Classifying P2P Activity in Netflow Records: A Case Study on BitTorrent

Ahmed Bashir[1], Changcheng Huang[1], Biswajit Nandy[2], Nabil Seddigh[2]
1 – Dept. of Systems and Computer Engineering, Carleton University, Ottawa, Canada
2 – Solana Networks, Ottawa, Ontario, Canada
E-mail: {abashir1, huang}@sce.carleton.ca[1], {bnandy, nseddigh}@solananetworks.com[2]

Abstract— **The ability to accurately classify various types of Internet traffic within a network using Netflow traces represents a major challenge as there is no payload information available with Netflow. P2P applications represent a very large portion of the internet traffic and are becoming more difficult to classify, as some of these applications tend to use port masquerading techniques and encrypted payloads, rendering the traditional classification approaches obsolete. In this paper, a simple yet effective classification method is proposed using a set of heuristics based on the discriminating features and the operation nature of P2P applications. We mainly focus on identifying BitTorrent activities using Netflow records. The presented scheme has been tested with a collection of real data sets. The results of the classification have shown to be accurate even when applied to data sets with complex Internet traffic. The results of the proposed scheme were tested against two other existing approaches and were observed to have improved classification accuracy – BitTorrent traffic was identified with 91-95% accuracy for the five data sets tested.**

I. INTRODUCTION

Accurate traffic classification is important to network operators for a number of reasons. The ability to accurately identify P2P applications finds utility in provision of QoS (Quality of Service), network security and network planning. In recent years, the increased complexity of Internet applications and the growth of P2P services have reduced the accuracy of current traffic classification techniques. This has led to a quest to develop new, scalable and robust techniques to classify P2P traffic. Traffic classification approaches can be broadly divided into four categories: (i) Port based classification [1] (ii) Deep packet inspection (DPI) [2], (iii) Machine learning [3] (iv) and Host behavior approaches [4].

Section II is a review of the current approaches proposed for Internet traffic classification, discussing both their advantages and disadvantages. Section III introduces BitTorrent applications explaining their functionality and their nature of operation. Section IV presents the heuristics used for classification and the proposed classification model. Finally, the results of applying the model to a collection of data sets are presented in Section V and the conclusions are presented in Section VI.

II. RELATED WORK

Recently, researchers have studied the utility of combining port-based classification with other identification techniques in a quest to achieve greater accuracy for classifying P2P traffic. Karagiannis et al. proposed the BLINC technique [1] where they combined port classification with host behavior to accurately classify different types of Internet traffic. The results from this technique could only classify traffic from a general perspective labeling all P2P traffic as Peer to Peer without specifying which specific P2P application generated the traffic.

In [2], for example, Meo et al. implemented a DPI classification apparatus for P2P traffic by matching the payload signatures to different P2P applications but the limitations of DPI approaches prevent this method from being deployed on Netflow records or transport layer information.

In [3], Rui et al. proposed a P2P classification approach using machine learning with SVM but the accuracy of the results solely depends on the accuracy and the quality of the given trainingdata sets that the algorithms rely on for classification.

Zhang et al. in [4] proposed a host behavior approach by exploiting the connection behavior of different applications and converting them to graphical and statistical representations based on flow cardinality and directions. This approach also cannot specify different types of P2P traffic and the classification accuracy for P2P traffic is significantly low when compared to the other traffic (85%).

After studying the current approaches suggested for Internet traffic classification, it was found that DPI fails to classify applications and services that utilize encrypted payloads. On the other hand, machine learning techniques are not reliable when the application features overlap. Moreover, port based approaches do not yield sufficient accuracy when applied against the current generation of P2P traffic.

The most promising approach for identifying P2P traffic appears to be the host behavior approach, due to the fact that it is application specific. This is important because our main objective is to identify BitTorrent activity. By analyzing BitTorrent connection patterns, heuristics and classification schemes are demonstrated in this paper to accurately identify

978-1-4673-3122-7/13/$31.00 ©2013 IEEE

them. The classification model is constructed by integrating different flow features unique to P2P applications, and the connection patterns that P2P applications exhibit whilst they are operating, in order to achieve a solid identification mechanism capable of accurately classifying the required traffic.

