VERSITA

BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **13**, No 3

Sofia • 2013

Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2013-0026

P2P Traffic Identification Based on Host and Flow Behaviour Characteristics

Jinghua Yan, Zhigang Wu**, Hao Luo**, Shuzhuang Zhang***

** School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, 100876, China*
*** Institute of Network Technologies, Beijing University of Posts and Telecommunications, Beijing, 100876, China*
Emails: jhyan@bupt.edu.cn wuzhigang@bupt.edu.cn luohao@bupt.edu.cn zhangshuzhuang@bupt.edu.cn

Abstract: *Peer-to-Peer (P2P) networks have been widely applied in file sharing, streaming media, instant messaging and other fields, which have attracted large attention. At the same time P2P networks traffic worsens the congestion of a network significantly. In order to better manage and control P2P traffic, it is important to identify P2P traffic accurately. In this paper we propose a novel P2P identification scheme, based on the host and flow behaviour characteristics of P2P traffic. First we determine if a host takes part in a P2P application by matching its behaviour with some predefined host level behaviour rules. Subsequently, we refine the identification by comparing the statistical features of each flow in the host with several flow feature profiles. The experiments on real world network data prove that this method is quite efficient to identify P2P traffic. The classification accuracy achieves 93.9 % and 96.3 % in terms of flows and bytes respectively.*

Keywords: *P2P traffic identification, host behaviour, flow behaviour.*

1. Introduction

With the rapid development of Peer-to-Peer (P2P) technology, the P2P traffic has accounted for more than 60 % of Internet traffic [1]. However, P2P applications have resulted in a significant consumption of the network bandwidth and brought a great impact on the quality of service for other network traffics. Therefore, the problem how to accurately identify P2P traffic has attracted great attention from the research and industry community. P2P identification can allocate the network resource more feasibly and can improve the quality of a service in a network.

A number of approaches have been proposed for P2P identification. The most traditional method is port-based, which inspects the ports number in the packet headers and then identifies the application according to the well known port number

[2]. However, this approach is becoming increasingly inaccurate since more and more P2P applications do not use the standardized port number. This implies that the traditional port-based approach has not been suitable for the identification of P2P traffic any more. An alternative way is using DPI (Deep Packet Inspection) technology to examine the payload of P2P network flows and then create signatures for each application [3]. However, it generates several limitations. First, it is only useful as long as the packet payload is unencrypted. Second, it is hard to keep the signature databases up-to-date, since new P2P applications are born every day and the current ones evolve constantly. In order to address these limitations, the research community has recently proposed several machine learning techniques for traffic identification by using the statistic features of each flow (e.g. the packet size, flow duration, inter-packet time, packet numbers, etc.) [4-7]. The start point of the machine learning method is that flows of different applications have distinct statistical properties. On the other hand, the researchers have proposed some heuristic methods based on the characteristics of the host behaviours as well [8-14]. These approaches are based on the analysis of the host behaviour instead of the individual traffic flow. The basic idea of this solution is that different applications have different communication patterns, for instance, a client application has a few outgoing connections, while a server has numerous incoming connections. The P2P applications have many connections in both directions, and so on.

Both the machine learning methods based on statistical features, and the heuristic method based on host behaviour do not need to inspect the packet payload. Therefore, these methods provide a promising alternative for P2P traffic identification. Although the above methodologies have played an important role in traffic identification, they still suffer from some limitations. The machine learning methods are relying on the statistical features of the flow, which may be sensitive to the network condition and easy to spoof, while the host behaviour based methods are coarse identification methods. They can determine whether a host takes part in an application, but a host may participate in different applications at the same time. So the classification accuracy of the host behaviour method may be decreased for this reason.

