POLITECNICO DI TORINO
Repository ISTITUZIONALE

Inferring undesirable behavior from P2P traffic analysis

(Article begins on next page)

19 May 2024

Inferring Undesirable Behavior from P2P Traffic Analysis

Ruben D. Torres, Mohammad Y. Hajjat,
Sanjay G. Rao
Purdue University
{rtorresg,hajjat,sanjay}@purdue.edu

Marco Mellia, Maurizio M. Munafò
Politecnico di Torino
{mellia,munafo}@tlc.polito.it

ABSTRACT

While peer-to-peer (P2P) systems have emerged in popularity in recent years, their large-scale and complexity make them difficult to reason about. In this paper, we argue that systematic analysis of traffic characteristics of P2P systems can reveal a wealth of information about their behavior, and highlight potential undesirable activities that such systems may exhibit. As a first step to this end, we present an offline and semi-automated approach to detect undesirable behavior. Our analysis is applied on real traffic traces collected from a Point-of-Presence (PoP) of a national-wide ISP in which over 70% of the total traffic is due to eMule [19], a popular P2P file-sharing system. Flow-level measurements are aggregated into "samples" referring to the activity of each host during a time interval. We then employ a clustering technique to automatically and coarsely identify similar behavior across samples, and extensively use domain knowledge to interpret and analyze the resulting clusters. Our analysis shows several examples of undesirable behavior including evidence of DDoS attacks exploiting live P2P clients, significant amounts of unwanted traffic that may harm network performance, and instances where the performance of participating peers may be subverted due to maliciously deployed servers. Identification of such patterns can benefit network operators, P2P system developers, and actual end-users.

Categories and Subject Descriptors

C.2.3 [**Network Operations**]: Network Monitoring; C.2.4 [**Distributed Systems**]: Distributed Applications

General Terms

Measurement, Performance, Security

Keywords

P2P, Measurement

1. INTRODUCTION

Peer-to-peer (P2P) systems have rapidly emerged in popularity in the last few years, and they have matured to the point we have recently seen several commercial offerings, including file sharing, VoIP and multimedia applications. Recent studies [13] indicate that over 60% of network traffic is dominated by peer-to-peer systems, and their emergence has drastically affected traffic usage and capacity engineering.

With the growth of P2P systems, many of which involve millions of hosts, and complex interactions between participating peers, it becomes critical to monitor these systems, and to ensure they are behaving as intended. Indeed, several reports are emerging about potential vulnerabilities in these systems either due to implementation bugs, or design flaws [4, 7, 9, 18, 30]. The behavior may be undesirable either from the perspective of the performance of the system, or in terms of unwanted traffic (malicious or otherwise) generated by the systems.

Detecting undesirable behavior is of interest to network operators, P2P system developers, and actual P2P end-users. Network operators may wish to identify causes for large traffic consumption, or they may want to optimize P2P traffic delivery, e.g., limit traffic peering costs. Knowledge of undesirable behavior and its causes can aid P2P system developers in augmenting the design of the systems. Finally, end-users seek to ensure that their host is not being exploited for malicious purposes, and care about application performance.

While the ultimate objective is automated identification of undesirable behavior, there is limited understanding in the community today on the patterns of undesirable behavior that P2P systems may exhibit, and the prevalence and seriousness of such behavior in real networks. Our primary contribution in this paper is to create such understanding by systematically analyzing real traffic traces collected from a Point-of-Presence (PoP) of a nation-wide ISP. In the ISP we consider, 70% (95%) of inbound (outbound) traffic is due to eMule [19], a popular file-sharing system, and the associated Kad network, one of the largest DHT-based deployments. We analyze a 25 hour trace, comprising about 2TB of data. Another interesting aspect of this dataset is the use of a modified Kad system - called KadU - within the ISP network that was optimized by a large community of ISP users to exploit the peculiarities of the ISP architecture.

One of the key challenges we faced in our study is that it is hard to distinguish undesirable behavior from normal usage in a completely automated fashion, given the intrinsic heterogeneity of P2P traffic, and given there are few assumptions that can be made about the underlying nature of undesirable behavior in P2P systems. Undesirable behavior can be predominant, given it can arise due to flaws in the design or implementation of the system. This complicates the use of automated techniques widely adopted in the detection of anomalies of general network traffic such as [8, 25, 26], which assume most data-points are normal, and which identify

anomalous behavior by detecting sudden and significant deviations from normal values.

Consequently, our methodology employs a combination of data-mining techniques, and manual inspection through domain knowledge. The behavior of individual hosts is characterized with respect to a wide range of metrics over multiple time samples. The set of metrics chosen is broad, since there is limited a priori knowledge of the types of undesirable behavior that may be present. Standard clustering algorithms are utilized to identify homogeneous groups of samples. Finally, the clusters are manually inspected, correlated and interpreted using domain knowledge to identify undesirable patterns.

Our methodology reveals several interesting findings, both confirming already known types of undesirable behavior of P2P systems, as well as highlighting new undesirable patterns. Some of our most relevant findings include:

• We show evidence of real DDoS attacks being conducted on DNS servers by exploiting P2P systems.

• We show that stale membership information and presence of hosts behind Network Address Translators (NATs) can result in the failure of 15% of TCP connections and 18% of UDP flows incoming to the PoP. This may hurt peer performance, introduce unnecessary traffic, and may waste significant computation resources of state-full network devices, such as firewalls or NAT boxes.

• We show instances where maliciously deployed servers can subvert the performance of hosts participating in the P2P system.

While much of our analysis is conducted with Kad, and KadU given their predominant usage in the ISP network, we extended the analysis to consider other popular P2P systems, BitTorrent [12] and DC++ [17]. Given these systems are not widely used in the network, our analysis is conducted on a separate one-week long trace in which sufficient data samples are present. Our analysis exposes undesirable behavior in these systems as well. Overall, our results shed light on the prevalence and impact of undesirable behavior in P2P systems, and demonstrate the potential of a systematic traffic-analysis approach in uncovering such behavior.

2. METHODOLOGY OVERVIEW

The methodology we propose in this paper seeks to infer undesirable behavior of P2P systems, by identifying possibly atypical traffic. Our methodology may be viewed as consisting of the following steps, as depicted in Figure 1.

In our analysis, we assume that data is collected at the edge of a network, for instance at the edge of an enterprise network. We assume that flow-level records of all UDP and TCP data traversing the network edge is available. While well-known flow level loggers such as Cisco NetFlow [14] can be used to generate flow records, a key requirement for our study is that flow-level records are classified based on application, and flows corresponding to the P2P system of interest are clearly identifiable. Several techniques have been developed for classification of traffic as P2P (for instance [10, 15, 16, 21, 23, 24, 29]), which may be leveraged. In this paper, we use datasets where traffic is classified using Tstat [28], a passive sniffer with deep packet inspection (DPI) capabilities. Raw packets are sniffed from the link of interest, flows are passively rebuilt, and classification is performed in an online fashion based on the application layer payload.[1]

While we begin with per-flow measurement information, we aggregate this information to capture per-host behavior. We conduct

[1]In our context, encrypted payload has not been a major issue, but in general one approach to deal with it is using behavioral classifiers.
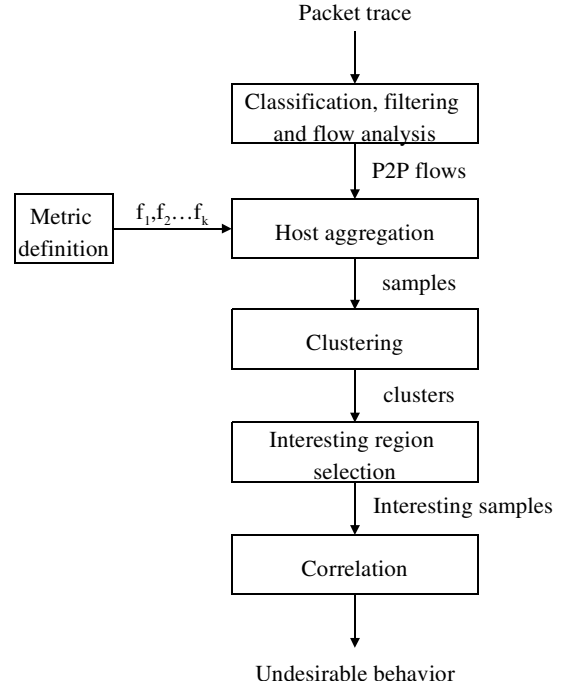

Figure 1: Schematic overview of the proposed methodology.

our analysis at the host level since our goal is to characterize peer activity - for instance, we are interested in capturing peers that exhibit undesirable behavior such as searching aggressively, or generating large amounts of traffic. We capture host behavior using several metrics such as the number of active flows, the total number of received connections, and the average size of packets sent and received. For any given host h, and in a given time window $[i\Delta T, (i+1)\Delta T]$, and for each metric f_m, $m = \{1, 2, \ldots, k\}$, a sample of the metric $f_m(h, i)$ is obtained for that time window. We study host behavior in various time windows, since the host might be demonstrating normal behavior overall, but might exhibit interesting behavior for certain periods of time.

