Independent Comparison of Popular DPI Tools for Traffic Classification

Bujlow, Tomasz; Carela-Español, Valentín ; Barlet-Ros, Pere

Published in:
Computer Networks

DOI (link to publication from Publisher):
10.1016/j.comnet.2014.11.001

Publication date:
2015

Document Version
Accepted author manuscript, peer reviewed version

Citation for published version (APA):
Bujlow, T., Carela-Español, V., & Barlet-Ros, P. (2015). Independent Comparison of Popular DPI Tools for Traffic Classification. *Computer Networks*, *76*(0), 75-89. https://doi.org/10.1016/j.comnet.2014.11.001

Independent Comparison of Popular DPI Tools for Traffic Classification

Tomasz Bujlow[a,*], Valentín Carela-Español[b], Pere Barlet-Ros[b]

[a] *Section for Networking and Security, Department of Electronic Systems, Aalborg University, DK-9220 Aalborg East, Denmark*
[b] *Broadband Communications Research Group, Department of Computer Architecture, Universitat Politècnica de Catalunya, ES-08034 Barcelona, Spain*

Abstract

Deep Packet Inspection (DPI) is the state-of-the-art technology for traffic classification. According to the conventional wisdom, DPI is the most accurate classification technique. Consequently, most popular products, either commercial or open-source, rely on some sort of DPI for traffic classification. However, the actual performance of DPI is still unclear to the research community, since the lack of public datasets prevent the comparison and reproducibility of their results. This paper presents a comprehensive comparison of 6 well-known DPI tools, which are commonly used in the traffic classification literature. Our study includes 2 commercial products (*PACE* and *NBAR*) and 4 open-source tools (*OpenDPI*, *L7-filter*, *nDPI*, and *Libprotoident*). We studied their performance in various scenarios (including packet and flow truncation) and at different classification levels (application protocol, application and web service). We carefully built a labeled dataset with more than 750 K flows, which contains traffic from popular applications. We used the Volunteer-Based System (VBS), developed at Aalborg University, to guarantee the correct labeling of the dataset. We released this dataset, including full packet payloads, to the research community. We believe this dataset could become a common benchmark for the comparison and validation of network traffic classifiers. Our results present *PACE*, a commercial tool, as the most accurate solution. Surprisingly, we find that some open-source tools, such as *nDPI* and *Libprotoident*, also achieve very high accuracy.

Keywords: Deep Packet Inspection, PACE, nDPI, Libprotoident, NBAR, L7-filter

1. Introduction

Network communication became the standard way of exchanging information between applications located on different hosts. The exchanged application-layer data is segmented and encapsulated into IP packets, which are transmitted through the network. Deep Packet Inspection (DPI) tools analyze the content of the packets by searching for specific patterns (i.e., signatures). Thus, DPI became one of the fundamental traffic analysis methods for many tools performing traffic classification, network management, intrusion detection, and network forensics.

DPI-based traffic classification relies on a database of characteristic signatures of protocols (e.g., HTTP, or POP3), applications (e.g., Skype, or BitTorrent), and web services (e.g., Facebook, or YouTube). These signatures must be initially extracted and kept up to date in order to adapt them to the continuous evolution of the applications (e.g., new applications, new versions, new obfuscation techniques). They are later employed, usually in an online phase, to classify the traffic flowing in a network. The pattern matching algorithms for online classification are computationally complex and usually require expensive hardware. Nevertheless, DPI is commonly considered as the most accurate technique for traffic classification, and most commercial solutions rely on it [1, 2, 3, 4].

The scientific literature shows that the existing traffic classification techniques give accurate results. Unfortunately, as pointed out in [5], there are numerous problems with the comparison and validation of these techniques. The quality of the validation process directly depends on the methods used to build and pre-classify the ground-truth dataset, so it is not an easy task to compare how various classifiers perform. The researchers must choose between two approaches of ground-truth establishment: creating their own dataset, or using an already existing dataset created by someone else.

The first approach requires the researcher to build and label the dataset by himself. Both of these tasks are challenging. Building the dataset involves the selection of the applications to be included in the dataset. Data labeling is usually carried out by DPI tools given their presumably high accuracy. However, the actual accuracy of DPI-based tools is still not clear, as the previous works that tried to compare the performance of different DPI tools showed that the ground-truth used to validate the proposals was obtained through port-based classifiers, other DPI-based tools, or methodologies of unknown reliability [6, 7, 8, 9, 10]. Thus, the results of the comparison are arguable. In addition, most commercial tools are black boxes that claim high accuracy based on their own studies, which cannot be validated, because they were performed using private

[*] Corresponding author. Tel.: +45 5026 2494

Email addresses: `tomasz@bujlow.com` (Tomasz Bujlow), `vcarela@ac.upc.edu` (Valentín Carela-Español), `pbarlet@ac.upc.edu` (Pere Barlet-Ros)

URL: `http://tomasz.bujlow.com` (Tomasz Bujlow), `http://personals.ac.upc.edu/vcarela/` (Valentín Carela-Español), `http://people.ac.upc.edu/pbarlet/` (Pere Barlet-Ros)

datasets.

The second approach to the proper validation of traffic classification techniques is the use of publicly available datasets. This way, it is easier to compare the results obtained from different tools. Examples are the datasets published by the Cooperative Association for Internet Data Analysis (CAIDA) [11] or the Internet Measurement Data Catalog [12]. Although the datasets are pre-classified, the actual reliability of this labeling is unknown, which is a very important factor for testing traffic classifiers. Most of the traces have no payload or just the first bytes of each packet. The MAWI repository [13] contains various packet traces, including daily 15-minutes traces made at an trans-Pacific line (150 Mbit/s link). The bandwidth of this link has changed through the years. The traces contain the first 96 bytes of the payload and the traffic is usually asymmetric. Another useful data source is the Community Resource for Archiving Wireless Data At Dartmouth (CRAWDAD) [14], which stores wireless trace data from many contributing locations. Another interesting project is The Waikato Internet Traffic Storage (WITS) [15], which aims to collect and document all the Internet traces that the WAND Network Research Group from the University of Waikato has in their possession. Some of the traces can be freely downloaded and they contain traffic traces from various areas and of different types (as DSL residential traffic, university campus traffic, etc). Most of the traces do not have payload (i.e., it is zeroed) or truncated. All these datasets although useful for many network evaluations are of limited interest for DPI validation given that no correct labeling can be performed on them.

Szabo et al. [16] introduced a method for validation of classification algorithms, which is independent of other classification methods, deterministic, and allows to automatize testing of large data sets. The authors developed a Windows XP driver based on the Network Driver Interface Specification (NDIS) library. Another approach to obtain the ground-truth was taken in [17]. The authors created a tool, which collects the data from the network and labels the flows with the real application names (e.g., *Thunderbird*) and application protocol names (e.g., *SMTP*). This tool is similar to our Volunteer-Based System (VBS), which was used in our work for the ground-truth generation and is further described in Section 3.1.1. Yet another way of establishing the ground-truth was shown in [18], which describes a system developed to accelerate the manual verification process. The authors proposed Ground Truth Verification System (GTVS) based on the DPI signatures derived from the databases available in the Internet, including L7-filter. GTVS, however, does not collect the application names from the operating systems, so the established truth cannot be completely verified.

In a former conference paper [19], we tried to face the problem of ground-truth reliability. We described various methods of obtaining the ground-truth for testing various traffic classification tools. As part of the evaluation, we tested several DPI-based traffic classifiers and assessed if they can be used for obtaining reliable ground-truth. This paper is not only an extension but a more broad and comprehensive validation of DPI-based tools. The focus of this paper is different, as we directly compare the selected DPI tools regarding their per-class accuracy. Reference datasets used in this paper as well as the previous one are generated by the same methodology further explained in Section 3. However, both datasets are significantly different. The dataset used in this paper is significantly larger than the one used in the conference paper. Furthermore, the new dataset also contains labeled non-HTTP flows belonging to various web services, which is a unique feature. The methodology of testing the classifiers is also different. In our previous paper, we tested the accuracy of the classifiers on a single level. In the current paper, we evaluate different levels (i.e., application, web service, etc), so the new evaluation method is more detailed and complete.

This paper compares and validates six well-known DPI-based tools used for network traffic classification. In order to allow the validation of our work, we publish the reliable labeled dataset used to perform our study. Two main aspects have been carefully addressed when building this dataset: the reliability of the labeling and the representativeness of the data. We used the VBS tool [20] to guarantee the correctness of the labeling process. This tool, described in Section 3, is able to label the flows with the name of the process that creates them. This allowed us to create a reliable ground-truth that can be used as a reference benchmark for the research community to compare other proposals. The selection of applications for our dataset was made based on well-known indexes of the most commonly used Internet applications and web services. In order to allow the publication of the dataset and avoid any privacy issues, we created the traffic by running a large set of applications and meticulously simulating common behaviors of the users.

The main contributions of this paper can be summarized as follows:

- We published a reliable labeled dataset with full packet payloads. The dataset, further described in Section 4, contains traffic from a diverse set of commonly used applications. Although artificially created, we carefully simulated human behaviors in order to produce a dataset that is as much realistic as possible. This dataset can be used by the research community to test and compare various traffic classifiers.

- We compared six well-known DPI-based tools widely used for network traffic classification. Using the previous dataset, we evaluated the precision of PACE, OpenDPI, nDPI, L7-filter, Libprotoident, and NBAR, and compared their results on various classification levels and in different scenarios. We are aware that OpenDPI and L7-filter are abandoned projects, on which development stopped several years ago. However, we decided to include them into our evaluation as a reference and for completeness, as many existing scientific papers base their results on these two classifiers.

