VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

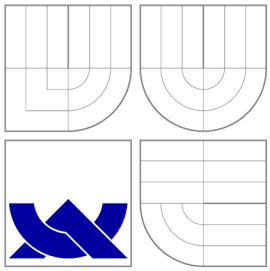
USER ACCOUNTING IN NEXT GENERATION NETWORKS

TEZE K DISERTAČNÍ PRÁCI
PHD THESIS NOTES

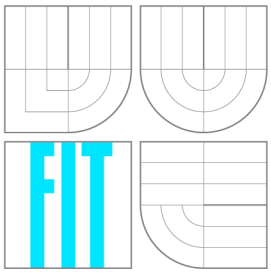
AUTOR PRÁCE
AUTHOR

Ing. MATĚJ GRÉGR

BRNO 2016



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

ÚČTOVÁNÍ UŽIVATELŮ V SÍTÍCH NOVÉ GENERACE

USER ACCOUNTING IN NEXT GENERATION NETWORKS

TEZE K DISERTAČNÍ PRÁCI
PHD THESIS NOTES

AUTOR PRÁCE — Ing. MATĚJ GRÉGR
AUTHOR

VEDOUCÍ PRÁCE — prof. Ing. MIROSLAV ŠVÉDA, CSc.
SUPERVISOR

BRNO 2016

Abstrakt

Velikost sítě Internet dosáhla takového rozměru, že globálně jednoznačná adresace všech připojených zařízení již není možná při zachování současné architektury TCP/IPv4. Tímto problémem se začalo zabývat již v 90. letech a od té doby bylo představeno několik návrhů nových architektur a síťových protokolů, které mají či měly ambice omezení adresace vyřešit. V současné době, v roce 2016, je jediným globálně nasazovaným řešením problému adresace protokol IPv6. Tento protokol zvětšuje velikosti síťové adresy čímž umožňuje adresovat téměř libovolné množství zařízení, ovšem za cenu nekompatibility se současným protokolem IPv4. Rozdílně se také staví ke způsobu automatické konfigurace koncových zařízení, proměnlivé velikosti síťové hlavičky a omezení nekompatibility řeší různými přechodovými mechanismy. Tato práce diskutuje dopady, které tyto změny mají na oblast monitorování a účtování uživatelů. Zejména změny způsobu konfigurace adresy vyžadují jiný přístup než v současných monitorovacích systémech, které ukládají pouze metadata o síťové komunikace pomocí protokolu NetFlow/IPFIX. Práce je zaměřena primárně na vyřešení problému účtování uživatelů v sítích kde jsou souběžně nasazeny protokoly IPv4 i IPv6, použity tunelovací přechodové mechanismy nebo překlad adres. Část práce je zaměřena na měření globálního vývoje a nasazení protokolu IPv6 mezi koncovými poskytovateli internetového připojení, poskytovateli obsahu a páteřními operátory.

Abstract

The number of devices connected to the Internet is such enormous that it is impossible to assign a globally unique address to every device in today's TCP/IPv4 architecture. Since the discussion how to solve the problem began in 1990s, there has been several proposals of new protocols and architectures trying to solve the problem. However, the only proposal that is widely deployed today is the IPv6 protocol. The IPv6 protocol enlarge the network address size, thus, it is possible to assign a globally unique IPv6 address to unlimited number of devices. Furthermore, the protocol introduces a paradigm shift, especially for address assignment and length of the IPv6 header. The IPv6 protocol is, however, incompatible with IPv4. To overcome the limitation, several transition mechanisms were proposed. The thesis discusses issues introduced by IPv6 protocol to user accounting process. In particular, it focuses on new approaches that can eliminate problems of current accounting methods that use NetFlow or IPFIX protocols. The aim of the thesis is to solve user accounting process for networks running a transition mechanism or a network address translation. Part of the thesis discusses the global IPv6 deployment and measures IPv6 penetration among content providers, internet service providers and transit operators.

Klíčová slova

NetFlow, NAT, IPv6, monitorování sítě, zavádění IPv6

Keywords

User accounting, IPv6, measuring, deployment, NAT, CGN, NetFlow, IPFIX

Citace

Matěj Grégr: User accounting in next generation networks, teze k disertační práci, Brno, FIT VUT v Brně, 2016

Contents

1

Introduction

The Internet has traditionally been used by research and educational organizations. End users had a free access to the network, with the actual costs absorbed by budgets of their research or government agencies [1]. Because of funding by research organizations and government, there has been little or no requirement to collect data at the level of an individual user, because an accounting of the user was not necessary. The network operators usually collected only aggregated data that were used for network management.

This model has been abandoned by the commercialization of the Internet in the 1990s. Commercial network operators need more detailed information to prevent violation of their service-level agreements or enforcing quality of service. The requirement to collect data at the level of an individual user has been increasing since then and the lack of detailed data usage for individual users becomes a serious shortcoming. Another reason for user accounting is given by legal requirements.

This fast growth of mobile devices in recent years together with the overall rapid growth of the Internet have a serious impact on the number of available IP addresses. With the IPv4 address exhaustion, the ISPs are forced to add support for the IPv6 protocol, which is, however, not backward compatible with the IPv4. This migration from one incompatible protocol to another is probably the second time in the history of the Internet, where the first was the transition from NCP to TCP/IP in January 1983 [2]. The incompatibility forbids the ISP to migrate to IPv6 only networks because the overall majority of content is available using IPv4 protocol only. The IPv6 protocol also changes the paradigm of address assignment for connected devices such as several IPv6 addresses per one interface, temporary addresses or different usage of DHCPv6 protocol. The ISP must monitor both protocols and cope with new features of IPv6, which is not an easy task, because there is a lack of information available, e.g., Is it possible to use similar approach for user accounting in IPv6 network as in IPv4 network?, Is it enough to just store an IPv6 address per user? How does the monitoring of dual-stack infrastructure scale in large networks?

1.1 Goal of the thesis and research questions

The thesis deals with the issues of user monitoring and accounting especially in next generation networks. The definition of a next generation network or future Internet is cumbersome because the term has been used from several point of views, e.g., a complete redesign of the Internet (clean slate approach) or an evolutionary approach. The thesis defines the future network as a network, which is incompatible with the today's network architecture

(TCP/IPv4). The example of a future network can be IPv6 protocol, but the thesis is not limited only to the IPv6. The aim of the thesis can be summarized in the following points:

- **Describe current approaches in user accounting.** The thesis summarize the background information needed for understanding the development presented in later chapters. Overview of address configuration techniques will be provided.
- **What are challenges and approaches to reach the next generation network?** With the incompatible architecture and several different transition mechanisms, the transition from the IPv4 protocol to the IPv6 protocol can be seen as an example of the transition to a next generation network. Understanding of the transition between IPv4 and IPv6 protocols can help us to understand more general questions. How to run an user accounting process in a potential network that come after IPv6? How long does it take to move from one incompatible network architecture to another one?
- **Which solution for users accounting in next generation networks can be used?** A different architecture comes with different requirements for user accounting. Novel approaches, how to handle these different requirements will be presented. Using transition between IPv4 and IPv6 as an example and solving the user accounting problem for this transition process, can bring us closer to a better understanding of the transition process between IPv6 and future network architecture.
- **Are the proposed techniques for accounting feasible for deployment and scalable?** The approaches used to cope with user accounting in a next generation network should be deployable and scalable. As a measurement of scalability, the solutions should be tested in a reasonable big network and the thesis should present statistics to support the statement of scalability and deployability.

1.2 Structure of the thesis

The objectives of this thesis are described in the previous section. This section describes the overall structure of the thesis and individual chapters.

- Chapter 2 discusses in detail issues connected with a transition of a network from one network protocol to another and how to measure the progress of the transition. IPv6 is used as an example of the future networking protocol. The chapter presents an analysis of IPv6 support in routing infrastructure, content distribution and among end users.
- Chapter 3 describes several issues connected with user accounting in dual-stacked networks. Transition techniques between IPv4 and IPv6 protocols are discussed as well. We show how we can build a scalable system that is able to provide necessary information for user accounting process in these networks.
- Chapter 4 presents issues with users accounting in networks, where network address translation techniques are used. We extend the system presented in chapter 3 to be able to account users even without the support of middleware that provides the translation.
- Chapter 5 concludes this thesis, describes main contributions and discusses the future work.

2

Toward Next Generation Network

"If you cannot measure it, you cannot improve it."
– William Thompson

The TCP and IP protocols were developed as protocols that could replace the NCP and the transition from NCP to TCP/IP started in 1982. This could be perceived as the first transition between two incompatible architectures. The NCP was officially obsoleted on January 1, 1983. All hosts (approximately 250) had to be TCP/IP capable till this date, or they could not connect to the network. This hard deadline is also called a flag day. IP protocol extended the 8 bits address to 32 bits where 10.0.0.0/8 was reserved for ARPANET nodes. The 32-bit address was perceived as large enough in the 1980s because no one could imagine such number of computers will ever exist.

In the early 1990s, it was obvious, that the Internet is growing much faster than anyone expected. The discussion over how the 32-bit IP address could be expanded began and took almost eight years. At the same time, another effort how to restrain IPv4 addresses was in full swing. This effort introduced several techniques - classless interdomain routing (CIDR) that made usage of address space much more efficient, and network address translation (NAT) that allowed multiple devices to share a single public address. These techniques preserved the global IPv4 address pool till 2011 when the Internet Assigned Numbers Authority (IANA) allocated the remaining last five `/8` address blocks of IPv4 address space to the Regional Internet Registries (RIRs). APNIC, LACNIC, RIPE NCC and ARIN already depleted their pools of addresses. The only remaining RIR is AFRINIC where the projected exhaustion date is in 2019. We are thus facing the second transition from one incompatible protocol to another. The situation is however very different from the previous one (NCP to TCP/IP).

The number of networks, devices and connected users using IPv4 protocol is tremendously big. Also the fact, that there are still IPv4 addresses available in several regions together with heavy usage of NAT technique and lack of features parity between IPv4 and IPv6 across software and hardware devices hold back the transition from IPv4 to IPv6 even more. The consequence is that every network device and network supports IPv4 protocol in these days, but the same is not true for IPv6. There are also additional costs connected with the transition. These additional costs and the fact that everything works right now with IPv4 lead to a situation that network operators do not deploy IPv6 and play a waiting game. This situation was well described in the Geoff Huston's paper *Is the Transition to IPv6 a „Market Failure?"* [4].

These two examples of transitions use different approaches for switching from an old to

a new architecture. Dual stack hosts and relays between NCP and TCP/IP architectures were used in the NCP to TCP/IP transition. The transition also had a hard deadline. The transition from IPv4 to IPv6 also uses dual-stacked hosts as a transition technique. However, there are many more different transition technologies and there are also several techniques how to prolong the operation of IPv4 architecture. There was also an attempt to set a transition plan in RFC 5211 [6]; unfortunately, it failed. The consequence is that the IPv6 transition is going much slower than it was expected. The another observation is that a flag day is not a viable option in such a large network as current Internet.

Furthermore, every network architecture that has been proposed so far uses clean slate design. The clean slate is an approach where everything is designed from scratch without maintaining compatibility with the previous architecture. It was the case of TCP/IP architecture, which was designed as incompatible with NCP. The same approach was also used with IPv6. Recursive InterNetwork Architecture (RINA), Content Centric Networking (CCN), Named Data Networking (NDN) are other examples of new networking architectures that are currently proposed. All these new architectures are, however, incompatible with IPv4/IPv6. We can expect that these architectures will face the same issues with migration as IPv6.

ISPs or content providers ask in essence the same following questions when they are thinking about migration to a new architecture.

- What is the actual number of content available over a new architecture?
- How is the availability of the content measured? Which dataset is used?
- If we deploy a new architecture, how many of our users will support it? How much traffic will flow over it? How much it will cost?
- How many users are using the new architecture?

The priority of these questions depends on the type of the network.The rest of this chapter tries to answer these questions using historical data, statistics and real experience from the transition of Brno University of Technology (BUT) network. A part of this chapter was submitted and presented at International Conference on Networking and Services [7] and International Conference on Network Protocols [8] conferences.

2.1 State of the Art

There are several approaches to measuring IPv6 adoption and quality of IPv6 service. Measurement can be performed on the content provider's side [9], [10],[12] . The analysis of requests can reveal the number of clients that can or cannot connect to dual-stack Web servers and their latency. This methodology measures clients. It shows, if IPv6 is supported by an application (web browser), operating system and client's ISP (Internet Service Provider). The numbers are between `9 - 11 %` in July 2016, as it is shown in Google's global IPv6 statistics[1]. There are countries with much higher IPv6 penetration – currently Belgium, USA, Switzerland or Greece and Portugal as shown in Geoff Huston's [11] and Cisco's statistics[2].

[1] `https://google.com/ipv6`
[2] http://6lab.cisco.com/stats/

Another method is based on measuring the number of autonomous systems announcing an IPv6 prefix. The statistic informs how ISPs and transition networks are prepared to provide IPv6 connectivity for customers. One analysis was presented by Karpilovsky et al. [13]. Their study has shown, that almost half of assigned IPv6 prefixes is not used at all, and the rest of them is announced long after the allocation. Thorough analysis of global routing table, the number of IPv4/IPv6 ASNs, prefixes, etc., is done by Geoff Huston [14].

One way of measuring the quality of IPv6 service is to measure the one-way delay. Zhou et al. [15, 16] published a study comparing IPv4 and IPv6 one-way delay between several measurement points. Their conclusion was that native IPv6 paths had small 2.5 percentile and median end-to-end delay, and comparable delay to their IPv4 counterparts. The study [17] found that the latency is less over IPv4 than IPv6. The mean latency is 55 ms over IPv4 for destinations in the North America but substantially higher, 101 ms, for the same destinations over IPv6. The difference between the IPv4 – IPv6 performance is more likely correlated with a different forward AS-level path as was reported in [18]. The measurement [19] compares the performance of IPv6 and IPv4 protocol by measuring the web page download time. They found that control plane (routing) was responsible for differences between IPv4 and IPv6, because the data plane (implementations of IPv4 and IPv6 stacks) performs comparably.

Jakub Czyz et al. [20] analyzed the IPv6 adoption from several perspectives – allocation of IPv6 prefixes, clients readiness, etc. These different metrics give entirely different insight into the adoption of IPv6, and show orders of different magnitude progress. For instance, 12% of cumulative allocated prefixes are IPv6, but just 0.63% of average traffic is carried over IPv6 – a two-order-of-magnitude difference. This difference follows the prerequisites for IPv6 deployment (e.g., allocation precedes routing, which precedes clients, which precedes actual traffic).

All above-described techniques present IPv6 deployment from the Internet infrastructure point of view. However, the ISPs are also interested in the number of content providers that enabled IPv6. Several measurements have been published to describe this information – [21, 22, 23, 24, 25, 26]. These papers and methodologies will be analyzed in more detail in section 2.2.2.

The following section describes the IPv6 transition progress from several points of view, e.g., global routing or IPv6 content penetration. The measuring platform for gathering long-term statistics about IPv6 penetration will be described in the next section as well.

2.2 Measuring the IPv6 transition progress

The previous section gave an overview of approaches used for measuring IPv6 transition progress. This section describes two of these approaches in detail. Firstly, BGP and IPv6 prefix analysis are presented. We examine global BGP table to find out current trends in IPv6 adoption. We also correlate BGP analysis with NetFlow data from Brno University of Technology (BUT) and CESNET networks. It is a novel approach, as BGP analysis is usually presented without any correlation with real traffic flows.

Secondly, IPv6 penetration among content providers will be evaluated in detail.

2.2.1 BGP and IPv6 prefix analysis

The IPv6 allocation process maintains the same allocation hierarchy as with IPv4 addresses. IANA allocates IPv6 address blocks to the five regional Internet registries (RIRs). The

Table 2.1: Flow data statistics obtained on 13th January 2015 in Brno University of Technology and CESNET NREN network

	BUT	CESNET
Average Bit Rate	2.127 Gb/s	18.4 Gb/s
Maximum Bit Rate (peak)	3.604 Gb/s	32.3 Gb/s
Total Number of Flows	608 000 000	8 062 000 000
Average Flow Rate	7 000 flow/s	93 000 flow/s
Maximum Flow Rate (peak)	10 000 flow/s	152 000 flow/s
Consumed Disk Space (compressed)	25 GB	240.3 GB
Consumed Disk Space (uncompressed)	48 GB	482 GB

RIRs allocate the IPv6 addresses to various local registries (LIRs) which allocates the addresses to their customers and end users. RIRs publish a daily snapshot of allocated blocks of addresses which can be downloaded for further analysis. The first assessment of IPv6 deployment could be an analysis of these allocations. However, there could be a bias because each RIR has a different allocation policy. For example, RIPE NCC address policy states that IPv4 allocations will only be made to LIRs if they have already received an IPv6 allocation from an upstream LIR or the RIPE NCC. The consequence is, that LIRs allocate both IPv4 and IPv6, but use only IPv4 address space. This behavior of LIRs is also confirmed by research community [13].

Therefore, it is better to analyze the number of IPv6 prefixes found in the Internet's global routing table rather than just allocations. Unfortunately, even this approach can have a bias. The presence of an IPv6 prefix in BGP table does not mean that the prefix is used in a production network. ISPs often announce a prefix before the prefix is put in production. Twitter, Inc. can be used as an example of this behavior. Twitter, Inc. is using ASN 13414, which originates three IPv6 prefixes since the beginning of 2015[3] without IPv6 enabled on their websites or in mobile applications.

Dataset

As a dataset, we use BGP data collected by Route Views project [27] since January 2004. NetFlow data from Brno University of Technology (BUT) and CESNET networks are used to find out, how many IPv6 prefixes are used, how many users use IPv6 and which content is accessed over IPv6, etc. The BUT network has approximately 25 000 users (students and staff). CESNET is National research and education network in the Czech Republic. An example of data stored during a day is shown in Table 2.1.

Autonomous system analysis

The ASN allocation process is similar to allocation of IP prefix. IANA allocates blocks of AS numbers to RIRs. Each RIR then assigns AS numbers to entities within its region. Autonomous System number space use 16-bit field (possible 65,536 unique values). The size, however, became a problem, because RIRs were running out of the 16-bit ASN numbers. RFC 4893 [29] introduced support for four-octet (32-bit) AS number to overcome the problem. The compatibility for the 32-bit extension is negotiated during BGP session establishment. Both implementations (32-bit and 16-bit) can operate together thus allowing

[3] https://stat.ripe.net/widget/announced-prefixes#w.resource=13414

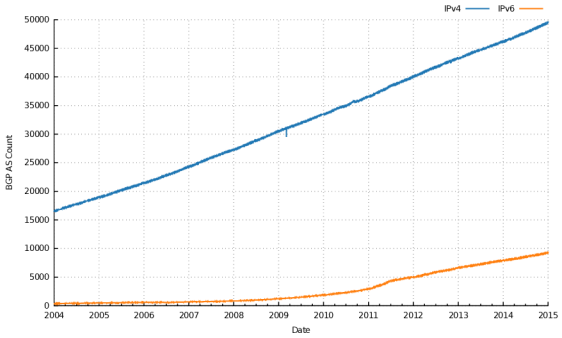


Figure 2.1: Number of unique ASes supporting IPv4 or IPv6

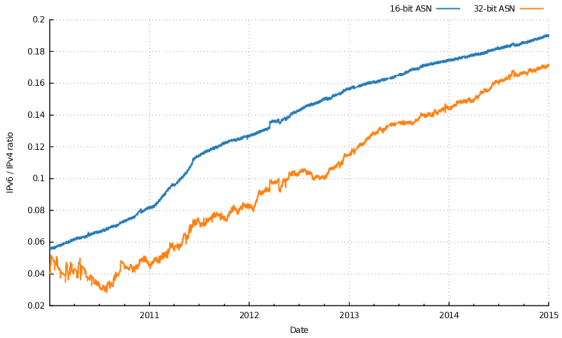


Figure 2.2: Ratio of IPv6 support among 16-bit and 32-bit ASes

gentle transition and incremental deployment.

It is interesting to see that the transition from ASN-16bit to ASN-32bit was driven by the same reason as the transition from IPv4 to IPv6; to have a larger number space. Contrary to transition from IPv4 to IPv6, the enlarged ASN space was designed to be backward compatible. The consequence is that ISPs can freely move to the new, larger AS Number space without any serious issues.

The global BGP table contained around 53 000 unique ASNs at the end of 2015. Figure 2.1 shows a number of unique ASes supporting IPv4 or IPv6 protocol. Although a few IPv6-only[4] ASes exists in the global BGP table, these ASes are tied with entities that have IPv4 ASN as well. The consequence is that there are not any entities (ISPs or large organizations) supporting IPv6 protocol only. The IPv6 line in Figure 2.1 thus should be seen as ASes that support both protocols (IPv4 together with IPv6).

The overall number of unique ASes depicted in Figure 2.1 can be split to 16-bit and 32-bit numbers. These numbers can indicate the support for IPv6 protocol among new companies because the default ASN allocation is currently a 32-bit ASN.