An important constraint has been added as a consideration when developing the classification model. We require all the features and attributes used in the classification model to be data that can be obtained from the Netflow protocol. Numerous vendors other than Cisco provide an equivalent technology capable of exporting IP flow information such as Juniper's jFlow and sFlow. The exported IP flow information from those other technologies is similar to Netflow in terms of the statistics they can provide regarding IP flow information such as IPs, ports, byte count and packet count. The adoption of this constraint means that the classification model developed can be applied to a wide diversity of traffic flow traces. It will also facilitate the identification of P2P activities despite missing payload information or in the presence of encrypted payloads. Netflow is the most popular protocol for exporting information regarding traffic flow statistics. Netflow contains over 40 different traffic statistics and defining attributes per traffic flow that is monitored.

III. Behavior Analysis

P2P applications have emerged as a dominant portion of today's Internet traffic since they provide a wide spectrum of services such as file sharing, video streaming, and creation of media hubs. The focus of this paper is on one of the most important P2P applications – BitTorrent. BitTorrent is widely considered as the most popular P2P file sharing service. It allows clients to download a file simultaneously from different peers. The following section introduces BitTorrent applications, describing the basic concepts behind their technology and focusing on their operational behavior inside a network.

A. *BitTorrent Clients*

BitTorrent [5] is a set of protocols and services that are used mainly for massive file sharing among nodes in a P2P network. To locate a desired file, two alternative sets of mechanisms are available: tracker based vs. DHT (Distributed Hashing Table) based.

In a tracker-based system, a client contacts a centrally maintained tracker for information regarding the nodes that have the desired file. All the nodes that have the desired file or parts of it form a group called a "swarm". If one peer desires to download from another peer in a swarm identified through the tracker, it contacts the peer using either TCP or UDP-based signaling mechanisms.

For a DHT based system, BitTorrent clients use DHTs for tracking the location information of different peers in order to download files without the assistance of a tracker. When DHT is enabled in a BitTorrent client, it connects to an initial bootstrap node to enter the DHT swarm. The bootstrap node provides the BitTorrent client with a set of initial peers to populate the DHT. Afterwards, the BitTorrent client queries those initial peers using UDP in order to find other peers that have the desired file. If a peer having the desired file is located, the BitTorrent client connects to the peer using TCP.

Figure 1 illustrates the connection patterns of a BitTorrent client downloading from peers via a tracker as well as the scenario where the client downloads from peers using DHT.

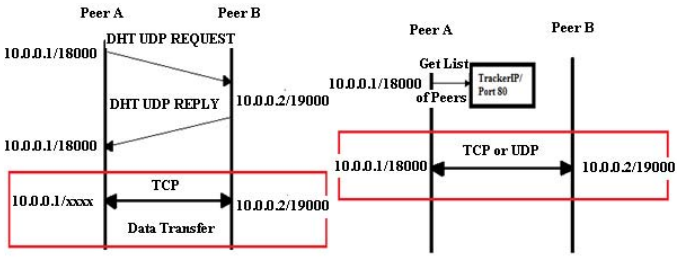

Figure 1 BitTorrent Client Connection to Peers via Tracker & DHT

IV. Classification Heuristics

After analyzing the behavior and connection patterns of BitTorrent applications, a number of classification heuristics were deduced based on the obtained results. These heuristics utilize a number of characteristics such as port number, packet size, number of flows, flow duration, etc.

As a starting point for the classification procedure, we wish to distinguish between P2P and non-P2P activities. This is carried out using the heuristic introduced in subsection (A).

Afterwards, it is necessary to identify a set of unique characteristics and connection patterns for BitTorrent applications in order to achieve accurate classification. The unique connection patterns for BitTorrent clients are introduced in both Subsection (B) and (C).