- We provide an independent and impartial analysis of the performance of these tools at different classification levels. These results can help researchers to better understand the accuracy of the different tools used to set their ground truth

Table 1: DPI tools included in our comparison

Name	Version	Released	Apps. identified
PACE	1.47.2	November 2013	1000
OpenDPI	1.3.0	June 2011	100
nDPI	rev. 7543	April 2014	170
L7-filter	2009.05.28	May 2009	110
Libprotoident	2.0.7	November 2013	250
NBAR	15.2(4)M2	November 2012	85

as well as network managers to better decide which DPI-based classifiers are more suitable for their needs and scenarios. Indirectly, we also provide the information about the reliability of those non-DPI classification techniques proposed in the literature that used one of the DPI tools compared in this paper to set their ground truth.

In this paper, we focus on a single performance parameter: the classification accuracy. We acknowledge that other performance parameters are also important, such as speed, scalability, complexity, robustness, or price of the solutions. However, given that the research community is mainly using DPI-based technique for offline ground-truth generation, we think that those metrics are less decisive. Furthermore, we believe that our dataset will be useful to evaluate most of these parameters as well.

The remainder of this paper is organized as follows. Section 2 describes the DPI-based tools and their configurations used for the evaluation. Section 3 presents the methodology used to obtain the reliable dataset that is described in Section 4. Section 5 presents the results of the performance evaluation of the different DPI-based tools. Section 6 discusses the results, compares them with the literature, and comments the limitations of our evaluation. Section 7 reviews the related work. Finally, Section 8 concludes and summarizes the outcomes and contributions of the paper.

2. Classification tools

On the market, there are many available DPI-based traffic classification solutions. For our experiment, we selected PACE, OpenDPI, nDPI, Libprotoident, NBAR, and L7-filter, which will be broadly introduced in this section. The selection was based on the popularity of the tools used in the scientific literature or embedded in the operating systems available on the market to classify the traffic or set the ground truth. In particular, we evaluated those tools available to researchers, including most popular open source tools, as well as those commercial tools whose vendors accepted to share with us free of charge. Table 1 summarizes these DPI-based tools along their characteristics.

PACE. It is a proprietary classification library developed by *ipoque* entirely in C, which supports classical DPI (pattern matching), behavioral, heuristic, and statistical analysis. According to its website, PACE is able to detect encrypted protocols as well as protocols which use obfuscation. Overall, more than 1000 applications and 200 network protocols are supported. It is also possible to include user-defined rules for detection of applications and protocols. To the best of our knowledge, PACE is the only commercial tool used in the literature to build the ground truth [8].

OpenDPI. It was an open-source classifier derived from early versions of PACE by removing support for encrypted protocols, as well as all performance optimizations. The project is now considered as closed. In [6, 7], the authors mention that OpenDPI is not a classic DPI tool, as it uses other techniques apart from pattern matching (i.e., behavioral and statistical analysis). Thanks to that, it should severely reduce the amount of false classification, but some traffic can remain unclassified [6]. Another interesting feature is flow association, which relies on inspecting the payload of a known flow to discover a new flow, as inspecting a control FTP session to obtain the five tuple of the newly initiated data session [10].

nDPI. It is an OpenDPI fork, which is optimized and extended with new protocols [21]. It re-introduced support for many encrypted ones due to analysis of session certificates. Overall, nDPI for now supports more than 100 protocols [22]. The current architecture is scalable, but it does not provide the best performance and results: each of the protocols has its own signature scanner, through which the packets are examined. Every packet is examined by each scanner, regardless, if a match was found. If there are multiple matches per flow, the returned value is the most detailed one [10]. Additionally, there is no TCP or IP payload re-assembly, so there is no possibility to detect a signature split into multiple TCP segments / IP packets [22].

Libprotoident. This C library [8] introduces Lightweight Packet Inspection (LPI), which examines only the first four bytes of payload in each direction. That allows to minimize privacy concerns, while decreasing the disk space needed to store the packet traces necessary for the classification. Libprotoident supports over 200 different protocols and the classification is based on a combined approach using payload pattern matching, payload size, port numbers, and IP matching.

Cisco Network Based Application Recognition (NBAR). It was developed to add the ability to classify the network traffic by using the existing infrastructure [23]. It is able to classify applications, which use dynamic TCP and UDP port numbers. NBAR works with Quality of Service (QoS) features, thanks to what the devices (e.g., routers) can dynamically assign a certain amount of bandwidth to a particular application, drop packets, or mark them in a selected way. The authors claim that NBAR supports a wide range of stateful protocols, which are difficult to classify.

L7-filter. It was created in 2003 as a classifier for Linux Netfilter, which can recognize the traffic on the application layer [24]. The classification is based on three techniques. At first, simple numerical identification based on the standard iptables modules, which can handle port numbers, IP protocol numbers, number of transferred bytes, etc. At second, payload pattern matching based on regular expressions. At third, the applications can be recognized based on functions. L7-filter is developed as a set of rules and a classification engine, which can be used independently of each other. The most recent version of L7-filter classification engine is from January, 2011, and the classification rules from 2009.

3. Methodology

Our experiment involved numerous steps, which will be defined and described in this section. We had two main goals – building the dataset and testing the classifiers. Each of them required an individual methodology.

3.1. Building of the dataset

3.1.1. The testbed

Our testbed consisted of 7 machines, which were used for running the selected applications and generating the traffic data, and of a server. We equipped the data generating machines with Windows 7 (3 machines), Ubuntu (3 machines), and Windows XP (1 machine). The additional Ubuntu server machine was equipped with a MySQL database for data storage.

To collect and accurately label the flows, we adapted the Volunteer-Based System (VBS) developed at Aalborg University [25]. The goal of the VBS project is to collect flow-level information from the Internet traffic (e.g., start time of the flow, number of packets contained by the flow, local and remote IP addresses, local and remote ports, transport layer protocol) together with detailed information about each packet (e.g., direction, size, TCP flags, and relative timestamp to the previous packet in the flow). For each flow, the system collects the process name associated with it, which is obtained from the system sockets. Additionally, the system collects some information about the types of the transferred HTTP contents (e.g., *text/html*, *video/x-flv*). The captured information is transmitted to the VBS server, which stores the data in a MySQL database.

On every data generating machine, we installed a modified version of VBS. The source code of the original system as well as the modified version is publicly available in [26] under *GNU General Public License v3.0*. The modified version of the VBS client captures, apart from the data described above, full Ethernet frames for each packet, and extracts the HTTP *URL* and *Referrer* fields – all the information is transmitted to the server, and stored in the MySQL database. We added a module called *pcapBuilder*, which is responsible for dumping the packets from the database to PCAP files. At the same time, INFO files are generated to provide detailed information about each flow, which allows us to assign each packet from the PCAP file to an individual flow.

3.1.2. Selection of the data

The process of building a representative dataset, which characterizes a typical user behavior, is a challenging task, crucial from the point of testing and comparing different traffic classifiers. Therefore, to ensure the proper diversity and amount of the included data, we decided to combine the data on a multidimensional level. Based on w3schools statistics [27], we found that most PC users use Windows 7 (56.7 % of users), Windows XP (12.4 %), Windows 8 (9.9 %), and Linux (4.9 %) - state for October 2013. Apple computers contribute for 9.6 % of the overall traffic, and mobile devices for 3.3 %. Because of the lack of the equipment and/or software for Apple computers, Windows 8, and mobile devices, we decided to include in our study Windows 7 (W7), Windows XP (XP), and Linux (LX), which cover now 74.0 % of the used operating systems.

The application protocols, applications, and web services selected for this study are shown below. To group them, we adopted the same classification categories used in the reports from Palo Alto [28].

1. File-sharing applications. According to [28], they account for 6 % of the total bandwidth. Inside that group, BitTorrent accounts for 53 %, FTP for 21 %, Dropbox for 5 %, Xunlei for 4 %, and eMule for 3 %. Based on the statistics found in [28], as well as those in the CNET [29] and the OPSWAT P2P clients popularity list, the CNET FTP clients popularity list [30], and the Direct Download popularity list [31], we selected the following applications:
   - BitTorrent: uTorrent (Windows), kTorrent (Linux).
   - eDonkey: eMule (Windows), aMule (Linux). The studied configurations were: outgoing-non-obfuscated-incoming-all, all-obfuscated.
   - FTP: FileZilla (Windows, Linux) in active mode (PORT) and passive mode (PASV).
   - Dropbox (Windows, Linux).
   - Web-based direct downloads: 4Shared (+ Windows app), MediaFire, Putlocker.
   - Webdav (Windows).
2. Photo-video group. According to the reports from Palo Alto [28], they account for 16 % of the total bandwidth, where YouTube accounts for 6 % of total, Netflix for 2 % of total, other HTTP video for 2 % of total, RTMP for 2 % of total, and others for 4 % of traffic in total. To select the applications in this category we also used the Ebizmba ranking of video websites [32].
   - YouTube: most watched videos from all the times according to the global ranking [33].
   - RTMP: around 30 random short live video streams (1–10 minutes) were watched from Justin.tv.
   - Vimeo – a web-based photo sharing solution.
   - PPStream (Windows) – P2P streaming video software.
   - Other HTTP video.
3. Web browsing traffic. Based on w3schools statistics [34], the most popular web browsers are: Chrome (48.4 % of users), Firefox (30.2 %), and Internet Explorer (14.3 %).

These browsers were used to generate the web traffic. According to the reports from Palo Alto [28], they account for 20 % of the total bandwidth. The selection of the websites was based on Alexa statistics [35], Ebizmba web statistics [36], Quantcast statistics [37], and Ebizmba search engines popularity [38]. In order to make the dataset as representative as possible, we simulated different human behaviors when using these websites. For instance, on Facebook, we log in, interact with friends (e.g., chat, send messages, write in their walls), upload pictures, create events or play games. Similar behaviors were simulated for other popular web services, such as Twitter, Google+, eBay, etc. The detailed description of actions performed with the services is listed in our technical report [39].

4. Encrypted tunnel traffic. According to the reports from Palo Alto [28], they account for 9 % of the total bandwidth, where 6 % of total is SSL and 2 % of total is SSH.
   - SSL (Windows, Linux): collected while using various applications and web services.
   - SSH (Linux).
   - TOR (Windows). First, we used TOR to browse various websites and download big files. Then, we configured TOR to act as an internal relay, so we participated in creating the invisible path for other users.
   - Freenet (Windows): for browsing the intranet and also as a relay for other peers.
   - SOCKSv5 (Windows). We created a SOCKSv5 server on Linux, and used it for Firefox and uTorrent.