The next step consists of detecting interesting, and potentially undesirable patterns of behavior that hosts may exhibit. To achieve this, samples corresponding to a given metric are fed to a clustering algorithm. In particular, we adopt a density-based clustering algorithm - clusters are regarded as regions in the data space in which the objects are dense, and which are separated by regions of low object density (noise). As output of this step, we get, for each metric, clusters of samples $\{f_m(h, i)\}$. Through manual inspection and domain knowledge, clusters are labeled as normal or possibly interesting. Interesting samples are then correlated across hosts to identify if they correspond to particular hosts, or are spread across multiple hosts. In addition the analysis may rely on correlating interesting behavior across multiple related metrics.

3. DATA SET

Real traffic traces are collected from a main broadband telecommunication ISP in Europe, offering telecommunication services to more than 5 millions families. Thanks to its *fully IP* architecture, and the use of both Fiber to the Home (FTTH) and Digital Subscriber Line (xDSL) access, the ISP offers converged services over a single broadband connection. No PSTN circuit is offered to end-users, so that only IP connectivity is adopted to offer data, VoIP and IPTV services over the same infrastructure [11]. The very pe-
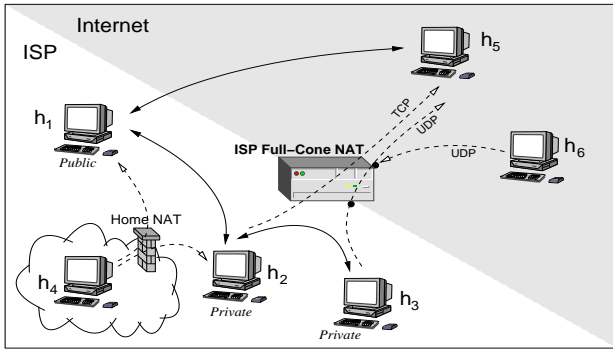
Figure 2: Schematic representation of possible connections from hosts in the MiniPop versus other hosts.

culiar mix of FTTH and high-quality ADSL access makes the ISP the leader in providing high speed access in its country, and the preferred ISP among high-end users.

3.1 Setup and Trace Collection

A Metropolitan Area Network (MAN) Ethernet-based architecture is adopted in the last mile. Residential and small business customers are connected to a Home Access Gateway (HAG), which offers Ethernet ports to connect PCs, the set-top box and traditional phone plugs. In case of FTTH access technology, HAGs are connected to Ethernet switches in the building basement, which are then interconnected to form Gigabit Ethernet rings. Rings are terminated at the so called MiniPoP routers, which offers connectivity to the ISP backbone. Customers are offered a 10Mbps half-duplex Ethernet link. In case of ADSL access, the HAGs are connected by the DSLAM to backbone routers. Customers are offered 1024kbps upstream and 6Mbps or 20Mbps downstream links.

Addressing and NATs: Both private and public addresses are offered to end users, as shown in Figure 2. A small number of hosts (for instance, host h_1), have public IP addresses and these hosts have unrestricted end-to-end IP connectivity with other Internet hosts. The vast majority of hosts (for instance hosts h_2 and h_3) are assigned private IP addresses. Whenever such hosts communicate with hosts in the external Internet (for instance, h_5 and h_6), the data communication involves traversal of an ISP-wide NAT. Note however that plain end-to-end IP connectivity is offered among hosts inside the ISP network, and communication between hosts inside the ISP (for instance, h_1, h_2, and h_3) does not involve NAT traversal. At the peering point, a Full-Cone NAT service [33] is implemented. This forbids any TCP connection initiated from the external Internet. However, it is possible that UDP flows initiated from the external Internet are permitted. In particular, once a host behind a full-cone NAT (for instance, host h_3) sends a UDP packet to the external Internet, it can receive UDP packets on the port from any arbitrary external host, (for instance, h_6). Finally, in addition to the ISP-wide NAT, individual users (for instance, host h_4) may also employ home NAT boxes. Host h_4 cannot be contacted by any host (unless proper configuration at the home NAT is provided).

Trace Collection: Traces have been collected at a MiniPoP router during March and April 2008. A probe PC running Tstat was used to analyze in real time all the packets going to and coming from all the users in the MiniPoP, and produce a flow level log that has then been post-processed later[2]. In this paper we report results ob-

tained focusing on a subset of the dataset, corresponding to about 25 hours, or about 2TB of information. About 2,200 different hosts were active in the MiniPoP, exchanging packets to about 782,000 different hosts in the Internet. Few hosts in the MiniPoP are using public IP addresses, which correspond in general to servers installed in small offices.
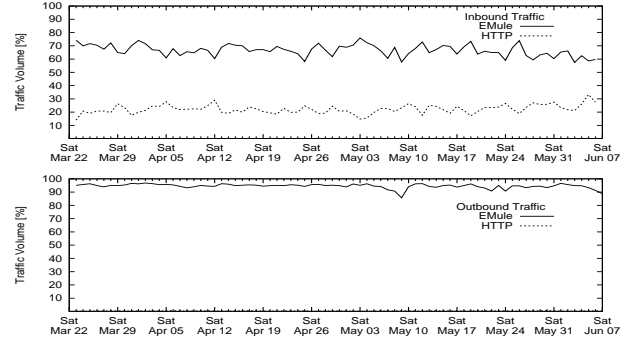

Figure 3: Traffic volume shares in the MiniPoP. The top plot reports the inbound traffic, while the bottom plot reports the outbound traffic. Only HTTP and eMule traffic is reported. On the bottom plot, HTTP traffic is small and not visible.

3.2 Description of P2P Systems

The most popular P2P system used among the users in the ISP is eMule [19], a widely deployed file sharing application. To demonstrate this, Figure 3 shows the byte-wise traffic volume percentage as measured during about three months. The top and bottom plots report results considering inbound and outbound traffic respectively (i.e., bytes destined to/sourced from a host in the MiniPoP). The two most popular protocols are HTTP and eMule, with other protocols accounting for no more than 10% of traffic. In particular, eMule accounts for about 60-70% of traffic on the inbound traffic, while it has a share of more than 95% on the outbound volume. HTTP traffic is predominant only in the inbound traffic, since hosts in the MiniPoP act as clients.

We focus our analysis on eMule traffic given its large predominance in the dataset. eMule supports both a centralized architecture, referred to as *eMule network*, and a DHT system, referred to as *Kad network*. In particular, eMule servers do not store any files, but they only maintain a database of files and users per file. The Kad network is a large-scale DHT-based system based on Kademlia [27]. A Kad client looks up for both content and peers sharing content using the DHT instead of relying on the eMule servers. Once a client has found a peer that is sharing the desired file, direct connections are used to download/upload the actual data using end-to-end TCP connections; communication with the eMule server goes preferentially over TCP, while Kad relies on UDP only.

The original eMule/Kad networks have mechanisms in place to identify clients behind NAT, and limit the performance of such clients when they try to download content. This impacts the performance of hosts with private IP addresses in the ISP, since a NAT is traversed when communicating with hosts in the Internet, e.g., eMule servers. Given that the large majority of ISP hosts have been given private IP addresses, the performance of eMule is severely

---

[2]A flow is identified by the traditional 5-tuple. In case of TCP, a flow starts when the SYN packet is observed. If the three-way-

handshake is (not) completed, then the TCP flow is said to be *(un)successful*. In case of UDP, a flow starts when the first packet is observed. If (no) packet is observed in the reverse path, then the UDP flow is said to be *(un)answered*. Flows end after no packets have been observed for 10 minutes.

Table 1: General Statistics of P2P Traffic

Direction	TCP		UDP	
	eMule		eMule/Kad/KadU	
	succ. connections	Bytes	flows	Bytes
MiniPoP to ISP	264k	512G	4.7M	820M
MiniPoP to Internet	377k	80G	412.7k	58M
ISP to MiniPoP	174k	341G	3.8M	735M
Internet to MiniPoP	0	0	208k	35M

limited. Therefore, a community of ISP users modified the original eMule client [6] to form a closed P2P network that we call *KadU network*. The network is closed in the sense that all KadU clients belong to the ISP network only. The Kad protocol has been modified, so KadU messages can only be exchanged among peers running the modified eMule version and using IP addresses actually used by the ISP. This ensures that a KadU client cannot operate in the Internet. Similarly, the peer selection mechanism has been modified to preferentially connect to other KadU clients. The KadU peers perform search operation on the KadU network by default, rather than relying on server-based search as in the default eMule configuration. No changes have been made to the eMule part, so that both server and P2P protocols are the same as in the original eMule, and the modified eMule client and original eMule client can perfectly interoperate.

Besides avoiding the NAT issues, running the modified client has several advantages. Indeed, it is desirable to download content from other clients in the ISP because the large percentage of hosts connected by FTTH access guarantees much higher upload capacity than the typical one offered by ADSL providers. Furthermore, given that all the ISP peers are in the same European country, the content that is available in the P2P system matches the interest of the community, and it is easier to trade content in the closed network than in a worldwide network. For these reasons, clients in KadU typically see much better performance than the one typically achieved by clients in the Kad network.

