PERFORMANCE OF OPENDPI TO IDENTIFY TRUNCATED NETWORK TRAFFIC

Jawad Khalife, Amjad Hajjar
Faculty of Engineering, IT Department, Lebanese University, Beirut, Lebanon

Jesús Díaz-Verdejo
Department of Signal Processing, Telematics and Communication, University of Granada, Granada, Spain

Keywords: Network traffic identification, Deep packet inspection, Payload truncation.

Abstract: The identification of the nature of the traffic flowing through a TCP/IP network is a relevant target for traffic engineering and security related tasks. Traditional methods based on port assignments are no longer valid due to the use of ephemeral ports and ciphering. Despite the privacy concerns it arises, Deep Packet Inspection (DPI) is one of the most successful current techniques. Nevertheless, the performance of DPI is strongly limited by computational issues related to the huge amount of data it needs to handle, both in terms of number of packets and the length of the packets. This paper addresses the sensitivity of OpenDPI, one of the most powerful freely available DPI systems, when truncation of the payloads of the monitored traffic is applied. The results show that it is highly dependent on the protocol being monitored.

1 INTRODUCTION

Network traffic identification aims to classify packets (packet-based identification) or flows (flow-based identification) in a given network according to the associated application protocol. Traditionally, this task has been considered quite simple as ports were assigned for many application protocols. In this scenario a simple inspection of transport layer header suffices to identify the underlying protocol. Nevertheless, this situation is changing, thus becoming traffic identification a hot research topic, as some Internet applications, such as P2P, are becoming more and more challenging to identification techniques by using port obfuscation, encryption, and/or tunnelling (Nguyen, 2007). One of the most successful methods currently available to identify traffic is based on the examination of the payloads to find known protocol patterns or signatures. This is the so-called DPI (*Deep Packet Inspection*) (Allot Communications, 2007).

However, in today's networks, *performance* and *privacy* issues are two important factors that are considered some of the weaknesses of DPI. On the other hand, DPI is not able to inspect ciphered payloads. This is what is pushing researchers for alternate solutions in which P2P identification is still considered a complex task, especially when DPI is not involved at all.

As such, one of the current research trends is to optimize current DPI based identification methods characterized by their high *accuracy*, while keeping at the same time an acceptable level of user *privacy* and *performance*.

One of the means in optimizing DPI is to *reduce the input size* through *partial payload inspection* or *payload truncation*. However, what impact would *payload truncation* have on DPI accuracy and how far could DPI based methods be optimized through payload truncation are important questions that still need further discussions and experimentation.

In an attempt to answer these questions, we present in this paper, a study on the effect of payload truncation on identification accuracy by using one of the best DPI-based tools: *OpenDPI* (OpenDPI, 2010). Our conducted identification experiments were based on full payload dataset traffic as captured through an institution's Internet link. We tested *OpenDPI* accuracy with different incremental truncation lengths keeping three goals in mind:

(i) To provide protocol oriented results for accuracy as a function of truncation length.

(ii) To show to what extent payload truncation could affect OpenDpi accuracy.
(iii) To draw conclusions on how far combined DPI methods could be optimized through partial payload truncation.

The remaining of this paper is organized as follows. Section 2 provides an overview of payload based identification tools, methods and optimizations. Section 3 describes *OpenDpi* tool in the way it analyzes and labels packets and flows. Section 4 provides a description of the testbed we used for the experiments. Section 5 shows our conducted experiments in running the *OpenDpi* tool with different truncation lengths and then highlights on some obtained results. Finally, Section 6 presents some conclusions and future work.