5. Storage-backup traffic. According to Palo Alto [28], they account for 16 % of the total bandwidth, where at least half of the bandwidth is consumed by MS-SMB, and the rest by many different applications. Therefore, the only tested application was MS-SMB (Windows, Linux).

6. E-mail and communication traffic. According to the reports from Palo Alto [28], e-mail traffic accounts for 3 % of the total bandwidth. E-mail market share from October 2013 [40] shows that only one desktop mail client, Microsoft Outlook (17 %), is in the top 10 of used mail clients. The rest is split between web-based clients (as GMail) and mobile clients (Mac, Android). The tested applications / web-based mail services include: Gmail, Hotmail, Windows Live Mail (Windows), and Mozilla Thunderbird (Windows). The desktop e-mail applications (Windows Live Mail and Mozilla Thunderbird) were tested to use various protocols: SMTP-PLAIN (port 587), SMTP-TLS (port 465), POP3-PLAIN (port 110), POP3-TLS (port 995), IMAP-STARTTLS (port 143), and IMAP-TLS (port 993). We also tested Skype between Windows and Android OS: video sessions, voice conversations, and file transfers.

7. Management traffic. DNS, ICMP, NETBIOS, NTP, RDP.

8. Games. Based on the most played online games in USA according to DFC Intelligence [41], we selected:
   - League of Legends (Windows) – with all launchers.
   - World of Warcraft (Windows) – including all launchers.
   - Pando Media Booster (Windows) – a process added by League of Legends to seed the game installer to other users, which offloads the servers, because the download is performed in the P2P mode. It generates enormous accounts of traffic and fills the connection.
   - Steam – delivers a range of games straight to a computer's desktop. Includes automatic updates, lists of games and prices, posters, plus access to a large number of games. We included Steam on the list as it is a platform for numerous games.
   - America's Army – a popular game from Steam.

9. Others. This category includes:
   - Music applications: Spotify, iTunes (Windows).
   - P2P Internet TVs: PPLive, Sopcast (Windows).

3.2. *Testing of the DPI tools*

The process of testing different DPI tools is complex and, therefore, we split it into several parts: labeling of the data, the classification process, and analysis of the classification logs. Some of the steps can be different for some DPIs than for the others – in these cases the differences are explicitly highlighted. We evaluate only one performance parameter: the classification accuracy. We acknowledge that other performance parameters are also important, such as speed, scalability, complexity, robustness, or price of the solutions, however, their evaluation is outside the scope of this paper.

3.2.1. *Labeling of the data*

All the flows stored in the database need to be properly marked by attaching to them the labels of the applications, application protocols, web services, types of the content, or Internet domains. One flow can be associated with multiple labels. Flows, which are not labeled, are not taken into consideration while extracting them to PCAP files.

We classify the flows at different levels. We start the labeling process by identifying the web flows and assigning them to the selected web services. Every web service is identified by a set of domains. The domains were chosen based on the number of their occurrences in the collected HTTP flows. The HTTP flows are marked with a web service label only if they contain the traffic from the matching domains. In case the flow contains traffic from domains belonging to multiple services (or to domains, which are not assigned to the selected services), the flow is left as unlabeled. The HTTP flows are also marked with the labels of the type of the transmitted content (e.g., *video/x-flv*), if they transmit audio or video. Those flows that are not HTTP belonging to the web services (e.g., SSL) are labeled using an heuristic method as follows. To be recognized as a non-HTTP web flow, the application name associated with the flow should be the name of a web browser (e.g., *chrome*), a name of a web browser plugin (e.g., *plugin-container*, *flashgcplay*), or the name should be missing. Then, we look at the HTTP flows, which were originated from 2 minutes before to 2 minutes after the non-HTTP flow. If all the corresponding (i.e., originated from the same local machine and reaching the same remote host) HTTP flows have a web service label assigned, and the service label is the same for all of the flows, the non-HTTP flow is classified with the same web service label.

Afterwards, we identify the application protocols. The application protocol label is applied only to those flows for which we are sure that transmit the specific application protocol. Finally, we identify the applications. We consider applications and protocols individually because, for example, a web-browser may use many different protocols besides HTTP, or a BitTorrent client can connect to websites to download files using HTTP or SSL, etc.

3.2.2. The classification process

The packets were extracted into PCAP files in 3 different modes: the normal one, with truncated packets (i.e., Ethernet frames were overwritten by 0s after the 70th byte), and with truncated flows (we extracted only 10 first packets for each flow). We designed a tool, called *dpi_benchmark*, which is able to read the PCAP files and provide the packets one-by-one to PACE, OpenDPI, L7-filter, nDPI, and Libprotoident. All the flows are started and terminated based on the information from the INFO files, which contain the timestamps. After the last packet of the flow is sent to the classifier, the tool obtains the label associated with that flow. The labels are written to the log files together with the flow identifier, which makes us later able to relate the classification results to the original flows in the database.

We used the default configurations of all classifiers except for *L7-filter*, which was evaluated in two different configurations. The first version (*L7-filter-all*) had all the patterns activated , but the patterns marked as *overmatching* by their authors have a low priority. For the second version (*L7-filter-com*) we adapted the methodology proposed in [42], which does not activate the patterns declared as *overmatching* and the default pattern priorities were modified.

Classification by *NBAR* required to build a complete working environment. We did not have any Cisco device that could be used for the experiment. Therefore, we used GNS3 – a graphical framework, which uses Dynamips to emulate our Cisco hardware. We emulated 7200 platform, since this is the only platform supported by GNS3 that can run the newest version of Cisco IOS (version 15), which contains Flexible NetFlow. Previous versions of Cisco IOS contain only traditional NetFlow, which does not support NBAR reporting on the per flow basis. We connected the virtual router to a real computer by using a virtual interface. The router was configured to use Flexible NetFlow with NBAR on the created interface.

Every flow recognized by Flexible NetFlow was tagged by the application name obtained from NBAR. On the computer, we used *tcpreplay* to replay the PCAP files to the router with the maximal speed that did not cause packet loss. At the same time, we used *nfacctd*, which is part of PMACCT tools [43], to capture the Flexible NetFlow records sent by the router to the computer.

The data stored in the classification logs was processed and imported back to the database. We matched the log records to the proper flows in the database using the flow identifier contained by each flow record. NBAR relies on Flexible NetFlow, which treats the flows in a unidirectional way. It means that we needed to assess the type of the bi-directional flow based on 2 unidirectional flows (inbound and outbound). In most of the cases the label from both unidirectional flows were the same. In a few cases there was only an inbound or an outbound flow, since there were no packets going in the opposite direction. In case, both unidirectional flows existed and the label of each of them was different, the bidirectional flow got the label from the unidirectional flow that accounted for more bytes.

3.3. Analysis of the results

The method for analysis of the results depend on the level of classification on which the flows were labeled:

- *Application protocol* level, such as DNS, HTTP, or POP3: To consider the classification as correct, the label reported by the classifier must be an application protocol (e.g., DNS, HTTP), but not at a different level (e.g., FLASH, YOUTUBE). This is useful to test if the tool can recognize the specific application protocol. If the result is given at a different level, the flow is considered as *unclassified*. However, the same flow will be classified as *correct* during other tests at different levels, when we for example look for a web service called *YouTube*. Those flows with labels belonging to different application protocols, and services and applications that do not belong to this application protocol are considered as *wrong*.

- *Web service* level, such as Yahoo or YouTube: the classification is considered to be *correct* only if the name of the web service is given. If it is given at a different level, such as HTTP or FLASH, the flow is considered as *unclassified*.

- *Application* level (when the application uses its proprietary application-level protocols). For example, uTorrent and Skype applications can use multiple protocols, including their proprietary protocols called respectively *Skype* and *BitTorrent*, and other protocols, such as *HTTP* or *SSL*. For example, *HTTP* and *SSL* can be used to connect to the web server to download the user's data or advertisements. Therefore, in the case of Skype, flows labeled by DPI tools as *Skype*, *HTTP*, *SSL* are all marked as *correct*.

- *Application* level (when the application does not use its proprietary application-level protocols, but directly uses HTTP, SSL, etc.) It concerns for example Spotify. Then, only the flows marked as *Spotify* are considered to be labeled correctly, as no specific application-level protocol exists for this application, so we expect the application name itself to be identified.

Table 2: Application protocols in the dataset

App. protocol	Flows	Megabytes
DNS	18251	7.66
HTTP	43127	7325.44
ICMP	205	2.34
IMAP-STARTTLS	35	36.56
IMAP-TLS	103	410.23
NETBIOS Name Service	10199	11.13
NETBIOS Session Service	11	0.01
SAMBA Session Service	42808	450.39
NTP	42227	6.12
POP3-PLAIN	26	189.25
POP3-TLS	101	147.68
RTMP	378	2353.67
SMTP-PLAIN	67	62.27
SMTP-TLS	52	3.37
SOCKSv5	1927	898.31
SSH	38961	844.87
Webdav	57	59.91

Table 3: Applications in the dataset

Application	Flows	Megabytes
4Shared	144	13.39
America's Army	350	61.15
BitTorrent clients (encrypted)	96399	3313.98
BitTorrent clients (non-encrypted)	261527	6779.95
Dropbox	93	128.66
eDonkey clients (obfuscated)	12835	8178.74
eDonkey clients (non-obfuscated)	13852	8480.48
Freenet	135	538.28
FTP clients (active)	126	341.17
FTP clients (passive)	122	270.46
iTunes	235	75.4
League of Legends	23	124.14
Pando Media Booster	13453	13.3
PPLive	1510	83.86
PPStream	1141	390.4
RDP clients	153837	13257.65
Skype (all)	2177	102.99
Skype (audio)	7	4.85
Skype (file transfer)	6	25.74
Skype (video)	7	41.16
Sopcast	424	109.34
Spotify	178	195.15
Steam	1205	255.84
TOR	185	47.14
World of Warcraft	22	1.98

Thanks to this multilevel testing approach, we obtained the knowledge of which classifier is able to provide results on each particular level of classification. This knowledge would allow, for example, the end user to adjust the choice of the DPI technique according to the desired level of classification.