The ratio of IPv6 support among 16-bit and 32-bit ASes is shown in Figure 2.2. The ratio is computed according to Equation 2.1. The same formula is used to compute ratio among 32-bit ASes.

$$ratio16bit = \frac{number\ of\ 16\text{-}bit\ ASNs\ supporting\ IPv6}{total\ number\ of\ 16\text{-}bit\ ASNs} \tag{2.1}$$

Figure 2.2 shows that approximately 21% of 16-bit ASes and 19% of 32-bit ASes support IPv6. The conclusion can be that support for IPv6 is slightly lower among new companies. These results support the conclusion of Livedariu et al. work [5] that some organizations (especially newly joining edge ASes) using IPv4 transfer market as a mechanism to avoid deploying IPv6 immediately.

[4] AS that originate IPv6 prefix only

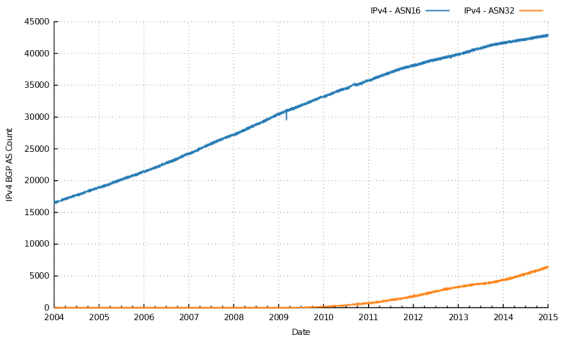


Figure 2.3: 16-bit ASN and 32-bit ASN supporting IPv4

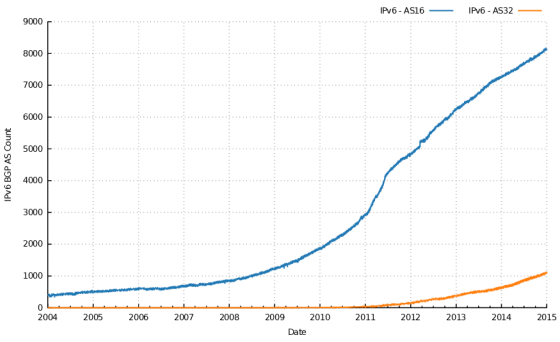


Figure 2.4: 16-bit ASN and 32-bit ASN supporting IPv6

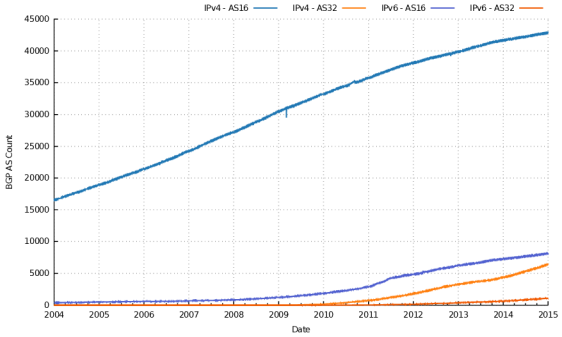


Figure 2.5: 16-bit and 32-bit IPv4 and IPv6 autonomous systems in BGP

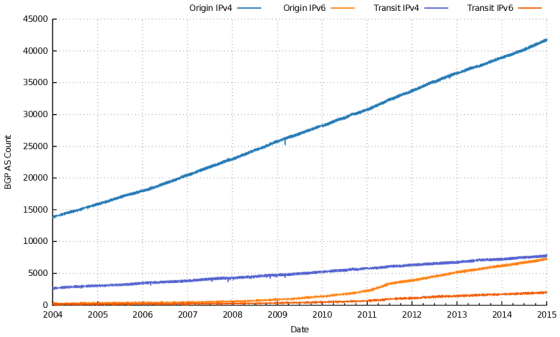


Figure 2.6: Origin and mixed autonomous systems supporting IPv4 or IPv6

Figures 2.3 and 2.4 show number of 16-bit and 32-bit IPv4 and IPv6 ASes. To put the growth of IPv6 in the context, Figure 2.5 compares these numbers with each other. These figures indicate that support for IPv6 is growing, but very steady.

The overall number of autonomous systems presented in previous figures can be further divided between origin only, mixed or transit ASes. By dividing the overall number of autonomous systems between origin, mixed and transit only, we can observe IPv6 adoption between edge networks (origin only ASes) and the core of the Internet (mixed and transit only ASes). Figure 2.6 shows IPv6 adoption between origin only and mixed ASes. Figure 2.7 depicts IPv6 adoption between transit only ASes and finally, Figure 2.8 shows the IPv6/IPv4 ratio among these autonomous systems. These Figures show that at the end of 2015 IPv6 is supported in 19% origin ASes, 26.1% mixed ASes and 59% transit only ASes. The IPv6 penetration is thus higher in the core of the network and lower at edges.

Table 2.2 analyzes IPv6 adoption in more detail by comparing the increment in IPv4 and IPv6 ASes over the 2009-2015 period. What this table indicates is that the growth of IPv4 ASes (new companies) is stable in last six years. IPv6 ASes, on the other hand, raise faster in percentage, but the actual growth in number is still low.

Figures 2.6, 2.7 and 2.8 show a rising trend in IPv6 adoption. However, we find the trend still quite slow. Despite the fact that IPv6 has been deployed among several biggest companies - Facebook, Google, Comcast, Verizon just to name a few, it appears, that other providers are still not convinced, that deploying IPv6 is a viable option for them.

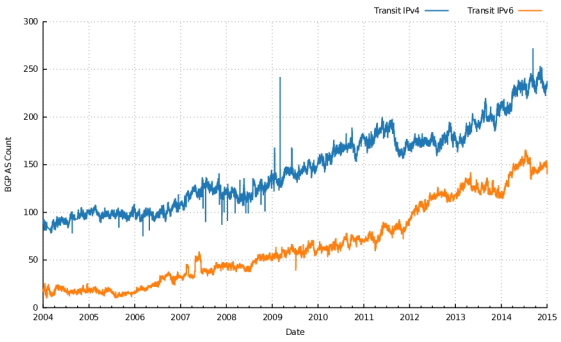


Figure 2.7: Transit only autonomous systems supporting IPv4 or IPv6

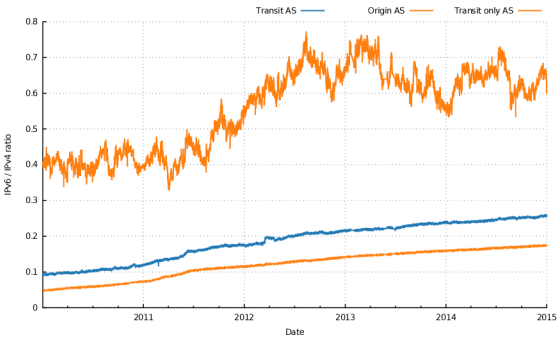


Figure 2.8: Ratio of IPv6 support among transit only, mixed and origin ASes

Table 2.2: Increment in IPv4 and IPv6 ASes over the 2010 - 2015 period

	Origin AS		**Transit AS**	
Date	IPv4	IPv6	IPv4	IPv6
2009	2500 (9.6%)	500 (55.3%)	500 (10.7%)	100 (42.5%)
2010	2500 (9.1%)	900 (62.9%)	600 (10.7%)	200 (45.1%)
2011	3000 (9.7%)	1600 (71.8%)	500 (8.6%)	400 (58.9%)
2012	2800 (8.2%)	1300 (33.3%)	400 (6.4%)	300 (30.3%)
2013	2400 (6.6%)	1000 (19.6%)	500 (7.3%)	300 (19.5%)
2014	2750 (7.1%)	1100 (17.5%)	550 (7.6%)	200 (6.9%)
2015	2900 (6.9%)	1300 (18.3%)	650 (8.3%)	430 (23.5%)

IP prefix analysis

The allocation process of IP prefixes is similar to allocation process of autonomous system numbers described above. Central coordination is required to ensure that different networks use unique non-overlapping IP prefixes. IP addresses are allocated by the IANA from pools of unused address space and delegated to the appropriate RIRs.

The status of IPv4 address pools at the beginning of 2016 is following. IANA address pool was exhausted in February 2011 followed by APNIC in April 2011, RIPE NCC in September 2012, LACNIC in June 2014 and ARIN in September 2015. Projected exhaustion for AFRINIC is at the beginning of 2019.

In this case, the definition of exhaustion is following: IPv4 address pool of available addresses reaches the threshold of no more general use allocations of IPv4 addresses. Each RIR defines the threshold differently. For example, APNIC and RIPE NCC set the threshold to the last `/8` of available IPv4 addresses in their address pool. ARIN, LACNIC and AFRINIC have different policies - ARIN sets the threshold to `/10`, LACNIC and AFRINIC to `/11`. The consequence is that almost all IPv4 addresses are allocated to end entities. Figure 2.9 confirms this situation by showing the number of IPv4 addresses assigned, allocated and advertised for each RIR. We can see that RIRs depleted their IPv4 address pools because a majority of IPv4 addresses are in the assigned state. One exception is AFRINIC with approximately 2.71 `/8` available IPv4 addresses in its pool. Despite the fact that RIRs pools were depleted, there is still a small room for growth – especially in ARIN region, because a lot of IPv4 addresses are not even advertised in the routing system. More precisely, around 25% of overall IPv4 addresses (about 1 billion of IPv4 addresses)

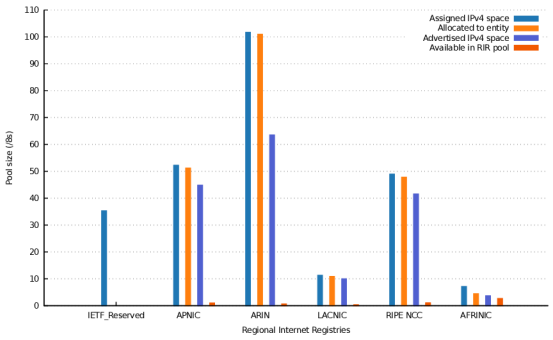


Figure 2.9: Regional Internet Registries IPv4 pool size on 17.2.2015

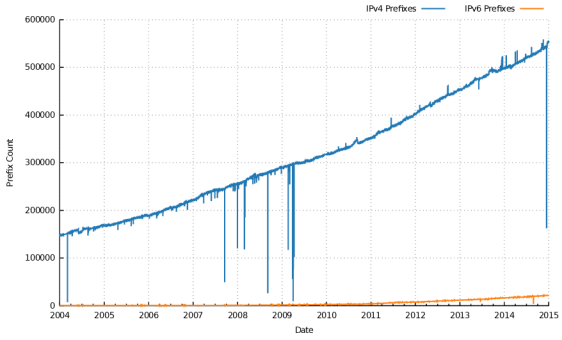


Figure 2.10: IPv4 and IPv6 prefixes in BGP table over 2004 - 2014 period

has not been advertised in BGP yet. Thus, these addresses are not used for end-to-end connectivity.

Figure 2.10 shows the overall number of IPv4 and IPv6 prefixes in BGP table over the 2004 – 2014 period. The figure shows similar trend line as we could see in the autonomous system number analysis – the number of IPv6 prefixes is growing. The growth is, however, steady. According to Geoff Huston [30], the rate of IPv6 growth has increased to the current level of some 15 to 20 new entries per day. It is still, however, lower than the IPv4 growth, which is growing by some 100 - 150 prefixes per day even today, where all main pools are depleted.

Based on routing table snapshots, we obtained the number of IPv6 prefixes announced in the global routing table as shown in Figure 2.10. Unfortunately, presence of IPv6 prefix in the global routing table does not say anything about the actual usage of the prefix in a production network. Can we obtain the information? Is it possible to find if an IPv6 prefix is used or if it is just advertised in the BGP table? We used the following approach to try to find the answer.

An organization usually obtains IPv6 prefix of `/32` or `/48` length from a RIR. If IPv6 is deployed in the organization, the actual IPv6 prefix size used for addressing end devices is, however, larger. The IPv6 Addressing Architecture described in RFC 4291 [31] requires a unique `/64` prefix for every individual network segment. To illustrate this approach, let us use the BUT network as an example.

BUT obtained IPv6 PI prefix `2001:67C:1220::/46` from RIPE NCC. This prefix is used for addressing end users, servers, VoIP phones, etc. Currently, there are approximately 150 `/64` networks in BUT internal OSPFv3 network as every server farm or campus VLAN use their own prefix. The conclusion is that a `/48` prefix should contain several `/64` prefixes if IPv6 is deployed in production. Of course, the global BGP table does not contain prefixes of `/64` size, because these prefixes are aggregated on ASNs boundaries. However, we can distinguish unique `/64` prefixes from the network traffic. NetFlow data collected on BUT and CESNET networks described in dataset section 2.2.1 can be used for this purpose. We can aggregate IPv6 addresses on unique `/64` and `/48` boundaries. If there are more `/64` prefixes per a `/48` prefix, we can conclude, that the network deployed IPv6 in production. If there are only 1 or 2 `/64` prefixes per a `/48` prefix, we can conclude, that the `/48` network is probably still in a testing phase.

Is this hypothesis right? Table 2.3 shows statistics obtained from NetFlow data on BUT network. We probably do not have to introduce Facebook or Google - big players in IPv6

Table 2.3: Unique /48 and /64 IPv6 prefixes seen by BUT network on 9.1.2015

	ASN	Prefix	/48	/64
Brno University of Technology	197451	2001:67c:1220::/46	3	99
Facebook, Inc.	32934	2a03:2880::/32	39	580
O2 Czech Republic, a.s.	5610	2a00:1028::/32	116	1545
Google, Inc.	15169	2a00:1450::/32	20	266
CESNET z.s.p.o.	2852	2001:718::/32	41	199

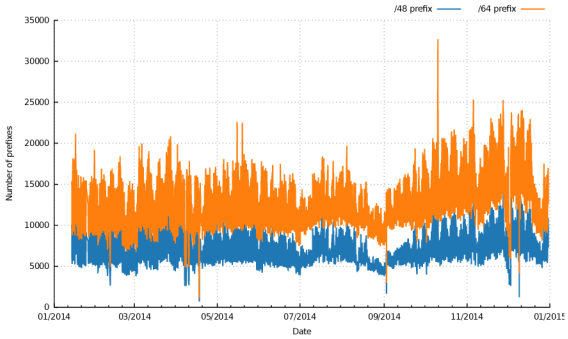


Figure 2.11: Number of unique /64 and /48 prefixes seen in CESNET NREN network

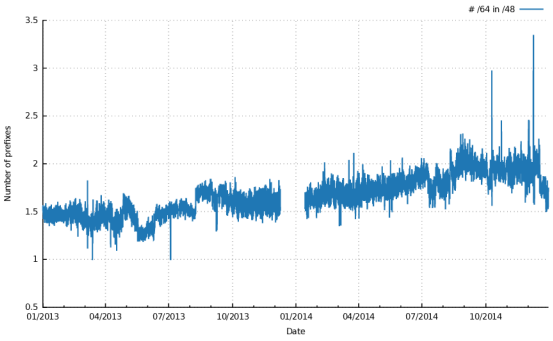


Figure 2.12: Number of unique /64 prefixes in one /48 prefix

deployment. The rest are CESNET – the biggest NREN network in the Czech Republic and O2, one of the biggest ISP in the Czech Republic. Both companies are known for promoting and deploying IPv6 in production networks in the Czech Republic. The table supports the hypothesis that a production network has several /64 per /48 – there are approximately ten times more /64 in a /48 in these networks.

We can also analyze the long-term data from CESNET and BUT datasets. CESNET NetFlow dataset was analyzed, and the results are depicted in Figures 2.11 and 2.12. Figure 2.11 shows the number of unique /64 and /48 prefixes in CESNET NREN network in 2014. The figure shows, that the number of these prefixes steadily increases. The number of unique /64 prefixes in one /48 prefix is depicted in Figure 2.12. The figure shows that there are around 1.5 unique /64 prefixes in one /48 prefix in 2013 and around two unique /64 prefixes in 2014.

BUT dataset is analyzed in Figures 2.13 and 2.14. These figures show a similar trend as figures analyzing CESNET dataset. The number of /64 prefixes in one /48 prefix is lower, but this is obvious, because BUT network is not as big as CESNET network.

We should, however, highlight possible biases that these data could have.

1. If the end network belongs to a content provider, traffic from BUT network could be routed just to the nearest datacenter, thus it could be seen that we are accessing always the same IPv6 network.

2. Network administrator can use only one /64 prefix for the whole network.

3. The traffic profile is different in every country as each country has different popular services. Different traffic profiles can be responsible for a different amount of IPv6

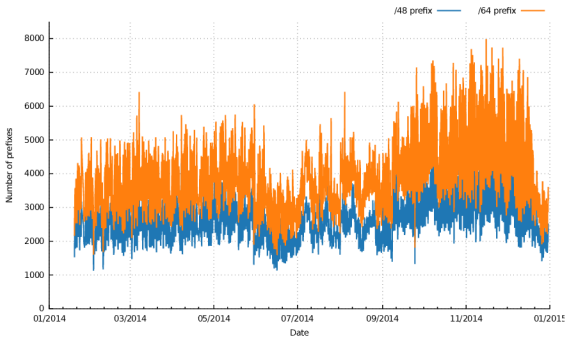


Figure 2.13: Number of unique `/64` and `/48` prefixes seen in BUT network

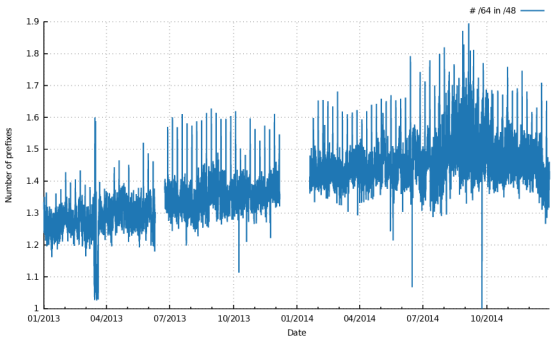


Figure 2.14: Number of unique `/64` prefixes in one `/48` prefix

traffic, but it should not cause differences in the number of `/64` prefixes in one `/48` prefix as network administrators tend to use similar network deployment practices.

Bias introduced in the first item is eliminated by using significantly large network dataset. The second item should not happen in the real world as it denies best practices in network design. The third item can have an impact on the volume of IPv6 traffic as there could be more services available over IPv6 in different countries, but different traffic profile has not the impact on the number of IPv6 prefixes.

Even though that analysis of BGP DFZ shows the increasing trend, correlation with the real world IPv6 traffic suggest, that these prefixes are probably still used mainly for testing and not for the real traffic. We discussed this hypothesis with David Plonka, the researcher from Akamai Technologies, and he suggested that there could be another bias we have not mentioned. An ISP can use `/48` prefix per subscriber. In that case, there is only a few `/64` prefixes in the `/48` prefix. However, if a subscriber receives `/48` prefix, there should be several `/48` prefixes in one `/32` prefix. We tried to run an analysis to find out if the bias has an impact on the presented hypothesis, but the analysis does not reveal any significant impact. However, we are going to study this bias further in the future work as the analysis was run only on a subset of our dataset.

2.2.2 IPv6 content analysis

To promote the deployment of IPv6, a significant effort was made by several operators and content providers. The IPv6 Day on June 8th, 2011 and IPv6 Launch on June 6th, 2012 were events that should motivate the activity of other service and content providers to enable IPv6. Observation confirmed an increase of IPv6 traffic and the number of users. Google kept IPv6 enabled for several YouTube servers, thus the main contributor for IPv6 traffic since then is YouTube. Thanks to the results from IPv6 Day, big content providers decided to turn on IPv6 permanently one year later on "IPv6 Launch day".

Nevertheless, big content providers such as Google, Akamai, Facebook or Netflix do not represent the whole Internet. Millions of other websites remain without IPv6 addresses. This situation raises several interesting questions.

i What is the ratio of enabled IPv6 websites and services (e.g., mail servers, name servers)?

ii Is the ratio increasing or stagnant?

Table 2.4: Resource records checked

Service	Record type	Test
Web	A	`www.<domain>`
	AAAA	`www.<domain>`
	AAAA	`www6|ipv6|www.v6.<domain>`
Mail	A for MX	`<domain>`
	AAAA for MX	`<domain>`
DNS	A for NS	`<domain>`
	AAAA for NS	`<domain>`

iii If IPv6 is enabled, is the quality of service (e.g., response time) better, worse or the same as in IPv4?

iv If we deploy a new architecture, how many users will use it? How much traffic will flow over it?

These questions are important to ask, since answers to these questions can help us to estimate, how long does it take to move from one incompatible network architecture to another.

Methodology and dataset for measuring IPv6 content

IPv6 penetration among content providers can be measured by checking the appropriate resource records (RR) in DNS. Table 2.4 briefly describes the RRs that can be tested. The web services should also be checked if an alternative record for IPv6 (e.g. `www6, ipv6, www.v6`) is available. The alternative records are sometimes used by network administrators for testing purposes.

