Journal of Network and Computer Applications 99 (2017) 1–9

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

A nonparametric approach to the automated protocol fingerprint inference☆

YiPeng Wang[a,b], Xiaochun Yun[a], Yongzheng Zhang[a,*], Liwei Chen[a,b], Guangjun Wu[a]

[a] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[b] University of Chinese Academy of Sciences, Beijing, China

ARTICLE INFO

Keywords:
Protocol fingerprints
Network traffic
Nonparametric method

ABSTRACT

Protocol fingerprints are a set of byte subsequences within packet payload that can distinguish individual application protocols. They play an important role for deep packet analysis in traffic normalization and network management. In this paper, we propose ProPrint, a network trace-based protocol fingerprint inference system. In ProPrint, we first build a protocol language model based on a modified nonparametric Bayesian statistical model. Second, we use the corresponding protocol language model to identify field boundaries in packet payload, such that we can segment each payload into a set of protocol feature words according to the hidden structure information. Third, we propose a ranking algorithm that selects true protocol fingerprints from the candidate protocol feature words. In evaluations, we measure ProPrint on real-world network traces, and also compare ProPrint to existing state-of-the-art solutions, ProWord and Securitas. The experimental results show that ProPrint performs better than ProWord and Securitas on f-measure for online application classification.

1. Introduction

1.1. Motivations

This paper concerns the automatic inference of protocol fingerprints from the packet payload of protocol traces. Protocol fingerprint specifications are referred to by us as a set of unique byte subsequences that can be used to distinguish protocol traces from mixed Internet traffic. Protocol fingerprint inference is a fundamental problem for a variety of networking and security services, such as network management, network-based Intrusion Detection and Prevention Systems (IDSes/IPSes), traffic classification, and measurement, *etc* (Luo and Yu, 2013; Najam et al., 2015; Reviriego et al., 2014; Rubio-Largo et al., 2014; Wang et al., 2012). For example, protocol fingerprints are helpful for Internet Service Providers (ISPs) to provide a better service experience for Internet end users. In practice, with fine protocol fingerprints, ISPs can formulate an in-depth understanding of the protocol traffic passing through their networks, such that they can impose different strategies (i.e., higher or lower priority) and appropriate policies on the protocol traffic they are concerning for better services.

1.2. Limitations of prior work

Prior works for protocol fingerprint inference fall into two categories: 1). executable code-based approaches and 2). network trace-based approaches. In this paper, we focus on the problem of automated protocol fingerprint inference based on the packet payload of protocol traces, and thus executable code-based approaches are beyond the discussion of this paper. Notice that many network trace-based approaches have been proposed for the deep understanding of protocol traffic in prior arts, such as Discoverer (Cui et al., 2007), KISS (La Mantia et al., 2010; Finamore et al., 2010), Veritas (Wang et al., 2011), ProDecoder (Wang et al., 2012), ProWord (Zhang et al., 2014) and so on. ProWord (Zhang et al., 2014) proposed by Zhang *et al* is the most recent and relevant work, and it is an elegant system for network trace-based protocol fingerprint inference. To infer protocol fingerprints, ProWord identifies possible word boundaries of protocol traces using entropy and then selects the most possible boundary positions for word partitioning. With the recognized word boundaries, ProWord partitions the payload of protocol traces into a set of candidate protocol words. The extracted protocol words are regarded as the building rules for deep packet analysis. However, when reconstructing the hidden structure of packet payload, ProWord ignores the temporal coherence of candidate protocol words in protocol messages. Taking message “250 OK” as an example, protocol word “250” is basically followed by “OK” in network communications, and both words together form a protocol fingerprint of SMTP. It is noteworthy that ProWord often breaks the aforementioned protocol fingerprint “250 OK” into two separate parts, “250” and “OK”. The divided protocol words are incomplete protocol fingerprints for SMTP. The main reason is that

☆ This work was supported by the National Natural Science Foundation of China under Grants No. 61402472.
* Corresponding author.
E-mail addresses: wangyipeng@iie.ac.cn (Y. Wang), zhangyongzheng@iie.ac.cn (Y. Zhang).

http://dx.doi.org/10.1016/j.jnca.2017.10.009
Received 2 March 2017; Received in revised form 4 July 2017; Accepted 5 October 2017
Available online 06 October 2017

ProWord only depends on the entropy information of protocol messages to construct protocol models, and thus misses the opportunity of exploiting temporal information among protocol words.

1.3. Proposed approach

In this paper, we propose ProPrint, a nonparametric and unsupervised approach that performs automated protocol fingerprint inference from the network traces of application protocols. Prior literatures in this field can be roughly classified into two categories, 1) statistics-oriented methods, and 2) string-oriented methods. ProPrint belongs to the second category – string-oriented methods. The input to ProPrint is the packet payload of a given protocol, and the output to ProPrint is the protocol fingerprints for the corresponding protocol. ProPrint is based on the key insight that application protocols can be regarded as a kind of formatted languages for application softwares, such that we can leverage statistical language models for robust and accurate protocol fingerprint inference. The key novelty of our work is that when reconstructing the hidden structure information of packet payload, we explore the temporal coherence (i.e., the Markov property) of protocol words, in which such information is often omitted in prior literatures.