3.3 Preliminary Traffic Analysis

In the considered 25 hours dataset (on a Wednesday), we identified 478 clients running KadU inside the MiniPoP, and exchanging traffic with about $229,000$ KadU clients outside the MiniPoP. For Kad, we identified 136 clients which were exchanging packets with about $300,000$ clients in the Internet. Table 1 presents details on the trace characteristics for the Kad, KadU and eMule systems. Most of the paper will focus on these systems. But, in Section 9, we extend the analysis to consider other systems.

Knowing the address space allocation for the ISP and for the MiniPoP, we are able to classify all hosts as being in the MiniPoP, in the ISP (but not in the MiniPoP) or in the Internet. We leverage this classification in Table 1. Each row provides statistics about traffic exchanged between hosts in two classes. The second and third columns give the number of successful TCP connections classified as eMule, and the amount of bytes they carried. The last two columns show similar numbers for UDP flows classified as eMule, Kad or KadU. For example, the MiniPoP to ISP row reports that (i) there was a total of $264k$ eMule TCP connections initiated from inside the MiniPoP to clients inside the ISP, which carried a total of $512GB$ of data; and (ii) around 4.7 million UDP flows (classified as eMule, Kad, or KadU) were initiated in the same direction, and about $820MB$ of data was exchanged.

From this table, we see that: (i) the bytes exchanged between hosts within the ISP is much larger than the bytes going to the Internet, for both TCP and UDP. This is due to the extensive usage

of the KadU network, and its efficiency in localizing traffic communication to within the ISP; (ii) there is a non-negligible amount of TCP and UDP traffic exchanged with the Internet. This is because the use of Kad clients is still prevalent. In addition, even clients that use the KadU network may need to rely on eMule if the content cannot be located within the ISP; and (iii) there are no TCP connections initiated from the Internet to the MiniPoP, but there are UDP flows though. This is due to the full-cone NAT at the edge of the network, as previously explained in Section 3.1.

4. TRAFFIC CLASSIFICATION

In this section, we describe the mechanisms we use to identify P2P traffic. We also present our methodology to aggregate flow metrics into samples per host.

4.1 Selecting Flows of Interest

In order to correctly identify eMule and Kad/KadU traffic, we employed an approach described in [28] which implements deep packet inspection (DPI), and produces a flow level log as output. When a new flow is identified, the DPI classifier looks at the application layer payload to identify a set of well-known protocols. All eMule and Kad/KadU protocol messages are included, and manual tuning has been adopted to guarantee conservative classification. While the performance of the DPI is out of the scope of this paper, we manually verified that the false positive and false negative probability is practically negligible.

The output of the classification and flow analysis phase is a flow level log, in which each flow that has been observed and classified as eMule, Kad or KadU is listed, along with a list of measurements. In particular, in this paper we exploit the following per-flow information: (i) *flow id* defined as <src_ip, src_port, dst_ip, dst_port, protocol_type>; (ii) *first and last packet time*; (iii) *number of sent and received packets*; (iv) *number of sent and received bytes*. Note that the above information can be easily derived by any flow level logger, such as NetFlow [14], running directly at routers.

4.2 Aggregating Flows into Host Samples

As described in Section 2, we aggregate flow measurements into host metrics $f_m(h, i)$. A sample f_m is obtained for host h at every time slot i of size $\Delta T = 5$ minutes. The latter choice enables us to track changes in host behavior over the order of minutes, and it is unlikely for host behavior to significantly shift over this period.

Given that clients could be part of either the Kad or KadU network each with very different properties, we would like to study each system in isolation by separately aggregating Kad and KadU flows into samples. However, while the two networks differ in the UDP protocol used in the Kademlia DHT, both employ identical TCP-based protocol on the data path, e.g., to exchange content. As a consequence, the DPI classification can successfully distinguish UDP-based control messages, but the TCP-based data flows are classified identically as eMule.

To handle this, we adopt the following heuristic. Consider a time slot i and a host h. If UDP flows are present, then we classify the sample as either Kad or KadU based on the classification of UDP flows. All TCP-based metrics are then classified accordingly. In the dataset, there are $12,963$ KadU samples and $1,519$ Kad samples. It is possible that a time slot includes both Kad and KadU - however, there are only 35 such samples, so that we can simply discard them.

Finally, it is possible to have samples with neither Kad nor KadU flows, but exclusively eMule TCP connections. There are $1,200$ such samples, most of which are due to 4 hosts running the centralized eMule protocol only. We do not consider them further.

5. METRICS

In this section, we present the list of the metrics considered in this paper, which is summarized in Table 2. All the selected metrics are very simple and intuitive metrics. Some of them have been previously proposed for both traffic characterization and classification considering both P2P systems, and traditional client/server applications. Some are specifically defined considering the scenario we are facing, e.g., to highlight eventual Kad and KadU dissimilarities, or to pinpoint possible atypical behavior.

If not otherwise specified, each metric is evaluated separately for UDP and TCP flows, given that the considered systems make use of both protocols. When needed, *TCP* (*UDP*) will be appended to the metric name, as appropriate. For relevant metrics, we consider the location of the flow initiator as either being inside or outside the MiniPoP. For ease of notation, metrics involving flows initiated inside (outside) the MiniPoP will be prepended with the term *inout* (*outin*) followed by the metric name.

In Section 5.1 we consider the metrics general to all flows first, and then the metrics to which initiator location is specified are detailed in Section 5.2.

5.1 Metrics Independent of Flow Initiator

These metrics consider various measurements that do not depend on the location of the flow initiator. We group them in two categories, *Flows*, which include per-flow basic statistics and *Data transfer*, which includes data exchange related metrics. Given a host h in the MiniPoP and a time slot i, we have:

• **Flow** related metrics: (i) *avg-duration* is the average duration of flows started during time slot i; (ii) *live-conn* is the total number of flows that were active during time slot i. This includes flows that have started in the current time slot and flows that started in previous time slots and are still active in the current one; (iii) *fract-incoming-conn* is the ratio of flows initiated from the outside to the total number of flows.

• **Data Transfer** related metrics: (i) *bps-rcvd* and *bps-sent* are the average bits per second (referred to as *bps*) received and sent respectively; (ii) *avg-pkt-size* is the average size of packets sent and received; (iii) *ratio-bytes-sent-to-rcvd* is defined as $\frac{B\_sent - B\_rcvd}{B\_sent + B\_rcvd}$, where B_sent (B_rcvd) is the total amount of bytes sent (received). *ratio-bytes-sent-to-rcvd* $= -1$ for hosts receiving data only, while *ratio-bytes-sent-to-rcvd* $= 1$ for hosts that send data only.

5.2 Metrics Dependent on Flow Initiator

In this case, four categories of metrics have been selected:

• **Flow** related metrics: *total-conn-attempts* is the total number of flows initiated (inout) or received (outin). This includes both successful and unsuccessful connections when considering TCP, and both answered and unanswered flows when considering UDP.

• **Destinations** related metrics: (i) *avg-conn-per-IP* is the ratio of all flows to the number of distinct destinations. A similar metric was used in [23] for P2P traffic classification, in which the authors showed that it is rare that P2P clients open concurrent connections to other peers; (ii) *total-peers* is the total number of distinct peers; (iii) *dest-ports* is the total number of distinct destination ports; (iv) *1024-dest-ports* is the total number of distinct reserved destination ports, i.e., ports from 0 to 1024. Since reserved ports should not be used by non standard application, we include this metric to highlight possible abuse.

• **Failures** related metrics: (i) *failure-ratio* is the ratio of unsuccessful TCP flows to total TCP flows [3] (ii) *fract-unanswered-appl*

---

[3]Note that unsuccessful TCP flows cannot be classified as eMule, since no payload can be inspected. Hence, we take a conservative

---

Table 2: List of Metrics

|		
Metrics Independent of Flow Initiator	Flows	*avg-duration*
		live-conn
		fract-incoming-conn
	Data Transfer	*bps-rcvd*
		bps-sent
		avg-pkt-size
		ratio-bytes-sent-to-rcvd
Metrics Dependent on Flow Initiator [inout, outin]	Flows	*total-conn-attempts*
	Destinations	*avg-conn-per-IP*
		total-peers
		dest-ports
		1024-dest-ports
	Failures	*failure-ratio* [TCP only]
		fract-unanswered-appl [TCP only]
		fract-unanswered [UDP only]
	ISP	*ISP-to-Internet-ratio*
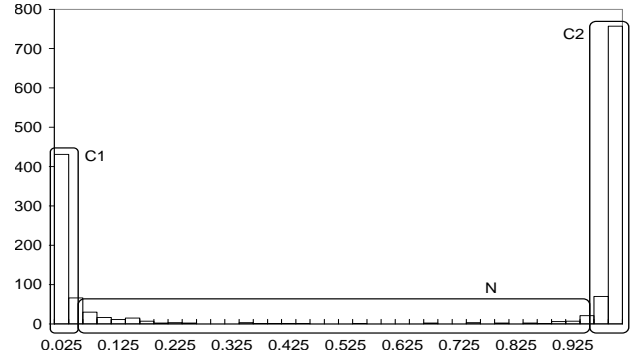
Figure 4: *outin-fract-unanswered* **is an example of multiple cluster metric.**

is the fraction of TCP flows where the TCP handshake is successfully completed, but the destination never sends any application data for the duration of the connection. (iii) *fract-unanswered* is the fraction of unanswered UDP flows to total UDP flows.