4. Dataset

Our basic dataset (without truncated packets or flows) contains 767 690 flows, which account for 53.31 GB of pure packet data. The application name was present for 759 720 flows (98.96 % of all the flows), which account for 51.93 GB (97.41 %) of the data volume. The remaining flows are unlabeled due to their short lifetime (usually below 1 s), which made VBS incapable of reliably establishing the corresponding sockets. The application protocols together with the number of flows and the data volume are shown in Table 2, while the applications are shown in Table 3 and the web services ae shown in Table 4. The data volume is presented here only for an overview – the rest of the paper uses only the number of flows as the reference value.

We are going to publish our basic labeled dataset with full packet payloads on our website [44]. Therefore, it can be used by the research community as a reference benchmark for the validation of network traffic classifiers.

5. Results

This section provides an overview of the classification results of the different types of traffic by each of the classifiers. The evaluation was performed on 3 datasets: a normal set, a set with truncated packets, and a set with truncated flows. This section presents a description of the most interesting results, but a discussion is later presented in Section 6 with the conclusions that can be drawn from them. The following subsections present the results for the normal dataset, while the results for the sets with truncated packets or flows are discussed separately. All the accuracy results are given in terms of flows. The methodology used to compute the accuracy is shown in Section 3.3. The interested reader may find the complete confusion matrix in the technical report [39].

5.1. Application protocols

The evaluation of the classification of application protocols is shown in Table 5. The columns present the percentage of correctly and wrongly classified as well as unclassified flows belonging to various application protocols by each classifier.

An important performance parameter of DPI-based techniques is the completeness of their results (i.e., number of applications they can classify). This section evaluates 17 different application protocols. As shown in Table 5, none of the techniques is able to classify all of them. Among the different techniques studied, nDPI and Libprotoident are the most complete, classifying 15 out of 17. At the far end, L7-filter only classifies 9 of 17.

Another important aspect of DPI techniques is their ratio of false positives (i.e., incorrect classifications). Usually techniques leave the non-recognized flows as unclassified, trying to decrease the number of false positives. Even though, both versions of L7-filter are characterized for producing a high number of incorrect classifications (e.g., L7-filter-all classifies 85.79% of HTTP traffic as Finger). Regarding the specific classifications, most of traditional application protocols (i.e., DNS, HTTP, IMAP-STARTTLS, POP3-PLAIN, SMTP-PLAIN and SSH) are generally well detected by all the techniques (e.g., accuracy between 70.92% and 100%). Unexpectedly, Libprotoident is the only classifier able to identify all the tested encrypted protocols. Regardless of the classifier, the undetected encrypted traffic is usually identified as regular SSL. An interesting case is presented by the classification of RTMP. Only nDPI and Libprotoident are able to properly classify it. PACE

Table 4: Web services in the dataset

Web service	Flows	Megabytes
4Shared	98	68.42
Amazon	602	51.02
Apple	477	90.22
Ask	171	1.86
Bing	456	36.84
Blogspot	235	10.53
CNN	247	3.66
Craigslist	179	4.09
Cyworld	332	13.06
Doubleclick	1989	11.24
eBay	281	8.31
Facebook	6953	747.35
Go.com	335	25.83
Google	6541	532.54
Instagram	9	0.22
Justin.tv	2326	126.33
LinkedIn	62	2.14
Mediafire	472	27.99
MSN	928	23.22
Myspace	2	2.54
Pinterest	189	3.64
Putlocker	103	71.92
QQ.com	753	10.46
Taobao	387	24.29
The Huffington Post	71	21.19
Tumblr	403	52.56
Twitter	1138	13.67
Vimeo	131	204.45
Vk.com	343	9.59
Wikipedia	6092	521.95
Windows Live	26	0.16
Wordpress	169	33.31
Yahoo	17373	937.07
YouTube	2534	1891.79

and OpenDPI classify this traffic as Flash. Although both traffics are usually related, the classification as Flash cannot be considered as being correct, as Flash is only a content container. Flash content (audio, video or any other binary file) can be transported using various applications protocols (e.g., HTTP, RTMP) or even different transport protocols (both TCP and UDP).

5.2. Applications

The second level of classification studies the *application* that uses its proprietary application-level protocols (e.g., BitTorrent, Skype). The evaluation of the classification of various *applications* is shown in Table 6. The columns present the percentage of correctly and wrongly classified as well as unclassified flows belonging to various *applications* by each of the classifier.

At *application* level the most complete technique is PACE, classifying 20 out of the 22 evaluted *applications*, followed by nDPI (17) and Libprotoident (14). Again, L7-filter is among the worst techniques (8), but it is overcomed by NBAR that classifies only 4 *applications*.

Regarding the false positive ratio, the lowest percentage of misclassified flows were obtained from PACE and OpenDPI. Contrary, the highest ratio of misclassified flows is again obtained from the classifications by both versions of L7-filter (e.g., 97% of America's Army traffic is classified as Skype and RTP).

As shown in Table 6, the classification at *application* level presents more problems than at *application protocol* level. It is particularly striking that almost no classifier is able to completely classify all the flows (i.e., 100%) from a specific *application*. This can be derived from the fact that usually *applications* use different internal operations that produce different traffic. Therefore, techniques need a specific pattern for every type of operation. For instance, the accuracy with Skype is always lower than 100% because none of the techniques is able to classify neither Skype filetransfers nor videos. Among the different studied techniques, PACE is the most accurate followed by nDPI and Libprotoident. Surprisingly, PACE presents severe problems with a traditional *application* as FTP, almost non classifying all its traffic. Another interesting observations extracted from the results are shown below:

- L7-filter, the most unreliable at this level, usually misclassifies the flows as Skype and Finger. However, around 1/3 of the Skype flows are misclassified by it as RTP, Finger, eDonkey, or NTP.
- The authors of traffic classifiers focus on popular applications, which either generate heavy data volume, or are critical regarding QoS requirements. Non-encrypted BitTorrent flows and Skype flows are the only groups of applications that are generally well detected by all the classifiers.
- America's Army game is not classified by any tool. The few correct classifications obtained by nDPI are due to the recognition of some flows originated by the TeamSpeak client integrated with the game.

5.3. Web services

The last level studied evaluates many different *web services*. Because of clarity and understanding, we do not present the results as a table but as a summary of the important outcomes.

The results with *web services* follow the outcomes obtained at previous levels. PACE is the most complete and accurate technique. The bad results of the rest of techniques are mainly due to a not enough specific classification (e.g., Facebook traffic classified as HTTP).

PACE recognizes 4Shared (84.69 %), Amazon (58.97 %), Apple (0.84 %), Blogspot (3.83 %), eBay (67.97 %), Facebook (80.79 %), Google (10.79 %), Instagram (88.89 %), Linkedin (77.42 %), Mediafire (30.30 %), Myspace (100 %), QQ.com (32.14 %), Twitter (71.18 %), Windows Live (96.15 %), Yahoo (54.70 %), and YouTube (81.97 %). PACE does not have problems with recognizing SSL flows belonging to these services, which means that PACE must use other techniques than just looking directly into the packets to associate the flows with the particular services, probably by analyzing the server certificates.

The commercial tool clearly improves upon its open-source version OpenDPI that recognizes only Direct Download websites: 4Shared (83.67 %) and MediaFire (30.30 %).

L7-filter recognizes only Apple (0.42 %). Furthermore, L7-filter (especially L7-filter-all) is characterized by a very high

Table 5: Evaluation of application protocols (% of flows)