2 IDENTIFICATION OF FLOWS BASED ON PAYLOADS

Deep Packet Inspection" (DPI) is defined in (Allot Communications, 2007) as being "...*a computer networking term that refers to devices and technologies that inspect and take action based on the contents of the packet (commonly called the "payload") rather than just the packet header.*"

The most important parts of DPI are *regular expression* matching and signature based scanning. In this technique the payload of all the packets is checked against the set of known protocol signatures (Erazm, 2007)**.**

Some well-known DPI technology based tools are *OpenDPI* (OpenDPI, 2011) —an open source traffic classification tool —, *L7-filter* (L7filter, 2011) — an open source application layer classifier for Linux's Netfilter— and *Snort* (Snort, 2011) — an open source network intrusion prevention and detection system —. In this paper, our choice was to use the *OpenDpi* tool since it includes the latest DPI technology combined with other techniques making it one of the most accurate classifiers (see Sec. 3).

Many authors attempted to enhance DPI *accuracy* by combining it with other methods, such as behavioural (Zhang, 2010), statistical (Dehghani, 2010), port based (Aceto, 2010) and DFI (*Deep Flow Identification*) based methods (Wang, 2008).

On the other hand, many recent works attempted to optimize DPI *performance* for high link speeds. Some of them apply software based optimization focused on enhancing DPI algorithms — e.g. (Yang, 2010) (Lin, 2008) — while others use hardware based optimization — e.g. (Rao, 2010) —.

In this paper, we will focus on a software optimization which consists on reducing the size of DPI input through partial payload inspection. In this context, different methods were proposed in the literature. For instance, ML (*Machine Learning*) identification methods (Nguyen, 2007) use the *feature selection algorithm.* On the other hand, *sampling techniques* are more general and easy to implement as they just try to reduce the size of the input data by simply taking samples or parts from the data according to a given criteria. This later approach could be jointly applied with DPI. In fact, this is the scenario considered in this work.

Sampling network traffic is the process of taking partial observations from the monitored traffic, and drawing conclusions about the behaviour of the system from these sampled observations. They are mainly used for network management and monitoring (Jurga, 2007) although may also be used in classification tasks — e.g. (Ficara, 2010) (Aceto, 2010) (Fernandes, 2009) —.

A detailed taxonomy of sampling techniques according to the used method is provided in (Jurga, 2007). Another way of categorizing sampling techniques is related to the target considered by the method. From this point of view, they can be classified as *per-flow packet sampling* (Carela, 2010) —*i.e.* sampling a subset of packets from within the whole traffic flow—, *per-packet payload sampling* (Ficara, 2010) (Aceto, 2010) —*i.e.* sampling bytes from within the packet payload— or a combination of both (Fernandes, 2009).

While sampling obviously provides a significant impact on the processing times by reducing the size of the input to process, it is a lossy process which may have an unexpected impact on the traffic classification techniques.

Apparently, few works apply sampling to network traffic classification, and even fewer works apply per-packet payload sampling to DPI classification —e.g. (Ficara, 2010) (Fernandes, 2009)—, a topic which is the most relevant to our work.

Per-packet payload sampling was shown in (Ficara, 2010), where authors proposed a novel approach that brings the sampling idea to the regular expression field. Their approach, called *payload sampling*, allows skipping a large portion of the text in the payload, thus processing less bytes. Their results show that the sampling approach is faster than previous advanced solutions. However, the price to pay is a slight number of false alarms which require a confirmation stage.

Another example of per-packet byte sampling was shown in (Aceto, 2010) which also combined the *port-based* method with the DPI approach. Using L7-Filter (L7filter, 2011) DPI tool, one of the paper's targets was to study the amount of payload information actually relevant in successful DPI matches. Their experimental results showed that 72% of the total attempts happen at the *first packet* of a flow. Moreover, they computed the offset of matching regular expression's first character and last character from the beginning of the packets respectively containing them. They showed that almost all matching strings start (99.98%) and finish (90.77%) in the first *32 bytes* of payload.

The combination of per-flow and per-packet sampling is addressed in (Fernandes, 2009). In this work the authors combined both sampling methods through the so-called LW-DPI. Results showed that most flows can be classified with only their first 7 packets or a fraction of their payload.

In this paper*, payload truncation* is used and explored as a sampling method. It can be thought of as a particular case of per-packet payload sampling for DPI methods. We consider it as an eventual mean of DPI optimization as only part of the payload (the initial truncated part) is to be inspected instead of the full payload.