In this paper we propose a new approach to identify P2P traffic, which can be considered as a combination of the machine learning method and host behaviour based method. It consists of two stages: First, we determine whether a host takes part in an application by matching with a set of predefined host level heuristic rules. We capture the connection patterns of flows sequences to or from a specific host instead of looking at an individual flow, and then regard them matched with a set of predefined heuristic rules (e.g., the number of ports/number of IP, ratio of failure connections, and so on). Second, we classify each flow in the host by comparing its statistical features with the flow level profiles of the applications (e.g., the flow duration, flow bytes and so on). In our two-stage matching method, we first locate these hosts that participate in the application and then classify their flows, which can refine the classification results.

The major contribution of the proposal is that we propose a finer P2P traffic identification framework, which combines the host level and the flow level behaviour characteristics of P2P traffic. By the method proposed we can successfully distinguish P2P traffic from the traditional non-P2P traffic.

The rest of the paper is organized as follows: The work related to P2P identification is discussed in Section 2. Section 3 outlines our two-stage method of P2P traffic identification based on the host level and flow level behaviour characteristics. Section 4 describes the implementation of our approach. Experiments based on a real-world traffic trace are shown in Section 5. Finally we make conclusions and discuss the future work in Section 6.

2. Related works

Port based and DPI methods are widely used today for P2P identification. Unfortunately, both turn out to be more and more inefficient, so we do not consider these approaches further. In this section we will concentrate on machine learning methods and host behaviour based methods.

The procedure of machine learning approaches includes two stages: First extract the statistical properties of flows, such as the average packet size, the average flow duration, and the inter-arrival times between packets and so on. Then employ clustering, classification or other machine learning methods to classify each flow. There are several limitations in identifying applications using flow properties. First, some statistical features, such as the packet size are easily spoofable. Second, some features, such as inter-packet time are sensitive to network dynamics like congestion or path changes. Last, the flows in different applications may have similar properties. Therefore, there will be false positive if per-flow statistics feature is only used. We also find that these machine learning methods do not exploit host-related properties which, we believe, contain a lot of valuable information.

The host behaviour based methods are first proposed by Karagiannis. In [8] K a r a g i a n n i s et al. propose a method named BLINC to discover the behavioural patterns (e.g., the number of hosts contacted, the transport layer protocol employed, the number of different ports used, etc.) of the hosts. They intend to obtain the inherent behaviour of a host at three levels with increasing details: the social, the functional and the application level. At the social level they consider the behaviour of a host in terms of the hosts it communicates with. At the functional level they consider the behaviour of the host in terms of its functional role in the network, whether it acts as a provider or a consumer of a service, or both. At the application level they regard the transport layer interactions between particular hosts at specific ports with the intent to identify the application of the origin. By using the above host level heuristic rules, they could accurately classify the traffic. Inspired by BLINC, more researchers attempt to identify P2P traffic using the host behaviour. C o l l i n s et al. [9] distinguish BitTorrent flows from other flows by using three metrics: the packet size (looking for small control

messages), the amount of data exchanged between hosts, and the rate of failed connections [9]. H u et al. [10] proposed a novel profile-based approach to identify the traffic flows belonging to the target application [10]. They construct the host profile by using an association rule method, which needs a certain amount of training samples. P e r e n y i et al. [11] proposed an updated set of six heuristics to identify and analyze P2P traffic, based on very similar ideas like K a r a g i a n n i s et al. [8]. P a o l a B e r m o l e n [12] proposed a method to identify Peer-to-Peer streaming (P2P-TV) application by calculating the number of packets and bytes exchanged among peers during small time-windows [12]. Similar to Paola Bermolen, J i e Y a n g et al. [13] analyzed the payload length distribution and payload length pattern of four popular P2P streaming media applications. Their approach concentrated only on several (P2P-TV) applications and needs to be expanded to classify more P2P applications. K e X u et al. [14] proposed a novel approach to identify P2P traffic by using the data transfer behaviour of P2P [14]. The behaviour investigated in this paper is that downloaded data from a P2P host will be uploaded on other hosts later.