Protocol	Classifier	% cor.	% wr.	% unc.	Protocol	Classifier	% cor.	% wr.	% unc.	Protocol	Classifier	% cor.	% wr.	% unc.
DNS	PACE	99.95	0.00	0.05	NETBIOS Session Service	PACE	100.00	0.00	0.00	SMTP PLAIN	PACE	100.00	0.00	0.00
	OpenDPI	99.99	0.00	0.01		OpenDPI	100.00	0.00	0.00		OpenDPI	100.00	0.00	0.00
	L7-filter-all	99.62	0.05	0.33		L7-filter-all	9.09	0.00	90.91		L7-filter-all	100.00	0.00	0.00
	L7-filter-com	99.62	0.02	0.36		L7-filter-com	9.09	0.00	90.91		L7-filter-com	100.00	0.00	0.00
	nDPI	100.00	0.00	0.00		nDPI	100.00	0.00	0.00		nDPI	100.00	0.00	0.00
	Libprotoident	99.96	0.00	0.04		Libprotoident	100.00	0.00	0.00		Libprotoident	100.00	0.00	0.00
	NBAR	99.99	0.00	0.01		NBAR	100.00	0.00	0.00		NBAR	100.00	0.00	0.00
HTTP	PACE	70.92	0.63	28.45	SAMBA Session Service	PACE	100.00	0.00	0.00	SMTP TLS	PACE	0.00	0.00	100.00
	OpenDPI	95.68	0.59	3.73		OpenDPI	100.00	0.00	0.00		OpenDPI	0.00	0.00	100.00
	L7-filter-all	3.58	96.04	0.38		L7-filter-all	100.00	0.00	0.00		L7-filter-all	0.00	0.00	100.00
	L7-filter-com	35.25	10.28	54.47		L7-filter-com	100.00	0.00	0.00		L7-filter-com	0.00	0.00	100.00
	nDPI	17.25	0.83	81.92		nDPI	100.00	0.00	0.00		nDPI	3.85	0.00	96.15
	Libprotoident	99.80	0.07	0.13		Libprotoident	100.00	0.00	0.00		Libprotoident	100.00	0.00	0.00
	NBAR	99.04	0.17	0.79		NBAR	0.00	0.00	100.00		NBAR	0.00	0.00	100.00
ICMP	PACE	100.00	0.00	0.00	NTP	PACE	100.00	0.00	0.00	SOCKSv5	PACE	78.26	0.00	21.74
	OpenDPI	100.00	0.00	0.00		OpenDPI	100.00	0.00	0.00		OpenDPI	0.00	0.00	100.00
	L7-filter-all	0.00	0.00	100.00		L7-filter-all	99.86	0.14	0.00		L7-filter-all	0.00	100.00	0.00
	L7-filter-com	0.00	0.00	100.00		L7-filter-com	99.86	0.13	0.01		L7-filter-com	0.00	100.00	0.00
	nDPI	100.00	0.00	0.00		nDPI	100.00	0.00	0.00		nDPI	92.99	0.00	7.01
	Libprotoident	100.00	0.00	0.00		Libprotoident	100.00	0.00	0.00		Libprotoident	100.00	0.00	0.00
	NBAR	100.00	0.00	0.00		NBAR	0.00	0.00	100.00		NBAR	0.00	0.00	100.00
IMAP STARTTLS	PACE	100.00	0.00	0.00	POP3 PLAIN	PACE	100.00	0.00	0.00	SSH	PACE	93.98	0.51	5.51
	OpenDPI	100.00	0.00	0.00		OpenDPI	100.00	0.00	0.00		OpenDPI	93.98	0.12	5.90
	L7-filter-all	100.00	0.00	0.00		L7-filter-all	100.00	0.00	0.00		L7-filter-all	94.19	0.36	5.45
	L7-filter-com	100.00	0.00	0.00		L7-filter-com	100.00	0.00	0.00		L7-filter-com	94.19	0.12	5.69
	nDPI	100.00	0.00	0.00		nDPI	100.00	0.00	0.00		nDPI	93.98	0.80	5.22
	Libprotoident	100.00	0.00	0.00		Libprotoident	100.00	0.00	0.00		Libprotoident	94.19	0.36	5.45
	NBAR	100.00	0.00	0.00		NBAR	100.00	0.00	0.00		NBAR	93.71	0.64	5.65
IMAP TLS	PACE	0.00	0.00	100.00	POP3 TLS	PACE	0.00	0.00	100.00	Webdav	PACE	3.51	0.00	96.49
	OpenDPI	0.00	0.00	100.00		OpenDPI	0.00	0.00	100.00		OpenDPI	0.00	0.00	100.00
	L7-filter-all	0.00	0.00	100.00		L7-filter-all	0.00	5.93	94.06		L7-filter-all	0.00	7.02	92.98
	L7-filter-com	0.00	0.00	100.00		L7-filter-com	0.00	0.99	99.01		L7-filter-com	0.00	7.02	92.98
	nDPI	0.00	0.00	100.00		nDPI	88.12	0.00	11.88		nDPI	0.00	0.00	100.00
	Libprotoident	100.00	0.00	0.00		Libprotoident	100.00	0.00	0.00		Libprotoident	0.00	0.00	100.00
	NBAR	100.00	0.00	0.00		NBAR	100.00	0.00	0.00		NBAR	0.00	0.00	100.00
NETBIOS Name Service	PACE	99.96	0.00	0.04	RTMP	PACE	0.00	0.00	100.00					
	OpenDPI	98.51	0.00	1.49		OpenDPI	0.00	0.00	100.00					
	L7-filter-all	0.00	5.63	94.37		L7-filter-all	0.00	23.54	76.46					
	L7-filter-com	0.00	9.15	90.85		L7-filter-com	0.00	23.54	76.46					
	nDPI	99.97	0.00	0.03		nDPI	70.90	15.87	13.23					
	Libprotoident	0.04	4.94	95.02		Libprotoident	86.51	0.26	13.23					
	NBAR	100.00	0.00	0.00		NBAR	0.00	0.26	99.74					

number of misclassified flows belonging to web services (usually 80–99 %). The flows are recognized in a vast majority as Finger and Skype.

nDPI recognizes Amazon (83.89 %), Apple (74.63 %), Blogspot (4.68 %), Doubleclick (85.92 %), eBay (72.24 %), Facebook (80.14 %), Google (82.39 %), Yahoo (83.16 %), Wikipedia (68.96 %), and YouTube (82.16 %) being the second best technique at this level.

Unlike at previous levels, Libprotoident recognizes only the Yahoo (2.36 %) *web service*. This result is understandable given that Libprotoident only uses the first 4 bytes of packet payload to classify a flow, making considerably more difficult a specific classification as web service.

The worst technique at this level is NBAR that does not recognize any web services.

5.4. *Impact of packet truncation*

An important characteristic of each DPI tool is the amount of information needed from each packet to identify the traffic. That significantly influences the classification speed and the resources needed. Furthermore, many traffic traces are published with payload truncated up to a certain number of bytes per packet for privacy reasons. As mentioned before, Libprotoident is the only tool, which is advertised to use the particular extent of the examined packets, namely first 4 bytes. Therefore, in order to discover the internal properties of each tool, we decided to test the impact of packet truncation. This subsection presents the differences between the classification results for the normal dataset and the dataset with truncated Ethernet frames to the first 70 B.

Truncation of packets has a considerable impact on the classification of most *application protocols* by all tools except Libprotoident and NBAR, which tend to maintain their normal accuracy. This suggests that NBAR can be somehow implemented as Libprotoident to classify *application protocols* while the rest of techniques base their classification on the complete flow. L7-filter is not able to detect DNS traffic on this set, while all the other classifiers present the accuracy of over 99 %. Unexpectedly, NBAR cannot detect NTP on the normal set, while it detects it 100 % correctly on the set with truncated packets. We can not present a verificable reason of this result given that NBAR is not an open-sourcer tool.

At *application* level, only Libprotoident is able to keep its normal accuracy whereas the rest of techniques considerably decreases their accuracies.

Regarding the *web services* level, only nDPI is able to recognize some *web services* in this set. Exceptionally, the detection rate is almost the same as for the normal set. Other classifiers tend to leave such traffic as *unknown*.

5.5. *Impact of flow truncation*

Another major concern is how many packets are needed in order to classify each flow. That depends on the classification tool as well as on the application or protocol, which we want to identify. However, the documentation of the traffic classifiers do not cover these issues, although they are very important for conserving disk space while publishing data traces used to test the tools. Therefore, we decided to study the impact of flow truncation. This subsection presents the differences between the classification results for the normal dataset and the dataset with truncated flows to the first 10 packets.

Truncation of flows does not have any noticeable impact on the classification of *application protocols*. This result suggests

Table 6: Evaluation of applications (% of flows)