To the best of our knowledge, except for (Aceto, 2010) and (Fernandes, 2009), no papers were found to be discussing per-packet payload sampling for DPI based methods. In addition, no results were found to be protocol oriented.

3 openDPI

As previously stated, the tool of choice for the classification of traffic is *openDPI* (Opendpi, 2010), which is derived from the commercial *PACE* product from *ipoque*. The core of openDPI is a software library designed to classify internet traffic according to application protocols. In (Opendpiwhite) the authors explain that the DPI-based protocol and application classification is achieved using pattern matching, behavioural analysis and statistical analysis.

Therefore, *openDPI* is not a pure-DPI product as it is not only signature-based but also incorporates information from other sources. This way, the classification accuracy is improved (no false classification according to ipoque's claims), although some packets and flows still remains unclassified. This, together with the availability and quality of the signatures, made us to select *openDPI* instead of any other similar product.

In its current version, up to 101 different protocols can be identified, including the most common ones as HTTP and DNS, and some P2P protocols as eDonkey, DirectConnect, etc.

Nevertheless, and according to its functioning, the capabilities of OpenDPI are mainly limited by the need to analyze the whole payload of all the packets in a flow in search of signatures (DPI behaviour) and to extract the behavioural and statistical information from the flows. Therefore, it is a basically full payload / full flow analysis which imply a high computational cost. This way, it would be desirable to reduce the size of the explored data in order to reduce this computational cost, but without degrading the performance of the classifier.

In this context, the target of this paper can be stated as analyzing how sensitive are the mechanism involved in OpenDPI to the truncation of the payloads.

4 TESTBED

In order to evaluate the effects of truncating the payloads in the traffic identification task, we have developed an experimental setup built from two main components. These components are a database of real traffic captured in an academic network, and a tool to automatically classify packets and flows according to their payloads by primarily using Deep Packet Inspection (DPI) which is based in OpenDPI.

Therefore, we have built a tool based on the *openDPI* library which is able not only to identify the application protocols but also to follow and differentiate the packets in each flow. To be able to handle UPD packets, we have generalized the concept of flow through the use of *sessions*. *Sessions* are considered as defined by the exchange of information associated to a tuple (IP addresses, ports and transport protocol). Nevertheless, throughout this paper, we will use the term *flow* to refer to a *session,* unless explicitly stated.

As the output of the tool, two levels of classifications are provided: flow-based (each flow is labelled) and packet-based (each packet is also labelled). The tool operates in batch mode.

On the other hand, the traffic database contains the data captured during 3 working days at the access link of a medium size institution. The data acquisition was carried out at a border router in order to be able to monitor all incoming and outcoming traffic. Therefore, apart from the boundaries of the caption, flows are captured complete and in both directions. Table 1 highlights some figures of the database.

Table 1: Figures for the captured traffic database.

Size of the database	~180 GB
Number of IP packets	278 Mpackets
Number of different IPs	822519
Number of flows	6.3 Mflows
Number of identified protocols	42

By using the customized OpenDPI tool over the whole database we have built the "ground truth", that is, the set of correctly labelled flows and packets that will be used as the reference for any further analysis. This procedure is adopted under the assumption that DPI is the best currently available method for traffic classification and that the number of errors is negligible. This is a common approach in the traffic identification field, the number of packets and flows that DPI is not able to classify being its major limitation.

The results provided by the classification tool show up to 42 protocols that have been identified in the database, being HTTP the most frequent one, while an important part of the flows and packets remain unclassified. The relative distributions of flows and packets for most relevant protocols are shown in Figure 1. A first inspection evidences big differences among the properties or frequencies at flow and packet levels. Therefore, the results can be different depending on whether we focus at flow or packet levels.

