# Observations of IPv6 Traffic on a 6to4 Relay

Pekka Savola
CSC/FUNET, Finland
psavola@funet.fi

## ABSTRACT

FUNET has been operating a public, globally-used 6to4 (RFC 3056) relay router since November 2001. The traffic has been logged and is now analyzed to gather information of 6to4 and IPv6 deployment.

Among other figures, we note that the number of 6to4 capable nodes has increased by an order of magnitude in half a year: in April 2004, there are records of about 2 million different 6to4 nodes using this particular relay. Vast majority of this is just testing the availability of the relay, done by the Microsoft Windows systems, but the real traffic has also increased over time.

While the observed 6to4 traffic has typically consisted of relatively simple system-level applications, or applications by power users, the emergence of peer-to-peer applications such as BitTorrent was also observed.

## Categories and Subject Descriptors

C.2.3 [**Computer Systems Organization**]: Computer communication Network—*Network Operations*

## General Terms

measurement

## Keywords

IPv6, 6to4, IPv6 transition.

## 1. INTRODUCTION

6to4 (RFC 3056 [1]) is an IPv6 transition mechanism which embeds the IPv4 address in the IPv6 prefix (e.g., 2002:0101:0101::/48 for the IPv4 address 1.1.1.1), and specifies automatic IPv6-in-IPv4 tunneling between 6to4 routers or hosts.

The mechanism can be used by the IPv6-capable hosts or sites which have a public IPv4 address when the ISP is not providing IPv6 connectivity, or by IPv6-connected nodes to create a "direct path" over IPv4 to any 6to4 node. The communication between 6to4 sites is tunneled directly, but connectivity to other IPv6 networks must be obtained through the use of 6to4 relay routers. 6to4 routers or hosts often default to using the global anycast address, 192.88.99.1; this way, 6to4 routers or hosts do not need configuration, always using the closest advertising relay to reach the rest of the IPv6 networks.

As 6to4 encapsulates IPv6 packets directly on top of IPv4, using protocol 41, 6to4 does not operate through Network Address Translators (NATs). For that, a different automatic tunneling mechanism (such as Teredo [2]) must be used. Obviously, upgrading the NAT to also support IPv6 (possibly including 6to4) is another fix in that scenario.

FUNET (Finnish University and Research Network) has been operating a public 6to4 relay since November 2001, advertising the anycast prefixes (2002::/16, 192.88.99.0/24) globally. The packets sent and received by the relay have been logged for the whole time, and this study analyzes these logs to gather information of 6to4 and IPv6 deployment.

The relay works in both directions: the 2002::/16 prefix attracts traffic from "native" IPv6 sites toward all 6to4 destinations, and the 192.88.99.0/24 prefix pulls in traffic from the 6to4 nodes. It is worth noting that advertising more specific routes of 2002::/16 is not allowed[1], so every advertising router is acting as a tunnel towards all the 6to4 sites.

The use of NAT has become very commonplace; those users can't use 6to4. Similarly, the direct traffic between 6to4 nodes does not show in this analysis as it does not go through the relay; some implementations also include a "local relay" on even the "native" IPv6 hosts, reducing the use for the 6to4 relays in the network. Also, one must note that dozens of public 6to4 relays exist in the Internet, so only a relatively small portion of the whole traffic is observed by our relay [4].

## 2. 6TO4 RELAY DEPLOYMENT

Malone [4] has studied the number of 6to4 relays in the Internet. If a relay advertises the anycast prefixes, 192.88.99.0/24 [3] and 2002::/16, the topology and routing policies determine the "active range" of the relay. For example, as long as the BGP route advertised by Autonomous System (AS) X remains the preferable in the remote autonomous systems, they use the 6to4 relay located at AS X. Unless explicitly configured otherwise, the first relevant tie-breaker in the BGP path selection process is the length of the AS-path which then determines that the remote AS's which have no 6to4 relay of their own use the one closest to them topology-

and policy-wise.

In other words, the topology, policies, and where the other relays are deployed determine the number of nodes which use the relay. These nodes can be IPv6 nodes wishing to reach 6to4 nodes, or 6to4 nodes wishing to communicate with non-6to4 nodes, depending on the advertisement.