Application	Classifier	% cor.	% wr.	% unc.
4Shared	PACE	27.08	0.00	72.92
	OpenDPI	27.08	0.00	72.92
	L7-filter-all	0.00	1.39	98.61
	L7-filter-com	0.00	0.00	100.00
	nDPI	0.00	0.00	100.00
	Libprotoident	0.00	0.00	100.00
	NBAR	0.00	0.00	100.00
America's Army	PACE	0.00	0.00	100.00
	OpenDPI	0.00	0.00	100.00
	L7-filter-all	0.00	97.71	2.29
	L7-filter-com	0.00	97.43	2.57
	nDPI	4.00	0.00	96.00
	Libprotoident	0.00	89.14	10.86
	NBAR	0.00	72.00	28.00
BitTorrent clients (encrypted)	PACE	78.68	0.05	21.27
	OpenDPI	0.27	0.00	99.73
	L7-filter-all	40.54	10.17	49.29
	L7-filter-com	40.62	7.30	52.08
	nDPI	54.41	0.18	45.41
	Libprotoident	60.31	0.02	39.67
	NBAR	1.29	0.63	98.08
BitTorrent clients (non-encrypted)	PACE	99.87	0.00	0.13
	OpenDPI	80.61	0.00	19.39
	L7-filter-all	94.56	0.49	4.95
	L7-filter-com	94.60	0.42	4.98
	nDPI	99.41	0.02	0.57
	Libprotoident	99.30	0.00	0.70
	NBAR	77.84	0.36	21.80
Dropbox	PACE	94.62	0.00	5.38
	OpenDPI	0.00	0.00	100.00
	L7-filter-all	0.00	0.00	100.00
	L7-filter-com	0.00	0.00	100.00
	nDPI	98.92	0.00	1.08
	Libprotoident	0.00	0.00	100.00
	NBAR	0.00	0.00	100.00
eDonkey clients (obfuscated)	PACE	36.06	7.26	56.68
	OpenDPI	0.00	0.00	100.00
	L7-filter-all	11.64	16.59	71.77
	L7-filter-com	11.64	11.09	77.27
	nDPI	11.04	2.67	86.29
	Libprotoident	11.47	0.00	88.53
	NBAR	0.00	15.93	84.07
eDonkey clients (non-obfuscated)	PACE	16.50	3.74	79.76
	OpenDPI	3.98	0.30	95.72
	L7-filter-all	17.97	16.32	65.71
	L7-filter-com	17.99	10.79	71.22
	nDPI	15.57	2.28	82.23
	Libprotoident	17.86	0.31	81.83
	NBAR	2.05	11.19	86.76
Freenet	PACE	79.26	0.00	20.74
	OpenDPI	0.00	0.00	100.00
	L7-filter-all	0.00	20.00	80.00
	L7-filter-com	0.00	14.07	85.93
	nDPI	0.00	3.70	96.30
	Libprotoident	0.00	0.00	100.00
	NBAR	0.00	15.56	84.44
FTP clients (active)	PACE	5.56	0.00	94.44
	OpenDPI	97.62	0.00	2.38
	L7-filter-all	5.56	92.06	2.38
	L7-filter-com	5.56	90.47	3.97
	nDPI	98.41	0.00	1.59
	Libprotoident	100.00	0.00	0.00
	NBAR	50.79	0.00	49.21
FTP clients (passive)	PACE	4.92	0.00	95.08
	OpenDPI	67.21	0.00	32.79
	L7-filter-all	4.92	76.23	23.77
	L7-filter-com	4.92	73.77	26.23
	nDPI	72.95	0.00	27.05
	Libprotoident	73.77	22.95	32.80
	NBAR	50.00	0.00	50.00
iTunes	PACE	77.45	0.00	22.55
	OpenDPI	0.00	0.00	100.00
	L7-filter-all	63.83	6.81	29.36
	L7-filter-com	63.83	0.00	36.17
	nDPI	13.19	0.00	86.81
	Libprotoident	0.00	0.00	100.00
	NBAR	0.00	0.00	100.00
League of Legends	PACE	0.00	13.04	86.96
	OpenDPI	0.00	0.00	100.00
	L7-filter-all	0.00	69.57	30.43
	L7-filter-com	0.00	4.35	95.65
	nDPI	0.00	13.04	86.96
	Libprotoident	0.00	4.35	95.65
	NBAR	0.00	0.00	100.00
Pando Media Booster	PACE	99.45	0.39	0.16
	OpenDPI	99.23	0.54	0.23
	L7-filter-all	0.00	0.74	99.26
	L7-filter-com	0.00	0.55	99.45
	nDPI	99.26	0.63	0.11
	Libprotoident	99.26	0.41	0.33
	NBAR	0.00	0.36	99.64
PPLive	PACE	88.21	0.00	11.79
	OpenDPI	0.07	0.13	99.80
	L7-filter-all	0.00	56.03	43.97
	L7-filter-com	0.00	17.15	82.85
	nDPI	43.91	1.05	55.04
	Libprotoident	43.91	1.05	55.04
	NBAR	0.00	0.40	99.60
PPStream	PACE	79.32	0.00	20.68
	OpenDPI	0.79	0.00	99.21
	L7-filter-all	0.00	38.39	61.61
	L7-filter-com	0.00	15.07	84.93
	nDPI	0.53	0.26	99.21
	Libprotoident	0.96	0.00	99.04
	NBAR	0.00	5.26	94.74
RDP clients	PACE	99.69	0.00	0.31
	OpenDPI	99.70	0.00	0.30
	L7-filter-all	0.00	92.25	7.75
	L7-filter-com	0.00	92.25	7.75
	nDPI	99.69	0.02	0.29
	Libprotoident	99.66	0.01	0.33
	NBAR	0.00	0.67	99.33
Skype (all)	PACE	83.51	5.05	11.44
	OpenDPI	38.49	0.32	61.19
	L7-filter-all	59.21	31.70	9.09
	L7-filter-com	62.52	24.67	12.81
	nDPI	99.82	0.00	0.18
	Libprotoident	88.75	0.00	11.25
	NBAR	70.37	3.40	26.23
Skype (audio)	PACE	100.00	0.00	0.00
	OpenDPI	0.00	0.00	100.00
	L7-filter-all	85.71	14.29	0.00
	L7-filter-com	100.00	0.00	0.00
	nDPI	0.00	0.00	100.00
	Libprotoident	0.00	0.00	100.00
	NBAR	0.00	0.00	100.00
Skype (file transfer)	PACE	0.00	100.00	0.00
	OpenDPI	0.00	0.00	100.00
	L7-filter-all	0.00	100.00	0.00
	L7-filter-com	0.00	100.00	0.00
	nDPI	0.00	0.00	100.00
	Libprotoident	0.00	0.00	100.00
	NBAR	0.00	0.00	100.00
Skype (video)	PACE	0.00	100.00	0.00
	OpenDPI	0.00	0.00	100.00
	L7-filter-all	0.00	100.00	0.00
	L7-filter-com	0.00	100.00	0.00
	nDPI	0.00	0.00	100.00
	Libprotoident	0.00	0.00	100.00
	NBAR	0.00	0.00	100.00
Sopcast	PACE	66.27	3.07	30.66
	OpenDPI	66.27	2.59	31.14
	L7-filter-all	0.00	99.06	0.94
	L7-filter-com	0.00	74.76	25.24
	nDPI	63.68	1.18	35.14
	Libprotoident	46.70	0.24	53.06
	NBAR	0.00	0.00	100.00
Spotify	PACE	37.64	2.25	60.11
	OpenDPI	0.00	0.00	100.00
	L7-filter-all	0.00	43.26	56.74
	L7-filter-com	0.00	10.11	89.89
	nDPI	0.56	3.93	95.51
	Libprotoident	0.56	0.00	99.44
	NBAR	0.00	0.56	99.44
Steam	PACE	55.19	0.75	44.06
	OpenDPI	0.33	0.00	99.67
	L7-filter-all	0.00	65.89	34.11
	L7-filter-com	0.00	4.73	95.27
	nDPI	76.02	0.42	23.56
	Libprotoident	75.85	0.00	24.15
	NBAR	0.00	0.58	99.42
TOR	PACE	85.95	0.00	14.05
	OpenDPI	0.00	0.00	100.00
	L7-filter-all	0.00	0.00	100.00
	L7-filter-com	0.00	0.00	100.00
	nDPI	33.51	0.00	66.49
	Libprotoident	33.51	0.00	66.49
	NBAR	0.00	2.16	97.84
World of Warcraft	PACE	27.27	0.00	72.73
	OpenDPI	0.00	0.00	100.00
	L7-filter-all	0.00	86.36	13.64
	L7-filter-com	0.00	22.73	77.27
	nDPI	13.64	13.64	72.72
	Libprotoident	13.64	0.00	86.36
	NBAR	0.00	0.00	100.00

that the classification of *application protocols* relies on patterns or signatures extracted from the first packets of the flows.

Similar behavior is obatained at *application* level. However, in this case the impact on the classification of *applications* is noticeable – the detection rate decreases. The only exception is Libprotoident, which provides the same results as for the normal dataset. Therefore, this insinuate that the classification of some *applications* probably rely on techniques based on statistics (e.g., Machine Learning). FTP in the active mode is a very interesting case, as Libprotoident maintains its 100 % accuracy, while the accuracy of the other classifiers drops to 5.56 %. An strange case is presented with Plain eDonkey traffic, as the best classification accuracy (45.28 %) we obtained by using PACE on the set with truncated flows, while the accuracy on the normal set was only 16.50 %.

The percentage of correctly classified *web services* is usually the same or nearly the same as for the normal set.

6. Discussion

This section extracts the outcomes from the results obtained during the performance comparison. Also, we discuss the limitations of our study.

As shown in the previous section, *PACE* is the best classifier for most of the studied classification groups. This high ranking is due to the ability of providing the results on various levels, as for example *HTTP:generic:facebook*. Other classifiers do not offer this ability at all and only one chosen level is given, so, for example, they do not offer the possibility to account the HTTP or SSL traffic, while they recognize the web service of the transported content. However, PACE also is not totally consistent in that matter. Facebook videos (which we observed as transported by HTTP) were detected as, for example, *FLASH:no_subprotocols:facebook*, while the live video streams from Justin.tv using RTMP, were classified as *FLASH:no_subprotocols:not_detected*. So, we do not have the knowledge from the results obtained from the classifier which application protocol was used (HTTP, RTMP, or other), because the content container level is returned instead. Ideally, the DPI techniques should provide results on all the possible levels, as *HTTP:FLASH:VIDEO:YOUTUBE*, so that kind of consistent accounting could be performed. However, PACE is a commercial tool not accessible for all the research community. Among the available open-source tools, *nDPI* and *Libprotoident* reveal as the most reliable solutions. Surprisingly for us, Libprotoident achieves very good results without giving a noticeable number of false classifications by using the first four bytes of payload for each direction. On the other hand, *L7-filter* and *NBAR* perform poorly in classifying the traffic from our dataset.

We did not observe large differences between the classifications performed on the normal dataset and the set with truncated flows to maximum 10 packets. The set with truncated packets is usually much worse classified than the other sets by all tools except Libprotoident, which maintains the same accuracy. We found that our modified version of L7-filter-com provides overall better results than the default L7-filter-all by increased number of correct classifications and greatly reduced rate of misclassifications (especially, regarding the web services).

Nonetheless, the previous conclusions are obviously tied to the particular application protocols, applications, and web services included in our dataset. Although we tried our best to emulate the real behavior of the users, many applications, behaviors and configurations are not represented on it. Because of that it has some limitations that we discuss next:

- In our study, we evaluated 17 well-known application protocols, 19 applications (including 4 in various configurations), and 34 web services. The results obtained from the different classifiers are directly related to those groups. Thus, the introduction of different groups could arise different outcomes. However, all the groups are evaluated individually and, therefore, the addition of new groups to our dataset would not change the results presented in this paper.

- The traffic generated for building the dataset, although has been manually and realistically created, is artificial. The backbone traffic would carry different behaviors of the groups that are not fully represented in our dataset (e.g., P2P clients running on port 80). The performance of the tools studied might not be directly extrapolated from the current results. However, the artificially created traffic allowed us to publish the dataset with full packet payloads.

- The poor performance of NBAR and L7-filter might be affected by the characteristics of our dataset. Thus, the reliability of previous works based on them is not called into question. Different configurations [42, 45, 10] and different or older classification groups would probably produce different results.

- The classification levels have considerable impact on the results. For instance, classifying Facebook, Google or Twitter is currently not possible by Libprotoident, however it is possible by nDPI and PACE.

- The amount of data available would also have impacted on the performance. The study presented in this paper is performed with full payload packets. However, in other works the traces are usually collected with a few bytes of data [46, 13, 47] (e.g., 96 bytes) in order to avoid packet loss, disk space, and privacy issues. For this scenario, it seems that Libprotoident is a more suitable solution, giving it only uses the first 4 bytes of every packet.

- The nature, distribution, and heterogeneity of the traffic would also impact the performance. The amount of classes detected by PACE is considerably bigger than detected by the rest of the classifiers, which makes PACE more suitable for heterogeneous scenarios. Also, PACE and nDPI are able to classify traffic in asymmetric scenarios.