The machine learning approaches rely mainly on the statistical features of the individual flow, while the host level behaviour methods intend to identify the traffic based on the host behaviours. Actually speaking, both the host level and the flow level characteristics are useful information for traffic identification. In this paper we summarize comprehensive heuristic rules for P2P applications, and our work combines the classification ability of both flow-level methods and host-behaviour based methods.

3. Host and flow behaviours of P2P traffic

In this section we exploit the behaviour profiles of P2P traffic with respect to host level behaviour and flow level behaviour. We will introduce the unique host-level and flow-level patterns of P2P traffic respectively. Our approach deploys only on flow records and requires no information about the signature of the individual packets.

3.1. Host behaviour profile of P2P traffic

In this subsection we summarize some host-level behaviour profiles (H1-H4) of P2P traffic. We focus on a host instead of a single flow, and can acquire enough information to depict the behaviour of a host. The heuristic rules proposed include a number of thresholds which might be tunable and we will set their values empirically. The host behaviour profiles are listed as follows:

H1: (IP popularity ratio). The IP popularity ratio is defined as the number of distinct hosts it communicates with, divided by the whole number of hosts it communicates with. We define a variable named ***ip_pop_ratio*** for this indicator, which is calculated as follows:

$$ip\_pop\_ratio = diff\_ipnum / all\_ipnum \quad (1)$$

where *diff_ipnum* is the number of different hosts it communicates with, *all_ipnum* is the number of all hosts it communicates with. Hosts interacting with a large number of other hosts in a short time period appear to participate in a P2P network. In traditional non-P2P applications, such as WEB and MAIL, the client hosts contact only with several fixed server hosts, while a P2P host will communicate with many different hosts to improve the network speed. Therefore, the value of *ip_pop_ratio* for the hosts running a P2P application will be close to 1. We set the threshold of *ip_pop_ratio* to 0.9 and we set a flag variable ***ip_pop_flag***. *ip_pop_flag* to 1, if *ip_pop_ratio* > 0.9, which indicates a potential P2P host.

H2: (Port pair difference). Suppose a client host *C* is running a non-P2P application (e.g., Web application); it will use a number of source ports to connect to a destination port of server *S*. By using multiple connections, *C* can fetch different objects simultaneously. However, P2P peers usually maintain only one connection to the other peer, which means that the number of source ports and the number of destination ports are nearly equal, or the difference will be small. Suppose that for a source and destination IP pair ⟨sIP, dIP⟩, the number of distinct source ports of sIP and the number of destination ports of dIP can be expressed as *source_portnum* and *dst_portnum*. We define a variable ***port_pair_ratio*** and ***port_pair_diff*** to measure the ratio and the difference of *source_portnum* and *dst_portnum*:

(2) $$port\_pair\_ratio = source\_portnum / dst\_portnum,$$

(3) $$port\_pair\_diff = source\_portnum - dst\_portnum.$$

We set a flag variable *port_pair_flag*, *port_pair_flag* to 1 if *port_pair_diff* < 2, *port_pair_ratio* < 1.5, and at least 5 different source ports exit for this host, which indicates a potential P2P host.

H3: (Ephemeral port ratio). Generally speaking, non-P2P applications, such as Mail and Web use well known privileged ports. Unlike non-P2P applications, a P2P application will have an ephemeral port number, which is above 1024. We define the ephemeral port ratio as *eph_ratio*, which can be calculated as

(4) $$eph\_ratio = eph\_flownum / all\_flownum$$

where *eph_flownum* is the number of flows which have a source and a destination port in the unprivileged range and *all_flownum* is the number of total flows at this host. For non-P2P traffic, such as Web and Mail, the expected value for the ratio *eph_ratio* is near to 0. For P2P application, the value will be close to 1. Here we set the minimum value of *eph_ratio* value 0.8. We set a flag variable ***eph_flag***, its value is 1 only if the value of *eph_ratio* of a host > 0.8, and the host is considered as a potential P2P host.