## 2.1 6to4 Relay at FUNET (AS1741)

The organization the author is associated with, FUNET (Finnish University and Research Network), has provided a PC-based 6to4 relay service since November 2001. For the whole time, it has advertised both 192.88.99.0/24 and 2002::/16 globally.

There is another public 6to4 relay in Finland, operated by Song Networks; a 6to4 relay in Estonia also attracts some traffic from Finnish sites. There are some more private, non-advertised 6to4 relays as well.

The prime areas where we have received traffic from has been our own network (obviously), Finland, other Nordic countries (Sweden, Norway, Denmark, etc.), and surprisingly also from further in the Internet, particularly from commercial sites, especially from the Unites States through our upstream provider's (NORDUnet) commodity transit links. On the other hand, most traffic from the other research networks in the world end up in different relays, especially the 6to4 relay operated by SWITCH in Switzerland, connected through the European research network GEANT, as many research networks are closer to that relay topology-wise.

The traffic of our 6to4 relay has been quite modest; in August 2004, the "steady state" was only about (a relatively constant) 300-500 kbit/s at 50-100 packets per second. However, quite often there are peaks to 10 Mbit/s and even beyond. Being well-connected, we have not observed any Denial-of-Service attacks, even though some other relay operators have stated that IPv6 DoS attacks have taken place, and in the process clogged up the relays.

We started storing daily snapshots of the aggregate amount of packets and bytes in 2004. Table 1 presents the summary of the traffic from 6to4 nodes to the relay (adjusted to remove 20 bytes of overhead from the packets), and from the relay to 6to4 nodes.

**Table 1: Packets and bytes transferred in April 2004**

| | From 6to4 | To 6to4 |
|---|---|---|
| Total packets in $10^6$ | 71.4 | 53.7 |
| Total bytes in $10^9$ | 26.8 | 22.8 |
| Average packet size (B) | 375 | 424 |
| Average packets/second | 27 | 20 |
| Average kbits/second | 83 | 70 |

Table 2 presents further breakdown of the traffic from the relay to 6to4 nodes to the amount of replies to Windows probing (as described in Section 4.1), "TCP established" traffic, and the rest.

Malone's [4] analysis of 6to4 relay routers places our relay among the smallest ones, measured with certain metrics (e.g., the 2002::/16 advertisement). With some others (e.g., the 192.88.99.0/24 propagation), it may be a bit more significant.

# 3. DATA COLLECTION

**Table 2: Breakdown of packets out to 6to4 in April 2004**

| | Probe replies | TCP est. | the rest |
|---|---|---|---|
| Total packets in $10^6$ | 30.4 | 22.6 | 0.677 |
| Total bytes in $10^9$ | 1.70 | 21.0 | 0.081 |
| Average packet size (B) | 55 | 929 | 119 |
| Average packets/second | 11.7 | 8.7 | 0.3 |
| Average kbits/second | 5.2 | 65 | 0.25 |

## 3.1 Background

When the 6to4 relay was installed in late 2001, the data collection was added as an afterthought, to get a round idea what's going on through the relay. The considerations for selecting the tools were:

- We did not even intend to analyze the data, much less publish any results,
- Extensive logging would have raised legal concerns, which would have taken some time settle,
- The data collection could not be too CPU-intensive, because the system was a 200 Mhz Pentium,
- The data collection could not require much storage, because the system only had 4 GB of disk space, and
- The data collection could not be complex or require implementation, because the relay was just an "on-the-side" hobby; either there would be no logging at all, or the logging would have to be very simple.

These considerations led to the use of syslog function of "ipfw" and "ip6fw". We also considered tcpdump, but it could not meet the contraints for the logging tool.

Looking back, one should obviously have spent more time on designing a better data collection methodology, and chosen better tools such as a Netflow-like [7] mechanism or something like NeTraMet [8].

## 3.2 Methodology

The traffic on the relay has been logged since 2001 using the syslog function of the in-built firewalling packages "ipfw" and "ip6fw". This logs only the IP addresses and TCP/UDP ports or ICMP types and codes.

Over two years later, when we started looking at the logs at more length, we noticed that the syslog daemon was apparently not capable of logging such amounts (or bursts) of traffic. Sometimes log lines seemed to get corrupted.

The most cases of log line corruption were obvious, such as a line being cut off prematurely in the middle. We wrote a script which checked the logs for syntactical correctness, and excluded corrupted lines from the analysis. The potential effect of corruption is analyzed in the next section.