• **ISP** related metrics: *ISP-to-Internet-ratio* is the ratio of the number of peers within the ISP that are contacted to the total number of contacted peers.

6. IDENTIFYING UNWANTED BEHAVIOR

Our goal is to identify undesirable behavior of P2P systems. The key challenge we faced in our study is that an exhaustive list of potential undesirable behavior is not available to us a priori. Moreover, the intrinsic heterogeneity of P2P traffic makes it hard to clearly distinguish undesirable behavior from normal usage. Consequently, our methodology employs a combination of data-mining techniques, and manual inspection through domain knowledge.

As a first step, we employ clustering algorithms [34] to obtain a set of *coarse* clusters of the data. Clustering algorithms are well known techniques in the data mining field that fall in the unsupervised machine learning category. Without the need of any training data, clustering algorithms aim at partitioning the data set into subsets - called "clusters" - so that samples in the same subset share common traits, i.e., they are close to each other according to a notion of distance. Clustering algorithms are often useful for outlier detection, where outliers may emerge as small clusters far apart from the others. As a second step, we extensively resort to domain knowledge and manual inspection to interpret the clustering results,

---

approach and only consider as eMule related failures those that are directed to the default eMule port.
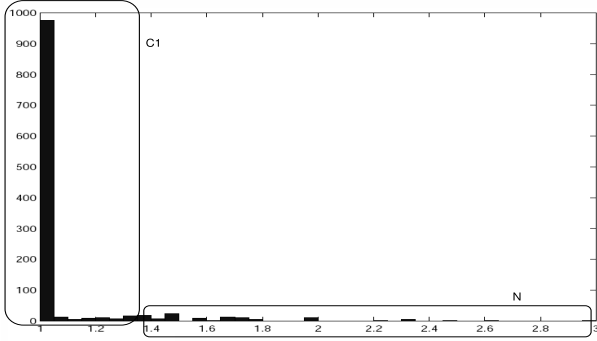
Figure 5: *inout-avg-conn-per-IP-TCP* **is an example of single cluster metric.**

zoom in on interesting patterns, and identify undesirable behavior.

6.1 Density based Clustering

Among clustering algorithms, density based clustering uses the concept of dense region of objects. In such schemes, dense regions of objects are considered a cluster and low density regions are considered as noise. In particular, we selected DBScan [20], since it is well known and offers several advantages: it automatically determines the number of clusters (contrary for example to the k-means algorithm); it is robust to noise, i.e., isolated samples; and finally it does not have any bias versus any cluster shape.

Intuitively, DBScan groups together points that fall in a dense region of the metric space. Given a point in the data set, density is estimated as the number of points within a specified radius of that point. There are three types of points: (i) **core point** is a point that has more than *Minpts* around it within a distance $d \leq \epsilon$; (ii) **border point** is a point that is within a distance ϵ of a core point but is not a core point; (iii) **noise point** is any point that is neither a core point nor a border point. With these definitions in mind, DBScan puts two core points in the same cluster if they are within a distance ϵ of each other. Also, a border point within distance ϵ of a core point is put in the same cluster as the core point. Finally, noise points are labeled as such.

For each metric m, we consider the set of all samples $F_m = \{f_m(h, i)\}$ collected during the desired observation period, for each host h and for all time slots i. In this paper, for a 5 minutes observation period and a 25 hour trace, $0 \leq i \leq 300$. We apply clustering algorithms to each metric individually, and define the distance between two samples as simply $d = |f_m(h_1, i_1) - f_m(h_2, i_2)|$. We choose to apply clustering on individual metrics rather than on multidimensional spaces for several reasons. First, each metric sample distribution is generally very skewed, which makes clustering difficult per se. When considering a multidimensional space obtained as the Cartesian product of skewed metrics, the result of clustering is hard to predict and control, e.g., to impose a coarse clustering. Further, distance in multidimensional space may be difficult to define, since each metric have very different support, e.g., $x \in [0, 1]$ and $y \in [0, \infty)$ make it hard to appreciate the spread on the x dimension. Note that this is typical of our scenario, e.g., considering metrics like *outin-fract-unanswered* and *avg-duration*. Although dimensional reduction and normalization techniques exist, the outcome from them may be difficult to control and interpret. Finally, possible undesirable behavior can be already identified when considering a single metric, while the correlation between undesirable

Table 3: DBScan Sensitivity

	Minpts					
	1%	5%	10%	20%	40%	60%
0.01	3(4%)	2 (15%)	2 (16%)	2 (19%)	1 (49%)	0 (100%)
0.05	2 (1.1%)	2 (2.3%)	2 (3.7%)	2 (4%)	1 (41%)	0 (100%)
ϵ 0.1	2 (0.5%)	2 (0.9%)	2 (1%)	2 (1.2%)	1 (40.8%)	0 (100%)
0.2	2 (0%)	2 (0.1%)	2 (0.1%)	2 (0.2%)	2 (0.3%)	0 (100%)
0.5	1 (0%)	1 (0%)	1 (0%)	1 (0%)	1 (0%)	1 (33%)

behavior across different metrics can be later checked exploiting the domain knowledge of the targeted scenario.

We illustrate the operation of the DBScan algorithm, and the impact of the parameters *Minpts* and ϵ with an example. Consider Figure 4 which shows the histogram of *outin-fract-unanswered* for UDP traffic considering the Kad dataset. More than 650 samples (around 36%) fall in the range $[0, 0.08]$, while more than 1,300 (around 58%) samples fall in the range $[0.98, 1]$.

Table 3 reports the DBScan result when applied to the dataset in Figure 4, for different values of *Minpts* and ϵ parameters. Each cell shows the number of clusters produced by DBScan, and the fraction of the samples that are classified as noise. For instance, for ϵ=0.1, and *Minpts*=40% of the total samples, there is 1 cluster, with 40.8% of the samples classified as noise. For large values of *Minpts* (60%) we see that 0 clusters are produced for most ϵ values, and all 100% of the samples are classified as noise. This is because no point has a sufficiently large neighborhood or density to be classified as a core point. As we decrease *Minpts* however, the noise region decreases, and clusters emerge. For ϵ=0.1, and for *Minpts* 20% or lower, 2 clusters are always identified, which matches our intuition from the Figure. We observe that DBScan is relatively robust to the input parameter setting in our scenario, and that there are several parameter settings that can achieve a reasonable coarse clustering.

We employ a simple iterative search heuristic to identify a value of *Minpts* and ϵ that can achieve a reasonable coarse clustering. Our heuristic seeks to obtain a clustering result with noise region that is non empty but not too large, e.g., a small percentage of samples. The reason for requiring a small number of samples to be classified as noise is to avoid cases where many smaller clusters are merged into one larger cluster with no noise region (for instance, ϵ=0.5, *Minpts*=20% in Table 3), or to prevent clusters being formed with a small number of points. We start with $\epsilon = 0.1$, *Minpts*=50% and keep decreasing *Minpts*, until the resulting noise falls in the target region. Then, ϵ is decreased until the noise region is exceeds the target value. Large ϵ enlarge clusters adding noise points and eventually merging clusters, while small ϵ results in possible splitting of clusters that might not be of interest. The results we present employ a target noise region of 6%, but we have found that DBScan is relatively robust to the choice in our scenario, and any value in the range 2-10% would provide very similar results.

6.2 Interesting Region Selection

After getting the output from DBScan, intuitively we could consider the samples in the noise region to be the interesting ones. However, undesirable behavior could be so prevalent to form a whole cluster, and hence, only considering the noise points may cause information loss. We therefore believe the interesting region should be selected based on domain knowledge from the network operator, P2P developer, or the end-user. In particular, applying DBScan to each metric, two possible cases are obtained: (i) metrics exhibiting a single cluster and a noise region; and (ii) metrics exhibiting multiple clusters and one or more noise regions.

In cases where the metric exhibits a single cluster, the inter-

esting region typically coincides with the noise region. To illustrate this, consider Figure 5, which shows the histograms of values taken by the *inout-avg-conn-per-IP-TCP* metric considering the Kad dataset. As shown in the Figure, when DBScan is employed, a single cluster ($C1$) is produced which includes all samples in the range $[0, 1.4)$, and a noise region, including samples in $[1.4, 3]$. The noise region is interesting since eMule clients are not expected to open more than one TCP connection with the same host. We have further investigated the samples in this region, and have found them to be due to hosts being attacked by *fake servers*, as we explain in Section 8.3. We also note that for such metrics, DBScan enables choosing the thresholds for the noise region appropriately - simpler heuristics like selecting the top or bottom 10% of samples as showing interesting behavior do not take the distribution of data into account and may not be as effective in general.