5 TRUNCATION OF THE PAYLOADS

In this section, we will show the conducted experiments in running OpenDPI on partially truncated packet payloads. Our main targets at this level are, as mentioned in Section 1*: (i) To provide protocol oriented results for accuracy as a function of truncation length (ii) to show to what extent payload truncation could affect OpenDpi accuracy*

5.1 Methodology for Truncation Experiments

To achieve our targets, we customized OpenDPI tool to be able to parse only a specified length of byte (called *truncation length*) within each packet's payload. In order to obtain granular results, our choice was to iterate with incremental *truncation length* values with step of S Bytes, ranging from 0 Bytes (*no payload*) to 1500 Bytes (*full payload*). We have chosen S=128 Bytes.

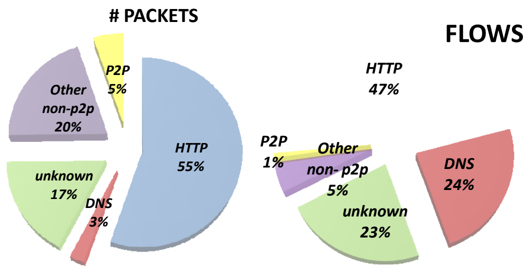

Figure 1: Distribution of packets (left) and flows (right) for most relevant protocols or groups of protocols.

Since the main dataset is very huge (63 captured *pcap* files totalling 177G), the customized OpenDPI tool was run on a subset of 17 randomly selected files only (totalling 45G, i.e. 25% of the full dataset). On the other hand, those packets and flows that were not classified by OpenDPI when using the whole payload are dismissed and not considered in the figures and percentages that will be shown.

The evaluation of the identification provided by OpenDPI is measured in terms of *accuracy* (Nguyen, 2007), that is, the percentage of detected packets/flows in regard to the full payload case. The results are shown as a function of the truncation lengths and grouped according to three different sets: per protocol, per protocol group, and for all the protocols. This way, the protocols where categorized into 12 groups that were defined according to (Ipoque, 2011).

5.2 Global Results

The results obtained for all the protocols are shown in Figure 2. At packet level, a sudden drop in the accuracy for truncation lengths lower than 1280 Bytes is observed. For 1280 Bytes, 47% of the packets were correctly classified, while for 1408 Bytes, 99% of all the packets were identified. On the other hand, the results at flow level, show that for truncation length equals to 512 Bytes, 57% of total flows were detected, while for 1280 Bytes, 91% all flows were detected. Therefore, the analysis is more tolerant to payload truncation at flow levels than at packet levels.
This way, in order to reach 50% of both flows and packets accuracy, truncation length must be at least 1280 Bytes. This is not a very encouraging result for DPI optimization through payload truncation as reducing only 15% of payload input would lead to a 50% drop in OpenDPI packet accuracy. However, results are encouraging if only flow accuracy is the main concern since still 57% of flows can be detected for 512 bytes of truncation.

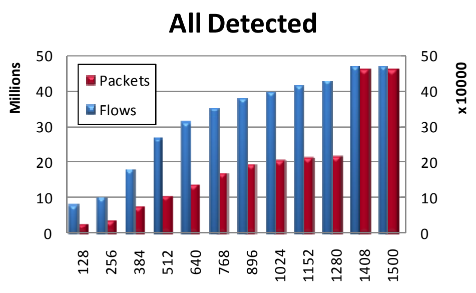

Figure 2: Global results for classification accuracy as a function of the truncation length of the payloads.

From a macroscopic point of view, OpenDPI showed a common behaviour for all protocols:

i. The number of detected packets/flows is increasing as the truncation length increases.
ii. For truncation length equal to 512 Bytes, 57% of flows were detected while only 22% of packets were detected.

Unless just a few bytes (not more than 128 Bytes) were truncated from the end of the packet payload, payload truncation with combined DPI/DFI will lead to many unknown flows and packets. Thus, optimizing DPI/DFI methods through payload truncation could not be considered generally effective for all protocols (especially when the set includes stateful protocols, which are the more affected) unless packet accuracy is not considered.

5.3 Results per Protocol Group

When varying the truncation length, OpenDPI shows different behaviour for different protocol groups.