H4: (Failed connection ratio). In P2P networks the mechanisms which track the presence of peers are not always perfect, and the current information of peers will quickly become out of date. As a result, P2P peers always connect to peers that have disconnected from the P2P network, which leads to failed connection. This behaviour is not common among other traditional Client/Server applications. In Client/Server protocols, the servers often work well and almost all the connections

are successful. We capture the failed connection behaviour of a given host and express it as *fail_ratio*, which is calculated as follows:

$$fail\_ratio = fail\_flownum / all\_flownum \quad (5)$$

where *fail_flownum* is the total number of new outgoing connections that failed and *all_flownum* is the whole number of new outgoing connections of the host. Values of fail_ratio tend to be low for normal applications, higher for P2P hosts. We set the threshold of *fail_ratio* to 0.2 to indicated P2P. At the network level, these failed connections will result in RST messages or multiple SYN packets intending to start a connection, and timing out. We can determine the failed connection situation by checking multiple SYN and RST flags. We set a flag variable *fail_flag*, its value is 1 only if the value of *fail_ratio* > 0.2. The host meeting this condition is considered as a participant in the P2P application.

3.2. Flow behaviour profile of P2P traffic

We can judge whether a host participates in a given application by comparing its host behaviour with the host heuristic profiles of this application. While a host may run different applications at the same time, we then compare each flow of the host to determine which flow belongs to this application and which one does not. A flow is defined as consecutive packets sharing the same 5-tuple (source IP and destination IP, source port and destination port, IP protocol). In this subsection, we will shift to focus on the flow-level behaviour profiles (F1-F2) of P2P traffic. The flow behaviour profiles are listed as follows:

F1: (Flow bytes and flow duration of a large flow). A large portion of hosts running P2P application will download large files from other peers. Therefore there are many large flows in P2P traffic. Flows which carry more than 1 MB of data in one direction and have flow durations longer than 10 minutes are identified as P2P flows. This heuristic rule is proposed by Perenyis [11]. We set a flag name *large_flag*, the value of *large_flag* is 1 only if the feature of a flow satisfies the former requirements.

F2: (Byte ratio of forward and backward direction). Since peers in P2P networks are equivalent, this means that each peer can initiate and receives packets. While Client/Server hosts primarily either initiate connections (clients) or receive them (servers). Generally speaking, the number of bytes of the backward direction is much bigger than that of the forward direction for a Client/Server application. Unlike Client/Server applications, an obvious behaviour of the hosts in a P2P application is the balance of both forward and backward connections. We set a variable named *byte_direction_ratio*, which can be expressed as

$$byte\_direction\_ratio = \frac{byte\_forward}{byte\_backward} \quad (6)$$

byte_forward denotes the number of bytes in the forward direction of a flow, *byte_backward* denotes the number of bytes in the backward direction of a flow. We set the region of *byte_direction_ratio* between 0.5 and 1.8 to indicate a P2P flow. We set a flag variable *byte_direction_flag*, its value is 1 only if 0.5 < *byte_direction_ratio* < 1.8.

4. Implementation of the proposed scheme

In Section 3 we have summarized host Heuristic rules (H1-H4) and Flow heuristic rules (F1-F2). Our identification framework can be divided into two processes, at first we evaluate the status of a host, and assign a label named *host_label* for this host to indicate whether it is a possible P2P host. Then associate the flows of this host with a label named *flow_label* to determine its final application, and the *flow_label* is the final result of the identification.

In the last section we list four rules to analyze a host, and there are four flags (*ip_pop_flag*, *port_pairt_flag*, *eph_flag*, *fail_flag*). Each rule has the ability to judge a host, suppose we assign a weight w_i for each rule, and set a variable *host_rule* to measure the overall possibility of a host belonging to a P2P host. The *host_rule* can be calculated as follows:

$$host\_rule = w_1 \times ip\_pop\_flag + w_2 \times port\_pair\_flag + \\ + w_3 \times eph\_flag + w_4 \times fail\_flag \quad (7)$$

where w_i is the weight for each rule, here we set $w_i = 0.25$, which means we treat each rule equally. The higher the value of *host_rule* is, the more confident we are. If the value of *host_rule* >= 0.5, we label the host as "P2P", that is *host_label*="P2P". If 0< *host_rule*< 0.5, the case is not very obvious, and we will mark the host as "unclassified". We will further analyze this case at the flow-level stage. The last situation is if *host_rule* = 0, it is quite sure that the host is "non-P2P", and we filtered it out.

