Flow-Based Identification of P2P Heavy-Hitters

Arno Wagner* Thomas Dübendorfer* Lukas Hämmerle† Bernhard Plattner*
*Communication Systems Laboratory, Swiss Federal Institute of Technology Zurich,
{wagner, plattner}@tik.ee.ethz.ch, thomas@duebendorfer.ch
† SWITCH, Switzerland, haemmerle@switch.ch

Abstract

One major new and often not welcome source of Internet traffic is P2P filesharing traffic. Banning P2P usage is not always possible or enforcible, especially in a university environment. A more restrained approach allows P2P usage, but limits the available bandwidth. This approach fails when users start to use non-default ports for the client software. We have developed the PeerTracker algorithm that allows detection of running P2P clients from NetFlow data in near real-time. The algorithm is especially suitable to identify clients that generate large amounts of traffic and can easily be used to find P2P heavy-hitters. A prototype system based on the PeerTracker algorithm is currently used by the network operations staff at the Swiss Federal Institute of Technology Zurich. We present measurements done on a medium sized Internet backbone and discuss accuracy issues, as well as possibilities and results from validation of the detection algorithm by direct polling in real-time.

1. Introduction

P2P filesharing is a newer Internet application that generates large amounts of traffic. It seems even to be one of the driving factors for home-users to get broadband Internet connections. It also has become a significant factor in the total Internet bandwidth usage by universities and other organisations. While in some environments a complete ban on P2P filesharing can be a solution, this gets more and more difficult as legitimate uses grow. The Swiss Federal Institute of Technology at Zurich (ETH Zurich) has adopted an approach of allowing P2P filesharing, but with limited bandwidth. The default ports of the most popular P2P filesharing applications are shaped to a combined maximum Bandwidth of 10Mbit/s. There is a relatively small number of "heavy hitters" that consume a large share of the overall P2P bandwidth and avoid the use of default ports and hence the bandwidth limitations. In fast network connections, such as the gigabit ETH Internet connectivity, it is difficult to identify and monitor P2P users for their bandwidth consumption. If heavy hitters can be identified, they can be warned to reduce their bandwidth usage or, if that does prove ineffective, special filters or administrative action can be used against them. In this way P2P traffic can be reduced without having to impose drastic restrictions on a larger user population.

To this end, we have developed the *PeerTracker* algorithm that identifies P2P users based on Cisco NetFlow [4]. It determines hosts participating in the most common P2P networks and detects which port setting they use. This information can then be used to determine P2P bandwidth usage by the identified hosts. We present the PeerTracker algorithm as well as results from measurements done in the SWITCH [3] network, a medium sized Internet backbone in Switzerland. We discuss detection accuracy issues and give the results of work done on validation of the PeerTracker algorithm by real-time polling of identified P2P hosts. Note that the PeerTracker cannot identify which files are shared.

A prototypical implementation of the PeerTracker algorithm, fitted with a web-interface, is currently in use at the central network services of ETH Zurich in a monitoring-only set-up for hosts in the ETH Zurich network. A software release under the GPL is planned.

The paper is organised as follows: Section 2 gives an overview of the data this work is based on and a short description of the DDoSVax project that this work is part of. Section 3 briefly discusses current trends in P2P filesharing. Section 4 presents the PeerTracker Algorithm followed by PeerTracker measurements in Section 5. Section 6 discusses validation work. Section 7 has a brief survey over related work and the paper finishes in section 8 with a conclusion.

2. Backbone Flow-Level Traces

2.1. DDoSVax project

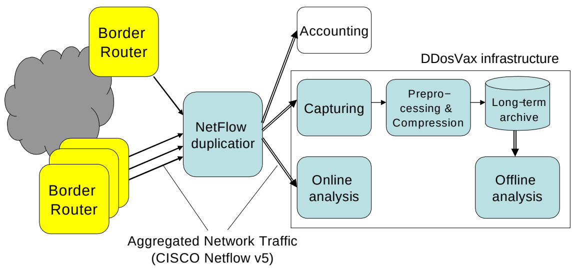

Figure 1. The DDoSVax infrastructure

The DDoSVax[5] project maintains a large archive NetFlow[4] data which is provided by the four border gateway routers of the medium-sized backbone AS559 network operated by SWITCH[3]. This network connects all Swiss universities, universities of applied sciences and some research institutes.