After the BitTorrent host has been identified, it is essential to extract all the associated flows generated by the BitTorrent client. This is accomplished by applying the heuristic displayed in Subsection (D).

Our classification model is developed by integrating the heuristics displayed in the following subsections. The combination and the ordering of the heuristics are summarized in Subsection (E) where the classification model is presented.

A. *DHT Probing Heuristic*

Since most BitTorrent clients currently utilize UDP for DHT probing, it is essential to identify the UDP port associated with the BitTorrent client in order to extract all the flows related to the P2P client. This is carried out by identifying all the outgoing UDP flows from a host that utilize the same source port and which have sent out more than 100 requests to different destinations. The number 100 is chosen based on experimentation and testing results – the summary of which is displayed in Figure 2. We ran 10 different tests and studied the behavior during the probing phase – where the application was idle (no file downloads) during a 10 minute duration.

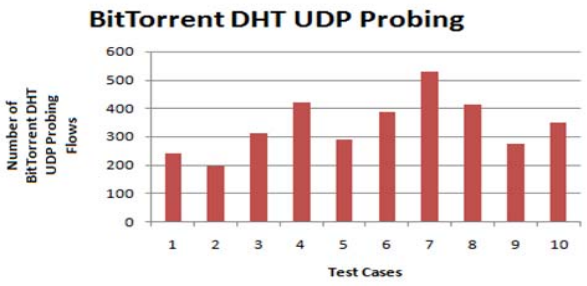

Figure 2 BitTorrent UDP Probing Flows

Another metric that is used to differentiate between BitTorrent and other applications is the packet size of the UDP DHT probing. After extensive analysis of the behavior of BitTorrent clients, the following heuristic was proposed:

The percentage of UDP probing flows that have 1 packet and a size greater than 100 bytes is significantly more than the ones having size less than 100 bytes in BitTorrent UDP probing

By applying this heuristic to a data set where BitTorrent activities were generated in a controlled environment, we can verify the validity of this heuristic by using the port numbers of the BitTorrent clients that were running during the process. The threshold of 100 bytes was selected based on monitoring the DHT UDP probing for duration of 6 hours and documenting the packet size distribution for each case. Table 1 presents the packet size distribution for BitTorrent UDP probing flows that have 1 packet. During the testing, several BitTorrent clients were active and their UDP ports were known.

Table 1 Packet Size Distribution for DHT UDP Probing of BitTorrent

BitTorrent DHT UDP Packet Size Distribution

	Data Set 1			Data Set 2			Data Set 3		
Packet Lengths	Packet Count	Distribution Percentage	Threshold	Packet Count	Distribution Percentage	Threshold	Packet Count	Distribution Percentage	Threshold
0-50	548	2%	13%	496	2%	15%	527	1%	12%
50-75	1921	7%		1354	5%		1854	6%	
75-100	1097	4%		1418	8%		1336	5%	
100-125	7684	28%	87%	5982	23%	85%	6314	22%	88%
150-200	9879	36%		6534	25%		7859	31%	
200-400	6311	23%		10053	38%		9873	35%	

From Table 1 we notice that the distribution of the UDP probing flows for BitTorrent is concentrated in the 100-400 byte range, representing approximately 87% of the total distribution. By exploiting the difference between the packet size distribution for BitTorrent and other applications, we can utilize the packet sizes as a discriminating feature to identify the DHT UDP probing of BitTorrent clients.

B. *Tracker Contacting Heuristic*

As mentioned, some BitTorrent clients use a tracker based system rather than DHT based. Figure 1 illustrates a BitTorrent client contacting a tracker to obtain a list of all available peers using the predefined UDP port in the BitTorrent client`s settings. The presence of a UDP flow outgoing from the suspected BitTorrent port to a destination port equal to 80 is a unique connection pattern for BitTorrent clients as most applications that utilize port 80 operate on TCP rather than UDP.