At the flow-level identification stage we calculate the statistical feature of a flow incoming and outgoing from a "P2P" or "unclassified" host, and compare it with rule F1 and F2. If one of these rules is established, we consider this flow to be a P2P flow. For a host which is marked as a P2P host, if the flow is determined as a P2P flow, *flow_label*="P2P", else *flow_label*="non-P2P". For a host which is labeled as unclassified host, if the flow is determined as a P2P flow, *flow_label*="P2P", else *flow_label*="unclassified".

Typically we deal with a flow table, which is the fundamental information of the network traffic. The flow table maintains the information, such as IP address, ports number, timestamp, flow duration, flow bytes. We process the flow table, evaluate each rule of all flows over a sliding window. We set a variable named *interval* to indicate the window size; in this paper *interval* = 1 minute. The whole identification algorithm can be described as Algorithm 1.

Algorithm 1: Traffic identification based on host and flow behaviour characteristics

Input: Flow Table
Processing:
Step 1. Examine the host heuristic rules
1.1. For a host, calculate the value of **host-level** flag according to rules H1, H2, H3, and H4.
1.2. Compute *hos_rule*, if *host_rule* >=0.5, label the host as "P2P", if 0 < *host_rule* < 0.5, label the host as "unclassified", otherwise label it as "non-P2P".

Step 2. Examine the flow heuristic rules
2.1. For a flow incoming or outgoing from a P2P or unclassified host, we calculate the value of the flow-level flag according to rules F1, F2.
2.2. If one of rules F1 and F2 is established for a "P2P" host, we label the flow as P2P flow, otherwise label it as non-P2P. For "unclassified" host, we label the flow as P2P flow; otherwise label it as "unclassified".
Output: *flow_label* for each flow

5. Experiments

In order to evaluate the performance of the method proposed, we applied our method to a real network dataset. The experiment results confirm the efficiency of the method suggested.

5.1. Dataset description and evaluation metrics

The dataset we used for the experiment is Unibs dataset [15]. This dataset was collected at the edge router of the campus network of the University of Brescia in three consecutive working days (2009.9.30-2009.10.02). The dataset is composed of traffic generated by a set of twenty workstations. The traffic is collected by running tcpdump on the faculty's router, which is a dual xeon linux box that connects the network to Internet through a dedicated 100 Mb/s uplink. The dataset consists of Web (http and https), Mail (pop3, pop3s, imap, imaps) and P2P (bittorrent, edonkey, skype). We have divided the network data in two main groups: P2P and non-P2P applications. The composition of UNIBS dataset is given in details in Table 1. From Table 1 we can observe that although non-P2P applications occupy a large portion of the connections of the dataset, the P2P applications account for the majority bytes of the dataset.

Table 1. The composition of UNIBS dataset

Application	Flow number	Flow ratio	Byte (GB)	Byte ratio
P2P	19683	29.0%	19.7	87.6%
Non-P2P	48254	71.0%	2.8	12.4%

There are several metrics to evaluate the accuracy of the identification task, such as True Positive, False Positive and False Negative. For a given class, the number of correctly classified samples is referred to as True Positive (TP). The number of samples falsely identified as a class is referred to as False Positive (FP). The number of objects from a class that are falsely labeled as another class is referred to as False Negative (FN). We measure the performance of a given method in terms of the following metrics.