Figure 1 shows the overall set-up. The SWITCH IP address range contains about 2.2 million addresses, which approximately corresponds to a /11 network. In 2003, SWITCH carried around 5% of all Swiss Internet traffic [10]. In 2004, we captured on average 60 million NetFlow records per hour, which is the full, non-sampled number of flows seen by the SWITCH border routers. Our data repository contains the SWITCH traffic data starting from the beginning of 2003 to the present.

2.2. NetFlow Data

Cisco's NetFlow [4] format version 5 that we use defines a "flow" as a *uni*directional stream of packets from one host to another. A flow is reported as a tuple of source/destination IP address, source/destination port, IP protocol type (i.e. TCP, UDP, other), packets/flow, number of network layer bytes/flow, time of first/last packet in this flow, routing-specific and other parameters.

For example for each TCP connection crossing the backbone border routers at least two flows are reported, one for each direction. Due to asymmetrical routing, it is possible that the two unidirectional flows forming one connection are reported by different routers. Also single packets, such as the single or double SYN from a failed TCP connection attempts create one flow. In addition the router sometimes accumulate several UDP, ICMP or other packets into one flow record.

3. P2P Filesharing

The P2P community has developed and continues to develop many different and incompatible file sharing network technologies at a fast pace. As the P2P users are mostly interested in getting access to a large selection of interesting files, P2P clients that support simultaneous file sharing in multiple networks were developed that shielded the user somewhat from the idiosyncrasies of the underlying networks. Figure 2 gives an overview of the interrelationship of P2P clients and P2P networks as of mid-2004.

Our PeerTracker can track the population and detect heavy hitters of the eDonkey, Overnet, Kademlia (eMule), Gnutella, FastTrack, and BitTorrent P2P networks.

4. PeerTracker: Algorithm

The measurements were done with an UPFrame [6] plug-in. UPFrame is a plug-in framework for processing UDP packets. It is well suited for NetFlow data processing in real-time, since NetFlow data is exported from routers as a stream of UDP packets. The P2P traffic itself can be TCP or UDP. We use the term "default port of a P2P system" to also include the choice of TCP or UDP. The plug-in collects data about the active hosts in the SWITCH network and the kind of traffic they generate. This data form the basis of the peer detection.

Figure 3 shows the state diagram for each individual peer in the PeerTracker. When a network connection is detected each endpoint host becomes a *candidate peer*. A *candidate peer* that has more P2P traffic it becomes an *active peer* and is reported as active. Otherwise is becomes a *non-peer* after it has had no P2P traffic for a probation period (900 seconds) and is deleted. Each *active peer* is monitored for further P2P activity. After a maximum time without P2P traffic (600 seconds) it becomes a *dead peer*. Each *dead peer* is still monitored for P2P activity but not reported as active anymore. When a *dead peer* has P2P activity, it becomes active again. After a second time interval, the maximum afterlife (1 hour) without P2P activity a *dead peer* is considered gone and is deleted from the internal state of the PeerTracker.

The decision whether a network flow is a P2P flow is made based on the port information collected. If a P2P client uses a non-default listening port (e.g. in order to circumvent traffic shaping) the peer still will communicate with other peers on using the default port(s) from time to time. The last 100 local and remote ports (TCP and UDP) that were used to communicate are stored