In cases where the metric exhibits multiple clusters, the choice of interesting region can only be supported by the knowledge of the considered application, metric, and scenario. For example, in Figure 4, DBScan identifies two clusters $C1, C2$ and a noise region, which confirms the visual intuition. In this case, we consider the interesting region to include cluster $C2$, since it represents samples in which most externally initiated UDP connections are unanswered. We analyze this in further detail in Section 8.2. More generally, the interesting region could include a combination of multiple clusters and noise regions.

6.3 Correlation Across Interesting Samples

Having identified the interesting samples for each metric, we employ several simple heuristics to identify correlations across the samples, which in turn can aid making inferences of undesirable behavior. We describe these below:

● *Hosts dominating interesting samples:* We consider the number of distinct participating hosts (or IP addresses) to which the interesting samples for a given metric correspond. If the entire interesting cluster for a metric can be attributed to a small number of participating hosts, it is an indication that the interesting behavior is a property of those hosts. If however the interesting cluster is spread among several hosts, it is an indication that the interesting behavior is more general and not due to a few hosts.

  ● *Correlations across metrics:* We consider whether interesting behavior seen across multiple metrics are correlated, and are due to the same underlying cause. We typically rely on domain knowledge to determine such correlations. For instance, in Section 8.2, we used domain knowledge to reason that a large number of interesting samples seen in four of the metrics we considered were directly related. Likewise, in Section 8.4, we isolate hosts that generate a large number of samples in the interesting region across multiple metrics, and use these observations to reason about the potential behavior of the hosts.

7. RESULTS

In this section, we present high level characteristics of the Kad and KadU networks. We then discuss results with DBScan and the selection of interesting regions for various metrics.

7.1 High Level Characteristics of Systems

In this section we provide high level background on the Kad and KadU networks, highlighting key differences between them.
● *In contrast to Kad, KadU traffic typically stays within the ISP*: Kad clients mostly contact peers in the Internet while KadU clients mostly contact peers within the ISP. The *inout-ISP-to-Internet-ratio* metric was 1 for almost all KadU samples when UDP traffic was considered. Interestingly, KadU clients did contact more peers in

the Internet when TCP traffic was considered. This was not entirely expected and will be further investigated in Section 8.4.
● *In contrast to Kad, KadU clients use default UDP/TCP ports:* When the *dest-ports* metric is considered, the median value of KadU samples is 1, while it is 33 for Kad. This is because KadU clients run in a friendly environment in which no throttling is imposed on P2P traffic by the ISP. Hence there is no need to try masquerading P2P traffic by using random ports. On the contrary, Kad clients run in the Internet, where ISPs may block P2P traffic, and there is a greater tendency for users to adopt random ports (and possibly protocol obfuscation).
● *In contrast to KadU, Kad clients see almost no incoming TCP traffic due to a NAT at the edge of the ISP*: The metric *fract-incoming-conn-TCP* is equal to 0 for almost all Kad samples, while it has a bell distribution for KadU samples. The reason for this is that there is a NAT at the edge of the ISP, as discussed in Section 3.2, which forbids incoming TCP connections from the Internet. Interestingly, Kad clients can still receive UDP flows initiated in the Internet. This is because the NAT at the edge of the ISP is a Full Cone NAT.
● *KadU clients exchange much more data, with a prominent seed-like behavior*: When the *bps-rcvd* and *bps-sent* metrics are considered, the $90\%ile$ for KadU samples is 164kbps and 674kbps respectively. In contrast, the $90\%ile$ for Kad samples is only 36kbps and 54kbps. The much higher performance in KadU is due to the effectiveness of the optimizations in the KadU client, as well as the large installation of high-speed FTTH users in the ISP. Further, we noticed that KadU clients present a predominant seed-like behavior (for example, the $90\%ile$ of the *bps-sent* metric is 4 times the $90\%ile$ of the *bps-rcvd* metric). We believe this may be attributed to the high-speed upload bandwidth of the FTTH users in the ISP.

7.2 Interesting Region Selection

In this section, we present the results of applying DBScan to our dataset and the interesting regions we identified based on manual inspection. For single cluster metrics, we simply selected the noise region as interesting, as discussed in Section 6.2. Hence, we focus on metrics that involved multiple clusters.

The sensitivity of the interesting regions was tested in our dataset by splitting the 25 hour trace into two halves and then running DBScan over each portion, as well as running DBScan over the entire trace. One half corresponded to day-time activity and the other half to night-time activity. For single cluster metrics, the results of clustering were similar, with only marginal changes to clusters' width and noise regions. The multiple clusters metrics, on the other hand, had minor changes in clusters for some metrics, but overall, the final trend of the interesting regions was preserved. In the rest of the section, we focus on clusters obtained using the entire trace.

7.2.1 Kad

In this section we present results for Kad, which are reported in Table 4. The first column shows the metric name and transport protocol. The second column identifies a region as a cluster or noise, in which we highlight the interesting one in bold. The third column shows the actual range of sample values in each cluster, while the fourth column reports the percentage of the samples that are in the cluster. Finally, the fifth column shows the explanation why the selected region is interesting.

We summarize key observations as follows:
● *Samples with predominantly control messages:* The first row of Table 4 shows the clusters found by DBScan for the *avg-pkt-size-TCP* metric. There are three clusters for this metric. Cluster $C1$ contains 16.28% of the samples, and it refers to samples whose flows exhibited "small" average packet size. $C3$ corresponds on

Table 4: Metrics with multiple clusters - Kad

Name	C/N	Range	percent	Explanation
avg-pkt-size-TCP $[Bytes]$	**C1**	[55 250]	16.28%	Primarily control
	C2	[726 955]	17.05%	
	C3	[956 1,348]	61.66%	
	N	[296 723]	5.01%	
outin-fract-unanswered-UDP	C1	[0 0.08]	36.14%	Left group or home NAT
	C2	[0.93 1]	58.04%	
	N	[0.08 0.92]	5.82%	
ratio-bytes-sent-to-rcvd-UDP	**C1**	[-1 -0.63]	17.98%	Left group or home NAT
	C2	[-0.62 0.45]	78.06%	
	N	[0.5 1]	3.96%	
inout-1024-dest-ports-UDP	C1	[0 0]	73.72%	DDoS attack
	C2	[1 1]	18.12%	
	N	[2 6]	8.16%	
ratio-bytes-sent-to-rcvd-TCP	**C1**	[-1 -0.62]	13.49%	Selfish hosts
	C2	[-0.4 0.62]	55.06%	
	C3	[0.62 1]	27.66%	
	N	[-0.62 -0.41]	3.79%	

Table 5: Metrics with multiple clusters - KadU

Name	C/N	Range	percent	Explanation
inout-ISP-to-Internet-ratio-TCP	**C1**	[0 0.62]	51.77%	Traffic within ISP
	C2	[0.62 1]	48.23%	
outin-fract-unanswered-UDP	C1	[0 0.22]	33.82%	Left group or home NAT
	C2	[0.8 1]	60.21%	
	N	[0.22 0.8]	5.97%	
ratio-bytes-sent-to-rcvd-UDP	**C1**	[-1 -0.71]	49.42%	Left group or home NAT
	C2	[-0.25 0.37]	44.69%	
	N	[0.37 1]	5.89%	
fract-incoming-conn-UDP	C1	[0 0.32]	18.85%	Left group or home NAT
	C2	[0.57 1]	75.58%	
	N	[0.32 0.56]	5.57%	
outin-failure-ratio-TCP	C1	[0 0]	62.33%	Left group or home NAT
	C2	[1 1]	33.06%	
	N	[0.01 0.97]	4.61%	
ratio-bytes-sent-to-rcvd-TCP	**C1**	[-1 -0.36]	5.61%	Selfish hosts
	C2	[-0.36 1]	91.8%	
	N	[-0.62 -0.36]	2.59%	

the contrary to "large" average packet size, while $C2$ corresponds to a cluster with "mid-sized" packets. These clusters correspond to hosts exchanging mostly control messages, mostly data messages and a mix of control and data messages respectively. Among those, cluster $C1$ is interesting since it corresponds to samples where only control messages were exchanged. This could be for benign reasons, for instance, a host that does not download or upload content. But it could also indicate undesirable behavior, for instance a host being part of a P2P botnet. One potential indication of malicious activity is a host that is persistently sending only control messages in all its samples. We did not find evidence of this in our trace, leading us to believe there was no malicious activity.

● *Samples of peers that do not reply to incoming requests*: When the *outin-fract-unanswered-UDP* metric is considered, it is striking that there is a cluster ($C2$) with samples in the range 0.93 to 1 and which includes 58.04% of the samples. This cluster corresponds to samples where almost every UDP flow initiated from the outside is unanswered, indicating potentially anomalous behavior. Likewise, considering metric *ratio-bytes-sent-to-rcvd-UDP*, cluster $C1$ corresponds to samples where UDP packets are mostly received, indicating again that the peer inside the MiniPoP is not responding to external queries. These two clusters are related, and we analyze further in Section 8.2.

● *Communication with reserved ports*: We consider metric *inout-1024-dest-ports-UDP*, which intuition suggests should be close to 0, since P2P applications are not expected to run using a reserved port. But both cluster $C2$ and the noise region N refers to values of this metric larger than 0, accounting for 26.29% of the samples. In Section 8.1 we investigate this metric further and present evidence of a DDoS attack on DNS servers.