Overall accuracy: the ratio of all samples correctly classified. This metric is defined as the sum of all TP to the sum of all TP and FP for the whole class, that is

$$overall\ accuracy = \sum_{i=1}^{n} \mathrm{TP}_i \Big/ (\mathrm{TP}_i + \mathrm{FP}_i). \tag{8}$$

Recall: the ratio of samples from a given class that are properly attributed to that class. Recall is the ratio of TP to the number of TP and FN, that is

(9) $$recall = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN}).$$

Precision: the ratio of samples correctly attributed to a class over the total samples attributed to that class. Precision is the ratio of TP to TP and FP, that is

(10) $$precision = \mathrm{TP}/(\mathrm{TP} + \mathrm{FP}).$$

Both of the above metrics can be computed for classification performance in terms of flows and bytes. For example, the overall flow accuracy is the ratio of all flows correctly classified, while the overall byte accuracy is the ratio of all bytes correctly classified.

5.2. Experimental results

The method proposed can be divided into two stages: host level classification and flow level classification. The host level classification stage determines whether a host is running a given application. All flows to and from this host are marked as flows of this application, since a host may take part in different applications even in a short time. So we further analyze each flow to and from a host using the flow level heuristic rules to boost the classification accuracy.

To illustrate the accuracy of our method, Table 2 shows the confusion matrix for the host level classification stage. In this matrix the value $C_{i,j}$ indicates the number of samples from class *i* that were classified as class *j*. We can compute the recall for class *i* by looking across the row of the confusion matrix at a given class *i*. We can calculate the precision of class *j* by looking down a column at the given class. To the standard confusion matrix we add an extra column ("unclassified") that depicts the percentage of samples which cannot be classified by our method.

5.2.1. Host-level stage results

Table 2 shows the flow classification accuracy results of the host level stage. We observe that 87.4 % of P2P flows are correctly classified as P2P flows, while 9.2 % of P2P flows are wrongly classified as non-P2P flows, 3.4 % of P2P flows cannot be classified by host-level heuristic rules. For non-P2P flows, 87.3 % of non-P2P flows are correctly classified, 8.6 % of non-P2P flows are wrongly classified as P2P flows, 4.1 % of non-P2P flows cannot be classified. From Table 2 we can calculate that the overall flow accuracy of the host-level stage is 87.5 %. The results of recall and precision rate of P2P and non-P2P are listed in Table 3. Table 3 shows that the precision of P2P is 81.8 % which is relatively lower than other metrics. We look into the dataset carefully and find that 4108 non-P2P flows are classified as P2P. The reasons are twofold: (1) a non-P2P host follows several bad links to servers which are down, the failed connections will raise the value of *fail_ratio*, thus increase the probability of the identified as a P2P host; (2) in a short time a host may run non-P2P applications, although it is classified as a P2P host. Similarly, there are 1815 P2P flows wrongly classified as non-P2P flows, the reasons can be categorized as follows: (1) a P2P host only connects to another constant peer, so the distinct number of different hosts it communicates with is very small, which will decrease the value of *ip_pop_ratio*. In this way it is more likely to be determined as a non-P2P host; (2) in a short time a host may have several P2P flows, although it is classified as a non-P2P host.

Table 2. Confusion matrix of host level classification stage (flow accuracy)

Application	P2P (%)	Non-P2P (%)	Unclassified (%)
P2P	17205 (87.4%)	1815 (9.2%)	663 (3.4%)
Non-P2P	4108 (8.6%)	42234 (87.3%)	1912 (4.1%)

Table 3. Confusion matrix of host level classification stage (flow accuracy)

Application	Recall	Precision
P2P	87.4%	81.8%
Non-P2P	87.5%	95.6%

We can also obtain similar results in terms of bytes and the results are described in Tables 4 and 5. We can calculate that the overall byte accuracy of the host-level stage is 91.3 %, which is a little higher than the overall flow accuracy. We think that the reason is that P2P applications carry more data than non-P2P applications.