7. Related work

This section reviews the literature related to the comparison of DPI tools. The OpenDPI tool amounts for most of the publications [6, 48, 10, 7, 49]. According to [6], the test performed by the European Networking Tester Center (EANTC) in 2009 resulted in 99 % of detection and accuracy for popular P2P protocols by OpenDPI. The big amount of flows marked as *unknown* by OpenDPI was confirmed in [48], where the authors made an effort to calculate various parameters for traffic originated from different applications: number of flows, data volume, flow sizes, number of concurrent flows, and inter-arrival times. The study was based on 3.297 TB of data collected during 14 days from an access network with connected around 600 households. 80.1 % of the flows, amounting for 64 % of the traffic volume, were marked as *unknown* by OpenDPI.

In [6], the authors study the impact of per-packet payload sampling (i.e., packet truncation) and per-flow packet sampling (i.e., collect only the first packets of a flow) on the performance of OpenDPI. The results show that OpenDPI is able to keep the accuracy higher than 90-99% with only the first 4-10 packets of a flow. The impact by the per-packet payload sampling is considerably higher. Their results use as ground-truth the dataset labeled by OpenDPI with no sampling. Thus, the actual classification of the dataset is unknown and no possible comparison with our work can be done.

Similar work, performed by the same authors, is described in [7]. The goal was to find out what is the suggested number of packets from each flow, which needs to be inspected by OpenDPI in order to achieve good accuracy, while maintaining a low computational cost. The focus was on Peer-to-Peer (P2P) protocols. The test was performed on a 3 GB randomly selected subset of flows from the data collected at an access link of an institution over 3 days. The authors found that inspecting only 10 packets from each flow lowered the classification abilities of P2P flows by OpenDPI by just 0.85 % comparing to the classification of full flows, while saving more than 9 % of time.

In [10], the authors tested the accuracy of L7-filter and OpenDPI, and they also built their own version of L7-filter with enhanced abilities of classification of the UDP traffic. The data used in the experiment were collected by Wireshark, while the applications were running in the background. The data were split into 27 traces, each for one application, where all the applications were supported by both L7-filter and OpenDPI. Other flows were removed from the dataset. However, they do not explain how they validate the process of the isolation of the different applications. The obtained precision was 100 % in all the cases (none of the classification tools gave a false positive), while the recall deviated from 67 % for the standard L7-filter, through 74 % for their own implementation of L7-filter, and 87 % for OpenDPI.

Fukuda compared in [47] the performance of L7-filter and OpenDPI on the backbone traffic. The dataset used is characterized as being in majority asymmetric and containing the packets truncated after 96 Bytes. The ground-truth is labeled using a port-based technique and then the three DPI-based techniques are compared. The results show that the DPI-based techniques are only able to classify 40-60% of the traffic in this scenario.

In [8], the developers of Libprotoident evaluated the accuracy of the classification of this tool and compared the results with OpenDPI, Nmap, and L7-filter. The ground-truth was established by PACE, so only the flows recognized by PACE were taken into account during the experiment. The accuracy was tested on two datasets: one taken from the Auckland university network, and one from an Internet Service Provider (ISP). On the first dataset, Libprotoident had the lowest error rate of less than 1 % (OpenDPI: 1.5 %, L7-filter: 12.3 %, Nmap: 48 %.). On the second dataset, Libprotoident achieved the error rate of 13.7 %, while OpenDPI 23.3 %, L7-filter 22 %, and Nmap 68.9 %. The authors claim that Libprotoident identified 65 % of BitTorrent traffic and nearly 100 % of HTTP, SMTP, and SSL. Same authors also compared in [9] four open-source DPI-based tools (i.e., nDPI, Tstat, Libprotoident, and L7-filter). Similarly to us, they artificially built a labeled dataset using a complicate mix of filters in an isolated host. Unlike us, their trace is not available to the community so no further comparison is possible. However, their results confirms some of the findings of our paper presenting nDPI and Libprotoident as the most accurate open-source DPI-based tools.

Another lightweight packet inspection approach was proposed in [50]. The authors developed PortLoad, which was designed to be characterized by the speed of port-based classifiers, while maintaining the accuracy of DPI tools. The authors showed that almost all the matching strings start (99.98 %) and finish (90.77 %) in the first 32 bytes of payload. Only the first packet in each direction is processed. PortLoad was compared against L7-filter and the port-based approach. The experimental evaluation showed that the processing time is cut down by more than 97 % comparing to L7-filter, while the accuracy was assessed to be 74 %.

To the best of our knowledge, there are no accessible research studies or reports about the accuracy of NBAR. However, an experiment was made to assess how big amount of network traffic is classified by NBAR and L7-filter, and how big amount of traffic is left as *unknown* [51]. The authors captured by Wireshark all the packets flowing in a local network of an IT company during 1 hour. From 27 502 observed packets, 12.56 % were reported as unknown by NBAR, and 30.44 % were reported as unknown by L7-filter.

A very comprehensive review of different methods for traffic classification was made in 2013 by Silvio Valenti et al. [52]. The authors refer to 68 different positions in the literature and cover the topic from the basis to more advanced topics, mostly dealing with Machine Learning Algorithms (MLAs).

8. Conclusions

This paper presents a reliable evaluation of the accuracy of some of the most well-known DPI-based network traffic classifiers. We compared the precision of six tools (i.e., *PACE*, *OpenDPI*, *L7-filter*, *nDPI*, *Libprotoident*, and *NBAR*), which are usually used for traffic classification. The results obtained in Section 5 and further discussed in Section 6 show that *PACE* is, for the majority of protocols, applications, and web services included in our dataset, the most reliable solution for traffic classification. Among the open-source tools, *nDPI* and *Libprotoident* present the best results. The choice between them would depend on the scenario or the level on which we would like to obtain the results. On the other hand, *NBAR* and *L7-filter* present several inaccuracies that make them not recommendable for network traffic classification in their current form.

In order to make the study trustworthy, we created a dataset using VBS [20]. This tool associates the name of the process to each flow making its labeling totally reliable. The dataset of more than 750 K flows contains traffic from popular applications. Trying to be as representative as possible, we selected the groups and applications based on Internet rankings. However, the robustness of our evaluation methodology is independent of the applications selected, as we provide the accuracy per application. Also, this dataset allows the validation of different techniques on different levels (i.e., application protocol, application, and web service). The total amount of data properly labeled is 51.93 GB. Furthermore, and more important, we released to the research community this dataset with full payload, so it can be used as a common reference for the comparison and validation of network traffic classifiers.

Although this study is complete, the continuous evolution of the network applications and the DPI-based techniques allows a periodical updated of the evaluation. For instance, this evaluation can be updated by adding new applications and web services to the dataset (e.g., Netflix) and by introducing new classification tools to the study (e.g., NBAR2 or Tstat). In this paper, we focused on the reliability of the DPI tools, however, a possible line of future work can be related to their deployment for real-time classification (i.e., scalability and computational cost).

9. Acknowledgments

This research was funded by the Spanish Ministry of Science and Innovation under contract TEC2011-27474 (NOMADS project) and by AGAUR (ref. 2014-SGR-1427). This work was also co-financed (grant no. 8–10100) by Aalborg University, the European Regional Development Fund (ERDF), and Bredbånd Nord A/S.

References

[1] NBAR2 or Next Generation NBAR – Cisco Systems, [Online]. Available: http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6558/ps6616/qa_c67-697963.html (2013).

[2] PACE | Network Analysis with Layer-7 Deep Packet Inspection, [Online]. Available: http://www.ipoque.com/en/products/pace (2013).

[3] Qosmos | Deep Packet Inspection and Metadata Engine, [Online]. Available: http://www.qosmos.com/products/deep-packet-inspection-engine/ (2013).

[4] Paloalto Networks, [Online]. Available: https://www.paloaltonetworks.com/ (2013).

[5] A. Dainotti, A. Pescapè, K. C. Claffy, Issues and future directions in traffic classification, IEEE Network 26 (1) (2012) 35–40, dOI: 10.1109/MNET.2012.6135854.

[6] J. Khalife, A. Hajjar, J. Díaz-Verdejo, Performance of OpenDPI in Identifying Sampled Network Traffic, Journal of Networks 8 (1) (2013) 71–81, dOI: 10.4304/jnw.8.1.71-81.

[7] J. Khalife, A. Hajjar, J. Díaz-Verdejo, On the Performance of OpenDPI in Identifying P2P Truncated Flows, in: AP2PS 2011, The Third International Conference on Advances in P2P Systems, IARIA, Lisbon, Portugal, 2011, pp. 79–84.

[8] S. Alcock, R. Nelson, Libprotoident: Traffic Classification Using Lightweight Packet Inspection, Tech. rep., University of Waikato, accessible: http://www.wand.net.nz/publications/lpireport (August 2012).

[9] S. Alcock, R. Nelson, Measuring the Accuracy of Open-Source Payload-Based Traffic Classifiers Using Popular Internet Applications, in: IEEE Workshop on Network Measurements (WNM), the 38th IEEE Conference on Local Computer Networks (LCN), IEEE, Sydney, Australia, 2013.

[10] C. Shen, L. Huang, On detection accuracy of L7-filter and OpenDPI, in: 2012 Third International Conference on Networking and Distributed Computing (ICNDC), IEEE, Hangzhou, China, 2012, pp. 119–123, dOI: 10.1109/ICNDC.2012.36.

[11] CAIDA Internet Data – Passive Data Sources, [Online]. Available: http://www.caida.org/data/passive/ (2012).

[12] CAIDA Internet Data – Internet Measurement Data Catalog (IMDC), [Online]. Available: http://www.caida.org/projects/trends/imdc/ (2012).

[13] MAWI Working Group Traffic Archive, [Online]. Available: http://mawi.wide.ad.jp/mawi/ (2013).

[14] CRAWDAD, [Online]. Available: http://crawdad.cs.dartmouth.edu/ (2013).

[15] WITS: Waikato Internet Traffic Storage, [Online]. Available: http://www.wand.net.nz/wits/ (2013).