The total number of tested domains will determine the precision of the measurement. The results will be more accurate with the larger number of tested domains. A favorite source of domains is a list of the most visited domains provided by Alexa The Web Information Company. However, we believe that only the top list is not enough. For example, there are about 4 700 domains within Czech TLD in the Alexa's top list. The total number of domains in Czech TLD is approximately 1 200 000 thus the list contains less than `0.4 %` registered CZ TLD domains. Another drawback of using Alexa dataset only is that all sub domains are aggregated to appropriate TLD.

Are there other data sources available for creating a list of popular domains? One possible way is to use mail servers logs and do a reverse lookup for addresses sending the emails to our network. Domains can also be collected from DNS cache of a resolver.

We developed the following solution to extend the number of domains in database and overcome the drawbacks of using only Alexa top sites and DNS cache. Several probes were deployed in the BUT campus network. These probes are still running and the monitoring process is still in place. Probes listen to all HTTP requests performed by users. The output is sent to a collector where the requests are stored in a database. Once per day, the update process adds the new unique domains into the central database. In April 2016, the database contained approximately 12 million of working domains [40].

The collected list of domains is also used for the quality of service measurement. Web domains supporting both protocols are checked and the response times for both protocols are measured and stored in the database. Using IPv4 and IPv6, the system tries to connect

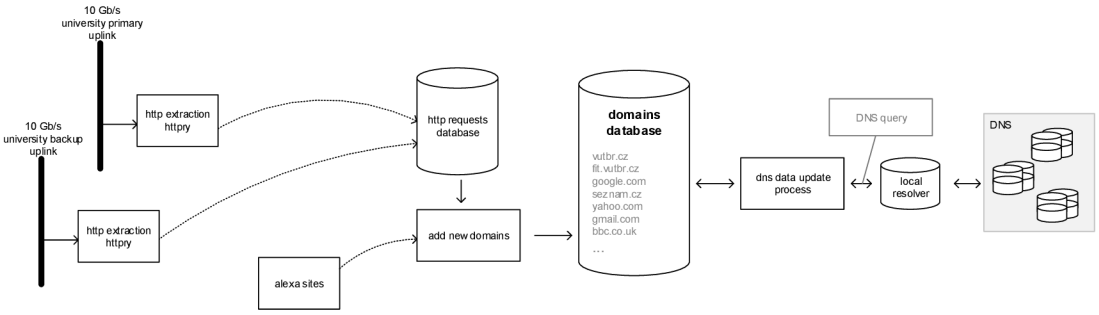


Figure 2.15: Architecture of the system

to the remote web server. The time is measured between the first packet initiating relation (SYN) and the answer from the server (SYN, ACK). Comparing the results obtained via IPv4 and IPv6, it can be observed, which protocol has a better response time.

To summarize, we tried to avoid to use a limited dataset as used by other projects measuring IPv6 adoption among content provides. The benefits of our approach are following:

- Independence on third party datasets (Alexa list).
- Visibility of IPv6 enabled sites, which are interesting for our users.
- Visibility of subdomains in a TLD domain. The visibility is useful because Alexa list does not provide information, about visited subdomains, only aggregated data.
- Ability to check the quality of connection and compare IPv4 and IPv6 connection speeds.
- Long term solution with minimum maintenance.

Architecture and Implementation

The architecture of the monitoring system presented in the previous section is divided into several blocks and it is depicted in Figure 2.15. The core of the system is a database containing a list of domains and statistical data. There are two subsystems connected to the database. The first one is responsible for querying DNS system. It takes the list of domains from the database and periodically updates data with the information gathered from DNS. The history of changes for each record is stored as well, allowing us to provide current and historical data for each domain in the database. The next subsystem performs the check of IPv6 quality by measuring the one path delay as was described in the previous section. It also updates information into domains database and stores historical information about each measurement.

All dual stacked domains in the database are periodically checked once a week for IPv4 – IPv6 speed comparison. The whole update process takes close to ten hours for the database containing approximately 800 000 of dual stacked domains.

Data Analysis

Data collection is still ongoing. This thesis presents data collected up to April 2015. The total number of web domains is defined as `NWT`, the number of web domains supporting web

Table 2.5: IPv6 penetration among web, mail and DNS services - ratio on 8th of July 2016

	IPv4 only	**IPv6 only**	**Dual stack**	**IPv6 alt. name**
Web service	91.77 %	0.003 %	7.77 %	0.45 %
Mail service	84.19 %	0.011 %	15.8 %	
DNS service	70.13 %	0.008 %	29.86 %	

services over IPv6 as NWV6, the number of web domains supporting dual stack as NWDS and the number of web domains supporting IPv6 web through alternative name as NWA6.

- NWT - domains having at least one IPv4 or IPv6 record announced for www.<domain>.
- NWV6 - domains having at least one IPv6 (AAAA) record and not having IPv4 record (A) announced for www.<domain>.
- NWV4 - domains having at least one IPv4 (A) record and not having IPv6 record (AAAA) announced for www.<domain> and not having alternative IPv6 record - e.g. www6|ipv6|www.v6.<domain>
- NWDS - domains having both IPv6 and IPv4 records announced for www.<domain>.
- NWA6 - domains announcing any of www6|ipv6|www.v6.<domain> via IPv6 (AAAA) and not announcing IPv6 for a record www.<domain>.

The penetration ratio of IPv4 only sites is computed using Equation 2.2. Other ratios are computed using the similar formula, but the numerator is changed accordingly to NWV6 for IPv6 only sites ratio, NWDS for dual stack ratio, etc.

$$NWV4ratio = \frac{NWV4}{NWT}100 \tag{2.2}$$

Based on these rules, we can analyze the data in our database to obtain the IPv6 penetration for the web, mail and DNS services. Table 2.5 shows that all these services are accessible using IPv4 protocol. More precisely, 99.997 % of web domains, 99.992 % of NS domains and 99.989 % of mail domains are accessible over IPv4. There is a slight number of IPv6 only domains (both web, mail or DNS), but these domains are without any meaningful content for end users. IPv6 only domains are mainly test websites and servers used for testing user's IPv4/IPv6 connectivity [11], sites where a www.<domain> has only AAAA record, but there is also a record for <domain> that is accessible via IPv4 or sites that are used for testing mail or DNS services over IPv6 only connection. The availability of DNS and mail services over IPv6 protocol is higher in comparison to websites. The higher penetration of these services corresponds to deployment strategy for a new service, where an administrator goes from the vital services to the less important ones, or he updates services where minimum changes are required.

Figures 2.16 and 2.17 show the progress of IPv6 penetration for mail, web and DNS services since 2012. These numbers confirm the observations described above. As we measure the content for a long period, all the main IPv6 events are visible. For example IPv6 Launch day in June 2012 and Cloudflare „IPv6 on by default“ in March 2014 are clearly visible in Figure 2.16. There was a high jump in IPv6 penetration for MX and NS domains as well as shown in Figure 2.17. According to our statistics, the main reason for

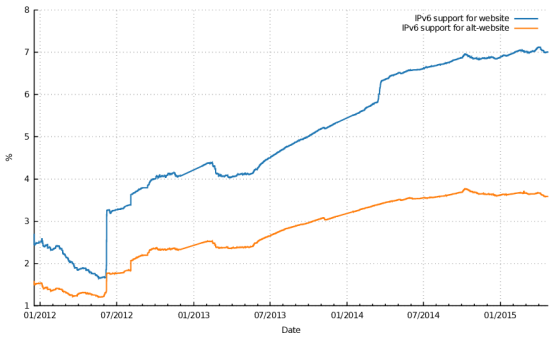


Figure 2.16: IPv6 penetration for web services, January 2012 - August 2015

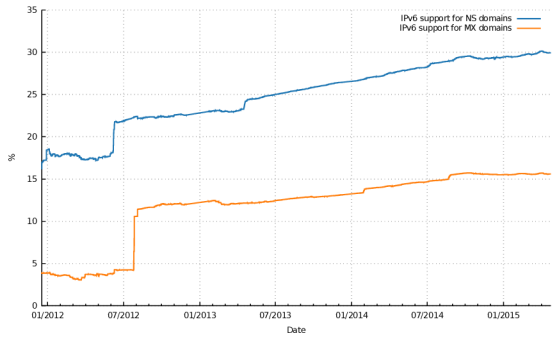


Figure 2.17: IPv6 penetration for NS and MX services, January 2012 - August 2015

the bounce in August 2012 for MX and July 2012 for NS records is that Google enabled IPv6 support for their Apps. Every website which uses Google Apps, e.g., for their work mails, started to be accessible over IPv6 since that time.

Validity of the Results

Every measuring system must prove that data provided by the system are trustworthy. If we would like to know the most accurate penetration of IPv6 services among content providers, we would need to collect all DNS data available in DNS system. This approach is, however, impossible due to the decentralized way how DNS system works. Fortunately, there are several projects trying to measure IPv6 penetration among content providers. Thus, we can compare our data with these projects. All similar projects use a list of domains provided by Alexa as a dataset, and usually only a subset of the top 1 million websites is used. It has all benefits and drawbacks described in the previous section. Hurrican Electric is an exception as they use the whole TLD zones for analysis. However, using only TLD zones, they cannot access information about subdomains.

The projects datasets and methodologies are compared in Table 2.6. The **Tests** column describes which of the following tests the project run.

- `web` test obtains the evidence of an `AAAA` record for selected domain.
- `alt` test checks an existence of alternative names for the domain (e.g., `v6.<domain>`).
- `MX` and `NS` tests are testing presence of `AAAA` record for mail and DNS services
- `DNSSEC` and `SPF` tests verify the support for these services.
- `avail` test measures the quality of connection using both IPv4 and IPv6.

There are other projects measuring IPv6 penetration that are not part of the comparison, e.g., RFC 6948 [25] or Dan Wing's statistics [42]. The reason for this is that these projects present the same results as others; they are using Alexa dataset, test the same metrics, etc. Thus, it would be just a duplication of information.

Results if IPv6 penetration for web services of selected TLDs that have the largest IPv6 penetration are compared in Table 2.7. The comparison is based on the data from the 1st of July 2015 or latest available. As we can see in Table 2.7, the obtained results are very different. Figure 2.18 shows one of the reasons for such distinction. The chart can be

Table 2.6: Comparison of projects measuring web content available over IPv6

Project	Data	Records	Tests	Freq.
[21] IPv6matrix	Alexa	top 1 million	web, alt, MX, NS	month
[22] IPv6observatory	Alexa	top 500/TLD	web, alt, MX, NS	defunct
[23] Eric Vyncke	Alexa	top 50/TLD	web, alt, MX, NS	daily
[24] Hurrican Electric	Alexa, zones	171 million	web, MX, NS	daily
[26] 6lab.cisco.com	Alexa	top 500/TLD	web, alt	daily
[41] cz.nic	.cz TLD	1.2 million	web, MX, NS, DNSSEC	month
[40] 6lab.cz	Alexa, Users	12 million	web, alt, MX, NS, avail, DNSSEC, SPF	daily

Table 2.7: Results of IPv6 penetration provided by different projects, July 2015

Project	.com	.cz	.de	.fr	.be	.ch
IPv6matrix	-	11.7 %	12.98 %	3.23 %	3.59 %	0.85 %
IPv6observatory	-	15.9 %	9.3 %	9.6 %	9.6 %	-
Eric Vyncke	-	65.46 %	30.76 %	33.8 %	30.6 %	32.68 %
Hurican Eletric	2.6 %	-	18.08 %	-	-	-
6lab.cisco.com	-	62.61 %	45.72 %	50.5 %	50.43 %	52.2 %
cz.nic	-	24.6 %	-	-	-	-
6lab.cz	5.89 %	20.01 %	17.4 %	8.21 %	4.74 %	3.94 %

interpreted as follows: IPv6 penetration (y-axis) is calculated for every number of domains (x axis). For example, IPv6 penetration for first three domains in Alexa list is 100 % for .com TLD. If the ratio is evaluated for top 5 domains, it drops to 80 %. If we use the same data source and top 500 domains, the penetration decreases to 8.4 %. If we want to receive meaningful numbers of IPv6 penetration, the number of analyzed domains must be increased at least to 10 000 records per TLD or more in the case of global IPv6 penetration. Using a smaller number of domains, we do not receive any precise number, and the IPv6 penetration is over-estimated.

Another reason for such a big difference between the projects' IPv6 penetration is that projects use different methodologies to compute IPv6 penetration. There are two main distinctions.

- **Geolocation:** 6lab.cisco.com and Eric Vyncke measurements use geolocation to identify a country for a particular domain [43]. This approach is useful for generic domains such as .com, where it is not clear to which country the domain belongs. This approach is, however, problematic as there are a lot of domains with the local content served outside of a country – a domain served by CDN (Content Delivery Network), a domain of a branch of a foreign company, etc. The problem is also a quality of geolocation databases for IPv6 addresses. We believe that the geolocation of domains is unnecessary and creates more problems than it solves. Furthermore, domains are usually also registered in the country's TLD thus they will be counted in the country's IPv6 penetration anyway.

- **Weight:** Some projects (e.g. 6lab.cisco.com or Eric Vyncke stats) use a weight of a domain to compute the IPv6 penetration in a country. The weight of a domain is an approximation based on position of a domain in Alexa list. This is based on an

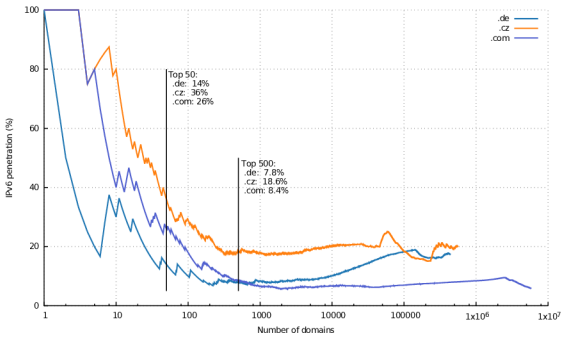


Figure 2.18: Dependence of the IPv6 ratio on the number of domains

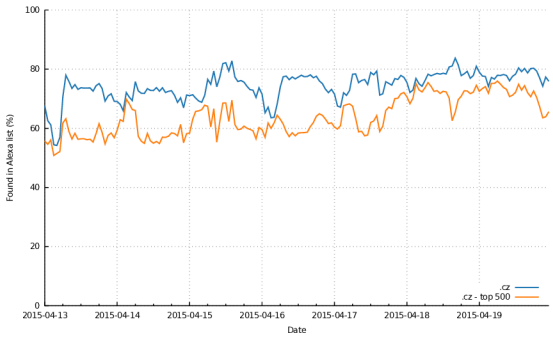


Figure 2.19: Percentage of .cz requested domains found in in Alexa list

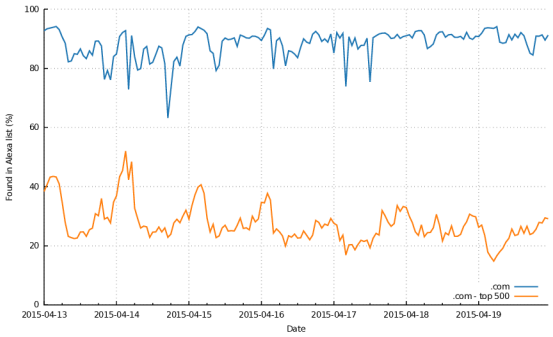


Figure 2.20: Percentage of .com requested domains found in in Alexa list

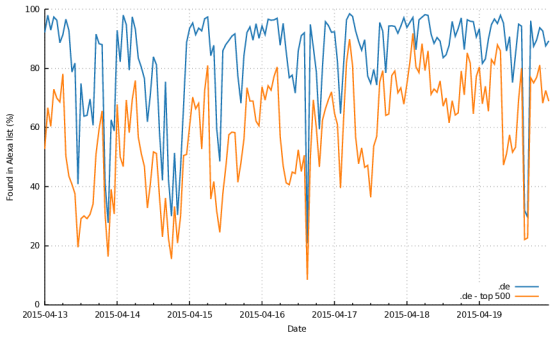


Figure 2.21: Percentage of .de requested domains found in in Alexa list

idea that users are more likely to connect, spend time and access content on a slight number of sites.

These ideas, however, introduce several questions. Can we just use the number of top N domains from the Alexa list because users mainly visit these sites? What is the ratio of domains that are found and not found in the Alexa list? Furthermore, what is the difference of IPv6 penetration between popular domains and domains not included in the Alexa list?

To analyze the assumptions, we used all HTTP requests made by all users in the BUT network during `13.4.2015 - 19.4.2015` period. We analyzed all these requests and selected domains (`.cz`, `.com` and `.de`) were analyzed in more details. These domains were chosen as they were the most visited domains by our users, thus provided the largest dataset. Furthermore, `.cz` and `.de` domains were reported to have the highest IPv6 penetration. There were approximately 620 million unique HTTP requests during the period. We divided the HTTP analysis into two steps. Firstly, we tried to answer the question, if users visited mainly sites in Alexa list. Secondly, we measured the IPv6 penetration for domains found or not found in Alexa list to see if there was any difference.

The requested domains were aggregated to the first subdomain after TLD, e.g., domain `maps.google.com` was aggregated to `google.com`. It was due to the fact, that Alexa list did not contain subdomains. Figures 2.19, 2.20 and 2.21 show the ratio of requested domains found or not found in the Alexa list. Each figure displays the ratio of a domain found in top 500 per TLD and the whole Alexa list. We can see that there is a probability between 60 – 80 % that a user's request will be found in Alexa list for `.cz` and this probability

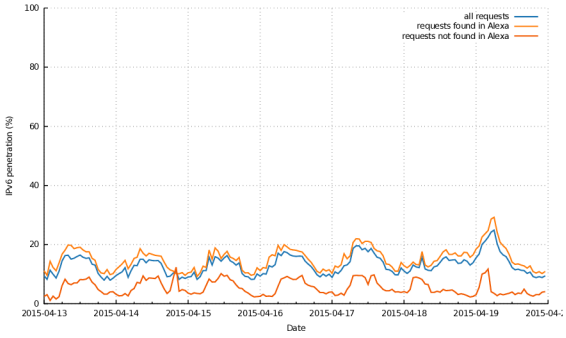


Figure 2.22: IPv6 penetration measured among all requests.

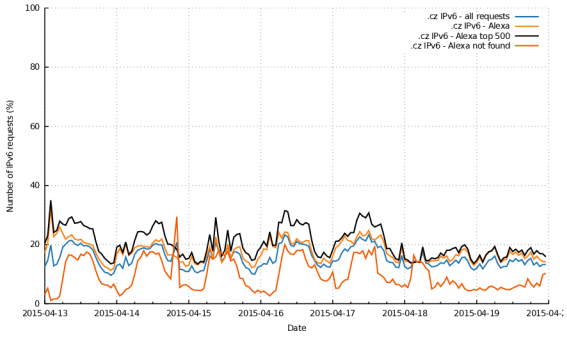


Figure 2.23: IPv6 penetration measured for .cz domain.

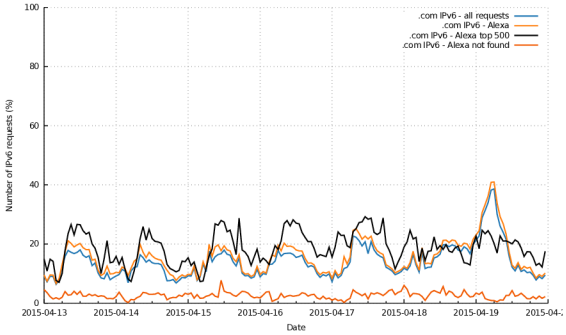


Figure 2.24: IPv6 penetration measured for .com domain.

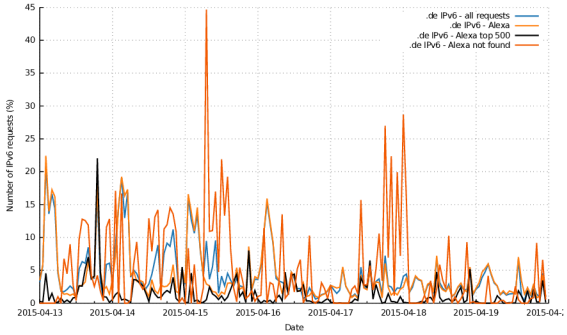


Figure 2.25: IPv6 penetration measured for .de domain.

is approximately 10 % smaller if we use only top 500 domains from Alexa dataset. The probability that a domain from `.de` TLD requested by BUT users will be found in Alexa dataset is between 20 – 90%. The reason for such a big difference lies in the fact, that `.de` TLD is the second largest TLD with approximately 16 million domains. Alexa list, however, contains only a small subset of `.de` domains (0.18 %). Even with our rather a small dataset, that contains 2.77 % of all `.de` domains, we can see, that there are a lot of domains regularly visited by BUT users that are not included in the Alexa list. The results confirm that a majority of requests is found in the Alexa top list, but there is still a significant number of requests that is not found. We can also see, that using only subset of domains, e.g., top 50 or top 500, provides very different results.