● *Selfish versus seed behavior*: Considering the metric *ratio-bytes-sent-to-rcvd-TCP*, three clusters are shown. $C1$ represents samples for hosts with selfish behavior (mostly receiving data), $C2$ represents samples for hosts that are both receiving and sending and $C3$ shows samples for hosts with seed behavior (mostly sending data). Considering P2P file sharing application, a user is expected to contribute fairly to the community, so cluster $C1$ represents possibly undesirable behavior.

7.2.2 KadU

In this section we focus on metrics where DBScan found multiple clusters for KadU metrics, which are reported in Table 5. We summarize key observations as follows:

● *Degree of communication within ISP*: Here we focus on met-

ric *inout-ISP-to-Internet-ratio-TCP*, for which DBScan found two clusters. Cluster $C2$ corresponds to samples for which peers within the ISP are predominantly contacted. Cluster $C1$ represents those samples for which mostly peers in the Internet are contacted. The presence of cluster $C1$ is not expected since KadU is optimized for communication with peers inside the ISP. We further analyze $C1$ in Section 8.4.

● *Samples of peers that do not reply to incoming requests*: Like in Kad, DBScan found cluster $C2$ for metric *outin-fract-unanswered* and cluster $C1$ for metric *ratio-bytes-sent-to-rcvd* for UDP, which characterize peers that do not reply to incoming requests. In addition to these metrics, two more related metrics were found to have multiple clusters in KadU which we believe is related to the same issue. First, for the metric *fract-incoming-conn-UDP*, cluster $C2$ contains all samples for which hosts mainly receive UDP flows. We note the cluster had a prominent spike around 1, which indicates that for a large number of samples, flows are only being received. Second, for the metric *outin-failure-ratio-TCP*, cluster $C2$ corresponds to samples in which all incoming TCP connections failed. A detailed analysis is presented in Section 8.2.

● *Selfish versus seed behavior*: The metric *ratio-bytes-sent-to-rcvd-TCP* has a very different distribution considering KadU, showing that the large majority of peers have a seed-like behavior, which are clustered in $C2$. Also, there is a cluster of samples that suggests a subset of peers act as selfish clients, not willing to share content. We therefore select again this latter cluster as interesting.

7.3 Host Distribution in Interesting Region

Having identified the interesting regions, we next consider the number of distinct participating hosts (or IP addresses) to which the samples correspond. If the entire interesting region for a metric can be attributed to a small number of participating hosts, it is an indication that those hosts are particularly abnormal. If however the interesting cluster is spread among several hosts, it is an indication that the interesting behavior is more general.

Figure 6 shows, for each Kad metric, a point reporting the fraction of hosts that generate 90% of the samples versus the fraction of samples in the interesting region. For example, the metric *inout-1024-dest-ports* for UDP has 26% of its samples in the interesting range. 90% of these interesting samples have been generated by 24% of the hosts running Kad. We have circled those metrics for which we present key findings later. In addition, a similar plot is shown for KadU in Figure 7.

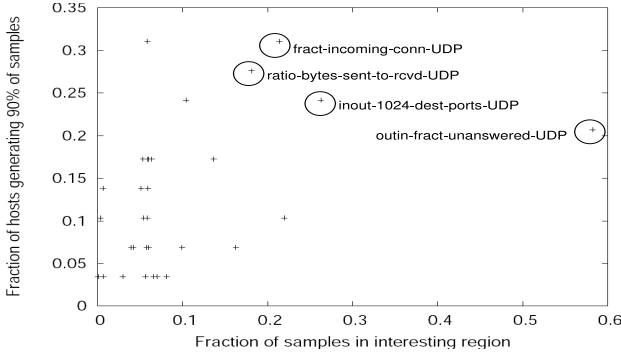We focus on metrics in the right side of Figures 6 and 7, which

Figure 6: Kad: fraction of samples in the interesting region versus fraction of clients generating them, for various metrics. Circled are metrics with most relevant results.
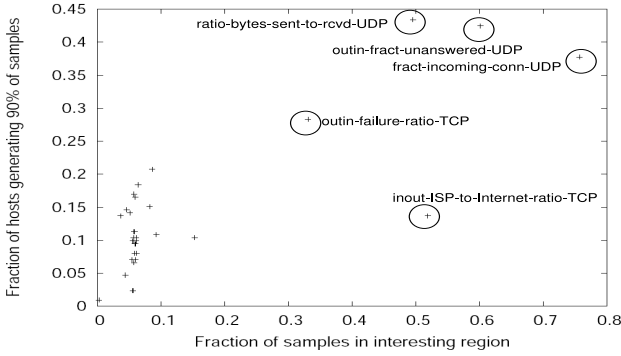

Figure 7: KadU: fraction of samples in the interesting region versus fraction of clients generating them, for various metrics. Circled are metrics with most relevant results.

correspond to those with a large fraction of interesting samples spread across many hosts. These metrics are the most interesting and we present and discuss our findings on them in Section 8. For most metrics in the bottom left of the figures, corresponding to those with interesting samples generated by a few hosts, we found the causes were usually benign and did not point to undesirable activity. However, a few cases deserve to be mentioned and we discuss them further in Section 8.

8. KEY FINDINGS

In this section, we present examples of undesirable behavior exposed by our methodology.

8.1 DDoS Attacks Exploiting P2P Systems

In this section, we describe our findings when studying the metric *inout-1024-dest-ports-UDP*, which was specifically added to observe undesirable traffic directed to reserved ports. Referring to Figure 6, Kad clients contacted peers to restricted ports for 26.25% of the samples, which is suspicious. We therefore isolated the samples in the interesting regions and looked at the destination port of those samples. It turns out that port 53 was the most common destination port, receiving $1,711$ out of $3,087$ flows destined to port 1024 or below. Note that no other port in the reserved range received more than 175 flows in total.

We further investigated and verified that flows destined to UDP port 53 were valid Kad flows, and not actual DNS flows misclassified by the DPI. Moreover, the destination IP address of the flows
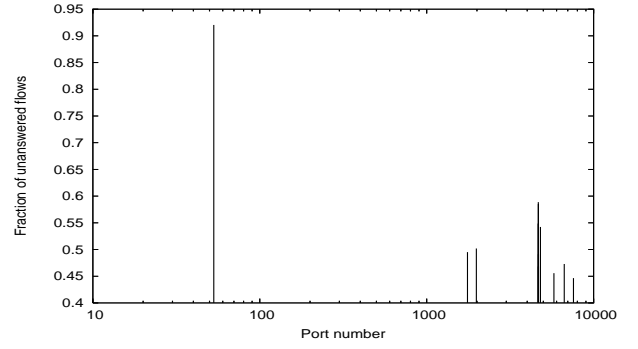

Figure 8: Fraction of unanswered flows per destination port.

corresponded to actual DNS servers not managed by the ISP, but serving domains in countries far away from the location of the MiniPoP. Finally, we noticed that most of the suspicious flows were unanswered. To better highlight this, Figure 8 shows the fraction of unanswered flows as a function of the destination port number. Notice the spike at port 53, which indicates that this port has the highest ratio of unanswered flows of more than 90%. Other spikes refer to typical Kad ports found in the dataset.

As a final observation, we noticed from Figure 6 that 25% of Kad peers were generating the interesting samples for *inout-1024-dest-ports-UDP*. On further study, we found that across all these peers, while less than 2% of Kad flows initiated are destined to reserved ports, more than 30% of these flows target port 53. This indicates that the problem is not specific to a small subset of Kad peers, but is more predominant.

We believe these results show evidence of DDoS attacks on well known DNS servers exploiting the Kad network. In such an attack, a malicious client in the Kad network, spreads contact information (IP address and port) about the victim (an actual DNS server) as if it were part of the Kad network. Later, innocent clients send regular Kad messages to the DNS server. Finally, we note that there has been some awareness of such attacks in eMule technical forums [2, 36], and in fact, the top most destination in our trace was mentioned in [2] as being under attack.

8.2 Unnecessary P2P traffic

Consider the 3 metrics on the top right corner of Figure 7. These correspond to *ratio-bytes-sent-to-rcvd-UDP*, *outin-fract-unanswered-UDP* and *fract-incoming-conn-UDP*. For each metric, 40% to 60% of the samples are in the interesting region, and about 60% of the KadU hosts are involved. The same three metrics are also highlighted in Figure 6 when considering Kad.

This clearly indicates some unexpected behavior, and points to a potentially significant problem. Investigating further, we observed that all metrics hint to a large number of UDP flows incoming to the MiniPoP that are never answered. In particular, 28% of UDP flows coming to the MiniPoP are unanswered, and 65% of this is due to Kad and KadU clients.

In addition, with the KadU dataset a high fraction of TCP failures is observed. Investigating further, $116,000$ TCP connections coming to the MiniPoP failed, which accounts for 30% of all TCP incoming connection attempts. Roughly 50% were due to KadU. Recall that for Kad peers, no incoming TCP connection is possible due to the ISP NAT.