[16] G. Szabó, D. Orincsay, S. Malomsoky, I. Szabó, On the Validation of Traffic Classification Algorithms, in: Proceedings of the 9th International Conference on Passive and Active network Measurement PAM 2008, Springer Berlin Heidelberg, Cleveland, Ohio, USA, 2008, pp. 72–81, dOI: 10.1007/978-3-540-79232-1_8.

[17] F. Gringoli, L. Salgarelli, M. Dusi, N. Cascarano, F. Risso, K. C. Claffy, Gt: picking up the truth from the ground for internet traffic, ACM SIGCOMM Computer Communication Review 39 (5) (2009) 12–18, dOI: 10.1145/1629607.1629610.

[18] M. Canini, W. Li, A. W. Moore, R. Bolla, GTVS: Boosting the Collection of Application Traffic Ground Truth, in: Proceedings of the First International Workshop on Traffic Monitoring and Analysis, TMA 2009, Springer Berlin Heidelberg, Aachen, Germany, 2009, pp. 54–63, dOI: 10.1007/978-3-642-01645-5_7.

[19] V. Carela-Español, T. Bujlow, P. Barlet-Ros, Is our ground-truth for traffic classification reliable?, in: Proceedings of the 15th Passive and Active Measurement Conference (PAM 2014), Proceedings Series: Lecture Notes in Computer Science 8362, Springer International Publishing Switzerland, Los Angeles, USA, 2014, pp. 98–108, dOI: 10.1007/978-3-319-04918-2_10.

[20] T. Bujlow, K. Balachandran, T. Riaz, J. M. Pedersen, Volunteer-Based System for classification of traffic in computer networks, in: Proceedings of the 19th Telecommunications Forum TELFOR 2011, IEEE, Belgrade, Serbia, 2011, pp. 210–213, dOI: 10.1109/TELFOR.2011.6143528.

[21] L. Deri, M. Martinelli, T. Bujlow, A. Cardigliano, nDPI: Open-Source High-Speed Deep Packet Inspection, in: Proceedings of the 10th International Wireless Communications & Mobile Computing Conference 2014 (IWCMC 2014), IEEE, Nicosia, Cyprus, 2014, pp. 617–622, dOI: 10.1109/IWCMC.2014.6906427.

[22] D. De Sensi, M. Danelutto, L. Deri, Dpi over commodity hardware: implementation of a scalable framework using fastflow, Master's thesis, Università di Pisa, Italy, accessible: http://etd.adm.unipi.it/t/etd-02042013-101033/ (2012).

[23] M. Ott, Intelligent network based application recognition, uS Patent 6,961,770 (November 2005).

[24] Application Layer Packet Classifier for Linux, [Online]. Available: http://l7-filter.sourceforge.net/ (2009).

[25] T. Bujlow, K. Balachandran, S. Ligaard Nørgaard Hald, T. Riaz, J. Myrup Pedersen, Volunteer-Based System for research on the Internet traffic, TELFOR Journal 4 (1) (2012) 2–7, accessible: http://journal.telfor.rs/Published/Vol4No1/Vol4No1.aspx.

[26] Volunteer-Based System for Research on the Internet, [Online]. Available: http://vbsi.sourceforge.net/ (2012).

[27] w3schools.com, OS Platform Statistics, [Online]. Available: http://www.w3schools.com/browsers/browsers_os.asp (2013).

[28] Palo Alto Networks. APPLICATION USAGE AND THREAT REPORT, [Online]. Available: https://www.paloaltonetworks.com/resources/whitepapers/application-usage-and-threat-report.html (2013).

[29] CNET, P2P & File-Sharing Software, [Online]. Available: http://download.cnet.com/windows/p2p-file-sharing-software/?tag=nav (2013).

[30] CNET, FTP Software, [Online]. Available: http://download.cnet.com/windows/ftp-software/?tag=mncol;sort&rpp=30&sort=popularity (2013).

[31] Top 15 Most Popular File Sharing Websites, [Online]. Available: http://www.ebizmba.com/articles/file-sharing-websites (2013).

[32] Top 15 Most Popular Video Websites | December 2013, [Online]. Available: http://www.ebizmba.com/articles/video-websites (2013).

[33] Popular on YouTube – YouTube, [Online]. Available: http://www.youtube.com/charts/videos_views?t=a (2013).

[34] w3schools.com, Browser Statistics, [Online]. Available: http://www.w3schools.com/browsers/browsers_stats.asp (2013).

[35] Alexa Top 500 Global Sites, [Online]. Available: http://www.alexa.com/topsites (2013).

[36] Top 15 Most Popular Websites | December 2013, [Online]. Available: http://www.ebizmba.com/articles/most-popular-websites (2013).

[37] Quantcast – Top Ranking International Websites, [Online]. Available: https://www.quantcast.com/top-sites (2013).

[38] Top 15 Most Popular Search Engines – eBizMBA, [Online]. Available: http://www.ebizmba.com/articles/search-engines (2013).

[39] T. Bujlow, V. Carela-Español, P. Barlet-Ros, Extended Independent Comparison of Popular Deep Packet Inspection (DPI) Tools for Traffic Classification, Tech. rep., Department of Computer Architecture (DAC), Universitat Politècnica de Catalunya (UPC), accessible: https://www.ac.upc.edu/app/research-reports/html/research_center_index-CBA-2014,en.html (January 2014).

[40] Email Client Market Share and Popularity – October 2013, [Online]. Available: http://emailclientmarketshare.com/ (2013).

[41] DFC Intelligence, [Online]. Available: http://www.dfcint.com/wp/?p=343 (2013).

[42] V. Carela-Español, P. Barlet-Ros, A. Cabellos-Aparicio, J. Solé-Pareta, Analysis of the impact of sampling on NetFlow traffic classification, Computer Networks 55 (2011) 1083–1099, dOI: 10.1016/j.comnet.2010.11.002.

[43] pmacct project: IP accounting iconoclasm, [Online]. Available: http://www.pmacct.net/ (2013).

[44] Traffic classification at the Universitat Politècnica de Catalunya (UPC)., [Online]. Available: http://www.cba.upc.edu/monitoring/traffic-classification (2014).

[45] M. Dusi, F. Gringoli, L. Salgarelli, Quantifying the accuracy of the ground truth associated with Internet traffic traces, Computer Networks 55 (5) (2011) 1158–1167, dOI: 10.1016/j.comnet.2010.11.006.

[46] H. Kim, K. C. Claffy, M. Fomenkov, D. Barman, M. Faloutsos, K. Lee, Internet Traffic Classification Demystified: Myths, Caveats, and the Best Practices, in: Proceedings of the 2008 ACM CoNEXT conference, ACM

New York, Madrid, Spain, 2008, p. 8, dOI: 10.1145/1544012.1544023.

[47] K. Fukuda, Difficulties of identifying application type in backbone traffic, in: 2010 International Conference on Network and Service Management (CNSM), IEEE, Niagara Falls, Ontario, Canada, 2010, pp. 358–361, dOI: 10.1109/CNSM.2010.5691234.

[48] S. Gebert, R. Pries, D. Schlosser, K. Heck, Internet access traffic measurement and analysis, in: Proceedings of the 4th international conference on Traffic Monitoring and Analysis (TMA'12), Springer Berlin Heidelberg, Vienna, Austria, 2012, pp. 29–42, dOI: 10.1007/978-3-642-28534-9_3.

[49] P. Megyesi, S. Molnár, Finding Typical Internet User Behaviors, in: Proceedings of the 18th EUNICE Conference on Information and Communications Technologies (EUNICE 2012): Information and Communication Technologies, Springer Berlin Heidelberg, Budapest, Hungary, 2012, pp. 321–327, dOI: 10.1007/978-3-642-32808-4_29.

[50] G. Aceto, A. Dainotti, W. De Donato, A. Pescapé, PortLoad: taking the best of two worlds in traffic classification, in: INFOCOM IEEE Conference on Computer Communications Workshops, 2010, IEEE, San Diego, California, USA, 2010, pp. 1–5, dOI: 10.1109/INFCOMW.2010.5466645.

[51] R. Goss, R. Botha, Deep Packet Inspection – Fear of the Unknown, in: Information Security for South Africa (ISSA), 2010, IEEE, Sandton, Johannesburg, South Africa, 2010, pp. 1–5, dOI: 10.1109/ISSA.2010.5588278.

[52] S. Valenti, D. Rossi, A. Dainotti, A. Pescapè, A. Finamore, M. Mellia, Reviewing Traffic Classification, in: Data Traffic Monitoring and Analysis, Springer Berlin Heidelberg, 2013, pp. 123–147, dOI: 10.1007/978-3-642-36784-7_6.

Tomasz Bujlow is a Ph.D. Student in the Department of Electronic Systems at Aalborg University in Denmark. He received his Master of Science in Computer Engineering from Silesian University of Technology in Poland in 2008, specializing in Databases, Computer Networks and Computer Systems. Previously, he obtained his Bachelor of Computer Engineering from University of Southern Denmark in 2009, specializing in software engineering and system integration. His research interests include methods for traffic classification in computer networks. He is also a Cisco Certified Network Professional (CCNP) since 2010.

Valentín Carela-Español received a B.Sc. degree in Computer Science from the Universitat Politècnica de Catalunya (UPC) in 2007 and a M.Sc. degree in Computer Architecture, Networks, and Systems from UPC in 2009. He is currently a Ph.D. Student at the Computer Architecture Department at the UPC. His research interests are in the field of traffic analysis and network measurement, focusing on the identification of applications in network traffic.

Pere Barlet-Ros received the M.Sc. and Ph.D. degrees in Computer Science from the Universitat Politècnica de Catalunya (UPC) in 2003 and 2008, respectively. He is currently an Associate Professor with the Computer Architecture Department of UPC and co-founder of Talaia Networks, a University spin-off that develops innovative network monitoring products. His research interests are in the fields of network monitoring, traffic classification, and anomaly detection.