Counterfeit Fingerprint Detection of Outbound HTTP Traffic with Graph Edit Distance

1st Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

2nd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

***Abstract*—**

***Index Terms*—Anomaly Detection, Data Exfiltration, Data Leakage, Application Fingerprinting, Network Security**

I. Introduction

II. Related Work

III. Proposed Approach

This section gives the details about our proposed method which aims at detecting counterfeit fingerprints from applications' outbound HTTP traffics. Before going further, all PCAP files collected by an enterprise's host is network activities generated by a set of browsers $B = \{b_1, \quad ... \quad , b_n\}$, and which are all installed in hosts. Each browser b_i has several PCAP files which contain specific network characteristics, and our proposed approach possibly create a fingerprint $f_{b_i} = F(b_i)$ for each browser. The proposed counterfeit fingerprint detection process consists of training and testing phases. In training phase, we assume enterprise hosts aren't compromised. This method mainly arises from the first one that is a data-driven and unsupervised flow responsible for a browser's fingerprint [1] and referrer correlation construction. This step takes the fields of a PCAP file as input and classifies browser traffics, and then construct fingerprints and referrer correlation graphs. In the testing phase, given a browser outbound HTTP traffic reconstructed by fingerprint and referrer correlation graph, and the second step filters benign browser traffics through fingerprint matching. Continuously, compare its and trained referrer correlation graph using Graph Edit Distance (GED) for counterfeit fingerprint detection. The proposed method is depicted in figure 1 and following paragraphs describe the details of each component.

A. Data Preprocessor

B. Fingerprint Constructor

C. Fingerprint Matching Module

D. Referrer Correlation Graph Constructor

E. Graph Similarity Estimator

IV. Experiment Results

Tony, pliz write overview here!!!

TABLE I
Datasets used in this paper's experiments

Datasets	Operation Type	Malware Families	Malware
Ahmadi et al. [?]	API	4	3,829
Ki et al. [?]	Opcode	9	10,867

A. Experimental Settings

Tony, pliz write like the followings in this subsection!

Malware analyzed in following experiments are malicious behavioral sequences from collected malicious executable files in real world. In proposed method implementation, collected malware were paresed and information of four major fields in table I. Table I also lists the the number of malware families, the size of gathered data, and operation types in malicious behavioral sequences.

B. Evaluation Metrics

Essentially, malware clustering is a multiple classification problem aim to identify malware comes from which families. Four well-known metrics for evaluating effectiveness of proposed method are adopted as followings: "true positive"(TP) means the number of malware which belong to same malware families. "False negative"(FN) is the number of malware which's families are wrongly predicted. Similarly, "true negative"(TN) means the number of malware which aren't same families and being viewed as others, while "false positive"(FP) is the number of false alarms that other families' malware being detected as the same ones. Based on accumulation of TP, FN, TN, and FP, one extended metrics ($accuracy$) popularly used in machine learning problems are also adopted here to evaluate proposed method and listed in equations below. Note that the optimal $accuracy$ of 1.0 means all of malware are successfully classified by proposed approach.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

C. Effectiveness Analysis

Tony, pliz show your exp graph and table here!!!

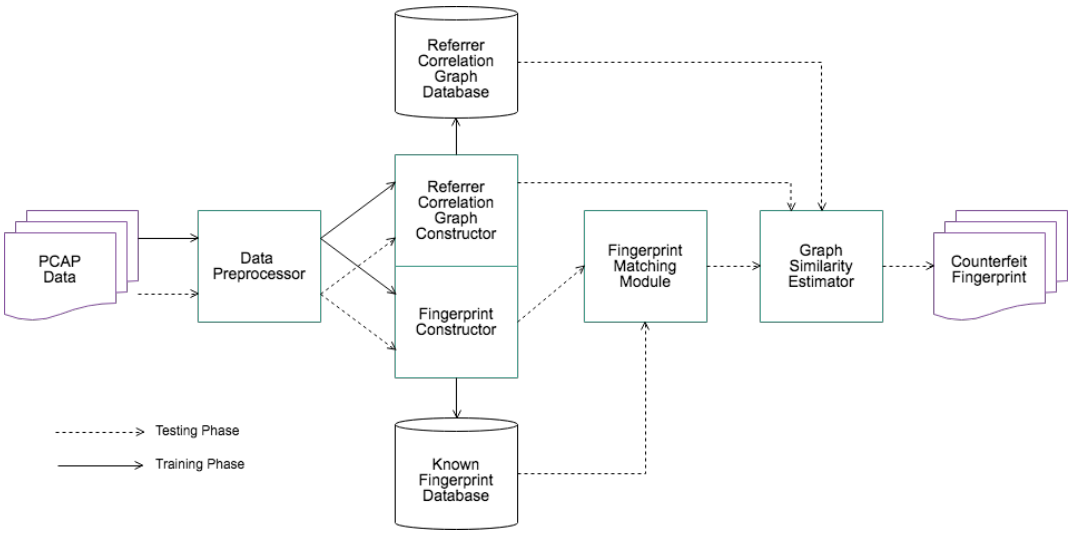

Fig. 1. An overview of our counterfeit fingerprint detection system. Five subsystems are depicted: (1) data preprocessor subsystem, (2) fingerprint constructor subsystem, (3) fingerprint matching subsystem, (4) referrer correlation graph constructor subsystem, and (5) graph similarity estimator subsystem. The system only takes the PCAP files of outbound HTTP traffics as input. In training phase, subsystem (1) and (2) passively extract the benign fingerprint from an application's outbound HTTP traffic, and subsystem (3) could use fingerprints to classify benign traffic in the testing phase. We note that referrer correlation extraction in the subsystem (4) is a key step, in the sense that if it can extract discriminative features for counterfeit fingerprint detection, the detection in the subsystem (5) is relatively straightforward.

V. Conclusion

Acknowledgment

References

[1] R. Bortolameotti, T. van Ede, M. Caselli, M. H. Everts, P. Hartel, R. Hofstede, W. Jonker, and A. Peter, "Decanter: Detection of anomalous outbound http traffic by passive application fingerprinting," in *Proceedings of the 33rd Annual Computer Security Applications Conference*. ACM, 2017, pp. 373–386.