The HTTP requests were further analyzed to help us understand what is the IPv6 ratio of domains found and not found in the Alexa list. Figure 2.22 shows IPv6 penetration of all requests, requests found in Alexa list and requests not found in the list. It can be seen that IPv6 penetration of domains found in Alexa list is higher than for domains that are not included in the list. However, the difference between IPv6 penetration of all requests and requests found in Alexa list is small. Figures 2.23, 2.24 and 2.25 show detailed measurements for `.cz`, `.com` and `.de` TLDs. We can see, that `.cz` and `.com` domains have similar patterns, e.g., domains found in Alexa list have higher IPv6 penetration, domains in top 500 have even higher IPv6 penetration and domains not found in the list have much lower IPv6 penetration. The exception is `.de` TLD where we can see, that there are a lot of IPv6 enabled domains that are not found in Alexa list.

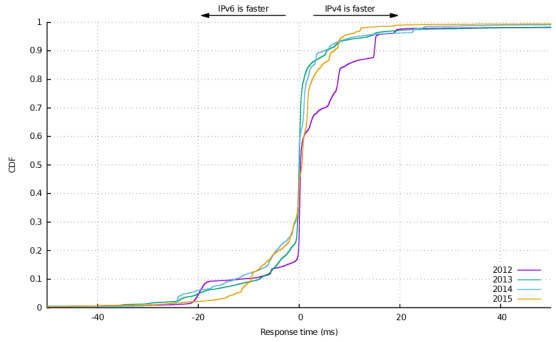


Figure 2.26: IPv4,IPv6 performance - CDF function.

Table 2.8: Number of measurements per year

Year	# measurements
2012	6,108,164
2013	19,198,299
2014	30,603,493
2015	26,660,007

Performance and availability of IPv6 domains

Implementing IPv6 support for a domain can introduce several new issues. Unfortunately, finding and debugging these problems is very hard thus it takes a long time before they are fixed[5]. Missing robust routing infrastructure for IPv6 is another problem. Currently, there are fewer peering contracts and redundant paths for IPv6 compare to IPv4. These issues can lead to non-optimal traffic routing, increase in latency or missing a redundant path if there is a route failure. An example of this behavior is missing IPv6 peering between Cogent and Hurricane Electric [45]. These two large companies are perceived as Tier 1 operators. Missing peering or transit contract between them create a splitted IPv6 Internet as BGP announcements from Cogent do not reach Hurricane Electric.

On contrary, there are several examples, where IPv6 outperforms IPv4 [46]. The study shows that Facebook had seen users' *News Feeds* loading 20 percent to 40 percent faster on mobile devices using IPv6. Unfortunately, they was no explanation why it happened as the data analysis was still ongoing.

To obtain more precise results, we regularly check availability and quality of RTT of domains with `A` and `AAAA` RRs. Using IPv4 and IPv6, we try to connect to a remote web server and measure the time between the first packet initiating relation (SYN) and the answer from the server (SYN, ACK).

Figure 2.26 depicts the difference between IPv4 and IPv6 response times using data from January 2012 – August 2015 period. The number of measurements in each year is shown in Table 2.8. The round trip time is measured as described in section 2.2.2. The RTT difference between IPv4 and IPv6 protocols is counted using Formula 2.3, thus negative values represents measurements where an IPv6 response is faster than a response over IPv4.

$$RTTdiff = RTT\ IPv6 - RTT\ IPv4 \tag{2.3}$$

The graph in Figure 2.26 plots cumulative distribution function (CDF) in the interval <`-50 ms, 50 ms`>. We use the following Formula 2.4 to compute the CDF function:

$$CDF(x) = P(X \leq x) = \frac{1}{n}\sum_{i=1}^{n} x_i \leq x \tag{2.4}$$

[5]For example, fixing non-working IPv6 connection for several government websites in the Czech Republic took more than two years![44]

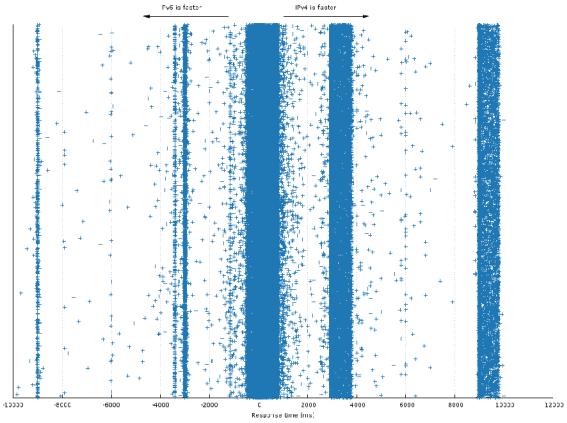


Figure 2.27: All measured RTT values in 2012.

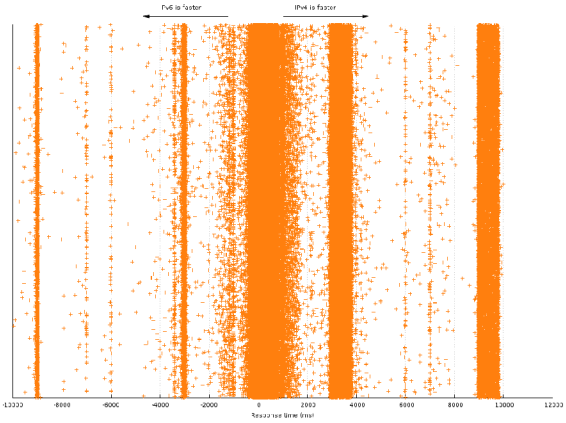


Figure 2.28: All measured RTT values in 2013.

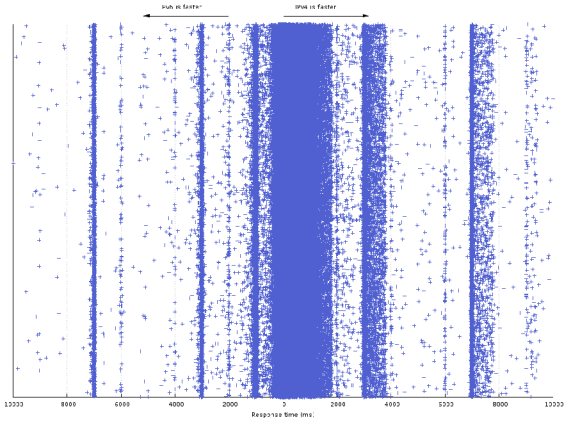


Figure 2.29: All measured RTT values in 2014

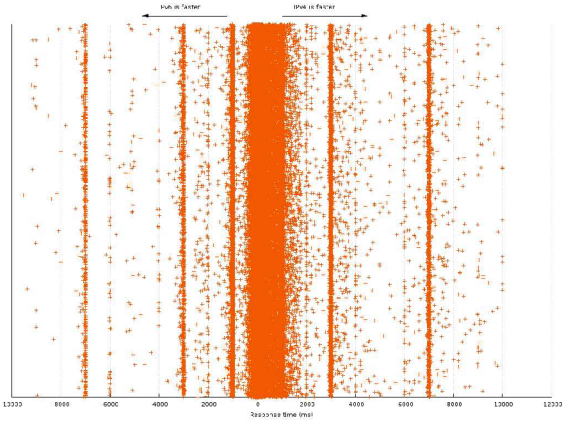


Figure 2.30: All measured RTT values in 2015

where n is number of measurements and x is response time of measured event. We can see, that round trip time values between IPv4 and IPv6 are very similar in confidence interval between 10 – 90 %. There were disproportions between response times in 2012, where IPv4 was faster more than 1 ms in 40 % of measurements but since then the performance of IPv6 improved and is very similar to IPv4.

Figure 2.26 clearly shows, that the performance of IPv4 and IPv6 are very similar. There are however cases, where IPv6 performs significantly better or worse than IPv4. As these cases are not visible in Figure 2.26, we depict all measurements in each year as a scatter plot. Figures 2.27, 2.28, 2.29 and 2.30 were created by shifting each data point in a year vertically by a random amount. Such jittering by a random number can be used to detect clusters. It can be seen, that there are patterns in each figure. For example, there were a lot of RTT values, where IPv4 response time was between 3000 – 4000 ms or 9000 – 10000 ms faster than IPv6, especially in 2012 and 2013 period. We can find similar patterns for IPv6 protocol as well. It is also visible that these patterns are converging to 0 during the years, which means that IPv6 performs similarly as IPv4. We analyzed differences between RTT values in detail and found, that they were caused by different paths in the routing system.

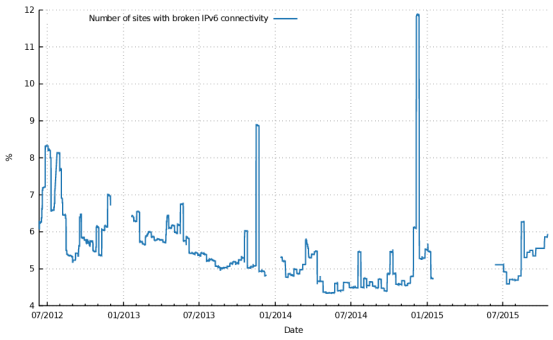


Figure 2.31: Number of domains we are not able to connect to, 2011 – 2015 period

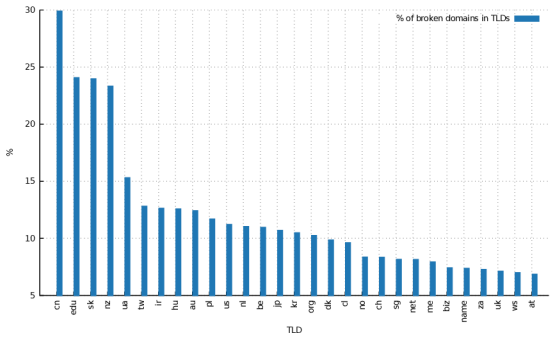


Figure 2.32: Percentage of broken domains in TLDs on 21.10.2015

Our dataset and analysis thus confirm Paul Saab's claim that IPv6 can perform significantly better than IPv4. The reason, however, is probably not a middleware (e.g. NAT) as it is believed, but differences between IPv4 and IPv6 paths. There is still a substantial number of sites we are not able to connect to. Figure 2.31 shows the percentage of all domains to which we were not able to establish a connection in 2012 – 2015 period. We can see, that there was around 5 % of all websites with broken IPv6 connectivity. The reasons can vary. According to our experiences, the main problems involve routing, security filtering, misconfiguration of DNS or application issues.

Figure 2.32 shows the number of broken domains per TLD. The figure is based on data from October 2015 and shows 30 TLDs with the highest number of broken domains. Only TLDs with more than 10 000 domains are considered. We can see a significant number of broken websites from `.cn`, `.edu` or `.sk` TLDs. We analyzed `.sk` TLD more deeply to find out, why there was almost 25 % of domains inaccessible. We discovered that a large content and hosting provider in the Slovak Republic published `AAAA` RR for many of their hosted web pages. Unfortunately, they probably misconfigured routing or security filters in their datacenter thus all our RTT probes failed. The issue was not resolved even after half a year. The users' browsers usually switch from non-working IPv6 to working IPv4 protocol without user intervention thus everything works fine from user's perspective. The underlying network infrastructure is, however, broken.

2.2.3 User penetration analysis

This section will be focused on users' support for IPv6 protocol, thus trying to answer the following questions.

- If we deploy a new architecture, how many of users will support it?
- How much traffic will flow over it?

The dataset used for generating following statistics is based on data collected in our campus network. We have been measuring the IPv6 support among clients since 2013. Figure 2.33 shows a percentage of IPv6 support among clients on our campus network. We can see, that there are approximately 80 – 85 % of users having IPv6 enabled and supported. There are drops in IPv6 support to 40 % during December and 40 – 50 % in June - August 2013 and 2015. These are caused by the fact that a lot of students and

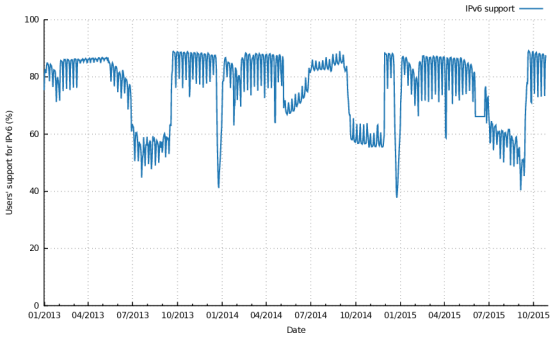


Figure 2.33: IPv6 support among BUT users devices, 2013 – 2015 period.

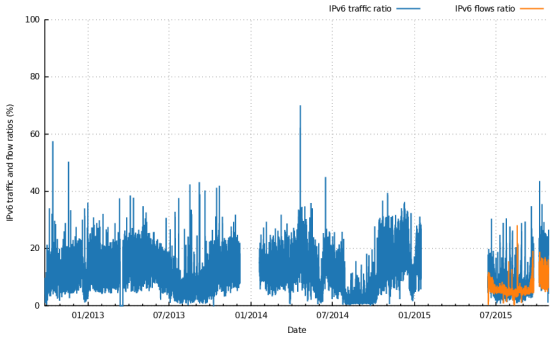


Figure 2.34: Ratio of IPv6 flows and IPv6 traffic

staffs go on holiday. Drop in IPv6 support during November – December 2014 is caused by changes in our routing infrastructure thus the numbers from this period are not relevant.

We only consider dual-stacked eyballs networks in our dataset, thus Figure 2.33 can be used as the answer to the first question. We can see, that the support for IPv6 is very high among end-user devices. However, we have to highlight the fact, that these numbers are relevant only for campus network or similar network (e.g. enterprise), where users' devices are connected directly to the network. ISPs providing Internet access via cable, FTTx, xDSL or even Ethernet are in different position as there is a middleware (CPE device) between users and ISP core network. IPv6 support for CPE devices is unfortunately very problematic and typically only a small number of CPE devices have a decent IPv6 support.

What about the second question? How much traffic will flow over a new protocol? To answer that question, we have been measuring IPv6 traffic volume as well . Figure 2.34 shows the ratio of BUT traffic carried by IPv6 protocol. Since July 2015, we have been also measuring a ratio between IPv4 and IPv6 flows. We can see, that the traffic volume oscillates around 20 % and flow ratio around 10 %. There are drops in IPv6 traffic during summer vacation and Christmas as there are not so many users connected to our campus network. The average IPv6 traffic volume was following – 12.6 % in 2013, 13.04 % in 2014 and 13.25 % in 2015.

It can be seen there is no big difference between traffic volume now and three years ago. It is caused by the fact that a lot of popular websites producing and delivering a large amount of content deployed IPv6 already in 2013. We believe that it will be very hard to raise the ratio from the current level as introducing IPv6 support for all small websites that comprises the rest of the network traffic will be a tremendous effort.

We have heard at several conferences during an informal discussion that IPv6 support for small, rather static websites is not a big issue as users tend to visit only a few websites (social, mail, news) where IPv6 is already enabled. According to our statistics, this is not correct. We analyzed all individual HTTP requests of 9000 users in a day. There were 9.8 % of users that visited less than ten unique websites. The average number of unique domains visited per user was 231. We split the number of visited domains to the intervals as shown in Figure 2.35.

We see that 68.4 % users visited more than 100 unique domains in a day. It does not correspond with the premise that users visit only a few websites. On the other hand, the premise is quite valid if we consider only mobile devices. We have analyzed traffic and flows in our wireless networks. Figure 2.36 shows that the percentage of traffic and flows

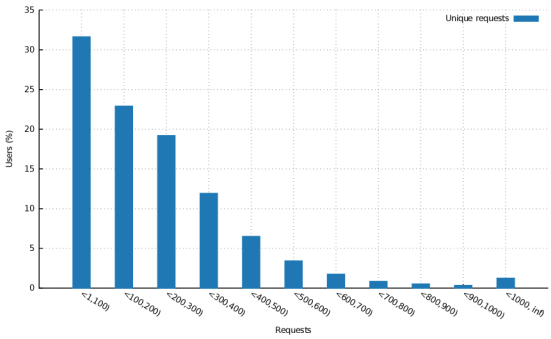


Figure 2.35: Number of unique websites visited by users in a day

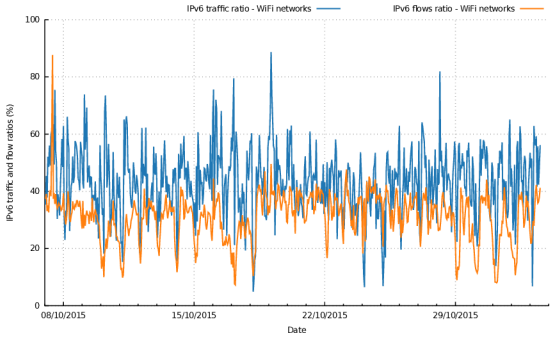


Figure 2.36: Ratio of IPv6 flow and IPv6 traffic in WiFi networks only

is much higher compare to desktop and laptop usage. The traffic volume ratio is 43 % in average, and flows ratio is 30.87 % in average. It is roughly two times more than the flows and traffic ratio in a wired network. Why are these ratios higher? Deeper analysis of traffic shows that Facebook, Google (Youtube), Microsoft and Akamai are responsible for the majority of traffic. Mobile devices are thus mainly used for consuming content rather than for work. This observation corresponds to the number of users as presented by Google[6]. Google monitors IPv6 user penetration very carefully and see a difference between IPv6 penetration during working days (12 %) and weekdays (10 %). One can argue, that mobile market is rising while PC market is falling; thus IPv6 gains more and more traffic in future. On the other hand, people still have to work thus the differences between IPv6 support at work and home will probably remains.

[6]https://google.com/ipv6

2.3 Summary

This chapter discussed issues with the transition to the next generation network. We used IPv6 as an example of a next generation network protocol to describe the process of transition from the old architecture to the new one. This summary highlights the most interesting results.

Routing infrastructure analysis: The overall trend is, that deployment of IPv6 is steadily growing. The support for IPv6 is slightly lower among 32-bit ASes than 16-bit ASes (see Figure 2.2). Currently, a new company receives 32-bit ASN. Thus, it is interesting that support among new companies is low. The number of IPv4 32-bit ASes (new companies) is rising at a higher speed than IPv6 support 16-bit and 32-bit ASes together (Fig. 2.5). The growth of transit only ASes supporting IPv6 drops significantly in 2014 - 2015 period. Slower growth of IPv6 support in the routing core means smaller path diversity and robustness. There are still a lot of prefixes not announced in BGP and transfer markets emerged as an alternative to obtaining additional IPv4 addresses. The number of IPv6 prefixes is growing, but the IPv4 growth is still far faster than IPv6.

We did a correlation of BGP analysis with NetFlow data from BUT and CESNET networks. It is a novel approach, as BGP analysis is usually presented without any relationship with the real network traffic. We found out that there were around 1.5 unique `/64` prefixes in one `/48` prefix in 2013 and around 2 unique `/64` prefixes in 2014. If IPv6 is deployed in a production network, the number of `/64` prefixes in one `/48` prefix should be much higher. The correlation thus suggests that IPv6 prefixes are probably still used mainly for testing purposes and not for the real traffic.

Content availability and quality analysis: We presented several figures showing the speed of IPv6 support among the web, mail and DNS services since 2012. All these numbers are based on our dataset we have been actively building for several years. We performed analysis of the availability of IPv6 domains and measured the quality of connection. The conclusion is that IPv4 and IPv6 perform similarly in most cases. We also found out that occurrences where one protocol performs better than other are mainly caused by differences in routing paths. Our measurements show that there is a substantial number of sites (around 5 %) we are not able to connect to.

User support and traffic volume analysis: We presented long-term statistics of user IPv6 support in our campus network. IPv6 support is very high – around 80 % of devices (laptops, PCs and mobiles) actively use IPv6. The IPv6 support stays almost on the same level as three years ago. By analyzing traffic volume, we did not observe substantial differences between traffic volumes today and three years ago. This is caused by the fact, that main content providers enabled IPv6 in 2012. We also argue that an assumption about users visiting only a limited number of websites is not entirely correct. We found out that 68.4 % users visit more than 100 unique domains in a day (average is 231 unique domains). On the other hand, users tend to visit a smaller number of sites on mobile devices – mainly social networks, email services and chat.

We can draw the following conclusions from these observations. IPv6 deployment introduces changes for user accounting that will be encounter by more and more ISPs and enterprises. We will cover issues in IPv6 accounting in chapter 3. To accommodate rising demand for IPv4 addresses, ISPs will have to employ more and more network address translation devices. Network translation is however introducing problems with user identification and accounting as well. We will cover these issues and introduce a scalable solution in chapter 4.

3

Transition technologies and users accounting

> *"Engineers always deliver the minimum so if you asked for a network that spans the galaxy you can hope that at least you can get one that spans the world."*
> – Bret Victor, *Future of programming*

This chapter describes the principles used for users accounting in the current TCP/IP architecture. The reader is expected to have an advanced understanding of IPv4 protocol. The first part of this chapter describes user accounting in today's ISPs networks. The rest of the chapter discusses differences between users accounting in networks that migrated to the IPv6 protocol. The incompatibility between IPv4 and IPv6 protocols introduces several problems for the current accounting methods. We will highlight these issues and present a system that is able to process all necessary information for user accounting in these networks at scale.