In order to download a file using BitTorrent, the BitTorrent client contacts all the trackers within the .torrent file using a UDP/destination port 80 connection. The request is carried out using the IP address of the machine running the BitTorrent client and the UDP listening port predefined in the BitTorrent client's settings

C. *Heavy Hitters Heuristic*

We define the term 'Heavy Hitter' according to [7] as a TCP or UDP flow where the consumption of bandwidth is very high. The conditions for a flow to be considered a heavy hitter varies from one application to another and are determined by various features such as the presence of the TCP push flag; as downloads require frequent packet processing. Also The duration of the flow is greater than 5 seconds, the number of incoming bytes is greater than 0.5 MB and the average packet size is greater than approximately half the value of the MTU of both TCP and UDP (around 800 bytes per packet). The reason behind the last condition is due to the collectors setting where a maximum interval is set for each flow. If the flow is divided into multiple flows, the true value of the average packet size cannot be obtained. Another reason for reducing the threshold is that the MTU for Ethernet is 1500 bytes. However, not all links have the same MTU values as it depends on the physical media type and the MTU configuration which may be tweaked to different values. If a packet comes across a link with a MTU value less than its size, the packet is dropped and the sender is notified in order to reduce the packet size and accommodate the MTU value of the link. Furthermore, most BitTorrent clients have shifted towards implementing the uTP protocol. uTP offers a dynamic packet size shaper, where the sizes of the packets are adjusted dynamically according to the status of the link. The initial size of the packet is 1500 bytes. If the connection is slow or congested the packet size is reduced until a suitable value is reached according to the uTP specification [12]. All of these factors can result in smaller packet sizes in BitTorrent file sharing. Usually heavy hitting flows have relatively higher flow parameters such as duration, packet count and packet length due to the nature of their large payloads. We summarize the discussion in this subsection as the following heuristic:

A set of BitTorrent client flows will contain at least one 'heavy hitter' where the byte count, packet count and duration are much higher when compared to other flows.

D. *BitTorrent Traffic Categories*

Some torrent clients such as uTorrent [11] use both TCP and UDP as a transport layer so it is important to differentiate between DHT probing flows and actual data transfer flows when it comes to UDP. Packet size can be used to differentiate UDP probing flows from actual file transfer flows. To achieve this, we selected the UDP flows that are incoming to the known host where both the source and destination ports are greater than 1024, and the size of an incoming packet is greater than 1500 (TCP & UDP MTU) to exclude the small UDP probing flows and only acquire the UDP file transfer flows. UDP probing flows have significantly less byte values than file transfers. We note also that the incoming packets are destined for the suspected BitTorrent UDP ports discovered using the port classification method introduced earlier

As we demonstrated, peers that are connected by TCP and were contacted via DHT can be easily extracted by scanning the records and searching if the TCP flow has a peer UDP counterpart. And for the torrent clients that use UDP as a transfer protocol, we can identify them by extracting all the UDP records that are associated with the suspected port. The problem lies in determining the TCP flows that were not created via DHT. These flows represent the peers that were contacted via the .torrent trackers. To overcome this drawback we will only select the heavy hitters from the tracker flows, by excluding the small flows.

We summarize the discussion in this subsection as the following heuristic:

BitTorrent traffic can be observed in three major segments: (I) Traffic from peers contacted via DHT, (II) UDP traffic from peers contacted via trackers, and (III) TCP traffic from peers contacted via trackers.

E. *Integration of Heuristics*

When applied in a proper order, the above obtained heuristics are in general enough to efficiently classify whether a host and port are BitTorrent related or not.

Figure 3 represents a pseudo code of the classification algorithm summarizing the proposed scheme where all the heuristics are combined and integrated together.

```
List<IPs> = CollectIPs

While (IPs not checked)
{
 List<Ports>= UDP ports with more than 100 request

 While (Ports not checked)
  {
      X = Get Percentage of (UDP FLows, 1 Packet, Bytes>100)

     If (X in Torrent Range)
          {
              1st Flow = UDP FLow with least timestamp
              List<AsFlows> = All Flows associated with suspected port

                If (1st Flow.Dst_Port=80 AND AsFlows.Contains(Heavy Hitters)
                    {
                       Port is BitTorrent Port
                    }
           }

     Else {
            Port is not a BitTorrent Port
          }
      }

  }
```

