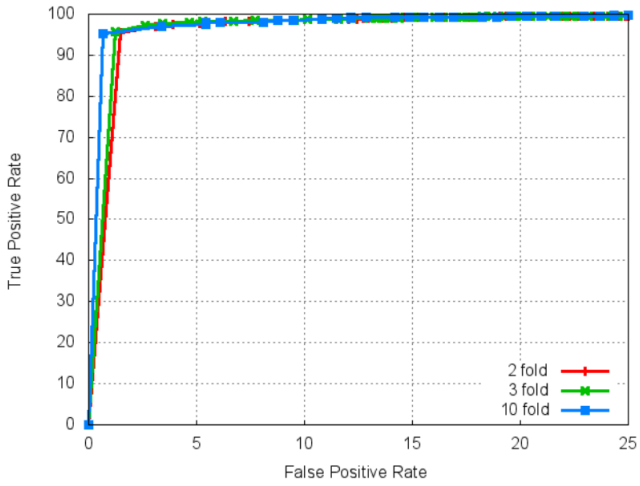

Fig. 1. ROC plots for different K-fold cross validations. For clarity, the X-axis ends at FPR = 25%. K = 10 performs the best..

Most of the FPs, i.e., benign domains identified as malicious by the approach, are of low degree, i.e., not business critical. The approach is also able to detect random looking malicious domains that are not part of the black lists, which are likely to be algorithmically generated by malware resident on the enterprise's hosts.

This approach shows that belief propagation is a reliable and scalable approach and can detect previously unknown malicious domains. The work is an example of big data analysis for security, i.e., analyzing enterprise event data to extract actionable security information.

REFERENCES

[1] P. K. Manadhata, S. Yadav, P. Rao, and W. Horne, "Detecting malicious domains via graph inference," in *Computer Security-ESORICS 2014*. Springer, 2014, pp. 1–18.

[2] A. Labrinidis and H. Jagadish, "Challenges and opportunities with big data," *Proceedings of the VLDB Endowment*, vol. 5, no. 12, pp. 2032–2033, 2012.

[3] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Understanding belief propagation and its generalizations," *Exploring artificial intelligence in the new millennium*, vol. 8, pp. 236–239, 2003.

[4] K. P. Murphy, Y. Weiss, and M. I. Jordan, "Loopy belief propagation for approximate inference: An empirical study," in *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 1999, pp. 467–475.