3.1 Current accounting techniques in IPv4 networks

Commercial ISPs in the early 1990's developed a business of providing Internet access for a fee. The first access usually took a form of a dial-up connection [47]. An IP address was assigned to a customer by his ISP for the duration of customer's call and then returned to the pool of addresses for subsequent reassignment for other customers. The ISP were storing the call detail records of the customer's call to be able to generate a telephone bill for him later.

If there was a demand to obtain the information who was using a particular IP address, the central `whois` registry was queried to receive information whom to ask (which ISP is responsible for the IP address range). The time of the day and the IP address were two pieces of information for the ISP to be able to pair the IP address with the telephone number and thus with the user.

The dial-up connection has been superseded by different technologies, e.g. cable connection, xDSL, WiFi, Ethernet or optical fiber. These connections put different requirements to which information must ISP preserve to comply with the data retention policies. These requirements can be summarized in the following points: [1]

[1] The following part of the text describes the situation where the **globally unique** IP address is assigned to an end user. Later sections will describe different scenarios.

- **IP address:** The IP address is usually an identifier from the data retention point of view [48] as well as from the ISP management point of view. If a user always has the same IP address for the whole period of the contract with the ISP, the ISP only needs to store information, which user has which IP address.
- **Timestamp:** The IP addresses can also be assigned in a dynamic way. If the dynamic assignment is used, a timestamp together with the IP address is required. The timestamp is necessary, because without the timestamp, the ISP could not pair the IP address with the correct subscriber.
- **Timestamp to IP address mapping:** The ISP must be able to pair a given IP address with the correct subscriber. Different address assignment methods for end devices can be used, but two are used the most – DHCP and PPP. If an ISP uses dynamic assignment, then the ISP must store logs from DHCP or BRAS[2] server. These logs provide information about the mapping between time and IP address.
- **Network layer to data link layer mapping:** If a dynamic address assignment model is used, the mapping between time and IP address is necessary but not sufficient. The another required information is a mapping between network and data link layer. Usually, it is the MAC address of subscriber's computer or CPE device that is stored. If PPP, PPPoE, PPPoA are used, a session-id or a username can be the information that maps an IP address to the correct subscriber.

All of the mentioned information must be stored by an ISP. Without the information, the ISP will not be able to fulfill the data retention request. We store MAC address of user's device, together with user's personal information. The MAC address is used in DHCP configuration for static IP address assignment. The similar model to BUT is used by other universities in the Czech Republic, as well, e.g., [50], [51] and [52].

Regardless of which accounting method is used, binding between an IP address and a user is becoming insufficient. For example, the data retention regulation in the Czech Republic also requires IP addresses to which the user communicated with[3]. To fulfill the requirement, ISP must store some level of packet activity of all subscribers.

A flow based monitoring stores only a part of packet headers without a payload. It provides a summary who communicates with whom and fits nicely for network management and data retention purposes. The flow based monitoring process stores several pieces of information – who communicates with whom (source, destination IP addresses), which application (source, destination ports) and time (start, end timestamps). The same information can be used for network management where NetFlow can facilitate identification of a new application, detect unauthorized WAN traffic, trace back security incidents, etc.

Because NetFlow data can be used for network management purposes and also for data retention, the flow monitoring is prevalent technique used by network administrators.

3.2 Address assignment in IPv6

This section describes address assignment techniques in IPv6 protocol. We expect that the reader is familiar with the address types and notation in IPv6, if not RFC 4291 [31] and RFC 5952 [56] could be used as references.

[2]Broadband Remote Access Server

[3]This is no longer the case, because the last revision of the data retention directive in the Czech Republic requires **only** the user identification – it means that ISP must not log the destination IP addresses.

One of the main differences between IPv4 and IPv6 protocols is that IPv4 uses only one address per interface. Contrary to IPv4, IPv6 capable device can have (must have in a real world deployment) multiple addresses per interface. Together with a link-local address which is mandatory, the interface may have several other global unicast, unique local or any other addresses from assigned address space [57].

3.2.1 Stateless address configuration

IPv4 protocol supports two ways of IPv4 address configuration - manual address configuration or DHCP (BOOTP). IPv6 maintains both methods (DHCP is replaced by DHCPv6) and introduces a new one – Stateless Address Autoconfiguration (SLAAC) [58].

Both SLAAC and DHCPv6 provide automatic address configuration. The main difference between these two approaches is that SLAAC allows a host to determine an IPv6 address by itself. It is the exact opposite of DHCPv6 operation where a DHCPv6 server maintains information about assigned addresses or prefixes.

The Router Advertisement message used in SLAAC is depicted in Figure 3.1. The important fields related to this thesis are `Router Lifetime`, flags `M`, `O`, `Prefix Information Option` and `A` flag in the `Prefix Information Option`.

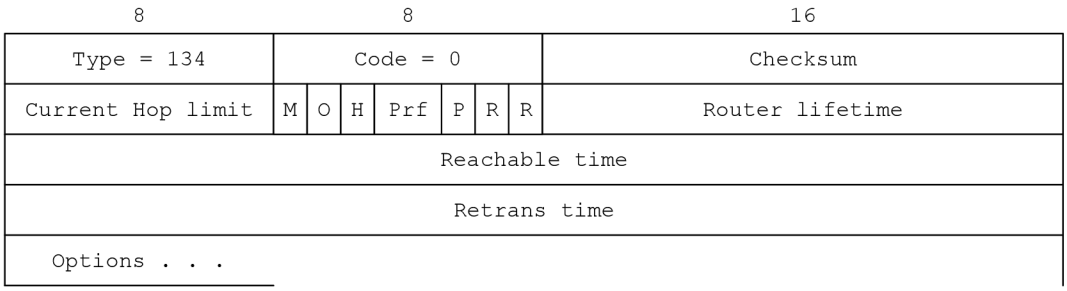


Figure 3.1: Router Advertisement message

- `Router Lifetime` – a value from interval $< 0, 9000 >$ seconds that indicates for how long the router is willing to behave as a default router. Lifetime of 0 indicates that the router should not be used as default one.
- `M` – Managed Address Configuration flag – when set, it indicates, that an address or addresses should be configured statefully using DHCPv6.
- `O` – Other Configuration flag – when set, it indicates, that other configuration information can be obtained from a DHCPv6 server, e.g., DNS servers.
- `Options` – The RA message allows several types of TLV-encoded options. Relevant option for the thesis is Prefix Information depicted in Figure 3.2 and fields `Prefix`, `Prefix Length` and `A` flag.
  - `Prefix` and `Prefix Length` – IPv6 prefix and prefix length.
  - `A` flag – Autonomous address-configuration flag – when set it indicates that this prefix can be used for stateless address configuration.

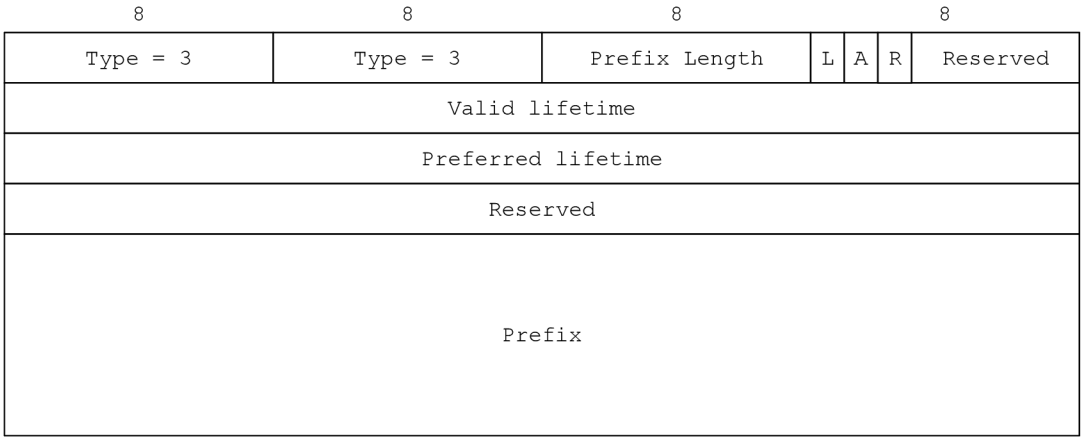


Figure 3.2: Prefix Information Option

Meaning of other flags and fields can be found in relevant RFCs – RFC 4861 [61], RFC 5175 [62] and RFC 6275 [63]. The typical scenario of address assignment using SLAAC in Ethernet networks can be then briefly described as follows:

- Firstly, a host must create a link-local address and assigns it to an Ethernet interface. The link-local prefix is `fe80::/10` or `fe80::/64` for Ethernet networks. The end user identifier (EUI) is derived from MAC address or generated randomly. The link-local address is assigned to the interface and marked as tentative. The discovery of uniqueness of the link-local address is done by Duplicate address detection (DAD) process. To be able to run DAD process, the host must also join *allhosts* and *solicited-node* multicast groups.
- If DAD ends successfully for the link-local address, the system marks the address as valid and use it for receiving a Router Advertisement (RA) message or sending a Router Solicitation message. If the RA message contains Prefix Information Option (PIO) and it is allowed to use the announced prefix for stateless configuration (`A` flag is set), the host will create another IPv6 address combining received prefix and EUI. It depends on host's operating system which algorithm will be used to generate the EUI. The majority of operating systems are using Privacy extension nowadays, thus, the EUI is generated randomly. The host must then joins to a corresponding *solicited-node* multicast group for the new IPv6 address and starts the DAD process to verify its uniqueness.
- The host adds the router, which sent the Router Advertisement message, to the host's default router list indicating that off-links packets could be forwarded via the router. Basically, the router sending the RA message is treated by the host as a default gateway.

The benefit of stateless address configuration is a very fast address configuration even in a large network. A host, however, needs to know also addresses of DNS resolvers. These addresses were originally available only by using stateless DHCPv6 server, but the options for DNS configuration were added to the SLAAC in RFC 6106 [64] in 2010.

3.2.2 Stateful address configuration

The stateful configuration in IPv6 is done by the DHCPv6 protocol. In general, the DHCPv6 protocol is very similar to the DHCP protocol used in IPv4, however, the details of DHCPv6 are very different from DHCP in IPv4.

DHCPv6 features two basic modes – stateless and stateful. The purpose of stateless DHCPv6 mode is to extend the SLAAC autoconfiguration and pass other information to a client based on simple, request – response interaction. The configuration options which DHCPv6 server can assign are listed at IANA webpage [65].

The stateful DHCPv6 can be used to assign IPv6 addresses to a host or other configuration options similarly to DHCPv4. The main differences between using DHCPv4 and DHCPv6 can be summarized into the following points:

- Using a link-local address for requesting an IPv6 address from DHCPv6 server is a cleaner way, how to implement the client. DHCPv4 has to use system specific implementation because it is usually a problem to sent a packet through an interface if the interface does not have assigned any IP address.
- The DHCPv6 messages are multicasted to `All_DHCP_Relay_Agents_and_Servers` multicast group. DHCPv4 uses broadcast, thus, the client must parse every broadcast message.
- DHCPv6 can configure multiple addresses and multiple interfaces in a single exchange.

Two special flags are used for signaling if there is a stateful or stateless DHCPv6 in the network; `M` -– managed flag and `O` -– other flag. If the `M` flag is set, a client should request information using stateful DHCPv6 protocol. If the `O` flag is set, a client can ask for necessary information using stateless DHCPv6. If both flags are set to zero, the end-users stations know that there is no DHCPv6 server available in the network.

We can see one of the biggest difference between DHCPv4 and DHCPv6: the DHCPv6 protocol is not a standalone protocol as in IPv4, but it is closely tight to the NDP protocol, specifically to the Router Advertisement message sent by routers in a network. Unfortunately, it is not clearly defined, whether a client should wait for receiving RA message and strictly follow the information inside the RA packet or not. Thus, the behavior of operating systems is very different. Some implementations take flags in RA message as a hint. Others strictly obey their presence. We will cover these issues and issues related to address assignment later in this chapter.

DHCP Unique Identifier (DUID)

Another fundamental change is a different identifier compare to DHCPv4. The DHCPv4 server uses client's MAC address as an identifier. DHCPv6 uses a concept of DHCP Unique identifier (DUID). According to RFC 3315 [66], the DUID identifier is designed to be unique across all DHCP clients and servers, and stable for any particular client or server. Furthermore, DUID must be treated as opaque values. Currently, there are four DUID identifiers defined.

- **DUID-LLT:** Link-layer Address Plus Time: This DUID is based on a combination of time and link layer address of any network interface that is connected to the DHCP device at the time that the DUID is generated. This is the default identifier in the majority of operating systems.

- **DUID-EN:** Vendor Based on Enterprise Number: The DUID is assigned by the vendor. It consist of the vendor's registered Private Enterprise Number and a unique identifier assigned by the vendor.
- **DUID-LL:** Link-layer Address: Link layer address of any one network interface is used.
- **DUID-UUID:** Universally Unique IDentifier [67] is used for this DUID.

3.3 Transition technologies

"We have more transition mechanisms than IPv6 packets."
– Randy Bush, *RIPE 62*

There are several several options that can be used to overcome the incompatibility between IPv4 and IPv6 protocols.

- **Dual-stack:** The most straightforward way for new nodes to remain compatible with the older nodes is by providing a complete implementation of both protocols and run both protocols simultaneously.
- **Tunneling:** A transition mechanism that provides a connection for the new nodes over the old protocol is another approach.
- **Address family translation:** There could also be a translation service or gateway between incompatible protocols.

We will cover dual-stack and tunneling mechanisms in this section. Address family translation between same or different address families will be covered in the next chapter. Tunneling transition mechanisms can be divided as follows:

- **IPv6-in-IPv4 tunneling:** IPv6 packets are transmitted over an IPv4 network. The general idea is to embed an IPv6 packet in the payload of an IPv4 packet as described in RFC 2473 [68]. The protocol number in the IPv4 header is set to 41. Several mechanisms were designed to use this approach; e.g., 6to4, 6in4, 6rd or 6over4. These mechanisms are considered as stateless tunneling mechanisms, as there is no extra configuration burden. There is also an approach where a shim header is inserted between IPv4 and IPv6 headers. Examples of a shim header can be GRE or MPLS headers.

  There are also stateless transition mechanisms that use a different approach. Teredo uses IPv4 protocol for tunneling, but the IPv6 payload is encapsulated in UDP and Teredo specific headers. It provides an IPv6 connection for users behind IPv4 NAT. ISATAP is similar as 6over4, but requires a particular configuration of network.

- **IPv4-in-IPv6 tunneling:** Several transition mechanisms were designed to transmit IPv4 packets over IPv6 networks. Examples are DS-lite [70], MAP-T [71] and MAP-E [72], A+P [73] or 4RD [74]. These mechanisms allow to deploy IPv6 only network in ISP core. Eyeball subscribers are connected to the IPv6 only network, and their IPv4 traffic is tunneled over the IPv6 only core.

- **Other type of tunneling:** There are transition mechanisms that use a different approach. LISP, BGP tunneling, or tunneling mechanisms as L2TP or VxLAN can be used during the transition from IPv4 to IPv6. These mechanisms create an overlay network that can be used for interconnection of IPv6 networks over IPv4 or L2 protocols.

The tunneling transition mechanisms have a severe impact on ISP accounting process. The flow accounting creates statistics only from the topmost header. If IPv6 is tunneled in IPv4 header, flow based accounting creates a flow according to IPv4 addresses and protocol numbers as depicted in Figure 3.3. In this case, the ISP loses the information about services and protocols transmitted inside the tunnel.

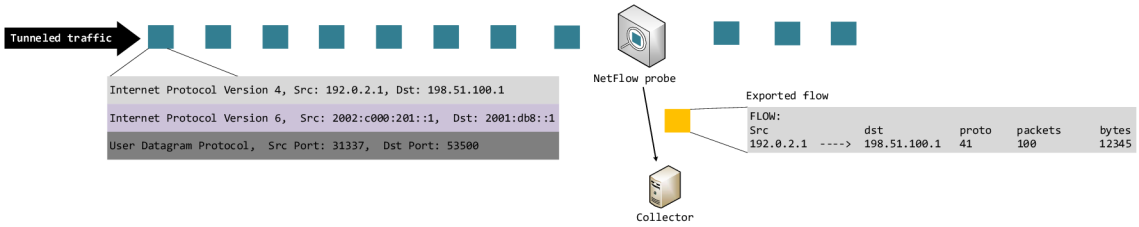


Figure 3.3: IPv6 tunneling over IPv4 protocol

3.4 Challenges in user accounting

This section describes the differences between users accounting of IPv4 and IPv6 protocols. We should highlight the fact that requirements for user accounting vary between ISPs. An ISP operating a cable network has different requirements compare to an enterprise or an academic networks. As we have ostly experience with academic, enterprise networks and eyeball ISPs, we focus on these types of networks in the rest of this section. We will, however, alert the reader if there are noticeable differences with other types of network.

3.4.1 Dual-stack

The dual-stack approach is currently a recommended approach for IPv6 deployment and it requires to run IPv4 and IPv6 simultaneously. We presented main methods for users accounting in IPv4 in section 3.1. Are the current approaches used for accounting IPv4 users valid for a dual-stack network? The short answers are „*No*“ or „*It depends.*“. There are several differences in the IPv6 address assignment that obsoletes the current techniques for accounting IPv4 users.

SLAAC

Let's take a typical SLAAC implementation as an example. User's device always has a link-local address as this address is mandatory. Using RS and RA messages, the user's device obtains a global IPv6 address. This address is generated according to EUI-64 algorithm or randomly. Even though that the address is random, another random address is generated. This address is a temporary address, and it is used for outbound communication. The address is valid for one day or one week (depends on the system configuration) then the


```
MAC:  08-00-27-6b-2a-ce
Global IPv6 addresses:
     2001:67c:1220:80e:bb:a4e9:5e45:c261   (DHCPv6)
     2001:67c:1220:80e:e98c:aac:5b07:6d60  (Stable private)
     2001:67c:1220:80e:51ec:7373:fdf4:777a (Temporary)
Link-local IPv6 Address:
fe80::e98c:aac:5b07:6d60%11 (Stable private)
```


```
MAC:  68-b5-99-ea-d4-8a
Global IPv6 addresses:
     2001:67c:1220:80e:98:3d70:925b:e28b   (DHCPv6)
     2001:67c:1220:80e:3f89:4943:87c5:7cc  (RFC 7217)
     2001:67c:1220:80e:ace7:613c:1aff:53ff (Temporary)
Link-local IPv6 Address:
fe80::6ab5:99ff:feea:d48a (EUI-64)
```

Figure 3.4: Different IPv6 addresses configured on Windows and Linux operating systems

system generates a new one. The older addresses are deprecated, but they can still be used for incoming communication for a while. Thus, a device with a long uptime can use several IPv6 addresses simultaneously.

Let us consider the standard installation of Windows and Linux operating systems to better understand the address assignment process. Figure 3.4 shows addresses configured on each system after the first boot. The Windows operating system configures four different IPv6 addresses – a link-local and three global unicast addresses. We can see that neither of these addresses is based on device's MAC address. IPv6 address `2001:67c:1220:80e:e98c:aac:5b07:6d60` is created randomly and never changes. Notice that the IID of the stable random IPv6 address (`e98c:aac:5b07:6d60`) is the same as the IID of the link-local address. Since Windows 7, the Windows operating system will reuse the random interface ID that was generated in the link-local IPv6 address in the Global Unicast and/or ULA addresses. In other words, the same IID will be used if the device is connected to a different network. IPv6 address `2001:67c:1220:80e:bb:a4e9:5e45:c261` was obtained from DHCPv6 server and the last one `2001:67c:1220:80e:51ec:7373:fdf4:777a` is a temporary address that changes regularly.

The situation in Linux operating system is different. There is a link-local address and three global unicast addresses as well. Notice, however, that the link-local address is created based on EUI-64 algorithm, thus, the address leaks device's MAC address. IPv6 address `2001:67c:1220:80e:3f89:4943:87c5:7cc` is a random IPv6 address created according to RFC 7217. It means that if the device is connected to a different network (there is a different IPv6 prefix, SSID, etc.), the address will be different. The DHCPv6 address and temporary address have the same meaning as in the Windows example.

We see that different operating systems use different mechanism to create their IPv6 addresses. We run an analysis in the BUT network to observe the number of unique IPv6 addresses per user. We analyzed an operating systems behavior of approximately 6000 dual-stacked users in years 2013, 2014 and 2015. Results are presented in Figure 3.5.