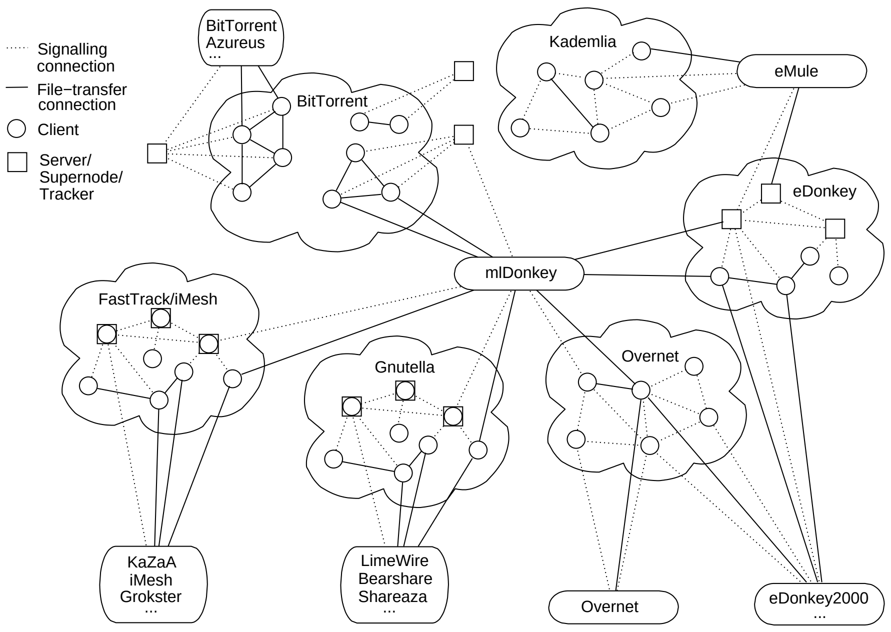

Figure 2. P2P Network and Client Overview

for every observed host, together with the amount of observed traffic on the individual ports. Traffic with one or both ports not in the range 1024-30000 (TCP and UDP) is ignored, since we found that most P2P traffic uses these ports. With reasonable threshold values on traffic amount (different for host within the SWITCH network and hosts outside) the most used local and remote ports allow the determination which P2P network a specific host participates in. This is done at the end of every measurement interval (900 seconds). Although some hosts can be part of several P2P networks (see Figure 2) they are accounted only to one P2P network, but can be in different networks in different measurement intervals.

For the P2P traffic estimation we determine a lower and an upper bound. The lower bound is all P2P traffic were at least one side uses a default port. The base assumption is that the main P2P default ports are well known enough by now and no other application has reason to use them.

The upper bound also counts all traffic were source and destination ports are above 1023 and one side was identified as P2P host. The effective P2P traffic is expected to be between these two bounds, and likely closer to the upper bound, because in particular P2P heavy-hitters rarely run other applications that cause large amounts of traffic with port numbers above 1023 on both sides. Typical non-P2P applications with port numbers on both sides larger than 1023 are audio and video streaming and online gaming, all of which do not run well on hosts that also run a P2P client.

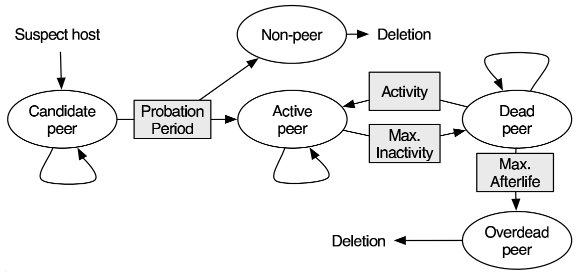

Figure 3. PeerTracker hosts state diagram

5. PeerTracker: Measurements

Due to traffic encryption and traffic hiding techniques used by some current P2P systems, the accurate identification of P2P traffic is difficult, even if packet inspection methods are used. Nevertheless, our NetFlow based approach can provide good estimations for the effective P2P traffic, even for networks with gigabit links that could hardly be analysed with packet inspection methods.

Identification of peers and their traffic is especially difficult if they have a low activity. This is an issue for all two-tier systems in which ordinary peers mainly communicate with a super peer and have few file transfers. Peers from one-tier systems like Overnet can be identified better because they communicate with many other peers even if no file transfers are in progress.