Having a large number of failed TCP connections or UDP flows is undesirable not only from the perspective of the introduced traffic, but also from the state that may need to be maintained by vari-
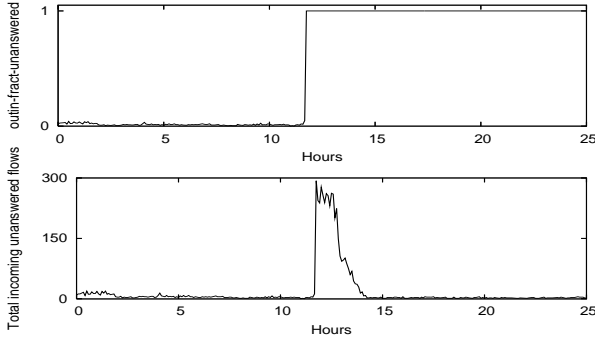
Figure 9: Host leaves the group about 11 hours after the beginning of the trace. Fraction of unanswered flows on the top and total number of unanswered flows on the bottom.

Figure 10: Connections made by the KadU client $h1$ towards $Server1$ (fake server) and $Server2$ (full server).

ous devices in the network (such as NATs and firewalls).

We believe there are two key reasons for unanswered flows. First, some P2P participants are behind home NATs. Other peers may learn about these participants through P2P membership management mechanisms, and may (unsuccessfully) attempt to communicate with them. Second, when a host leaves a P2P system, other peers may continue to attempt contacting it due to stale information in the P2P network.

Figure 9 shows an example of a host that left the P2P network, but which continues to receive packets for more than 14 hours after its departure. The top plot shows the time series for *outin-fract-unanswered-UDP*. Note the sharp transition from 0 to 1 which corresponds to node departure. The bottom plot depicts the total number of unanswered incoming UDP flows. Over 60 flows per minute are received during the next 2.5 hours, after which about 1 flow per minute is still observed for several hours until the end of the trace.

We have devised simple heuristics to identify flows that are unanswered due to the departure of a host. This is based on the observation that a host that leaves the P2P network will not initiate any new UDP or TCP flows; in contrast, hosts behind NATs are likely to initiate flows to other peers. We found that host departure is responsible for 41% and 48% of the unanswered UDP flows for Kad and KadU respectively, and the rest is due to hosts behind home NATs. For failing TCP connections, 75% were sent to hosts that appear to have left the P2P network. These results indicate that both factors (node departure and home NATs) play an important role in explaining the results.

Overall, these results indicate that better mechanisms must be designed to handle stale P2P membership, and hosts behind NATs for a P2P system to exhibit more friendly behavior to network operators. In particular, it is important for membership management algorithms to avoid propagating hosts behind NATs, and to ensure stale information is eliminated in a timely fashion.

8.3 Malicious P2P Servers

In this section we describe our findings when studying the metric *avg-conn-per-IP-TCP*. The interesting region for this metric in both Kad and KadU corresponds to samples where a peer contacts the same destination host more than once within a sample time window. We found that 94% of the interesting samples for KadU dataset were generated by only two hosts. In the following, we focus our analysis on one of the hosts which we call $h1$, with the results being similar for the other host.

We found that $h1$ generated a large number of flows to two servers, namely $Server1$ and $Server2$. Figure 10 shows the num-
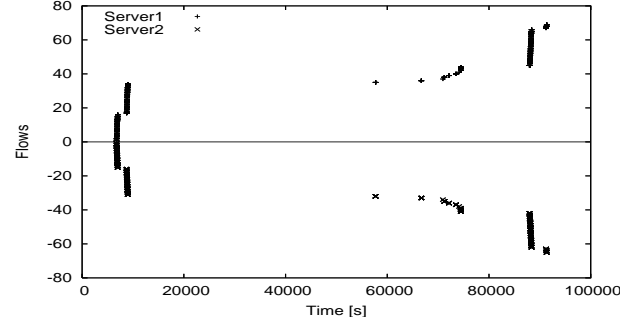
ber of connections $h1$ initiated to these servers during the whole trace. The X axis shows the connection start time, and the Y axis shows the connection ID. Positive IDs show connections opened to $Server1$, while negative IDs show connections opened to $Server2$. The average connection duration is 15 and 8 seconds respectively. For periods when the host was active, the inter-connection time to both servers is relatively small, i.e., 51 and 63 seconds for $Server1$ and $Server2$ respectively.

To further understand this behavior, we searched for information on the IP address of both servers and found that $Server1$ was reported as a *fake server* and $Server2$ was reported as a *full server* [1]. A fake server pretends to be a legitimate eMule server to fool clients with the goal of spying on them and to inject false information to disrupt the P2P system. These servers might be planted by parties such as the RIAA (Recording Industry Association of America) [5]. Fake servers may also impact the performance of victim peers since such peers cannot exploit the eMule network to search and exchange content. A full server is a legitimate server that has reached the maximum number of clients it can serve, so that further requests are denied. We believe the list of servers that host $h1$ has is limited, and possibly contains $Server1$ and $Server2$ only. This would result in $h1$ persistently initiating connections to both servers.

Considering the Kad dataset, the methodology pointed out an analogous problem. 92% of the interesting samples in the *avg-conn-per-IP-TCP* metric were generated by a single host. Once again, we found the host had a large number of connections to a particular server. Interestingly, we could not confirm from available manually maintained lists whether this host was a fake or full server, and we believe this is a hitherto unknown fake server. In general, we believe a traffic analysis approach such as ours can help in automatically identifying or inferring servers/peers with suspicious behavior, rather than relying entirely on manually maintained lists.

8.4 Other Interesting Findings

In this section we present some other examples of the findings highlighted by our methodology:

● *Inter ISP traffic - KadU*: As mentioned in Section 7.1, the metric *inout-ISP-to-Internet-ratio-TCP* for KadU shows a cluster in the range 0 to 0.62, with the majority of samples in the range 0 to 0.03. This represent clients where a large fraction of the connections was directed to peers in the Internet. In fact, 20% of the P2P traffic incoming to the MiniPoP is sent from the Internet. While some of the behavior is caused by clients that are searching for content not present in the KaU network, we believe there are several clients not using the KadU network to search. This is an undesirable behavior

considering that the KadU developers optimized KadU to maintain P2P traffic within the ISP.

• *Abnormal behavior with "buddy" maintenance mechanisms - KadU*: The metrics *outin-avg-conn-per-IP-TCP* and *outin-total-conn-attempts-UDP* highlighted an atypical region for which a host was receiving a lot of TCP and UDP flows in the KadU dataset. By investigating the anomalous samples for these metrics, we have found a single host which was responsible for 57% and 33% of interesting samples respectively. We looked further and found that a single external peer opened 825 TCP connections and $1,678$ UDP connections to this host in a 25 hours period. Looking at the message type exchanged among these two peers, we discovered that messages were related to the eMule "buddy" mechanism. To allow a client C behind a (home) NAT to upload content, C finds a "public" peer (or buddy) who forwards requests from other clients to it. Then, C can directly initiate a connection to the client requesting the content. Normally, clients behind a NAT use a single TCP connection to the buddy. The large number of connections initiated by this particular client is therefore atypical, and points to incompatibilities between the Kad/KadU protocol and the (home) NAT/Firewall, which repeatedly closes the connections.

• *Isolating very active peers - Kad and KadU*: Our methodology pointed out potentially interesting peers which account for a large number of interesting samples in several metrics. We isolated the hosts responsible for more than 10% of the interesting samples for at least 5 metrics, finding 3 KadU peers and 6 Kad peers. For example, a client was generating many interesting samples for the metrics *live-conn-TCP*, *inout-total-conn-attempts-UDP* and *bps-rcvd-TCP*, which show the host was aggressively searching and downloading content. Similar results were observed for other clients. While we did not find evidence of malicious activity, we believe our methodology was able to isolate very aggressive behavior, which is important from the ISP point of view, and also for the end users, e.g., to avoid leacher behavior.

9. GENERALIZING TO OTHER SYSTEMS

While much of our analysis is conducted with Kad, and KadU given their predominant usage in the ISP network, we are extending the analysis to consider other popular P2P systems. In this section, we present preliminary results reporting our findings with the BitTorrent [12] and DC++ [17] systems. Given these systems are not as widely used in the network, our analysis is conducted on a separate one-week long trace so sufficient data samples may be obtained.

• *Idle TCP connections in BitTorrent*: Our methodology showed that clients have a high *fract-unanswered-appl*, i.e., a large fraction of successful TCP connections to which the contacted peer never replied. We found that more than 40% of the samples are in the range 0.6 to 1. These connections are typically short lived, with more than 90% of them lasting less than 30 seconds. We believe this occurs when a client contacts a peer that is no longer sharing the file being searched. This is another example of how stale information leads to wasted network resources.

• *Unnecessary P2P traffic in BitTorrent*: Similar to Kad and KadU, we found that stale information and NAT presence could account for a large fraction of unanswered UDP flows in BitTorrent. 32% of all samples and 10% of all UDP flows were sent to hosts that left the P2P network. In addition, for BitTorrent we noticed that from the 4.2 million UDP flows initiated in the PoP, more than half are unanswered which can also be due to stale membership and NATs.