Table 4. Confusion matrix of host level classification stage (byte accuracy)

Application	P2P (%)	Non-P2P (%)	Unclassified (%)
P2P	18.45 (93.6%)	0.43 (2.2%)	0.82 (4.2%)
Non-P2P	0.42 (15.1%)	2.18 (77.8%)	0.2 (7.1%)

Table 5. Confusion matrix of host level classification stage (byte accuracy)

Application	Recall	Precision
P2P	93.6%	97.8%
Non-P2P	77.8%	83.5%

5.2.2. Flow-level stage results

From the above results we observe that there are still wrongly classified flows and unclassified flows. It is necessary to further classify each flow from a host. After the host-level classification stage, we process the flow-level classification stage by comparing with the flow behaviour rules. Table 6 reports the flow classification accuracy results at the flow level classification stage. Compared with Table 2, we find that the number of correctly classified flows increases, whereas the number of unclassified flows decreases. It can be implied that the flow level classification has improved the classification performance. Take P2P flows for example, there are still 663 flows unclassified in the host-level classification stage. After using the flow-level heuristic rules, we can identify 552 of them as P2P flows. For some non-P2P flows which are classified as P2P flows, we match their statistic properties with the flow behaviour rules, and successfully identify them as non-P2P flows. In this way the flow level classification stage promotes the classification performance. We can calculate that the overall flow accuracy of the flow-level classification stage is 93.9 %, and the recall and precision results are listed in Table 7.

Table 6. Confusion matrix of flow level classification stage (flow accuracy)

Application	P2P (%)	Non-P2P (%)	Unclassified (%)
P2P	17693 (89.9%)	1879 (9.5%)	111 (0.6%)
Non-P2P	1647 (3.4%)	46086 (95.5%)	521 (1.1%)

Table 7. Confusion matrix of flow level classification stage (flow accuracy)

Application	Recall	Precision
P2P	89.9%	91.5%
Non-P2P	95.5%	96.0%

The results in terms of bytes of the flow level classification stage are shown in Tables 8 and 9. The method proposed achieves promising performance and we can compute that the overall byte accuracy of the flow-level classification stage is 96.3 %. The results show that our approach is very promising and it has very high recall and precision rate. It achieves average accuracy for all flows and bytes 93.9 and 96.3 % respectively. On this dataset the proposed heuristics left as little as 1 % of the flows and 1.2 % of the data unclassified.

Table 8. Confusion matrix of flow level classification stage (byte accuracy)

Application	P2P (%)	Non-P2P (%)	Unclassified (%)
P2P	18.99 (96.4%)	0.49 (2.5%)	0.22 (1.1%)
Non-P2P	0.07 (2.5%)	2.67 (95.4%)	0.06 (2.2%)

Table 9. Confusion matrix of flow level classification stage (byte accuracy)

Application	Recall	Precision
P2P	96.4%	99.5%
Non-P2P	95.4%	86.5%

5.3. Discussion

The proposed P2P identification method in this paper is based on the host level and flow level heuristic rules of P2P network traffic. Compared to typical machine learning algorithms, our method needs to obtain only simple information about the traffic instead of complex statistical features. Besides, the identification procedure does not need any training samples because we use heuristic rules only. Furthermore, the work on the network is as short as 1 minute, which allows the operators to classify the traffic relatively fast.

Although the proposed approach is very promising, it still has some limitations. First, the method relies on the IP address of flows, the classification results will degrade if the flows go through NAT (Network Address Translators) or use dynamic IP addresses. Second, this method only makes coarse classification, since it can only identify broad P2P applications instead of sub-applications in P2P. Last, we adopt several related parameters (*ip_pop_ratio*, *port_pairt_ratio*, *large_ratio*, *fail_ratio*, *eph_ratio*, *byte_direction_ratio*) for identification, we set the thresholds for the parameters empirically. The parameter values that optimize the method may differ on different links, so we have to tune the parameter values carefully for different network datasets.