In a 30 day measurement (August 2004) a total of 2982 P2P hosts (unique IPs) could be identified in the SWITCH network, 53.4% of which were DHCP or VPN users. It could be shown that the number of web users is proportional to the number of P2P users, since both numbers decreased by 15% during the holiday season in a second measurement.

The P2P clients BitTorrent and Gnutella had a much higher fluctuation in number of active users than other P2P clients like Overnet. 160 peers (5.3%) were active longer than 10 days.

The amount of FastTrack peers and traffic found was surprisingly low in spite of the fact that FastTrack was at the time the measurements where made still he P2P network with the most users. The reason for this could be that FastTrack is not very popular among the SWITCH P2P users because of fake files, the RIAA P2P user prosecution or the universities traffic filtering measures. Another reason could be that FastTrack is the most difficult to identify network because its default port usage is very low in comparison to the other observed P2P systems as can be seen for TCP in Table 1. Note that the default port usage for a specific candidate peer is determined by how often the specific default port was seen in the last 100 port numbers (>1023) seen in connections from and to the candidate peer. Once a peer has been identified as belonging to a specific P2P network, this number is an estimation of how many peers in that network use the default port. The estimation is most accurate if the specific peer has no other network activity besides P2P traffic and degrades otherwise. Still, we believe these estimates are reasonable.

P2P System	Default port usage
BitTorrent	70.0 %
FastTrack	8.3 %
Gnutella	58.6 %
eDonkey	55.6 %
Overnet	83.9 %
Kademlia	66.6 %

Table 1. P2P ports, 8 day PeerTracker measurement end of 2004

P2P traffic in the SWITCH network is quite substantial. The lower bound for P2P traffic (stateless P2P default port identification) significantly lower than the upper bound (all traffic from PeerTracker identified hosts) for all observed P2P systems (Table 2), which means that quite some P2P traffic cannot be accurately estimated using only a stateless P2P default port method.

BitTorrent P2P users cause about as much traffic as eDonkey, Overnet and Kademlia users together, as can be seen in Figure 2. All peers of the SWITCH network generate 1.6 times more traffic to non-SWITCH hosts than incoming traffic, thus making the SWITCH network a content provider.

This is probably due to the fast Internet connection most SWITCH users have and the traffic shaping mechanisms that some universities in the SWITCH network use. Users within the university network hope to evade the traffic limiting by using non-default listening ports. However, this has no influence on their download rates because most external users use default listening ports. On the contrary, external users can download from internal users without being influenced by the traffic limitations because neither source nor destination port for these downloads are default ports.

6. Result Validation

The PeerTracker tries to identify P2P hosts and the used P2P network only on network flows seen, but makes no attempt to check its results in any other way. It is completely invisible on the network. There are two possible failure modes: *False positives* are hosts that the PeerTracker reports as having a P2P client running, while in fact they do not. *False negatives* are hosts that run a P2P client but are not identified by the PeerTracker.

It is difficult to identify false negatives. From manual examination of the flow-level data and comparison with the PeerTracker output we found that while there are unidentified P2P clients, these hosts have only very limited P2P activity and do not contribute significantly to the overall traffic. Manual examination involved looking at all flows generated by candidate peers that did not reach the threshold values. The typical situation found was that the suspect P2P traffic was clearly identifiable as P2P traffic, but low in volume.

In order to identify false positives, we have implemented an experimental extension to the PeerTracker that tries to determine whether hosts identified by the PeerTracker are actually running the indicated P2P client by actively polling them over the network. Table 3 gives a short overview of the polling methods used. In practice a three-staged approach was found to be most effective:

1. Use an ICMP "echo request" to determine whether an identified host is reachable.

2. Use a TCP "connection attempt" (a.k.a. "TCP-ping") on the identified P2P ports to determine whether the ports can be contacted.