Figure 3 Pseudo Code for Port Classification Algorithm

Individually, each heuristic is not sufficient to classify the required P2P traffic since some of these features are common between P2P applications and other services. For example, 'heavy hitters' exist in any kind of online communication that requires high bandwidth such as video streaming, downloading, and online gaming. However, when the heavy hitter's heuristic is applied after confirming that a host may be a suspected BitTorrent participant, we can validate whether the host is running a BitTorrent client or not, as BitTorrent clients always have heavy hitting flows associated with them.

Some of these heuristics can be considered as general features that most P2P applications have in common. This includes excessive UDP probing where different P2P applications utilize this technique and are not sufficient to be applied on their own for classification. These features must be combined with stronger heuristics based on the connection patterns of BitTorrent applications to obtain solid results with higher accuracy. By combining general P2P features with application specific heuristics, the accuracy is increased and the reliability of the classification model is enhanced.

V. Testing and Experimentation

In order to validate and measure the efficiency of the obtained heuristics and the chosen features for classification, the algorithm must be applied to a real life example to see how the classification apparatus will perform and obtain concrete results regarding its accuracy and performance.

A. *Testing Data Sets and Description*

Five data sets are used to test the accuracy of the obtained heuristics – see Table 2 for details. The 5 data sets were collected from several locations under varying conditions during different times of the day to obtain a variety of test sets. Each data set was collected from a different location to facilitate testing of different data sets under different conditions.

Dataset one consists of Netflow records acquired by monitoring office traffic from 3-4pm. Multiple hosts utilize the office network with a known host generating P2P activity through the capture duration. Generated traffic included downloads using 5 different BitTorrent file sharing clients: Bitcomet, Azureus, uTorrent, uTorrent Portable and BitTorrent [12]. The trace also included a 6 minute Skype voice call and 5 minute Skype video call.

Dataset two consists of traffic generated by a single known computer at a Carleton University lab. Traffic was captured from 11 am to 1 pm using Wireshark and converted to flow data. Throughout the capture duration, active applications included: (i) a BitTorrent file sharing client (ii) a p2p radio streaming application (iii) a Skype client (iv) a well known Trojan called 'SUS/UnkPacker' was active within a sandbox application in order to limit its threat within a quarantined zone. The SUS/UnkPacker's characteristics are similar to P2P applications where it sends out UDP requests to the attacker using random port numbers.

Dataset three was captured using Wireshark (with subsequent conversion to flow data) from a host computer connected to the Carleton University residential network. Traffic was captured from 12pm to 4pm. Active applications included a BitTorrent file sharing client, a Skype client and 2 P2P T.V streaming applications - Sopcast and PPStream.

Dataset four was captured from the Systems & Computer Engineering Lab at Carleton University. Traffic was monitored using NTOP from 10am to 6pm. The traffic consists of multiple users generating diverse traffic. A known computer utilized the following applications: (i) two BitTorrent file sharing clients (uTorrent & Vuze) and (ii) Skype for a 45 minute video call. The NTOP data was converted to flow data using Wireshark.

Dataset five was captured over a 6 hour period using NTOP at the Carleton University Residence network. Multiple users Internet traffic was captured along with a known host running the following applications: (i) File download via two

BitTorrent clients (Bitcomet and uTorrent) and (ii) Skype – a 20 minute voice call was conducted during the captured session. The NTOP data was converted to flow data using Wireshark.