6. Conclusion

Nowadays P2P applications are responsible for the majority of network traffic. P2P traffic identification has recently attracted great attention due to its importance for network management and network security. In this paper we propose a framework to identify P2P traffic by making use of host and flow behaviour characteristics of

P2P traffic. Experiments on real network data have shown that the result of the suggested method is promising. It can obtain classification accuracy of 93.9 % and 96.3 % in terms of flows and bytes respectively, leaving as little as 1 % of the flows and 1.2 % of the bytes unclassified. However, our method can only identify broad P2P applications rather than different applications within P2P (e.g. bittorrent, gnutella and so on). In the future we will use additional information about these specific applications and achieve fine-grained P2P traffic classification.

Acknowledgements. This work is supported by the National Science and Technology Support Program of China (Grant No 2012BAH37B02, 2012BAH46B02), National High Technology Research and Development Program of China (Grant No 2012AA013001), National Information Security Program of China (Grant No 2012A54, 2012A51, 2012A62).

References

1. MacManus, R. Trend Watch: P2P Traffic Much Bigger than Web Traffic, 2006. **http://www.readwriteweb.com/archives/p2p_growth_trend_watch.php**
2. Moore, A., D. Zuev. Internet Traffic Classification Using Bayesian Analysis Techniques. – In: Proceedings of 26th ACM International Conference on Measurement and Modeling of Computer Systems, SIGMETRICS 2005, 2005, 50-59.
3. Sen, S., O. Spatscheck, D. Wang Accurate. Scalable in-Network Identification of P2P Traffic Using Application Signatures. – In: Proceedings of 13th World Wide Web Conference 2004, 512-520.
4. Nguyen, T. T., G. Armitage. A Survey of Techniques for Internet Traffic Classification Using Machine Learning. – IEEE Communications Surveys and Tutorials, Vol. **10**, 2008, No 4, 56- 76.
5. Crotti, M., M. Dusi, F. Gringoli, L. Salgarelli. Traffic Classification through Simple Statistical Fingerprinting. – ACM SIGCOMM Computer Communication Review, Vol. **37**, 2007, No 1, 7-16.
6. Bernaille, L., R. Teixeira, K. Salamatian. Traffic Classification on the Fly. – ACM SIGCOMM Computer Communication Review,Vol. **36**, 2006, No 2, 23-26.
7. Chen, M., X. Wei. Hidden Markov Model-Based P2P Flow Identification: A Hidden Markov Model-Based P2P Flow Identification Method. – IET Communications, Vol. **6**, 2012, No 13, 2091-2098.
8. Karagiannis, T., K. Papagiannaki, M. Faloutsos. BLINC: Multilevel Traffic Classification in the Dark. – In: Proceedings of the ACM SIGCOMM 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, 2005, 229-237.
9. Collins, M., M. Reiter. Finding Peer-to-Peer File-Sharing Using Coarse Network Behaviours. – In: Proceedings of 11th European Symposium on Research in Computer Security, ESORICS 2006, 1-17.
10. Hu, Yan, Dah-Ming Chiu, John C. S. Lui. Profiling and Identification of P2P Traffic. – Computer Networks, Vol. **53**, 2009, No 6, 849-863.
11. Perenyi, M., D. Trang Dinh, A. Gefferth, S. Molnar. Identification and Analysis of Peer-to-Peer Traffic. – Journal of Communications, Vol. **1**, 2006, No 7, 36-46.
12. Bermolen, P., M. Mellia, M. Meo, D. Rossi, S. Valenti. Abacus: Accurate Behavioural Classification of P2P-TV Traffic. – Computer Networks, Vol. **55**, 2011, No 6, 1394-1411.
13. Yang, Jie, Lun Yuan, Yang He, Lu-ying Chen. Timely Traffic Identification on P2P Streaming Media. – Journal of China Universities of Posts and Telecommunications, Vol. **19**, 2012, No 2, 67-73.

14. Xu, Ke, Ming Zhang, Mingjiang Ye, Dah Ming Chiu, Jianping Wu. Identify P2P Traffic by Inspecting Data Transfer Behaviour. – Computer Communications, Vol. **33**, 2010, No 10, 1141-1150.
15. Unibs Dataset.
http://www.ing.unibs.it/ntw/tools/traces/