In June 2004, we and Malone ported and merged syslog scalability fix es from Linux to FreeBSD codebase, implementing buffered logging – i.e., not having to do fsync() after each syslogged line. This does not help with the existing data, but appears to have fixed the most dire problems for future studies.

Tcpdump was also used for the duration of a day to gain information about the number of hops ip-proto-41 packets had to traverse before reaching the relay, as described in section 4.3.

Aggregate amounts of traffic in section 2.1 were taken from ipfw and ip6fw rule counters. Ip6fw counters are still 32 bits, overflowing regularly, which was an annoyance for the analysis.

## 3.3 Applicability Analysis

Figure 1 shows the number of all log lines in a month and the number of lines where corruption was detected, in log scale.

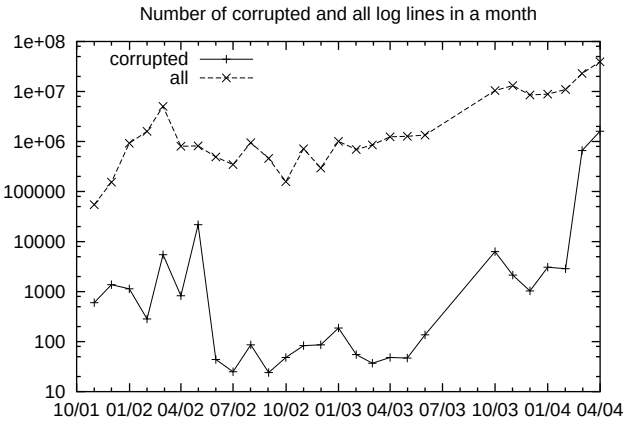


**Figure 1: The number of corrupted and all log lines, in log scale**

From the figure we see that the amount of corruption is typically 0.0% or 0.1%, and it is over 1 percent only in November 2001 (1.1%), May 2002 (2.7%), March 2004 (2.9%), and April 2004 (4.1%).

The number of log lines logged in March 2004 (about 23 million) was still reasonably modest, about 8.9/second in average. We do not believe this in itself could be causing problems. On the other hand, we believe the corruption may have been primarily caused by frequent bursts of UDP traffic (as TCP "established" is not logged). For example, a burst of 10 Mbit/s with the packet size of 1280 bytes could be generating about 1000 lines of log per second.

Based on this analysis and assumption, we believe that the logs are still reasonably accurate despite the shortcomings of the methodology. The current estimates of the number of 6to4 hosts is an underestimation as the corrupted log lines had to be excluded.

# 4. TRAFFIC ANALYSIS

## 4.1 Microsoft Windows Probing

### *4.1.1 The Phenomenon*

One particularly interesting thing to note is how many Microsoft Windows systems have used the 6to4 because many systems already ship with IPv6 support, and 6to4 support in particular. It is trivial to identify Microsoft systems, as their implementation forms 6to4 addresses by embedding the IPv4 address also as the IPv6 Interface Identifier, rather than just in the prefix. For example, with IPv4 address 1.1.1.1, Microsoft's 6to4 implementations use IPv4 addresses like 2002:0101:0101::0101:0101, where other systems use different identifiers and subnet numbers.

But that is not all; the Microsoft 6to4 implementation also implements what we call "relay probing" or "probing" in short. At predetermined intervals, it sends specific kind of packets to all the 6to4 relays in its "potential relay list"; this happens every time the IPv4 addresses or routes change, or when the timer (by default, 24 hours) expires [5].

The probing packet is an ICMP echo request with IPv6 Hop Limit set to 1, so it will not be forwarded; the IPv6 destination address is formed using the same algorithm and it may or may not exist. The implementation only expects to receive any ICMP message back (whether an Echo Reply, Destination Unreachable, Time Exceeded, or whatever) [5]. Based on that input, it picks the relay with best response time if it received replies from multiple relays. The source address of the reply seems to be used as the gateway address[1]. In the case of our relay the address does not exist, and an ICMP time exceeded message is sent back to the probing 6to4 node.

It is also interesting to note that there are nodes which probe 1000, 10000 or even 100000 times a month – up to every 5-30 seconds, repeatedly, without giving up. This seems to be most likely caused by changes in their IPv4 routes (or internal addresses) which triggers the probing process [5].