3. Use the appropriate application-layer handshake from from Table 3 to determine whether the Peer-

P2P System	P2P lower bound	P2P upper bound
BitTorrent	55.4 Mbit/s (12.2 %)	90.1 Mbit/s (19.9 %)
FastTrack	1.8 Mbit/s (0.4 %)	12.3 Mbit/s (2.7 %)
Gnutella	5.1 Mbit/s (1.1 %)	10.7 Mbit/s (2.4 %)
eDonkey, Overnet, Kademlia	47.7 Mbit/s (10.5 %)	82.1 Mbit/s (18.1 %)
Total P2P	110.0 Mbit/s (24.4 %)	195.2 Mbit/s (43.1 %)

Table 2. P2P traffic bounds and percentage of total SWITCH traffic (August 2004)

P2P System	Polling method	
FastTrack	Request:	`GET /.files HTTP/1.0`
	Response:	`HTTP 1.0 403 Forbidden` <number 1> <number 2>
	or	`HTTP/1.0 404 Not Found/nX-Kazaa-`<username>
Gnutella	Request:	`GNUTELLA CONNECT/`<version>
	Response:	`Gnutella` <status>
eDonkey, Overnet, Kademlia	Request:	Binary: 0xE3 <length> 0x01 0x10 <MD4 hash> <ID> <port>
	Response:	Binary: 0xE3 …
eMule	Same as eDonkey, but replace initial byte with 0xC5.	
BitTorrent	Unsolved. Seems to need knowledge of a shared file on the target peer.	

Table 3. Polling methods for different P2P clients (TCP, to configured port)

P2P System	TCP	P2P-client
eDonkey, Overnet, Kademlia	50%	41%
Gnutella	53%	30%
FastTrack	51%	41%
Total	51%	38%

Table 4. Positive polling answers

Tracker identified P2P client is really running on the target host.

The polling was done with up to 200 parallel threads to reduce the effects of unreachable hosts. The time needed to verify the peer sets did not exceed three minutes, while the delivery and processing delay of the NetFlow data in the PeerTracker was in the order of a few minutes.

The results of a representative measurement from February 2005 can be found in Table 4. The ICMP results are not listed, since 99% of the hosts were ICMP reachable. From the table it can be seen that roughly half of the identified hosts are not reachable via TCP, likely due to Network Address Translation (NAT) and firewalls that prevent connections initiated by outside hosts.

The unsuccessful polling attempts present an upper limit on the number of false positives. The main reasons for unsuccessful P2P client polling identified in a manual analysis are that the PeerTracker sometimes reports the wrong P2P network for a host, that especially Gnutella hosts answer in a variety of ways, some not expected by the polling code, and misdetection by the PeerTracker algorithm.

7. Related Work

While there are numerous measurements studies that use packet inspection [14, 8, 13, 9] for traffic identification, recently some have been published that use flow-level heuristics [7, 16, 11]. Flow-level identification is more robust against protocol changes of P2P clients and less resource demanding. It is relatively unproblematic with regard to privacy issues. These advantages come at the price of smaller accuracy and they might be circumvented by more sophisticated camouflage techniques in the future. Flow-based approaches are also slower, since they need a certain number of flows from a host to determine whether it runs a P2P client or not.

We are aware of the following flow-based efforts:

- In [15] signalling and download traffic was measured in a large ISP network using state-less default port number detection. Considered P2P networks were FastTrack, Gnutella and Direct Connect.
- An interesting approach is presented in [11]. The idea is to relate flows to each other according to source and destination port numbers using a flow relation map heuristic with priorities and SYN/ACKs to identify listening port. The method is not described very detailed though.

- In a campus network and the network of a research institute with about 2200 students and researchers, captured the first 64 Bytes [16]. The approach used for identification is, that unknown flows are induced by flows of known traffic. They used a time window to find induced flows. The higher the window the more ”identified” traffic resulted. They also stated, that false positives can’t be recognised without checking the payload.

- Flow measurements in the backbone of a large ISP were done in [7] for May 2002 and January 2003. The researchers determined the server port using the IANA[2] port numbers and the more detailed Graffiti [1] port table, giving precedence to well-known ports. Unclassified traffic was grouped in a ”TCP-big” class that includes flows with more than 100 KB data transmitted in less than 30 minutes. They noted that identified P2P traffic decreased in January 2003 but the TCP-big traffic dramatically increased (10.5 times for outgoing and 6 times for incoming directions). Moreover they found that the TCP-big traffic had a strong correlation with P2P traffic.