Table 2 Data Sets Description

	Data Set 1 (8 Host, 1 Hour)		Data Set 2 (1 Host,3 hours)		Data Set 3 (1 Host, 4 Hours)		Data Set 4 (16 Hosts, 8 Hours)		Data Set 5 (22 Hosts, 6 Hours)	
	Bytes	Flows	Bytes	Flows	Bytes	Flows	Bytes	Flows	Bytes	Flows
Total	1.8GBs	17943	5.6GBs	1900345	6.8GBs	2398789	16.3GBs	6587323	14.9GBs	5234974
TCP	1.2GBs	12564	4.7GBs	1634295	3.94GBs	1391294	11.5GBs	4696761	10.2GBs	3612132
UDP	0.59GBs	5379	0.76GBs	266048	2.84GBs	1007491	4.7GBs	1910323	4.61GBs	1622841
Traffic on Known Ports	0.41GBs	4126	4.3GBs	1459193	5.1GBs	1799091	5.8GBs	2371436	4.17GBs	1465792
Traffic on Unknown Ports	1.37GBs	13619	1.2GBs	407216	1.6GBs	564420	10.4GBs	4215886	10.728	3769181
Applications Running	BitTorrent, Skype, HTTP, HTTPS FTP,SNMP, NetBIOS, SMTP		BitTorrent, Skype, Virus, P2P radio, HTTP, HTTPS, SNMP, NetBIOS		BitTorrent, Skype, Sopcast, PPStream, HTTP Video Streaming, HTTPS, SMTP, NetBIOS		BitTorrent, Skype, HTTP, HTTPS, FTP, SNMP, SMTP, NetBIOS, DNS, Bootps		BitTorrent, Skype, HTTP, HTTP Video Streaming, FTP, SMTP, Bootps, NetBIOS, IMAP, HTTPS	
BitTorrent Traffic	1.3GBs 1 Host		1.85GBs 1 Host		0.94GBs 1 Host		9.8 GBs & 0.48 GBs 2 Hosts		9.18 GBs 1 Host	
Skype Calls	6 mins voice call, 5 mins video call		12 mins, 8 mins, 9 mins voice calls		20 mins video call		45 mins video call		20 mins voice call	

B. *BitTorrent False Positives, False Negatives and Accuracy*

Three key metrics were used to evaluate the classification model – False Positives, False Negatives and Accuracy.

The false positive is the probability that non-BitTorrent traffic counted in bytes is classified as BitTorrent traffic.

The false negative is the probability that BitTorrent traffic counted in bytes is classified as non-BitTorrent traffic.

Equations 1 and 2 summarize the values of the false positive probability and the false negative probabilities.

$$\alpha = \frac{x}{T} \quad (1) \qquad \beta = \frac{y}{T} \quad (2)$$

Where T is equal to the total number of download bytes via BitTorrent, x is equal to the non BitTorrent bytes classified as BitTorrent bytes and y is equal to the BitTorrent bytes classified as non BitTorrent bytes. While α and β are equal to the false positive probability and the false negative probability respectively.

The accuracy is defined as in Equation 3. The presented method calculates the byte wise accuracy, where π denotes the byte wise accuracy.

$$\pi = \frac{T-(x+y)}{T} \quad (3)$$

denotes the byte wise accuracy.

The importance of the byte wise accuracy in BitTorrent classification is displayed by Gossett et al in [8] where they are mainly concerned with the byte wise accuracy of their BitTorrent identification apparatus.

C. *BitTorrent Testing Results*

Table 3 represents the accuracy, false positives, and false negatives of the proposed scheme when applied to each of the 5 mentioned data sets.

Table 3 BitTorrent Classification Scheme Results

Data Sets	MB Accuracy	False Positives	False Negatives
Set 1	95.1%	1.6%	3.3%
Set 2	91.8%	2.9%	5.3%
Set 3	91.3%	5.6%	3.1%
Set 4	94.8%	1.2%	4%
Set 5	95.4%	0.9%	3.7%

In order to validate the results of the proposed scheme, the accuracy results are compared to 2 approaches suggested in other researches. The first method is a host behavior approach proposed by Basher et al. in [7]. The suggested approach utilizes the five features for identifying BitTorrent traffic. A detailed description of this scheme can be found in [7].

The second method selected for comparison is a machine learning approach proposed by Yuan et al. in [9]. The suggested approach utilizes SVM with a RBF kernel function for extracting BitTorrent flows. A detailed description of this scheme can be found in [8].

