

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 applications such as browsers $B = \{b_1, \dots, b_n\}$, 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} for each browser. The PCAP files of a host H include union of all browser fingerprints which is defined as $H = \cup_j^n f_{b_i}$. 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. Browser Traffic Extractor

For most cases of client-side attacking, hackers whose general goal is to steal valuable data before malware connects to C&C server. As a result, PCAP files, that contain specific network characteristics of an application (e.g., browser) for each host in the enterprise.

TABLE I
FIELDS AND VALUES OF DATABASE IN A PCAP FILE

Field	Value for Instance
<i>Domain</i>	yahoo.com.tw
<i>DstIP</i>	140.92.88.140
<i>User-agent</i>	Mozilla/5.0 (Windows NT 6.1; Win64; x64) ...
<i>Accept-Lang</i>	zh-TW
<i>Len</i>	1253
<i>Constant</i>	If-Modified-Since, Cookie, Connection, ...

To generate fingerprint for each browser, our approach first extracts various entities from PCAP files. Table I shows 6 heterogeneous fields which can be extracted from each one-line log, including domain (*Domain*), destination IP (*DstIP*), user-agent (*User-agent*), accept language (*Accept-Lang*), length (*Len*), and constant field (*Constant*). The reason for choosing these 6 fields for browser traffic classification can be summarized as followings and fingerprint construction is represented in next subsection.

In previous research [1], Bortolameotti et al. identified two types of HTTP applications (e.g., *browser* and *background*). This subsection aims to filter logs of a PCAP file according to the *User-agent*, because we focus on counterfeit fingerprints of browser network activities. To identify browser activities, *browserstring* we defined are "Mozilla", "Opera", "MQQBrowser", "UCWEB", "NOKIA5700", "Openwave", "Safari", and "Chrome", and which are used for string matching in field *User-agent*. Furthermore, in both training and testing phase, an implementation time-slot t is a fixed time window of T minutes, and the filtered logs is passed to the next module after t ends.

B. Fingerprint Constructor

C. Fingerprint Matching Module

D. Referrer Correlation Graph Constructor

E. Graph Similarity Estimator

IV. EXPERIMENT RESULTS

Tony, pliz write overview here!!!

A. Experimental Settings

Tony, pliz write like the followings in this subsection!

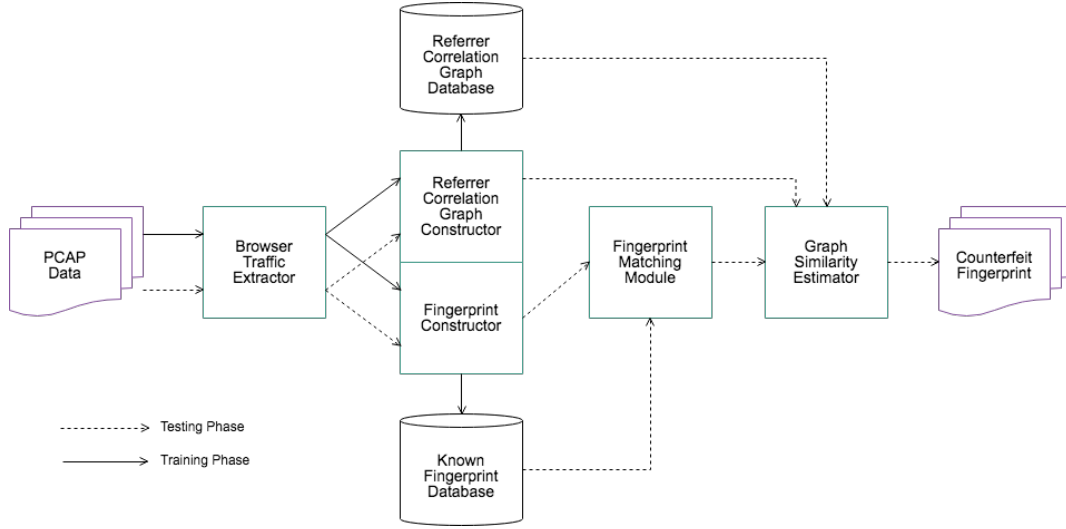


Fig. 1. An overview of our counterfeit fingerprint detection system. Five subsystems are depicted: (1) data preprocessor subsystem, (2) fingerprint constructor subsystem, (3) fingerprint matching subsystem, (4) referrer correlation graph constructor subsystem, and (5) graph similarity estimator subsystem. 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.

TABLE II
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

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

B. Evaluation Metrics

Tony, pliz replace malware to fingerprint!!!

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.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

C. Effectiveness Analysis

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

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.