In any case, the probing process is being considered to be changed so that if the node has only one relay candidate, the node expects the relay to be available, and will not probe [5]. Note that this does not affect the count of the systems in section 4.1.2 because that does not consider how many times a host has probed, just whether it has probed or not.

### *4.1.2 Counting the 6to4 Windows Systems*

Based on the identification process described above, it is possible to measure the number of Microsoft Windows systems which have enabled IPv6 within the topological range of the relay.

This is presented in figure 2 (in logarithmic scale). The Y-axis represents the number of distinct IPv4 addresses from which a probe has been received. This is close, but not quite the same thing as the number of distinct probing hosts. 6to4 nodes with a dynamic address result in a slight overestimation of the "host count"; unfortunately, it is impossible to measure how large this overestimation actually is.

As is obvious from the figure, the number of topologically close 6to4 users in Windows systems has changed dramatically during the last months of the analysis: in a year, the number has risen from about 100,000 to over 2 million.

Such a high number is quite interesting bearing in mind that the analysis is only done in one, relatively remote 6to4 relay. If dozens of relays were experiencing the similar (or even higher) probing rates, the total number of IPv6-capable 6to4 systems could be very large. Malone was able to count at least 46 relays world-wide in January 2004 [4], and most were more significant than ours. So, if the count of Windows systems at our relay was 2 million, and most of the about 50 relays had more users while some might have fewer, the number of 6to4 users could be even over 100 million. The number of potential IPv6 users may be even higher, because

[1]Yes, this does create problems if the the relay's 6to4 address is configured as an IPv6 anycast address, and the source address ends up being a non-6to4 address!

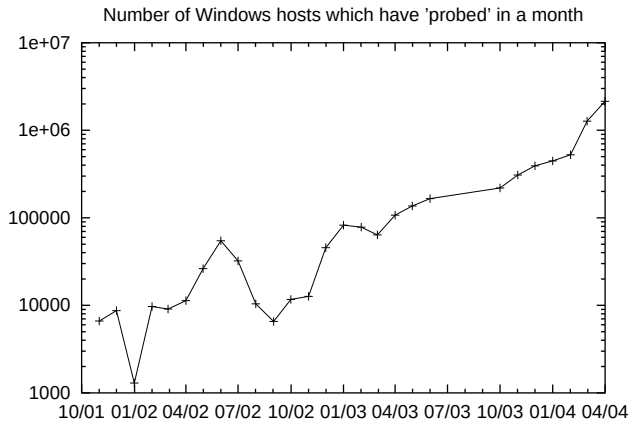


**Figure 2: The number of probing Windows 6to4 nodes per month, in log scale**

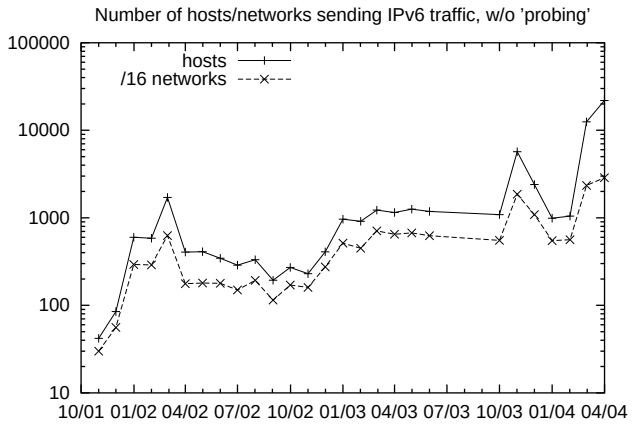


**Figure 3: The number of active 6to4 nodes per month, in log scale**

this does not include those systems which are behind a NAT where Teredo [2], for example, could be used instead of 6to4.

Microsoft's strategy [9] is apparently to push the IPv6 capability to all the nodes in the network, and to create a common platform on which application writers could create IPv6-only applications (e.g., for peer-to-peer systems) instead having to implement NAT traversal mechanisms for IPv4 on their own.

We were unable to see strong correlation between Microsoft's product releases and the number of Windows 6to4 hosts. The following events were noted: Windows XP release (October 2001), Windows XP service pack 1 (September 2002), ThreeDegrees peer-to-peer beta launch (February 2003), Windows 2003 server release (March 2003), and Advanced Network Pack for XP release (July 2003) [9]. The release of XP SP1 and ThreeDegrees service, however, apparently did start a longer term increase of Windows 6to4 hosts.