• *DDoS attack exploiting DC++*: We noticed that the interesting region for the metric *inout-1024-dest-ports-TCP* ranges from 2 to 9 ports contacted in a time slot. Further investigation showed that many of these connections were targeted to port 80 and did not receive a response from the destination. Manual inspection showed that the contacted IPs were real web servers and not DC++ clients. We hypothesize these flows are part of a DDoS attack exploiting DC++. Attacks of this nature have been previously reported [32]. In addition, we found that 95% of the DC++ connections stay within the ISP. Of the connections that leave the ISP, 21% are destined to ports below 1024 and potentially contribute to DDoS attacks, as described above.

10. RELATED WORK

Many recent works have focused on P2P traffic classification. In general, two main approaches have emerged: packet inspection techniques [22,28] and behavioral classification techniques [10,15, 21, 23, 24, 29]. Our work aims at analyzing the subset of traffic which has been already classified as P2P to identify any undesirable behavior these systems might have.

Anomaly detection of network traffic in general (for example, [8,25,26,35]), has been widely studied. Many of these works have developed automated techniques for detecting anomalies. The techniques typically leverage the fact that most data-points are normal, and flag anomalies based on sudden and significant deviations from baseline values. Our work differs in several ways. First, our focus is on obtaining better understanding on the types of undesirable behavior that P2P systems in the wild exhibit. Undesirable behavior can be predominant, and anomaly detection techniques based solely on deviations from baseline behavior are not sufficient in our context. This led us to rely on domain knowledge as part of our analysis. Second, our notion of undesirable behavior is broad, and includes not only malicious activities, but also many other patterns of undesirable behavior peculiar to P2P systems, for e.g., wasted resources caused by NATs and stale information in the system (Section 8.2). Third, we have considered a much wider range of traffic features than typical anomaly detection work, given limited a priori knowledge of the types of undesirable behavior that P2P systems exhibit. That said, it would be interesting to develop more automated analysis techniques for the identification of undesirable behavior in P2P systems in the future.

Our work both corroborates known patterns of undesirable behavior in P2P systems, and provides more insights into them. In particular, our findings on DDoS confirm recent works where researchers showed the feasibility of exploiting P2P systems to launch DDoS attacks on the Internet [7, 9, 18, 30]. While these works proposed attack heuristics and showed the feasibility of attacks, our work is one of the first to show evidence of real attacks taking place in the wild. Our findings on fake servers similarly support [5]. Our results (Section 8.3) have not only shown peers impacted by well-known fake servers [3, 31], but also shown the potential to automatically detect hitherto unknown fake servers.

11. CONCLUSIONS AND DISCUSSION

As a primary contribution of this paper, we have shown that P2P systems in the wild exhibit many types of undesirable behavior, and we have provided insights into the prevalence, characteristics and impact of such behavior. We have also shown the potential of a systematic approach involving P2P traffic analysis in uncovering such behavior. Our results include instances where the performance of the P2P system itself may be impacted (e.g. due to maliciously deployed servers), as well as examples where P2P system behavior can be detrimental to the network (e.g. DDoS attacks exploiting P2P systems, or unwanted traffic due to hosts behind NATs and

stale group membership). While there has been some prior aware-ness of these issues in the community, to our knowledge, our is the first work that systematically studies P2P traffic patterns with a view to identifying the undesirable behavior they exhibit.

Our analysis suggests that undesirable behavior may be exhibited by a range of P2P systems. Further, most examples of undesirable behavior that we found point to intrinsic design limitations in the underlying systems themselves, which leads us to believe that our findings are likely to hold if the systems are analyzed in other net-works as well. That said, generalizing the findings across multiple networks, and a wider range of P2P systems is an important aspect of our ongoing work.

In this paper, we have adopted a semi-automated methodology that combines data-mining with extensive use of domain knowl-edge in interpreting the results. This has been necessitated given that there is limited understanding in the community today on the characteristics of undesirable behavior that P2P systems may ex-hibit, and since the intrinsic heterogeneity of P2P traffic makes it hard to clearly distinguish undesirable behavior from normal usage. Undesirable behavior can be predominant, complicating the use of automated techniques which identify anomalous behavior by de-tecting significant deviations from normal values. An interesting avenue for future research is exploring more automated analysis techniques, for instance based on identification of significant shifts in P2P system behavior across networks and across time, and by employing rules general across an entire class of P2P systems.

12. ACKNOWLEDGEMENTS

13. REFERENCES

[1] eMule forum :: Fake Server List And Ip Numbers. http://forum.emule-project.net/index.php?showtopic=120066.

[2] eMule forum: Repeated Kad Errors. http://forum.emule-project.net/index.php?showtopic=133799.

[3] I-BlockList. http://www.IBlocklist.com.

[4] Is the Skype outage really a big deal? http://news.cnet.com/8301-10784_3-9761673-7.html.

[5] A. Banerjee, M. Faloutsos, and L. Bhuyan. The P2P war: Someone is monitoring your activities. In *ScienceDirect*, pages 1272–1280, 2008.

[6] Adunanza. http://www.adunanza.net/.

[7] E. Athanasopoulos, K.G.Anagnostakis, and E. Markatos. Misusing Unstructured P2P Systems to Perform DoS Attacks: The Network That Never Forgets. In *ACNS*, 2006.

[8] P. Barford, J. Kline, D. Plonka, and A. Ron. A signal analysis of network traffic anomalies. In *IMW*, 2002.

[9] S. Bellovin. Security Aspects of Napster and Gnutella. Invited Talk at USENIX Annual Technical Conference, 2001.

[10] L. Bernaille, R. Teixeira, and K. Salamatian. Early Application Identification. In *CONEXT*, 2006.

[11] R. Birke, M. Mellia, M. Petracca, and D. Rossi. Understanding VoIP from backbone measurements. In *INFOCOM*, 2007.

[12] BitTorrent. http://www.bittorrent.org.

[13] K. Cho, K. Fukuda, and H. Esaki. The Impact and Implications of the Growth in Residential User-to-User Traffic. In *SIGCOMM*, 2006.

[14] CISCO. Cisco IOS NetFlow. http://www.cisco.com/web/go/netflow.

[15] M. Collins and M. Reiter. Finding Peer-To-Peer File-sharing Using Coarse Network Behaviors. In *ESORICS*, 2006.

[16] F. Constantinou and P. Mavrommatis. Identifying Known and Unknown Peer-to-Peer Traffic. In *NCA*, 2006.

[17] DC++. http://dcplusplus.sourceforge.net/.

[18] K. Defrawy, M. Gjoka, and A. Markopoulou. "BotTorrent: Misusing BitTorrent to launch DDoS attacks". In *SRUTI*, 2007.

[19] eMule. http://www.emule-project.net.

[20] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *KDD-96*, 1996.

[21] M. Iliofotou, P. Pappu, M. Faloutsos, M. Mitzenmacher, S. Singh, and G. Varghese. Network Monitoring using Traffic Dispersion Graphs (TDGs). In *IMC*, 2007.

[22] IPP2P. http://www.ipp2p.org.

[23] T. Karagiannis, A. Broido, M. Faloutsos, and K. Claffy. Transport Layer Identification of P2P traffic. In *IMC*, 2004.

[24] T. Karagiannis, D. Papagiannaki, and M. Faloutsos. BLINC: Multilevel Traffic Classification in the Dark. In *SIGCOMM*, 2006.

[25] B. Krishnamurthy, S. Sen, Y. Zhang, and Y. Chen. Sketch-based change detection: Methods, evaluation, and applications. In *IMC*, 2003.

[26] A. Lakhina, M. Crovella, and C. Diot. Mining Anomalies Using Traffic Feature Distributions. In *SIGCOMM*, 2005.

[27] P. Maymounkov and D. Mazieres. Kademlia: A peer-to-peer information system based on the XOR metric. In *IPTPS*, 2002.

[28] M. Mellia, R. L. Cigno, and F. Neri. Measuring IP and TCP behavior on edge nodes with Tstat. *Computer Networks*, 47(1):1–21, 2005.

[29] A. Moore and D. Zuev. Internet Traffic Classification Using Bayesian Analysis Techniques. In *SIGMETRICS*, 2005.

[30] N. Naoumov and K. Ross. Exploiting P2P systems for DDoS attacks. In *International Workshop on Peer-to-Peer Information Management*, 2006.

[31] Phoenix Labs. PeerGuardian. http://phoenixlabs.org/pg2/.

[32] Prolexic. http://www.prolexic.com/content/moduleId/tPjJLKRF/article/aRQNVcBH.html.

[33] J. Rosenberg, J. Weinberger, C. Huitema, and R. Mahy. Simple Traversal of User Datagram Protocol Through Network Address Translation STUN, 2003. RFC-3489.

[34] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Pearson, 2006.

[35] Y. Zhang, Z. Ge, A. Greenberg, and M. Roughan. Network anomography. In *IMC*, 2005.

[36] M. Zhou, Y. Dai, and X. Li. A Measurement Study of the Structured Overlay Network in P2P File-Sharing Applications. In *ISM*, 2006.