The average number of unique addresses was 3.62 addresses per user in 2013, 4.01

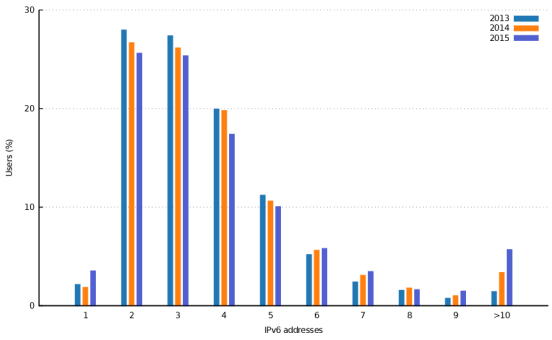


Figure 3.5: Number of IPv6 addresses per user, years 2013, 2014, 2015

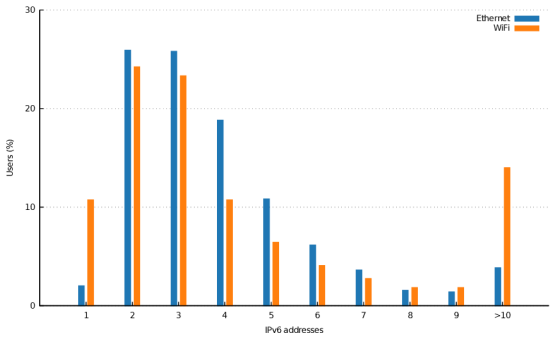


Figure 3.6: Number of IPv6 addresses per user in WiFi and Ethernet networks

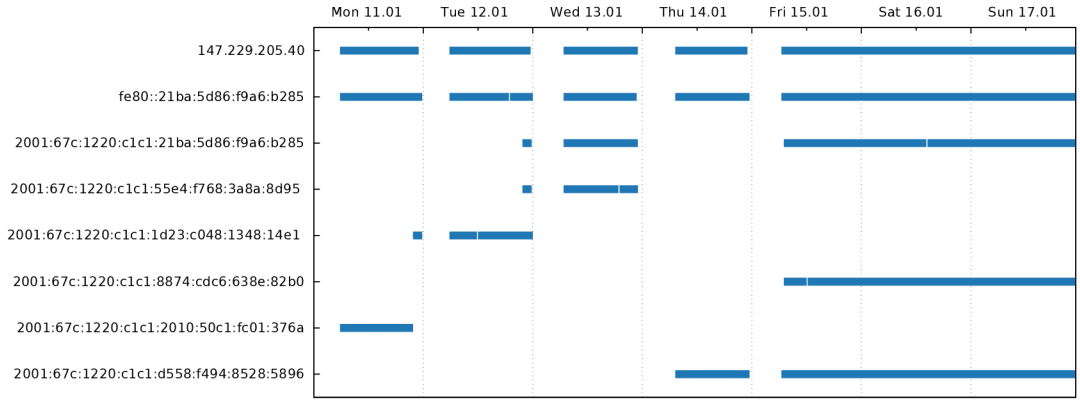


Figure 3.7: IPv6 addresses of a device during period of one week

addresses per user in 2014 and 4.57 addresses per user in 2015. The median remains the same – 3 unique addresses per user. Why is the average increasing and why devices use more and more addresses? Temporary addresses are the primary reason. However, there are other reasons as well. A device having a stable network connection during a day, for example, a desktop computer, will typically not exceed three IPv6 addresses per day (link-local, random and temporary random address). On the other hand, a device that regularly connects and disconnects from the network, generates sometimes a unique IPv6 address for every connection to the network. The increase in the number of addresses is, thus, driven by the rising number of mobile devices in our network. We have identified several devices that used more than 100 unique IPv6 addresses per day – the record holds a HTC device with 197 unique IPv6 addresses per day! Figure 3.6 confirms that there are more IPv6 addresses per device in WiFi compare to wired (Ethernet) network.

Figure 3.7 shows a behavior of an operating system on a wired network over a longer period of time. The computer obtained an IPv4 address from DHCPv4 server, generated a link-local IPv6 address and several unique global temporary IPv6 addresses. You can observe that IPv4 and link-local addresses are stable and active during the time when the device was turned on. Global IPv6 addresses were used randomly during the whole week. There were three unique IPv6 addresses simultaneously in use since Friday 15.1.2016. The device used these addresses for the entire weekend for transmitting and receiving traffic.

The behavior of operating systems presented above creates a serious problem for a

standard accounting process. If SLAAC is used in a network, the IPv6 address **cannot** be used as an identifier of a user's device. However, the problem is that all requests from law enforcement agencies, CSIRT teams, NOC[4] teams or companies complaining about unauthorized distribution of movies, music, etc. cannot use a different identifier than IPv6 address! The reason is simple. They do not have any other information from their side of the connection. They can extend their complaint with port numbers of L4 protocol and time of the incident, but this information is still not sufficient for a network provider to identify the user. We have seen these kinds of issues and complaints and we have received complaints containing IPv6 addresses only. Hence, the accounting process must be updated to reflect these issues.

DHCPv6

SLAAC introduces several issues for user accounting process. Is there a way how to solve these problems? Why cannot an ISP use DHCPv6 protocol for IPv6 address assignment similarly to IPv4? In the rest of this section, we introduce several reasons why we did not choose DHCPv6 for address assignment in our environment and why it is problematic to use DHCPv6 in general.

One of the problems is that DHCPv6 support is not mandatory for IPv6 end nodes according to RFC 6434 [92]. There can be devices without DHCPv6 support. If DHCPv6 is the only protocol used for address assignment in a network, there can be devices that will not be able to connect to the networks. Thus, an ISP is not motivated to deploy DHCPv6, as the ISP must run SLAAC anyway. Running two protocols that provide the same thing is an operational burden.

Android platform does not support DHCPv6 protocol. Android developers declined to implement the DHCPv6 support [93]. Their main objection was that DHCPv6 can limit the number of available IPv6 addresses for a device which breaks network tethering. A device must have several IPv6 addresses to be able to distinguish between a native IPv6 connection and connection that should be tethered to a different device. The another approach is using IPv6 NAT. Android developers do not want to implement IPv6 NAT, thus, they rely on protocols that can assure several IPv6 addresses for an end node. This can be achieved using SLAAC. A device can generate an IPv6 address from obtained prefix according to its needs. However, the same is not true for DHCPv6. Developing future applications can be, thus, cumbersome. Several people objected that the DHCPv6 protocol can allocate an entire IPv6 prefix using DHCPv6 Prefix Delegation, but there is no experience and guidance how an end node should handle an entire IPv6 prefix. Thus, all networks without a strict control over connected devices (public hotspots, campus networks, etc.) cannot use DHCPv6 if they do not want to cut off Android devices. Considering the fact that Android has a large mobile segment share (80%), it makes the problem even bigger.

There are other reasons why DHCPv6 is not used for address assignment of end nodes. DHCPv6 cannot provide information about a default gateway and IPv6 prefix. Thus, it must run together with NDP protocol. Why is there such a requirement? A layer violation. It is believed by several people in the networking community that DHCPv6 protocol, as a protocol of application layer, should not contain information from network layer. Thus, default gateway information or the prefix length information cannot be included in the DHCPv6 message as it would not be a „pristine“ design. Ironically, Router Advertisement message contains information about DNS servers (RFC 6106 [64]) or information about

[4]Network Operations Center

IPv6 address or a domain name of captive portal (RFC 7710 [94]) which are, by definition, services of application layer. The pristine design was, thus, broken in case of NDP, but people will argue that it is still necessary for DHCPv6. There were several proposals by ISPs to soften the refusal of default gateway in the DHCPv6 protocol. All these proposals were rejected. In the end, an ISP must run DHCPv6 protocol together with NDP protocol to assign an address to end nodes. ISPs, however, run their networks for business reasons and maintaining two protocols while there could be only one, increase their operating expenses. The practical consequence is that stateless address autoconfiguration is usually used in end networks. DHCPv6 is used mainly for IPv6 prefix delegation to user's CPE.

Furthemore, there is another obstacle in DHCPv6 protocol that hinders the use of the protocol for address assignment – DHCP Unique Identifier (DUID). The DHCPv6 protocol does not use MAC address as identifier of the client. There can be different types of DUID – link layer address + time, vendor specific identifier or link layer address only. In almost all current implementations, the first type is used – a link layer address + a time value when the DUID was generated. We have faced the following operational problems.

- **Reinstallation:** If a user reinstalls his/her operating system, DUID is changed.
- **Dual boot:** User can run several operating systems on a device. In this case, every operating system has a different DUID.
- **Cloning:** If an operating system is cloned, the DUID remains the same – it is a problem especially in virtual environments.
- **Firmware updating:** Updating firmware on CPE devices can lead to generating a new DUID as it is basically a new installation of operating system. CPE will, thus, receive a new IPv6 prefix as DHCPv6 Prefix Delegation is based on DUID identifier.
- **Forward knowledge:** It is currently not possible to determine the DUID for a device beforehand. The common workflow for configuration of user's CPE is following: Network administrator reads the MAC address or scan the barcode on CPE device and stores the value to a central configuration system. The system updates DHCPv4 configuration based on CPE's MAC address and the CPE is shipped to a customer. Only thing what a customer have to do is plug the device into the socket and the device download all necessary settings from DHCP server. This workflow is broken for DHCPv6.
- **Stability:** Several implementations of DHCPv6 clients change DUID value when the device is restarted. Fixing the problem is sometimes impossible for the ISP as the device is not under its control.

There are arguments that a link layer address of NIC can be easily changed as well which creates the same problems as unstable DUID value. Although it is true that a link layer address can be modified, we believe that the situation is different. If a user has to register his device before using it on the network and the user changes the link layer address, another registration is required. The user is not motivated to do that as he cannot gain any benefit. On the other hand, it is common that users reinstall their laptops, update the operating systems, e.g., from Windows 7 to Windows 10, or uses more operating systems. These actions create different DUIDs which trigger a new registration process. However, the device is still the same, thus, the user is only puzzled with the whole situation.

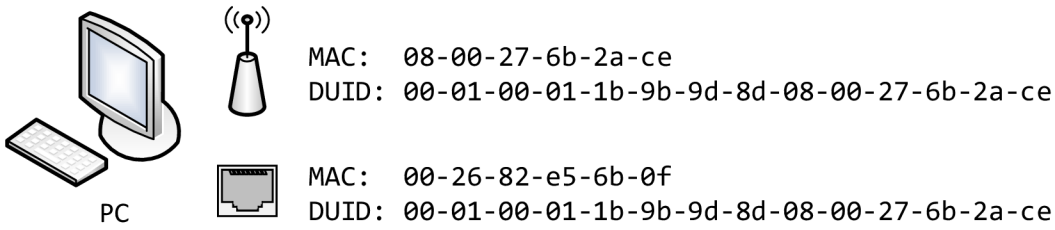


Figure 3.8: DUID and network interface cards

A robust accounting process can, in theory, handle problems with DUID instability, e.g., Interface ID option (similar to Option-82 in DHCPv4) can be inserted in DHCPv6 messages by access switches. However, the feature is usually not supported on current devices. Even if it was, does it mean that DHCPv6 is a valid option if we want user accounting in IPv6 networks?

Let us consider Figure 3.8. The figure shows a device with two interfaces – wireless and wired. During installation of an operating system, DUID is created. In the example, the following DUID is created `00-01-00-01-1b-9b-9d-8d-08-00-27-6b-2a-ce` (time + link layer address). One could be in temptation to use the link layer address from the DUID similarly as in IPv4 networks, but it is not a valid approach. RFC 3315 clearly states that clients and servers must treat DUIDs as opaque values and must not interpret the DUID in any other way. We can break this rule in practice, but it will not help us either because the choice of a network interface for creating DUID is entirely arbitrary. The consequence is that it is not possible to link the information from DHCPv4 server with logs of DHCPv6 server to obtain the full picture of all assigned IP address for a device.

Fortunately, there are other approaches. If DHCPv6 server is in the same segment as a client, the server can obtain the link layer address from the Ethernet header. This approach, however, does not scale very well. The best solution is probably RFC 6939 [95] where DHCPv6 relay adds the client link-layer address to DHCPv6 packets. Network equipment, mainly switches acting as DHCPv6 relays, must support RFC 6939. It is, however, not a commonly implemented feature yet.

Several people argue that IEEE 802.1X authentication framework should be used instead relying on link-layer addresses and all the necessary information should be extracted from IEEE 802.1X server logs. The address assignment process is, however, a layer above 802.1X. In other words, there must be a DHCP server that assigns an address to the user. 802.1X allows accounting, thus a NAS can export information about assigned IPv4 addresses (`Framed-IP-Address` attribute), but it is not supported on all platforms. RFC 6911 defines similarly attributes for IPv6 networks, but implementations are missing.

The consequence of these issues is that DHCPv6 is currently not a popular address assignment method for the end networks where SLAAC prevails. The DHCPv6 protocol is mainly used for IPv6 prefix delegation where ISPs push entire IPv6 prefix to customer's CPE.

3.4.2 Tunneling transition techniques

"Tunnels are evil."

– Geoff Huston, *RIPE 71*

Tunneling techniques are an inevitable part of the transition to the IPv6 protocol. User's IPv6 traffic can be tunneled inside IPv4 without a user or network admin intervention. Standard accounting techniques, such as NetFlow, cannot handle the inner traffic – see previous Figure 3.3. A network administrator loses an insight into the traffic which can cause problems.

3.5 Solving the challenges in user accounting

Is it possible to create an accounting system equipped with all the necessary information for user accounting? Is it feasible to deploy the system in a reasonable large network to test the system scalability? This section describes how it is possible to handle accounting of tunneling techniques. Later we describe how we overcame limitations of current accounting techniques in a dual-stacked network. All methods presented in the following sections were tested in a real, production BUT campus network. The core of the network uses 10 and 40 Gbps links, thus, the system is tested at reasonable speeds.

3.5.1 Tunneled traffic accounting

To overcome issues with tunneling transition techniques we developed two solutions. The first one is a hardware accelerated probe that is able to process packets on high utilized links. The second solution is a software probe with a slightly lower performance that is still able to account packets on BUT core links (if there is a standard Internet packet size distribution[5]).

Hardware probe

The hardware probe was developed by our collegues Martin Elich and Pavel Čeleda. The probe uses a hardware acceleration allowing to parse IPv6 transition tunnels at line rate. These probes were deployed at the CESNET2 network and result published in our paper *Monitoring of Tunneled IPv6 Traffic Using Packet Decapsulation and IPFIX* [96]. The probe is able to achieve high packet processing speed with no need to use packet sampling. The speed is achieved by distribution of packet to different processors.

The architecture of the probe consists of three layers as depicted in Figure 3.9. The first layer is a specialized hardware (NIC with FPGA module). The purpose of the layer is capturing packets and distributing the packets to different instances of flow exporter.

The developed solution uses a packet header parser directly on the NIC card. The parser extracts the following fields for flow identifications: source and destination IP addresses (128 bits per address), source and destination ports (16 bits per port), protocol number (8 bits), IP version (4 bits) and card's input interface. If the field is not present in the packet header, or the header cannot be parsed to find the field, all bits of the field are set to 0. The output of the parsing unit is a sequence of fixed length bits which is passed to a hash unit. The hash unit computes CRC hash with the length of $log_2(number\ of\ channels)$. Each packet is sent to one of the channels according to its hash (the hash is used to address a channel).

[5] see Frame Size Distribution at AMS-IX exchange point – https://ams-ix.net/technical/statistics/sflow-stats/frame-size-distribution

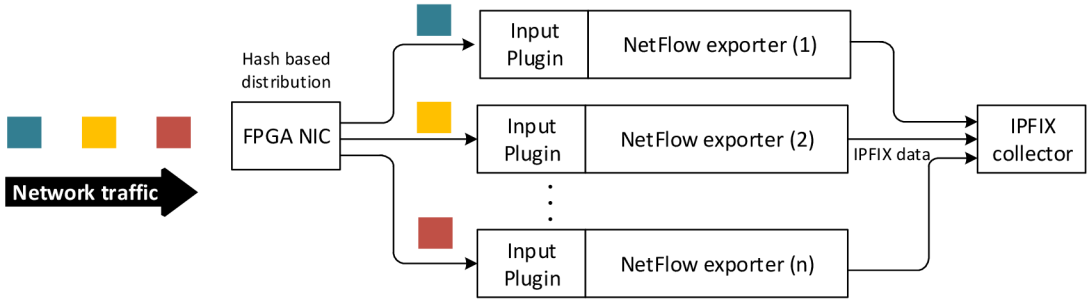


Figure 3.9: Architecture of the probe. Packets are captured by the NIC card and distributed up to 16 exporter instances.

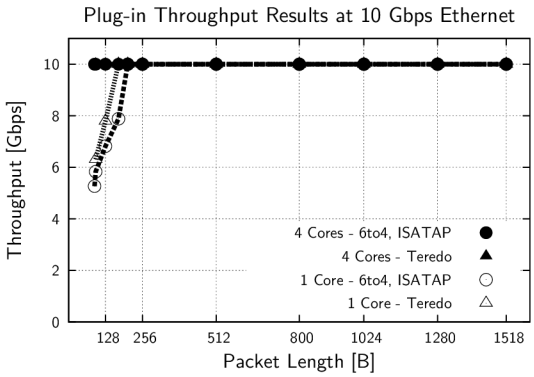


Figure 3.10: Throughput of the plugin at 10 Gbps Ethernet

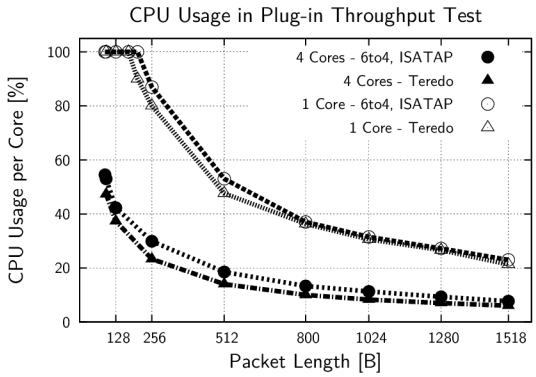


Figure 3.11: CPU usage during the test on single CPU core and multicore system

The second layer reads packets from the NIC and processes them. The NetFlow exporter is FlowMon Exporter developed by Flowmon Networks. The FlowMon Exporter can export NetFlow and IPFIX data and provides plugin support via defined API. My colleague Martin Elich designed and implemented a plugin for monitoring of IPv6 tunneled traffic. The plugin reads packets from the NIC card and processes them. Each IPv4 packet is parsed and analyzed to detect a presence of tunneling. Detection supports the following tunneling mechanisms: Teredo, 6to4 and ISATAP. The mechanism of the decapsulation is described later in this section.

Performance

Packet processing performance was measured on 2.0GHz quad-core CPU. We measured the overall throughput of 10 Gbps Ethernet network link and CPU usage. The throughput was measured for Teredo and 6to4 packets. The plugin performance for ISATAP traffic is the same as for 6to4 traffic, as the encapsulation is the same. A single instance of the FlowMon Exporter with loaded plugin was able to process packets with a size larger than 192 bytes at line rate. Four instances of the FlowMon Exporter with loaded input plugin were able to process all packets at line rate with medium to low CPU load on every core. Results are shown in Figures 3.10 and 3.11.

Contribution

The hardware probe was designed by my collegues Martin Elich and Pavel Čeleda. The original idea came from Tomáš Podermański. My contributions in this case were an analysis of captured NetFlow data (50 %), creation of statistics and participation in writing the paper (50 %) [96]. I did the presentation at the Traffic Monitoring and Analysis conference.

Software probe

The hardware probe has an advantage of creating statistics even on a link fully saturated with small packets. There are some disadvantages, though. Firstly, the hash for distribution to different CPU cores must be computed on the card. This can be a problem in some situations, e.g., a packet is fragmented. Secondly, the price of the solution is very high. We were in the situation where there was a demand for accounting of tunneled traffic on our campus network, but we could not afford several hardware accelerated probes. Fortunately, with the help of our colleagues, we were able to develop a software probe that is cheaper and provides sufficient performance to monitor user traffic on 10 Gbps links. The software probe is able to cope with 10 Gbps link saturated with standard Internet packet size distribution[6]).

The probe can be deployed on a service card module for HP 5406 switch. The module contains two internal 10 Gbps links connected directly to switch's backplane and out of band management port for external configuration and monitoring. A benefit of using a service card that is directly inserted into the switch's backplane is that we can configure the switch to copy traffic from selected ports to the service module. This feature gives us a possibility to monitor even inter VLAN traffic which is usually not available for a standalone probe. We developed an input plugin that is very similar as the one for hardware probe. Software probe uses network pseudo devices `rawnetcap` which is similar to `PF_RING`. It obtains packets directly from a network card. This approach bypass kernel network stack allowing much higher capture speed with less CPU overhead. On the top of these pseudo devices, we developed an input plugin for the exporter that is able to detect Teredo, 6to4, 6rd, ISATAP and AYIYA (Anything in anything) encapsulations.

The detection of 6rd, ISATAP and 6to4 tunnels is similar as they use the same encapsulation – IPv4 header is directly followed by IPv6 header. The plugin detects this encapsulation and passes a IPv6 packet to the IPv6 packet parser. The parser decides which tunneling mechanism is used according to the structure of the IPv6 address, e.g., if the IPv6 address is from the `2002::/16` prefix, 6to4 encapsulation is used, etc.