However, the Windows XP launch in October 2001 shows a start of a strong growth of IPv6-using hosts, as seen from figure 3.

## 4.2 Counting the Active 6to4 Nodes

The figures for "probing" are quite interesting but the figures of actual use of the relay are at least as important.

The number of distinct nodes (or more precisely, different IPv4 addresses) which did more than just probed the relay are presented in figure 3 (in logarithmic scale).

This, again, is quite interesting. It shows that for every 6to4 node actually communicating through the relay, about hundred are either completely idle, or only communicating directly between 6to4 nodes. This shows huge IPv6 potential waiting to be unleashed by IPv6 applications.

For reference, the figures also show the number of /16 IPv4 prefixes compared to the /32 host addresses. In other words, the figures show how many 6to4 nodes are typically active in an old "B-class" sized network[2]. There seems to be a slight trend for these two lines to deviate, i.e., the number of 6to4 user per network to rise, implying 6to4 getting more commonplace.

The steady number of active 6to4 relay users has risen to about 1,500 per month, but there are several peaks to higher values as well. It remains to be seen whether the latest rise in March and April 2004 is a stable rise or a temporary spike.

Again, these graphs do not show the traffic between 6to4 sites.

In both figures 2 and 3 there is some amount of discontinuity. We believe that the amount of 6to4 use from our relay's perspective is a function of roughly three components: the amount of 6to4 users, the amount of other 6to4 relays and their advertisement propagation, and the amount of log corruption. We believe that it is impossible to make exact estimates on this data in the face of this many variables, but instead just have to state that 6to4 use has been growing significantly.

## 4.3 The Range of 192.88.99.0/24 Advertisement

A sufficiently large number and wide distribution of relays is essential for the success of 6to4. Too few relays would result in an increased load in the few relays and less incentive to deploy one. Uneven distribution would mean that some users' traffic would experience a significant increase in round-trip time, which would be very undesirable.

The number of relays has been analyzed by Malone [4]. In this section, we analyze how far away in the network the users of our relay are, to get a measure of topological and geographical distribution of the relays from our perspective. This was done by analyzing the proto-41 packets sent to the relay by 6to4 nodes. From there, we looked at the IPv4 TTL field: if we could guess the packet TTL used by the sender, we could estimate the "range" of the 192.88.99.0/24 advertisement in IPv4 hops.

Microsoft systems default to using TTL=128, some others (such

[2]A more accurate methodology to measure networks would have been to map the IP addresses to AS numbers based on the global routing table (about 150,000 prefixes) instead of just looking at the first half of the address. However, we considered that fine level of detail irrelevant for the scope of this study.

as Linux) use TTL=64, and a few others (e.g., BSD derivatives) use TTL=40; a few have configured TTL to 255 as well; on some occasions, users have tuned these values (e.g., setting TTL=100).

The number of hosts using a different TTL than 64 or 128 was unacceptably low (around half a dozen hosts) and were excluded. Likewise, predicting the original TTL which was used after 20 hops became too unpredictable, so 20 was chosen as the upper bound of analysis.

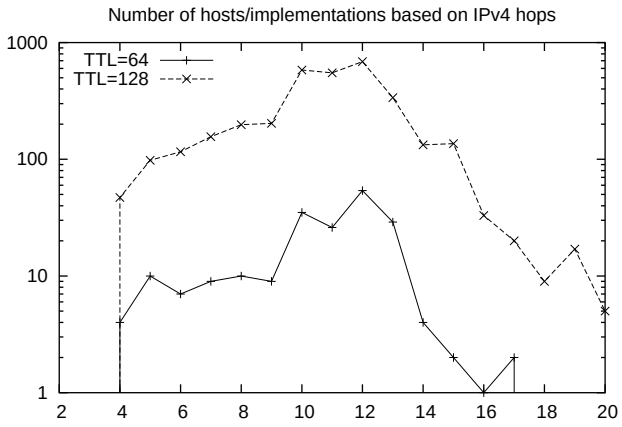


**Figure 4: Nodes with the number of IPv4 hops, in log scale**

Figure 4 displays the number of unique 6to4 nodes, measured during a 24 hour period in March 2004 from nodes that sent over 5 packets to the relay, with respect to the number of hops traversed to reach the 6to4 relay. The different (assumed) base TTL's have been graphed separately. This also gives a feel on the respective ratios of the 6to4 implementations used in the Internet: TTL=128 (Microsoft) appears to be 10-15 times as widely used as TTL=64 (Linux and probably many others).