As an example, results for web group packets and flows are shown in Figure 3.a. Web group results show that truncation, though differently, is affecting both packet and flow accuracy. In addition, web packet accuracy seems to be more affected by truncation than flow accuracy. It's noticeable that packet classification accuracy drops to around 50% for 1280 Bytes while for flow accuracy it drops to 50% only if less than 512 Bytes are truncated.

A different behaviour is observed for other groups. For example, if we consider the IM (Internet Messaging protocols) group –Figure 3.b– or DNS group –Figure 3.c– the classification accuracy is only slightly affected by truncation. In fact, for a truncation length equal to 256 Bytes, more than 50% of both packets and flows are detected. The same applies for DNS packets and flows.

The results for P2P protocols exhibit a mixed behaviour –Figure 3.d–: they are similar to those from the web group at packet level and to those from

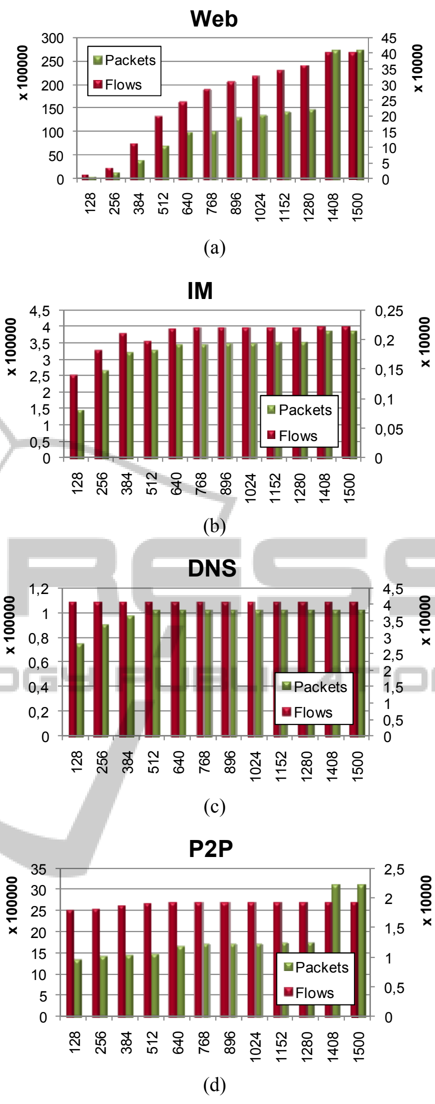

Figure 3: Results for various protocols/groups as a function of the truncation length for packets and flows. a) Web; b) Internet messaging; c) DNS; and d) P2P.

IM and DNS groups at flow level. In fact, packet accuracy drops to around 50% for 1280 Bytes while flow accuracy stays above 92% even for 128 Bytes only.

In summary, at a granular level, the experimental results showed different behaviour for OpenDPI with truncation for different protocols. This in fact could be based on two main factors: the stateful behaviour of some protocols combined with the detection algorithm used by OpenDPI which considers some behavioural and statistical information for the whole flow.

We can evidence this assert if we examine the obtained results for the web and DNS protocol groups. Since DNS is a stateless protocol, flows with truncated packets can still be detected. On the other side, as web is a stateful protocol, the detection of web flows drops for truncated packets. Though not shown, FTP results also were different since FTP protocol has a special behaviour.

Therefore, we can conclude that stateless protocols are less sensitive to payload truncation than stateful ones. Thus, optimizing DPI/DFI methods through payload truncation could be more effective for stateless and P2P protocols.

For interpreting the differences between flow and packet results for the same protocol, flow results are considered more significant since undetected flows may contain a huge number of packets thus affecting packet accuracy. We also noticed that flows detected at higher truncation length mostly contain a huge number of packets.