The detection of AYIYA tunneling is more difficult. The AYIYA tunneling uses UDP port 5072 in general, but the port can also be different. Furthermore, it is necessary to distinguish between AYIYA traffic and other traffic on port 5072. To detect AYIYA traffic, we use several fields in AYIYA header, such as the Epoch Time field. This field is used as a protection against the reply attack, and if the epoch time differs too much, the tunnel cannot be established. Thus, we can use it as a helper for detection of AYIYA traffic as we can use precise timestamp on the probe and the epoch field must be in a small interval around the timestamp. If the AYIYA header is validated and timestamp fits in the small interval around probe time, we can pass the next layer protocol in AYIYA header to appropriate parser. The parser extracts all necessary information about the inner flow and stores it in exporter's flow cache.

[6] https://ams-ix.net/technical/statistics/sflow-stats/frame-size-distribution

The most challenging task is a detection of Teredo traffic. The initial client communication is with Teredo server and it usually uses UDP port 3544. However, the communication between Teredo client and Teredo relay can flow on arbitrary ports. Furthermore, the detection is limited by the fact that the probe must be stateless – if we store stateful information, there would be a serious performance impact. Thus, we have to process every packet independently. The probe processes every unicast UDP packet trying to detect Teredo encapsulation. We search any of the Teredo specific headers, either Origin, Authentication or their mix, and Teredo specific IPv6 address field. If all headers and fields are consistent with the Teredo specification, we mark the packet as a Teredo packet. The information about the outer protocol is stored as well.

Using either software or hardware probe, we gain the visibility into tunneling traffic, thus accounting and backtracking of security incidents are quite easy tasks. Although usage of tunneling techniques, such as Teredo and 6to4, fall significantly during last few years, the monitor solution is available, and we can adapt it for techniques that can be used in future.

Contribution

I developed the input plugin for the software probe and integrated the probe into the central BUT monitoring system. Data captured by the probe were used several times in presentations at different conferences and lectures by myself or my collegues. The software probe source code was published [98], thus it can be incorporated into the commercial Flowmon probe.

3.5.2 Dual stack accounting

In the previous section, we introduced software and hardware probes. These probes can be used for user accounting even if a tunneling transition mechanism is used. The inner traffic can be still mapped to an IPv4 address of the user, thus, an admin can always find who is responsible for the traffic.

What will happen, if an ISP choose to deploy dual stack? In that case, user's traffic flows natively over IPv6 and the admin cannot correlate IPv6 traffic with the user as IPv4 and IPv6 protocols are completely independent. How to solve these issues to be able to account a user?

Firstly, we have to create a protocol agnostic identifier – an identifier which is not tied to any networking protocol. Why? It simplifies the implementation, queries and database. If every user has a unique identifier and all traffic is linked to this identifier, it is convenient for an accounting process to create statistics based on the user's identifier. We can also perceive the protocol agnostic identifier as an abstraction for the accounting process. The accounting process does not care if IPv6, IPv4 or other protocol is used as all these protocols are somehow linked to the identifier. Secondly, we have to create the link or mapping of all user's flows (IPv4, IPv6, ...) to the user's identifier. However, how to create such a mapping? What can be used as a common identifier for both IPv4 and IPv6 flows?

One solution can be to force a client to create such a mapping and publish all necessary information. For example, an ISP admin can demand that user's device must register itself to ISP's DNS by creating a dynamic DNS entry for every IPv4 and IPv6 addresses that the device generates. The solution has one advantage – the functionality is present in the end clients. It agrees with the end-to-end principle, but unfortunately, there are networks

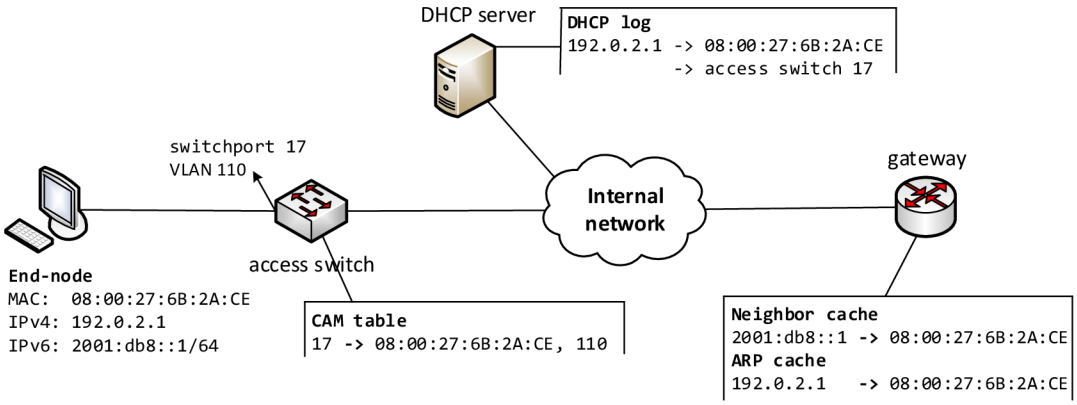


Figure 3.12: A simple network topology showing which information are available in different parts of the network.

where the solution does not fit. Let us consider a different approach presented in Figure 3.12. The figure describes a typical scenario how an end node is connected to a network.

The figure shows an end node connected through an access switch to ISP's internal network and through a gateway to the Internet. We can find several pieces of information on different devices. The access switch knows a MAC address and a switchport of the connected end node. There could also be information about the IPv4 address if First Hop Security is deployed (e.g., DHCP Snooping or SAVI[7] frameworks). The IPv4 address is also present in logs of a DHCPv4 server. The gateway holds mapping between IP address and MAC address. These mappings are stored in ARP cache table for IPv4 address and Neighbor cache table for IPv6 addresses.

We can see that information is repeating – mainly MAC address and switchport. Thus, we can use this information to glue necessary pieces together. We should, however, highlight the fact that the MAC address should not be used as the key! A user can change hardware (a common incident) or change its MAC address in operating system (less frequent, but it can happen). Thus, the identifier (UserID) must be different – it can be a random number, user's login or something else. It does not matter what, except that the identifier must be unique in the whole ISP network.

To put all the information together, we can reuse the infrastructure that is used in IPv4 world as described in 3.1. The accounting system must be extended with additional functionality that holds necessary information for IPv6. There should also be a possibility to insert additional information that is network specific, e.g., a Radius login or a PPP identifier.

The data structure for user identification can be the following tuple: *(UserID, Timestamp, L2 address, L3 address)*. If we collect and store this information, we can answer simple questions, e.g., *„Who used this IPv6 address on 1.1.2016?“* or *„Which addresses are connected with user John Doe?“*. Unfortunately, we cannot answer questions like: *Which servers were contacted by the user?* or *Can you confirm or deny that there was a communication between these two users?* This does not fulfill necessary law requirements for data retention, thus, additional information must be collected. Fortunately, we can reuse

[7]Source Address Validation Improvement

NetFlow/IPFIX protocol and extend the data structure with all the necessary information for every flow. Using NetFlow we can obtain the following tuple: *(Timestamp, source L3 address, destination L3 address, source port, destination port, protocol)*. If we extend the NetFlow tuple with UserID, we have the required information.

This solution provides all the required information and can be deployed easily as several key components are already set for IPv4 accounting. However, several interesting questions remain. Mainly:

- How to collect all necessary information about IPv4, IPv6 mapping, MAC to switchport mapping, etc?
- Does it scale? Does the solution have necessary performance?

Collecting the necessary information

There are several ways how to collect all the information. It depends on the type of the network, management tools that are used in the network, knowledge of network administrators, etc. We are going to introduce several possible approaches and highlight their benefits and drawbacks. The final solution probably varies with the network requirements, design and devices. A network administrator must choose a method that works for him best.

Intercepting management protocols

One possibility how to obtain a mapping between IPv6 and MAC addresses is to monitor ICMPv6 traffic. This approach was introduced in [99]. The method makes use of the DAD mechanism that must be triggered if a device configures a new IPv6 address. The method monitors traffic in a LAN and tracks all Neighbor Solicitation messages that carry information about the mapping between IPv6 and MAC addresses.

Benefits: The system does not require any support on end nodes. Furthemore, the address detection is very fast – there is only a minimal delay. This is convenient if there is a need to use the information immediately, e.g., a firewall rule is created based on the detected IPv6 address.

Drawbacks: The system must have a visibility to all VLANs present in the network. ICMPv6 messages are not reliable. They can be lost, especially in WiFi environment. If a message is lost, the system will not learn the mapping. It is not known how the system behaves on a larger network. It was tested only on a rather small network (`/24`). If MLD snooping is deployed in the network, there could be race conditions. The monitoring system can miss the Neighbor Solicitation message as system's port is not yet excluded in the MLD table of the switch. This drawback is only present in some networks with particular network equipment.

SNMP protocol

Another solution for collecting all the necessary information from switches and routers could be SNMP protocol. It is possible to collect L3 to L2 mappings only from `ipNetToPhysical` SNMP table. There are other tables necessary to query – `dot1qTpFdbTable` for mapping between VLAN, MAC address and switchport and `dot1qVlanCurrentTable` and

`dot1qVlanFdbId` on some switches to find information about VLAN ID. There are several either commercial or open-source software that can be used to gather this information, e.g., NAV[8] or netdisco [9].

Benefits: The SNMP protocol is widely supported by network devices. Network vendors often add additional info and statistics to SNMP MIBs that are not available otherwise. If a recent firmware is used, the network device can send a message with necessary information (a push-based approach), e.g., a SNMP trap or a Syslog message. This eliminates the delay of traditional pull-based approach.

Drawbacks: The SNMP protocol sometimes does not work well in large networks. Usually, it is caused by combination of slow CPU on the device and the fact that the response to SNMP request is sorted by OID value. If a large table is queried – which is the case of `ipNetToPhysical` and `dot1qTpFdbTable` tables, the device's CPU is overloaded, and the generating of SNMP responses can affect the normal behavior of the device. The mapping is also obtained with a delay as SNMP is usually used in pull-based mode.

Custom based tools

We overcame limitations of SNMP protocol and management protocol interception by writing a custom script that use device's command line interface via SSH protocol. The script connects to the device using SSH credentials, enters necessary commands and downloads the outputs. The script does not burden the CPU as the CLI implementation is much more efficient – the output does not have to be sorted, the device does not have to create SNMP packets, etc. Although we admit that there could be vendors that implement SNMP properly on their devices we had severe problems even with equipment from big vendors. According to discussion on several mailing lists, we are not alone, and others use a similar approach if query for a large amount of information.

Benefits: Similarly to SNMP, SSH access is widely supported. The solution, thus, can be deployed even in multi-vendor environment. It is effective even with large tables. The device's CPU processor is not overloaded even when we pull several thousands of records.

Drawbacks: Implementation is more complicated compared to SNMP protocol. There must be a specific parsing grammar for every vendor as different vendors use different commands and have different output syntax. It is still a pull-based approach. The mapping is obtained with a delay. Although it is possible to query devices reasonable often, e.g., every 1-2 minutes, it is still not as efficient as push-based approach.

Time interval consideration

The time dependency of gathering different data is crucial when accessing caches on a device. Caches hold information needed to build dependency between L3 and L2 addresses. Since IPv6 addresses change in time and have limited validity, in case an entry is lost, there is no way to detect the mapping. To ensure that all the information is stored properly, the polling interval has to be less that ND cache expiration timeout. Timeout for Neighbor Cache vary between vendors from one hour to several hours, e.g., Cisco and HP use four hours by default. It depends on requirements and use cases. If only what we want is user accounting we can use slightly less than four hours interval. If we wish to have the mapping earlier, the interval must be shorter. We query devices on our network every 15 minutes.

[8] https://nav.uninett.no/

[9] https://metacpan.org/pod/App::Netdisco

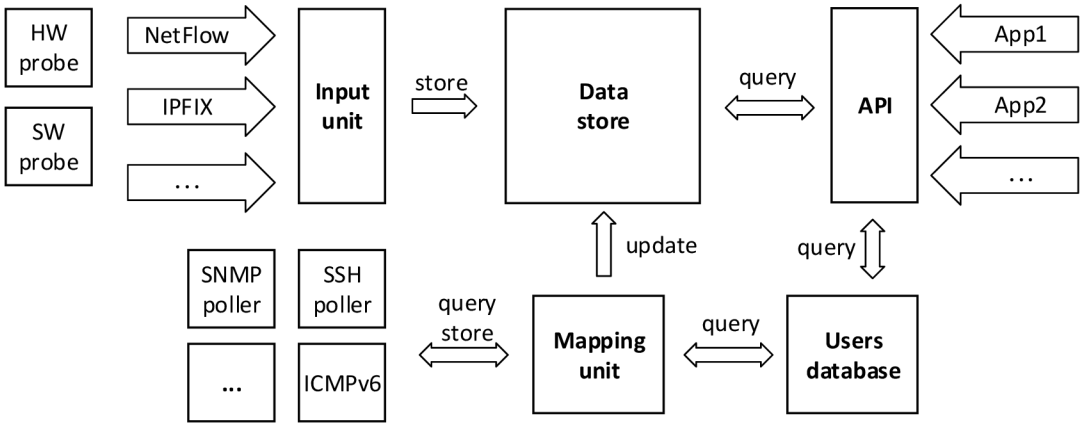


Figure 3.13: Scheme of the accounting system used at BUT network

Putting all the information together

Figure 3.13 shows an abstract view of the accounting system that is used at our university. The central point of the system is data store. We use a binary data format created by nfdump toolkit[10]. The reason is that the format is more efficient compared to traditional relational databases [102]. Although there are several limitations with the binary format, e.g., a lack of writing capabilities or missing indexes, we will show how it is possible to overcome these problems while maintaining the benefits.

The *Input unit* accepts NetFlow or IPFIX data from several sources and stores them to the data store. Any NetFlow exporter can be used. We used software probes introduced in the previous section to obtain visibility inside IPv6 transition tunnels. The SSH or SNMP pollers are used to provide information for the *Mapping unit*. The *Mapping unit* stores all the necessary information about the mapping between L2 and L3 addresses together with temporal information (start and end timestamps). It can use any standard rational database as a backend for storing this information. We recommend PostgreSQL as it has native support for IP address format (both IPv4 and IPv6). The *Mapping unit* communicates with *Users database* which holds UserID and other information about the user (e.g., name, contact, information about payment, etc.). If necessary, the *Mapping unit* can be combined with *Users database* to a single system.

The overall process works as follows: The pollers regularly update the mapping database. The *Mapping unit* uses collected information and extends and regularly updates the NetFlow data in the data store. The result is *Extended NetFlow data* structure – the standard NetFlow data extended with additional information. It is possible to query the data for accounting purposes, backtrack security incidents or create data retention reports. Any field can be used for a query, e.g., an application can ask for statistics about specific IPv4/IPv6 address, MAC address, switchport or all traffic belonging to a user based on UserID.

Figure 3.14 depicts an actual deployment of the accounting system at BUT network. The figure is a simplified view as the BUT network is much more complicated in reality, e.g., we are not presenting the underlying L2 interconnections, and there is also quite complex policy based routing in place. However, it should be clear to see how the system

[10] http://nfdump.sourceforge.net/

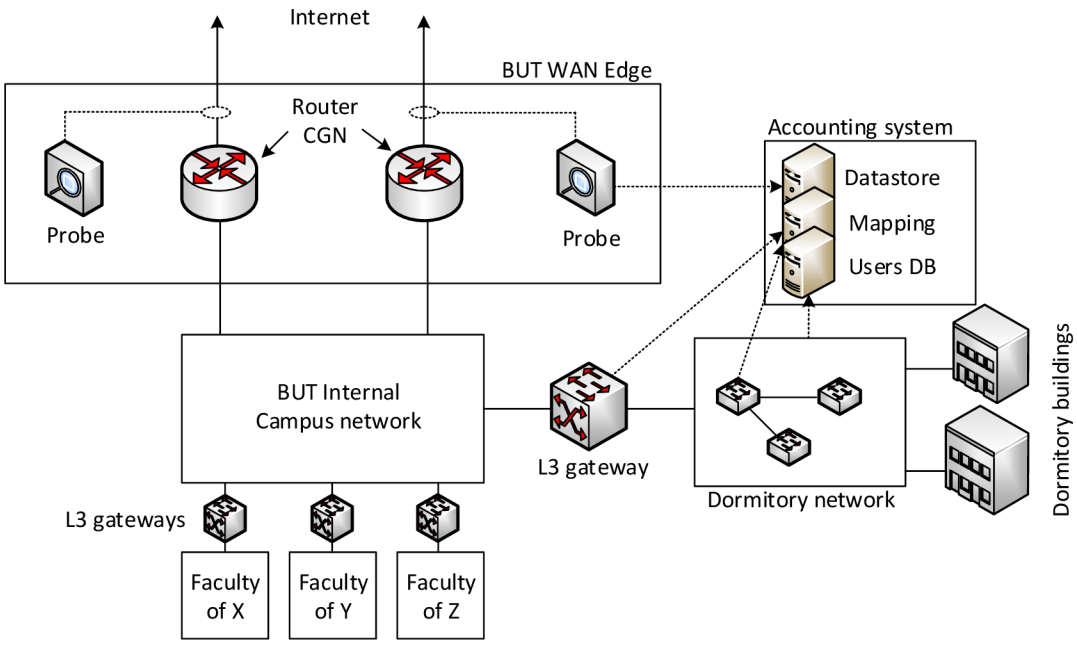


Figure 3.14: Real deployment of the accounting system used at BUT network

is deployed. We can divide the BUT campus design to different modules. BUT has several faculty buildings in different parts of Brno. The BUT Internal Campus network, thus, spans through city of Brno and interconnects all faculties and research buildings. Each faculty is connected via L3 gateway. There are two primary routers providing connectivity to the Internet using 40 Gb/s uplinks. These uplinks are monitored using NetFlow probes.

To obtain all the information, we monitor each L3 gateway for IP address to MAC address mapping. This mapping information is stored in the mapping database. NetFlow data is stored in the form of binary nfdump format on the central collector. If there is a necessity to monitor users' switchports, we collect the information using SNMP/SSH poller or from DHCP logs. Note that DHCP logs can only be used for IPv4 as there are currently problems with working implementation of RFC 6939 for IPv6 [95]. Although implementation exists, it is available only for selected platforms.

There are scripts that go through the data regularly and update or extend the data as necessary (MAC fields, UserID fields, switchport ID, etc.).

How it is possible to update the binary nfdump format with additional information? We use a special library for this purpose – `libnf`[11]. Basically, the library is a shim layer above nfdump sources. It provides necessary abstraction and allows to access each field of nfdump binary record. The benefit of this approach is that it is possible to develop an application that uses the library API. If the underlying data format of nfdump binary structure is changed, the application does not have to be rewritten. It is also possible to add different file format. The `libnf` library provides API for C language. It is also possible to use Perl API[12]. Using the library, the *Mapping unit* is able to go through the NetFlow records stored on a disc and updates all necessary information.

[11] http://libnf.net

[12] http://search.cpan.org/ tpoder/Net-NfDump/lib/Net/NfDump.pm

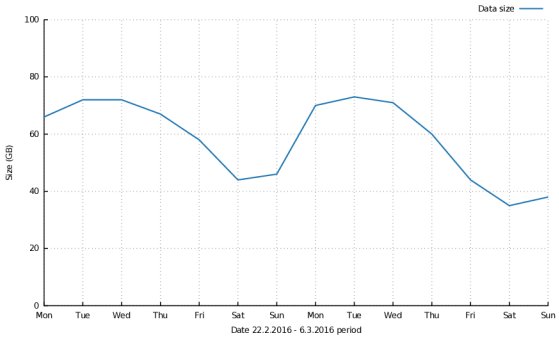


Figure 3.15: Size of NetFlow data in 14 days period

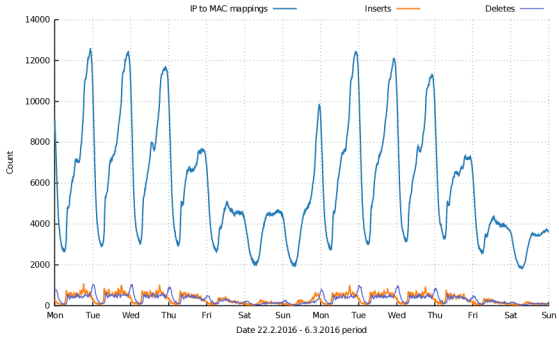


Figure 3.16: Number of open mappings, inserts and deletes during 14 days period

Another question from an observant reader can be: How the UserID is stored? The NetFlow version 9 standard and file format of nfdump do not contain any UserID field. It is true, however, there are several fields available that can be used for the purpose – e.g., `username` field from NSEL[13] extension. The `libnf` library supports writing to the field, thus, we can accept standard NetFlow data and extend it later on the collector. The ideal solution would be to use an IPFIX collector and define an own enterprise ID. Unfortunately, there is a lack of good open source IPFIX collectors. There is a promising `IPFIXcol`[14] [103], thus, the situation can change in future.

System statistics and performance

The amount of data necessary to process is higher than storing only simple NetFlow records. We show, however, that the overhead is not that significant and the accounting system can still run on a commodity hardware. What is the amount of information necessary to store?