1.4. Key contributions

In practice, ProPrint does not assume any prior knowledge on protocol specifications, such as word boundaries, and it can be effectively applied to both textual and binary protocols. Therefore, ProPrint is a more robust network trace-based system for automatic protocol fingerprint inference. In order to test and verify the effectiveness of ProPrint, we measure ProPrint and conduct extensive evaluations on six real-world protocol traces. Our experimental results show that ProPrint can accurately identify the protocol trace in terms of a precision of 99.01% and a recall of 93.72% on average. Furthermore, we also compare ProPrint to two state-of-the-art solutions, ProWord and Securitas. It is worthy to note that ProPrint reports more effective results in inferring protocol fingerprints than ProWord and Securitas. The main contribution are follows,

- We introduce and present a nested hierarchical Pitman-Yor process to build protocol language models. The proposed model explores the temporal coherence of protocol words and has no“unknown words” problem. In addition, it is a completely unsupervised learning of protocol language models directly from the byte sequences generated by an application protocol.
- We design and implement a lightweight tool called ProPrint, which automatically conducts trace-driven protocol fingerprint inference from the packet payload of protocol traces. As a general solution for fingerprint extraction, our technique is independent of the type of the target application protocol.
- We conduct extensive experimental evaluations on six stateful protocols, including SMTP, FTP, SopCast, BitTorrent, PPStream and PPLive. In addition, we compare our approach to two existing state-of-the-art systems, ProWord and Securitas. Our results of evaluation show that ProPrint is more effective than ProWord and Securitas on f-measure.

The remainder of this paper is organized as follows. In Section 2, we present and introduce the related work regarding protocol fingerprint inference. Section 3 is dedicated to an overview of our proposed system ProPrint. In Section 4 – 6, we present the technical details of each module of ProPrint. Next, in Section 7, we evaluate the whole system with the packet traces of different application protocols. We compare the experimental results of ProPrint to state-of-the-art algorithms in Section 8. Finally, we conclude the whole work in Section 9.

2. Related work

Protocol fingerprinting inference from packet payload is very important for application protocol identification as the inferred fingerprinting can be used as the building blocks for payload oriented protocol classifiers. It is worthy to notice that prior literatures in this field can be roughly classified into two categories, 1) statistics-oriented methods, and 2) string-oriented methods. Next, we give a brief introduction and discussion about the prior methods for the aforementioned two categories.

2.1. Statistics-oriented methods

The protocol fingerprints for statistics-oriented methods are regarded as statistical characterizations of the figures observed in a byte sequence of packets, and they are not particular strings.

The basic operating principle of statistics -oriented methods is as follows, statistics-oriented methods extract statistics information from the packet payload as classification features, and then use them to build machine learning classifiers for protocol identification. For example, KISS (La Mantia et al., 2010; Finamore et al., 2010) proposed by Finamore et al. is a statistics-oriented method that implements Internet classification explicitly targeting on UDP traffic. The protocol fingerprints of KISS are derived by means of a Chi-square like hypothesis testing, and then KISS builds an Support Vector Machine classifier for protocol identification using the learned fingerprints. Securitas is another statistical oriented method proposed by Yun et al. Contrasting KISS, Securitas (Yun et al., 2016) is a more robust system as there is no need for Securitas to assemble IP packets into TCP or UDP flows when carrying out payload oriented application protocol identification. Therefore, Securitas is able to handle more flows with limited memory resources. In addition, ProHacker (Wang et al., 2015) is a recent work that builds a statistics-oriented classifier based on *n*-grams of protocol traces. The advantage of ProHacker is that it focuses on solving the key problem of zero probabilities for *n*-grams not observed in the selected subset for protocol keyword inference.

In the end, we can conclude that statistics oriented methods use statistical quantities of packet payload to distinguish the protocol trace of individual protocols.

2.2. String-oriented methods

In those studies, the early mentioned protocol fingerprints are statistics information but not strings or regular expressions. An alternative way is to find the invariant fields among the packet payload of protocol traces, which are in the form of strings or regular expressions. The works that are closest to our solution ProPrint include ProDecoder and ProWord.

Kannan *et al.* proposed a Poisson process based method for extracting session structures of an application protocol based on the session logs between a pair of end hosts (Kannan et al., 2006). Session structure is referred to as a group of connections associated with a single network task. Kannan *et al.* claim that the session structure extraction focuses on higher-level abstractions of sessions, such as protocol state machine. Therefore, they aim to discover overall patterns of activity rather than detect individual instances, that is to say protocol fingerprint information is omitted in the analysis. In Levchenko et al. (2006), Ma et al. proposed another protocol fingerprint inference method that uses statistical and structural content models based on flow content. Ma's method assumes protocol message formats as a product of byte distributions. Therefore, each byte offset is treated independently. In addition, Ma's method needs to recover TCP flow information using traces of SYN, FIN and RST packets. In ProPrint, we do not need to assemble IP packets into higher application-level sessions, and therefore we can treat each packet independently.