It is also interesting to note that the number of 6to4 nodes jumps up at 10 hops. That seems to correspond to the typical hop count towards the U.S. from this particular relay, the lower hop counts being mostly traffic from the closer topological areas. The effective maximum range is around 16 hops – then even in the U.S., a closer relay could be found.

It is also worth noting that those implementations (such as Linux) which require manual configuration seem to be relatively more used with the lower hop counts. That is, those who explicitly toggle on 6to4 seem to be aware that 6to4 relay service is available close by. On the other hand, the curve for TTL=128 is rising monotonically also with the low hop counts 4-9.

## 4.4 The 6to4 Nodes' Use of Applications

We also analyzed the apparent applications used through the 6to4 relay by examining the ICMP types/codes, and all the TCP/UDP port numbers where either the source or destination port could be identified to be commonly used by an application.

Figure 5 depicts the number of distinct hosts (or more precisely, IP addresses) which have used a specific kind of application during April 2004. It shows the number of all applications, and the application "services" run on 6to4 systems (more of this below).

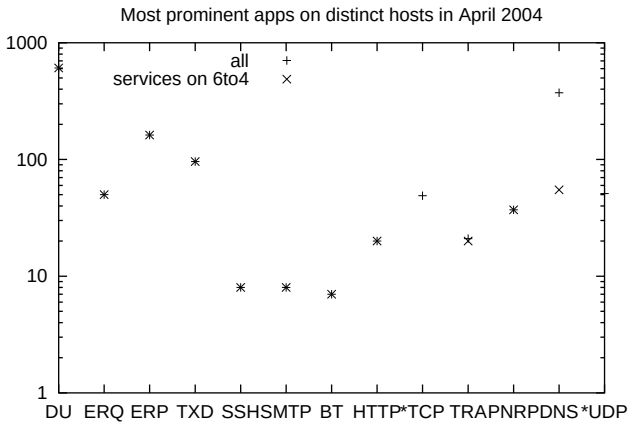


**Figure 5: Applications run by unique 6to4 nodes, in log scale**

The first four columns correspond to ICMPv6 messages: Destination Unreachable (DU), Echo Request (ERQ), Echo Reply (ERP), and Time Exceeded (TXD). The next columns correspond to typical TCP messages: SSH, SMTP, BitTorrent peer-to-peer filesharing (BT), HTTP and unidentifiable TCP. The last four columns correspond to UDP messages: traceroute (TRA), Microsoft's Peer Name Resolution Protocol (PNRP; a peer-to-peer rendezvous service [6]), DNS queries, and unidentifiable UDP.

The first interesting observation is that very few 6to4 users only use an application on native IPv6 hosts (and not provide the same application to native IPv6 users) is very small. Almost all the applications have been set up as services on 6to4 hosts as well, and in consequence, most points in figure 5 overlap.

The only major exception here is DNS: quite a large number of 6to4 nodes use it for name resolution only, rather than providing DNS server over IPv6.

It is interesting to note that DNS queries appear to actually be the most prominent application at the moment; "ping" also being common. However, these could be classified system level applications, which are automatically enabled whether controlled by the user or not.

It is also good to note the emergence of peer-to-peer applications on top of IPv6: BitTorrent (also works with IPv4, of course), and Microsoft's PNRP.

Certain other applications were also identified, but were excluded from the figure if they had no more than 5 users: ICMP Packet Too Big (5 users in total, 5 of which on 6to4 nodes), IPv6-in-IPv6 tunneling (1 and 1, respectively), POP3 (3 and 2), IDENT (2 and 0), NNTP (1 and 0), IMAP (4 and 4), FTP-data (1 and 0), FTP (5 and 5), HTTPS (1 and 1), and IRC (2 and 2).

In the current usage, it seems that some users are using 6to4 (and IPv6) for remote services: for example, using SSH to connect to nodes at home, running an SMTP server at home, going around the service provider's SMTP restrictions, or for example running an FTP, HTTP, HTTPS, IMAP or POP3 server at home; we call these "power users".