Figure 4 displays the accuracy of the proposed scheme when compared to the accuracy of the SVM approach suggested in [9] by Yuan et al. and the host behavior approach suggested in [7] by Basher et al.

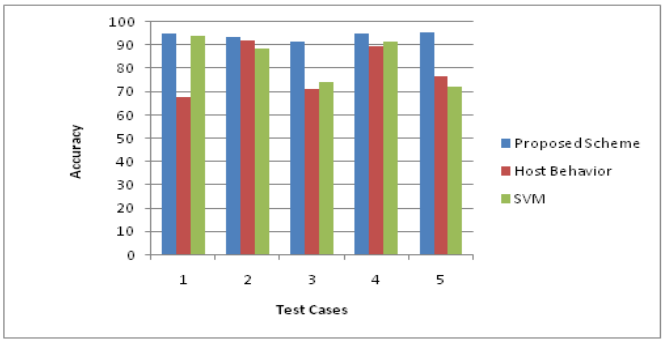

Figure 4 Accuracy Histogram of the 3 Approaches

The approaches suggested in [7] and [9] produce strong results when applied to data sets that do not contain challenging background traffic. The accuracy is significantly decreased when these methods are applied to data sets that contain BitTorrent traffic overlapped with applications that generate similar flows and behave in the same fashion.

Our proposed scheme outperformed the approaches suggested in [7] and [9] in all of the 5 data sets due to the following:

1. P2P T.V. streaming and radio applications utilize UDP probing in order to contact peers that are tuned in to the same channel by using a random UDP listening port predefined in the software's settings. The average bytes per packet of the data transfer flows generated by P2P T.V. streaming and radio application are within the 200-1200 according to [10]. If a host is concurrently running BitTorrent and P2P T.V. streaming applications, the TCP flows within the range of 200-1200 bytes per packet will contain a mixture of both P2P T.V. streaming flows and BitTorrent file transfer flows. The scheme proposed in this paper as well as both the approaches suggested in [7] and [9] cannot distinguish between these flows as they share similar flow features, leading to a high value of false positives. The proposed scheme outperformed the approaches in [7] and [9] due to the average bytes per packet condition where flows are

labeled as BitTorrent flows if the average bytes per packet value is greater than 800 bytes. This threshold does separate between BitTorrent and P2P T.V. streaming and radio flows; nonetheless it ensures capturing the majority of BitTorrent flows and only a small portion of the P2P T.V and radio. The accuracy of the proposed approaches in [7] and [9] is severely hindered due to the similarity of BitTorrent flows and P2P T.V. streaming flows (data set 3)

2. The SUS/Unkpacker Trojan and other similar viruses utilize a set of random UDP ports in order to communicate with the attacker using port 80. This may be similar to other P2P applications which causes mislabeling the flows generated from the Trojan as BitTorrent related flows. In the proposed scheme, a host must satisfy a collection of heuristics in a sequential manner in order to be considered a BitTorrent participant. The flows generated by viruses and Trojans are not mislabeled since they do not satisfy all of the heuristics and conditions. The proposed approaches in [7] and [9] may mislabel virus and Trojan flows as P2P flows due to the similarity between their flow features (data set 2).

3. P2P clients are not the only applications that use ports greater than 1024. Many other services such as Adobe flash video and Google Talk use port numbers that are greater than 1024 and are registered as official ports on the IANA port list. The authors in [7] state that in order for a flow to be classified as a P2P related flow, the source and destination port number must be greater than 1024. This assumption results in a high value of false positives when BitTorrent traffic overlaps with traffic generated from other services that use ports greater in 1024. In data set 5, a HTTP streaming page was open which uses the Adobe flash video port (1935). The accuracy of both selected approaches is heavily affected and produced inaccurate results when compared to the approach suggested in this paper as displayed in Table 3 for data set 5.