8. Conclusions

We presented an efficient P2P client detection, classification and population tracking algorithm that uses flow-level traffic information exported by Internet routers. It is well suited to find and track heavy-hitters of the eDonkey, Overnet, Kademlia (eMule), Gnutella, FastTrack, and BitTorrent P2P networks. We also validated detected peers by an application-level polling. Our results confirmed a good lower accuracy bound that is well suited for P2P heavy hitter detection. However, it is not optimally suited to detect low traffic P2P nodes. A validation of BitTorrent clients was not possible due to the specifics of this network. In addition we stated measurement results obtained with the PeerTracker and observations made during the validation efforts.

The PeerTracker algorithm was implemented as an UPFrame [6] plug-in and has an interactive web-based interface that shows the current list of the top P2P traffic receivers and senders. The PeerTracker tool is used since early 2005 by the network services group at ETH Zurich to detect P2P clients that do not respect the ETH network usage rules that allow P2P activity only on default ports. We also used NetFlow data of all border routers of a medium-sized Swiss Internet backbone (SWITCH, AS559) for stress tests. Our tool could process 60 million flow records per hour (i.e. the complete SWITCH traffic) in real-time on a commodity computer. We currently plan a release of the PeerTracker tool under the GPL.

As further work, we plan to adapt the PeerTracker to new P2P networks and to changes in existing P2P networks. We also plan to improve it further for easy use by network administrators.

References

[1] Graffiti. `http://www.graffiti.com/services/` (July 2004).

[2] IANA. `http://www.iana.com/assignments/port-numbers/services/` (July 2004).

[3] The Swiss Education & Research Network. `http://www.switch.ch`.

[4] Cisco. White Paper: NetFlow Services and Applications. `http://www.cisco.com/warp/public/cc/pd/iosw/ioft/neflct/tech/napps_wp.htm`, 2002.

[5] DDoSVax. `http://www.tik.ee.ethz.ch/~ddosvax/`.

[6] T. Dübendorfer, A. Wagner, and C. Schlegel. UPFrame: UDP Processing Framework. `http://www.tik.ee.ethz.ch/~ddosvax/upframe/` (July 2004).

[7] A. Gerber, J. Houle, H. Nguyen, M. Roughan, and S. Sen. P2P, The Gorilla in the Cable. Technical report, AT&T Labs - Research, June 2004.

[8] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan. Measurement, Modeling, and Analysis of a Peer-to-Peer File-Sharing Workload. Technical report, October 2003.

[9] T. Karagiannis, A. Broido, and M. Faloutsos. File-sharing in the Internet: A characterization of P2P traffic in the backbone. Technical report, University of California, Riverside Department of Computer Science, November 2003.

[10] O. Müller, D. Graf, A. Oppermann, and H. Weibel. Swiss Internet Analysis. `http://www.swiss-internet-analysis.org/`, 2003.

[11] J.-J. K. Myung-Sup Kim and J. W. Hong. Towards Peer-to-Peer Traffic Analysis. Technical report, POSTECH, Korea, October 2003.

[12] M. Ripeanu. Peer-to-Peer Architecture Case Study: Gnutella Network. In *Proceedings of the First International Conference on Peer-to-Peer Computing (P2P01)*. IEEE, 2001.

[13] S. Saroiu, K. P. Gummadi, R. J. Dunn, S. D. Gribble, and H. M. Levy. Analysis of Internet Content Delivery Systems. Technical report, University of Washington, December 2002.

[14] S. Sen and J. Wang. Accurate, Scalable In-Network Identification of P2P Traffic Using Application Signatures. Technical report.

[15] S. Sen and J. Wang. Analyzing peer-to-peer traffic across large networks. Technical report, AT&T Labs - Research, November 2002.

[16] R. van de Meent and A. Pras. Assessing Unknown Network Traffic. Technical report, University of Twente, Enschede, October 2003.