6 CONCLUSIONS AND FUTURE WORK

The effects of truncating packet payloads when using OpenDPI are explored in this paper. The experiments has shown that, unless just few bytes (not more than 128 Bytes) were truncated from the end of the packet payload, payload truncation for this method will lead to many unknown packets and flows decreasing the accuracy of the classification. The obvious interpretation is that by combining DPI with other technologies (such as behavioural and statistical modeling), the task of DPI optimization through truncation may render the identification method itself inefficient since the non parsed part of the data may still be needed for the other added technology. The truncation can still be useful as an optimization if, instead of classifying all the traffic, the target is to select some of them based on the application content and depending on the nature of the associated protocol.

An apparent contradiction emerged between the combination of identification technologies and the optimization through partial payload inspection procedures. Tradeoffs should be most probably one of the next steps to explore. Additional experiments have to be carried out to analyze the sensitivity to flow truncation, that is, to consider just a selected number of packets per flow.

ACKNOWLEDGEMENTS

This work has been supported by Spanish MICINN under project TEC2008-06663-C03-02.

REFERENCES

Aceto, G., Dainotti, A., de Donato, W., Pescapé, A., 2010. PortLoad: taking the best of two worlds in traffic classification, In Proc. of IEEE INFOCOM 2010.

Allot Communications, 2007. Digging Deeper Into Deep Packet Inspection (DPI). *White paper. Available at http://www.dpacket.org*

Carela-Español, V., Barlet-Ros, P., Cabellos-Aparicio, A., Solé-Pareta, J., 2010. Analysis of the impact of sampling on NetFlow traffic classification, *Computer Network* (In press), Elsevier.

Dehghani, F., Movahhedinia, N., Khayyambashi, M. R., Kianian, S., 2010. Real-time Traffic Classification Based on Statistical and Payload Content Features, *In Proc. IWISA 2010,* pp. 1-4.

Fernandes, S., Antonello, R., Lacerda, T., Santos, A., Sadok, D., Westholm, T., 2009. Slimming Down Deep Packet Inspection Systems, *In Proc. INFOCOM Workshops 2009*, pp. 1-6.

Ficara, D., Antichi, G., Di Pietro, A., Giordano, S., Procissi, G., Vitucci, F., 2010. Sampling Techniques to Accelerate Pattern Matching in Network Intrusion Detection Systems, *In Proc. 2010 ICC2010,* pp. 1-5.

Ipoque, 2011. http://www.ipoque.com/

Jurga, R. E., Hulbój, M. M., 2008. Packet Sampling for Network Monitoring, *Technical Report*, CERN | HP Procurve openlab project. Available at http://www.zdnetasia.com/whitepaper/packet-sampling-for-network-monitoring_wp-1828217.htm

La Mantia, G., Rossi, D., Finamore, A., Mellia, M., Meo, M., 2010. Stochastic Packet Inspection for TCP Traffic. *In Proc. ICC2010,* pp. 1-6.

Lin, P., Lin, Y., Lee, T., Lai, Y., 2008. Using String Matching for Deep Packet Inspection. *IEEE Computer*, vol. 41, pp. 23-28.

L7filter, 2011. http://l7-filter.clearfoundation.com/

Nguyen, T., Armitage, G., 2007. A Survey of Techniques for Internet Traffic Classification using Machine Learning, *IEEE Communications Surveys & Tutorials*, vol. 10, pp. 56-76.

Opendpi, 2011. http://www.opendpi.org/

Rao, A., Udupa, P., 2010. A Hardware Accelerated System For Deep Packet Inspection, *In Proc. MEMOCODE'10,* pp. 89-92.

Snort, 2011. http://www.snort.org

Yang, Y.-H. E., Hoang Le,Prasanna, V. K., 2010. High Performance Dictionary-Based String Matching for Deep Packet Inspection. *In Proc. of INFOCOM 2010*, pp. 1-5.

Wang, C., Zhou, X., You, F., Chen, H., 2008. Design of P2P Traffic Identification Based on DPI and DFI, *In Proc. of CNMT2009,* pp. 1-4.

Zhang, L., 2010. P2P-based Weighted Behavioral Characteristics Of Deep Packet Inspection Algorithm, *In Proc. of CMCE 2010,* pp. 468-470.