### 4.5 Packets per Applications

It would be interesting to know how much each application is used on each host, i.e., the amount of flows, packets, or bytes per application.

Unfortunately, we did not have sufficient data to properly analyze the amount of bytes each application transferred, as TCP "established" packets were excluded, and such an analysis would be biased towards UDP applications with a high packet rate.

However, we are able to compare TCP applications to other TCP applications, and consider the DNS as the only identified (and simple) UDP application.

Most of the TCP applications were comparable to the ratios of hosts using such applications in figure 5. The amount of DNS and ICMP packets were also comparable to each other and the numbers in figure 5.

The only major thing to note was that there were a couple of relatively heavy applications, resulting in a lot of packets and flows, which were only used by a couple of hosts; for example, IMAP and HTTPS (assumably set up by "power users" to securely access services at 'home' while away).

## 5. CONCLUSIONS

It seems reasonable to conclude based on section 4.1.2 that the number of IPv6-capable nodes in the Internet has risen dramatically during the last years, and the last months in particular. This facilitates developing IPv6-only applications, leveraging the IPv6 benefits and facilitating the IPv6 deployment. As noted based on section 4.2, only a small fraction of the IPv6-capable nodes actually use IPv6 applications (at least through the relay) at the moment. The number has risen dramatically in March and April 2004, and it remains to be seen whether this is a sign of the more active use of IPv6 applications or just a temporary spike.

The analysis of the "range" of the advertisement of 192.88.99.0/24 indicates that there are still too few relays deployed, possibly due to financial reasons of providing public service to the third parties. For example, a significant part of commodity traffic from the U.S. seems to end up at our relay in Finland – over 10-15 hops away from the 6to4 nodes. That's hardly an acceptable long-term situation because it could the increase of the load of relays and alienate the users due to poor latency, throughput and quality. Malone [4] also analyzes the number of relays, and one can note that there are still too few of them.

The applications used by 6to4 nodes, as described in sections 4.4 and 4.5 are mostly well-known from IPv4. In general, 6to4 nodes didn't use services they weren't also themselves providing (for example, SMTP sessions were only initiated from 6to4 nodes which had an SMTP server); this seems to be a sign of early-adopter "power users". Also the slow emergence of peer-to-peer applications appears to be in sight, as noted by the use of BitTorrent and Microsoft's peer-to-peer naming/rendezvous services. We also noted that just observing the packet counts per application would not give realistic view what are the most used applications, as a few power users may still be able to dominate that figure.

As a lesson learned, one should always try to spend time in properly designing the data collection and processing methodology first, even though there were no plans to use the data at all. We noticed this would make analysis much easier later on if the data turned out to be interesting.

For future work, it would be very interesting to observe the similar patterns by setting up a Teredo [2] server, as the 6to4 cannot be used by users behind a NAT, which we estimate is a mainstream scenario for most home networks. It will also be interesting to continue this study especially if the amount of IPv6 usage starts rising.

## 6. ACKNOWLEDGEMENTS

David Malone and the anonymous reviewers are acknowledged for their feedback and suggestions for improvement.

## 7. REFERENCES

[1] Carpenter, B. and Moore, K., *Connection of IPv6 Domains via IPv4 Clouds*, RFC3056, February 2001.

[2] Huitema, C., *Teredo: Tunneling IPv6 over UDP through NATs*, draft-huitema-v6ops-teredo-03.txt, work-in-progress.

[3] Huitema, C., *An Anycast Prefix for 6to4 Relay Routers*, RFC3068, June 2001.

[4] Malone, D., *Counting 6to4 Relay Routers*, Unpublished. An earlier version is available at http://ops.ietf.org/lists/v6ops/v6ops.2004/msg00253.html

[5] Thaler, D. and Talwar, M., *Personal Communication*, May 2004.

[6] Microsoft, *About PNRP*, Microsoft Platform SDK, Peer-to-Peer Infrastructure; on MSDN web site.

[7] Cisco, *NetFlow*, http://www.cisco.com/go/netflow, referred May 2004.

[8] Nevil Brownless, *NeTraMet*, http://www.auckland.ac.nz/net/NeTraMet/, referred January 2005.

[9] Microsoft, *Microsoft's Objectives for IP Version 6*, http://www.microsoft.com/windowsserver2003/techinfo/overview/ipv6.mspx, referred January 2005.