4. The host behavior approach proposed in [7] operates on the assumption that all BitTorrent file transfers take place only via TCP. This assumption leads to a very large value of false negatives when this approach is applied to traces generated by the new generation of BitTorrent clients using uTP where all data transfers use UDP. The approach presented in this paper and the SVM approach in [9] are not affected by the transport layer protocol of BitTorrent clients as displayed in Table 3 where 4 out of 5 data sets had BitTorrent clients using UDP for file transfers.

VI. Conclusions

In this paper, a set of heuristics and classification schemes based on the discriminating attributes and behavioral nature of different P2P applications are presented and integrated together in order to successfully identify the targeted BitTorrent applications. The proposed scheme relies solely on information that is captured by using any traditional Netflow collector which is the standard tool used to collect Internet traffic traces within a network. These heuristics have been integrated and combined together to produce a classification model capable of identifying BitTorrent flows with high accuracy.

The presented classification scheme has been tested on real life data sets where P2P and non P2P activities were conducted. The accuracy output for identifying BitTorrent activity was very high ranging from 91.3-95.4 %, in the case of byte comparison. The information regarding the total amount of downloaded bytes is very useful for network analysts and administrators as it can be used for bandwidth management and allocation, and in identifying 'bandwidth hogs' that cause congestion and deteriorate network performance.

The results of the proposed scheme have proven to be superior to other existing approaches in terms of accurately identifying BitTorrent flows within Netflow traces.

References

[1] T. Karagiannis , K. Konstantina, and A. Papagiannaki. " BLINC: Multilevel traffic classification in the dark", *In Proc. SIGCOMM'05*, Philadelphia, PA, USA, 2005.

[2] A. Finamore, M. Mellia, , M. Meo and M. Rossi. "KISS: Stochastic Packet Inspection" *in Proc. Traffic Measurement and Analysis (TMA) Workshop at IFIP Networking 2009*, Aachen, Germany,May 2009.

[3] W. Rui, L.Yang, Y.Yuexiang, "Solving the app-level classification problem of P2P traffic via optimized support vector machines" . *In Proc. Sixth International Conference on Intelligent Systems Design and Applications (ISDA '06)*: Vol 2, Oct 16-18, 2006, Jinan, China.

[4] S. Zhang, C. Gu, and X. Xue. "Encrypted Internet Traffic Classification Method based on Host Behavior", *JDCTA: International Journal of Digital Content Technology and its Applications*, Vol. 5, No. 3, pp. 167 ~ 174, 2011.

[5] B. Cohen, "BitTorrent Protocol Specification", *Retrieved: 21 October 2011*, http://bittorrent.org/beps/bep_0003.html

[6] Cisco, "Netflow", *Retrieved 4 February 2012*, http://www.cisco.com/go/netflow

[7] N. Basher, A. Mahanti, C. Willamson, M. Arlitt, "A Comparative Analysis of Web and Peer-to-Peer Traffic", *In Proc. WWW'08*, pages 287–296,Beijing, China, April 2008.

[8] A. Gossett , I. Papapanagiotou, and M. Devetsikiotis. "An apparatus for P2P classification in Netflow traces", *In Proc GLOBECOM Workshops (GC Wkshps)*, 2010 IEEE , vol., no., pp.1361-1366, 6-10, Miami, Florida, U.S.A, December 2010.

[9] R. Yuan, Z. Li, X. Guan, L. Xu. "An SVM-based machine learning method for accurate Internet traffic classification", *In Proc. Information System Frontier 12(2):149–156,Hingham*, MA, U.S.A, 2010.

[10] T. Silverston, O. Fourmaux, A. Botta, A. Dainotti, A. Pescap`e, G. Ventre, and K. Salamatian "Traffic analysis of Peer-to-Peer IP T.V. communities", *Computer Networks, vol. 53, no. 4*, pp. 470–484, New York, NY, U.S.A, 2009.

[11] BitTorrent Inc.,"uTorrent", *Retrieved: 26 October 2011,* http://www.utorrent.com

[12] B. Cohen. "BitTorrent Client", Retrieved: 28 April 2012, http://www.bittorrent.com/