Figure 3.15 displays the amount of NetFlow data stored each day. We see that the amount of data peaks around 70GB/day during a week and drops to 40GB/day during weekends. Note that the probes collect traffic twice – before entering to a NAT router and after leaving the NAT router. This is necessary for NAT accounting, and we will show in the next chapter why it is necessary and how the amount of data can be decreased. Figure 3.16 shows a number of open mappings in a temporal database and number of inserts and deletes. We can see that there are around 12 000 active mapping (both IPv4 and IPv6) during the evening. The numbers of inserts and deletes are rather stable – around 1000 during a day, less during night. Note that after the update of a central data store is triggered, the database can be cleared, thus, the size of the database that holds mapping is not important. It is not necessary to keep the data in the database as the mapping information are stored in extended NetFlow data. The overhead depends on the UserID size, e.g., if we choose a 64-bit integer to represent a UserID, the overhead is 4 or 8[15] bytes per flow.

[13] NetFlow Secure Event Logging – http://www.cisco.com/c/en/us/td/docs/security/asa/special/netflow/guide/asa_netflo
[14] https://github.com/CESNET/ipfixcol
[15] the size of 64 bit integer is implementation defined

3.6 Summary

Sections 3.1 describes the process of address assignment and user accounting in today's IPv4 networks. We discussed address assignment techniques for IPv6 protocol in section 3.2. Section 3.4 describes challenges that must be addressed if we want to have a robust accounting system both for IPv4 and IPv6 – mainly problems with different address assignment techniques, temporary addresses and automatic IPv6 tunnels. These issues were identified over several years of our experience with IPv6 deployment.

We introduced a system that is able to account users even in dual-stacked networks or network with a transition technology deployed. We described hardware and software probes that are able to detect IPv6 transition techniques and account the traffic inside the tunnel. The hardware probe has better performance, the software probe is cheaper and more flexible. Both can be used to monitor 10 Gbps links.

Data from these probes are collected on a central data store and extended with additional information. This information is gathered from various sources, e.g., L3 gateways, server logs or by passive monitoring techniques. We discussed benefits and drawbacks of these solutions as well as the performance of the solution and amount of data necessary to store.

This chapter presented several contributions. Firstly, we believe that detailed description of challenges for users accounting process is important as we are not aware of a publication that covers all these issues together. Secondly, there is a lack of information about the behavior of operating systems in a large network. We covered this topic and present several observations how operating systems behave, how many IPv6 addresses can the network administrator expect, etc. Thirdly, we introduced specialized probes that are able to cope with different transition techniques. The standard NetFlow probes or routers do not provide such functionality. Fourthly, we presented an accounting system that can process and store all this necessary information. The system is application-aware, meaning there is an API that can be used for future applications. The system is built using open source technologies that were extended to provide necessary functionality. It means that it is freely available to everyone and can be deployed cheaply. Furthermore, it is proven that the system can run in rather large and complex network as BUT campus. The system was used several times in practice to track IPv6 security incidents and malevolent users.

There were several papers presenting parts of the system – [96], [8], [104], [105] and [106]. This chapter summarizes these results and presents them in a compact form.

4

Address translation and user accounting

"Any problem in computer science can be solved with another level of indirection."
– David John Wheeler

We discuss several different statistics and measurements of migration from IPv4 to IPv6 protocol in chapter 2. The conclusion of the chapter is that support for IPv6 protocol grows slowly but steady and there are more and more clients connected over IPv6 and requesting IPv6 content. In chapter 3 we solved problems with user accounting in networks where dual-stacked or other transition technology is deployed. However, the IPv4 protocol grows as well and data in chapter 2 show that the speed of the IPv4 growth is higher than the growth of IPv6 (e.g. there are more new IPv4 ASN and IPv4 prefixes in BGP DFZ).

How is it possible that IPv4 Internet still grows if RIRs' IPv4 pools are depleted? The main reason why IPv4 Internet is still able to grow, ignoring the number of available IPv4 addresses, is the Network Address Translation (NAT) technology. NAT allows effective address sharing of one IPv4 address between several clients, thus, it is possible to connect much higher number of devices than the number of available IPv4 addresses. Even though the network translation is currently almost ubiquitous, e.g., 90% of residential customers from a major European ISP use NAT gateways to connect to the Internet [110], residential user accounting has not been an issue for ISPs as they assigned public IPv4 addresses to their customers. Unfortunately, the situation is now changing. ISPs are growing in the number of users despite the fact that public IPv4 addresses are scarce resources. ISPs solve the problem by adding another level of indirection (another level of NAT). The drawback of the solution is that user accounting is much harder as IPv4 address in a data retention request does not belong to a user, but to ISP's NAT box.

4.1 NAT, NAPT and CGN

The original proposal of NAT as described in the RFC 1631 [111] rewrites IP addresses of a inner network to the IP addresses of an outer network. The behavior is depicted in Figure 4.1. The mapping in the binding table is created either statically by network administrator or dynamically by NAT. The consequence of the IP address rewriting is that the inner network address block can be reused in a different part of a network. This NAT behavior is referred as Traditional NAT or Basic NAT [116].

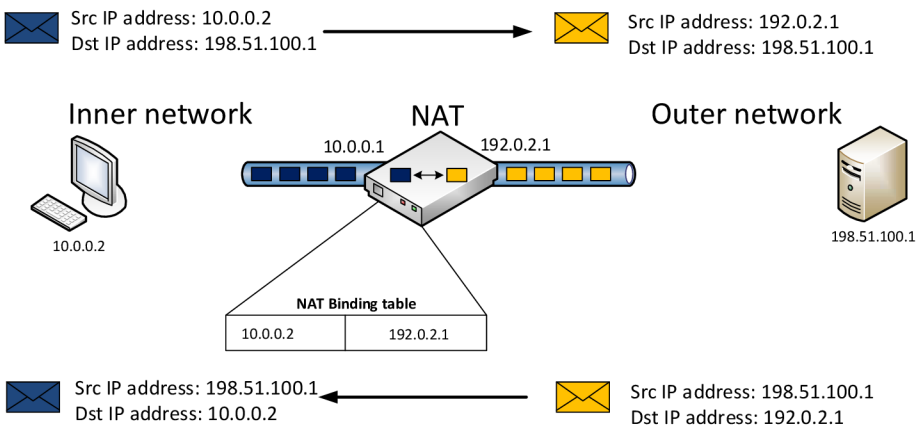


Figure 4.1: Operation of basic NAT – IP address of the device in inner network is rewritten to the IP address of outer network interface

The much more deployed mechanism today is NAPT - Network Address Port Translation, which is the Basic NAT extended with the identifiers from the transport protocol (UDP/TCP ports). Although NAPT is more precise abbreviation for the address and port translation, we will use NAT in the rest of this chapter. If we refer to address translation without port translation, we will use the Basic NAT term. Using IP addresses together with ports allow NAT device to serve more hosts per one IP address, because hosts are not identified only by their IP address, but by a tuple. It depends on NAT implementation, how the tuple is created. Some NAT implementations use only combination of IP address and port, other create a tuple with a protocol type, source and destination addresses, ports and binding time. The main difference between these approaches is the number of users that can be „squeezed“ per one IPv4 address. If only IP address and port is used, the number of clients per IPv4 address must be limited as there are only 65535 available ports (less in practice as ports < 1024 are usually not used). Modern NAT implementations use 5-tuple as shown in Figure 4.2. This approach allows to put a very large number of users per one IPv4 address as depleting ports is usually not a problem in this case – there could be 65535 simultaneous connection to the same port on the same destination.

Carrier Grade NAT, also called Large Scale NAT or NAT444, is a IPv4 address preserve technique used by ISP's to serve IPv4 connection to the clients. This technique is usually combined with dual stack, NAT64 + DNS64 or other IPv6 transition technique to provide IPv6 access. The CGN is basically NAPT in the ISP network, thus users' CPEs do not get public IPv4 as it still common today, but rather a private IPv4 address from RFC 1918 space [117] or an IPv4 address from the reserved prefix for shared address space (`100.64.0.0/10`) [118]. This cascading of NAPTs allows ISPs to serve more IPv4 clients with their current IPv4 address allocations and the whole translation process is illustrated in Figure 4.3.

4.1.1 NAT binding behavior

The creating of NAT binding is usually the same across all implementations – the binding is created by incoming interior SYN or UDP packet. The accessing the binding from an outer network and releasing the NAT binding from the binding table is, however, different among implementations. The reason is that the behavior of NAPT is not standardized.

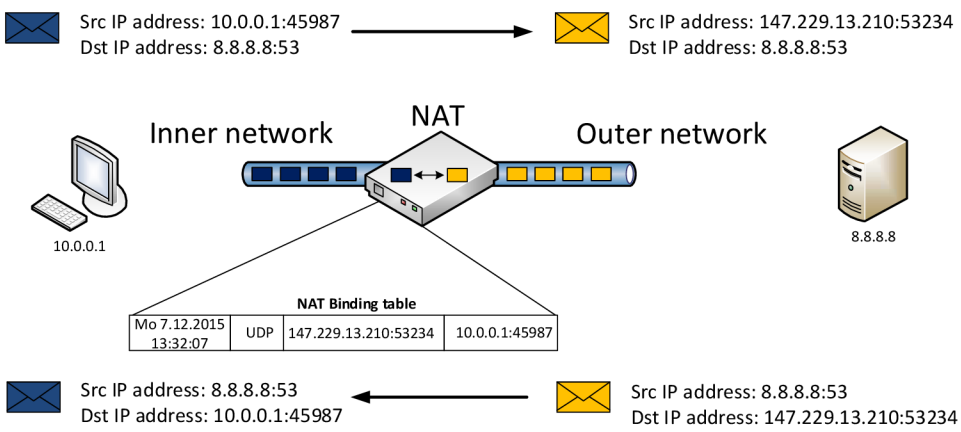


Figure 4.2: Operation of NAPT – combination of IP addresses and ports are used to create the binding between inner and outer network

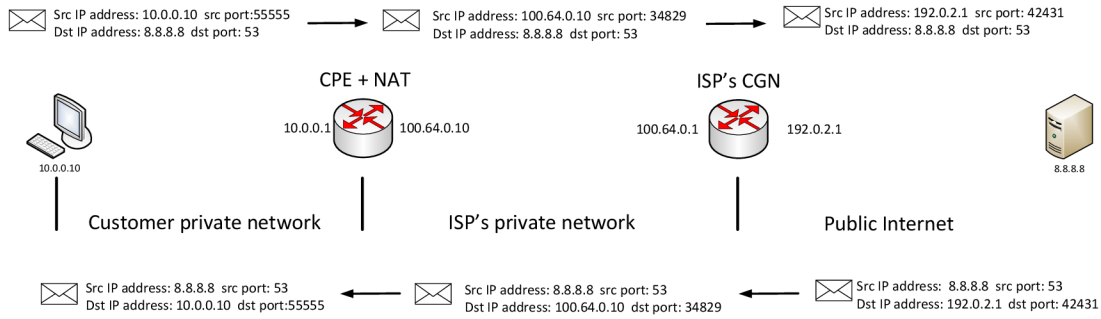


Figure 4.3: Operation of CGN and NAPT

The IETF chose to not standardize NAPT implementation and its operations as IPv6 was perceived as more appropriate and long term solution. The work on the IPv6 protocol, however, took much more time then expected.

The different implementation of NAT has led to an attempt to categorize NAT behaviors. The effort was focused mainly to distinguish different ways of accessing NAT bindings from outside networks and different approaches that NAT use to release the bindings. The terms symmetric NAT, full cone, restricted cone or port-restricted cone appeared first in RFC 3489 [119]. These terms were further studied in RFC 4787 [120], RFC 5382 [121] and RFC 7858 [122]. These different categories can be summarized as follows:

- **Symmetric NAT:** The binding tables stores a local address to a public address mapping and ties the mapping with the destination address for the whole lifetime of the binding. This behavior is usually default for TCP protocol.
- **Full cone NAT:** Any exterior IP address and any exterior port can initiate a connection to an internal host if there is a previous mapping for internal IP address and port in NAT binding table. A full-cone NAT behavior is desired especially for UDP VoIP application.
- **Restricted cone NAT** and **Port Restricted cone NAT:** These types of NAT

add additional restriction to the behavior of Full cone NAT. Device behind restricted cone NAT is accessible by the destination host only after the binding is created. The destination can use any ports as the NAT binding is restricted only to destination's IP address. Port restricted cone NAT acts as Restricted cone NAT but applies restriction also to ports. Restricted cone NAT accepts connections only from the IP address and port it sent the outbound request to.

Additional categories based on different binding behavior was published in [123]. The paper proposed the following terminology:

- **Port iteration:** Every new session receives a new port number. Port numbers start typically from 1024 (lower ports are reserved) and are incremented by one. This behavior is not described in the paper, but it is quite common, thus we extended the terminology.
- **Port preservation:** NAT attempts to preserve the local port number, if possible. It means that if there is a connection from internal IP address `10.0.0.1` and internal port `54321`, the NAT will try to use the same port for external mapping. If there are two connection with the same port, NAT preserves port for one connection and use a different port for the second connection.
- **Port overloading:** NAT uses port preservation at all times. The consequence is that a different host, establishing a new connection with a port that is already been used in binding, will usurp the existing binding.
- **Port multiplexing:** NAT attempts to preserve the port number as in the port preservation example. The difference is the whole five tuple is used which increase the chance that the source port will be preserved during the translation.

Different approaches are also used for releasing the binding from NAT binding table. A local host may have to use some form of keep-alive operation to maintain a NAT binding open. This is especially true for UDP traffic where UDP based protocols, such as for audio and video streaming, routinely send UDP keep-alive packets roughly every 15 seconds [124]. If the transport protocol is stateful, such as TCP, the binding is based on a transport session. The following table summarizes the different approaches of creating, releasing and accessing bindings.

Table 4.1: Design parameters of NAT - bindings [125]

	TCP	**UDP**
creating NAT binding	interior SYN packet	interior UDP packet
accessing NAT binding	symmetric	symmetric full cone restricted cone port-restricted cone
release NAT binding	timer interior RST or FIN exterior RST or FIN	timer

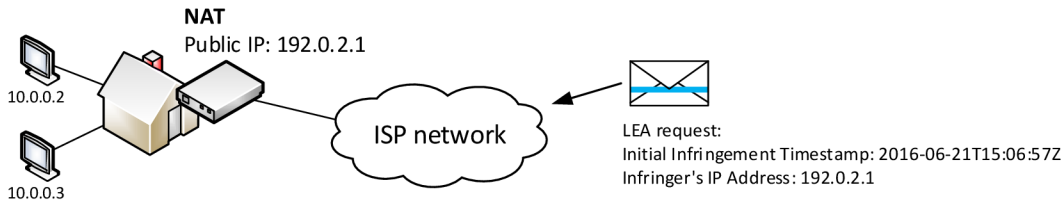


Figure 4.4: Lawful Enforcement Agency requests data originally generated by IP address `10.0.0.1`, but the only piece of information available from outside point of view is public IPv4 address of user's NAT.

4.2 Problem statement – NAT accounting

In the previous chapter, section 3.1 describes necessary information for accounting in IPv4 networks. The section discuss the reasons why providers store different mappings and information about assigned IP addresses. The following points just briefly repeat information presented in 3.1:

- **IP address:** The IP address is stored as it is used as an identifier from the data retention point of view [48] as well as from the ISP management point of view (information necessary for billing, etc.). ISP typically stores information about a contract with a customer, together with assigned IP address and other information in relational database.
- **Timestamp to IP address mapping:** Mapping between timestamp and IP address is stored as well if ISP provides dynamic address assignment. Typically logs from DHCP server (if address is assigned using DHCP protocol) or BRAS server (in case PPP protocol is used) are stored.
- **Network layer to data link layer mapping:** Mapping between network and data link layer is necessary as well if dynamic address assignment is used. It depends on a design of an ISP network – it can be mapping between IP address and MAC address of customer's CPE, logs from RADIUS server or session-id or a username in case of PPP, PPPoe or PPPoA protocols.

All these pieces of information are, however, insufficient if some form of address translation is deployed in the network – see Figure 4.4.

If an ISP receives a data retention request, the ISP is able to pair IPv4 address (`192.0.2.1`) in the request with the user that signed a contract with the ISP. It does not matter too much in practice that there are several computers/users connected in end network as the customer who signed the contract with the ISP is seen as responsible for the whole traffic. From ISP point of view, it is LEA's responsibility to correctly identify a real device or user from captured networking metadata that ISP provides.

If there is a CGN deployed in provider's network, the IP address in data retention requests is, however, maintained by the ISP as it is the address of CGN public interface – see Figure 4.5. ISP, however, cannot share all data generated by the CGN public interface as there are several customers behind CGN and sharing the date would break their privacy.

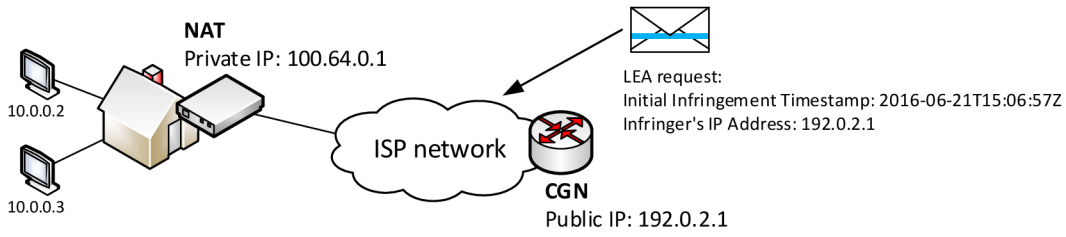


Figure 4.5: Lawful Enforcement Agency requests data originally generated by IP address `10.0.0.1`, but the only piece of information available from outside point of view is public IPv4 address of providers CGN.

Thus, there must be some kind of logging option present in ISP network to allow pairing between public address of CGN with private address assigned to customer.

4.2.1 NAT logging

There are several options how to enable logging in networks where NAT or CGN are deployed. Typically, these options requires support on NAT device itself. There is also a need for a protocol to transport logs to a central logging server. Two protocols are used the most - Syslog and NetFlow.

Syslog

NAT device can support logging via Syslog protocol. Every NAT translation created on the NAT device is then logged and sent to Syslog server. The format is not standardized, thus, every vendor can export different information. To solve this issue, there is a standardization effort in BEHAVE workgroup [126]. The following example Syslog export from Cisco IOS router where NAT translation is enabled.

```
13:16:10.543: %IPNAT-6-NAT_CREATED: Created tcp 10.0.0.1:23800 100.64.0.1:1024 192.0.2.1:80 192.0.2.1:80
13:17:52.251: %IPNAT-6-NAT_DELETED: Deleted tcp 10.0.0.1:23800 100.64.0.1:1024 192.0.2.1:80 192.0.2.1:80
```

Using Syslog for logging can have several issues. The following list is description of the most problematic issues:

- Syslog uses non-structured text-based protocol – the message can be parsed by regular expression but if the format of the message is changed, the parser must be changed as well.
- Minimum of two events (creating and deleting of the session) are logged for every session. It means that ISP must retain large amount of data. It can lead to overwhelming of Syslog server.
- Implementation of Syslog login uses CPU of the NAT device. If there are many sessions, there could be an operational impact of the NAT as the CPU is overloaded by generating Syslog messages.

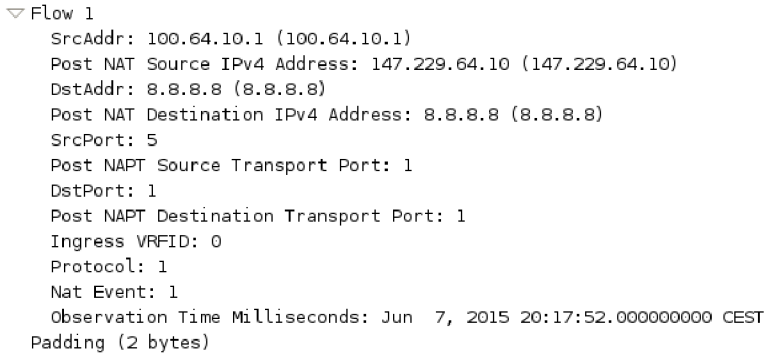


Figure 4.6: Export NEL logging information from Cisco CSR router.

NetFlow extension – NEL, NSEL and bulk port allocation

Similarly to Syslog, BEHAVE workgroup in IETF tries to standardize IPFIX template, but it is still a work in progress [127]. There are, however, several vendor proprietary extensions that are used in practice for NAT logging. Cisco NEL (NetFlow Event Logging, sometimes called HSL – High-Speed Logging) or NSEL (NetFlow Security Event Logging) can be example of such extensions. The following Figure 4.6 shows information exported using Cisco NEL from Cisco CSR Virtual router.

The following example of NetFlow records shows both NEL data (first two rows) and classic NetFlow data (rows three and four) as seen on a collector.

Date first seen	Event	Proto	Src IP Addr:Port		Dst IP Addr:Port	X-Src IP Addr:Port		X-Dst IP Addr:Port	Bytes
18:17:52.398	CREATE	ICMP	10.10.10.1:5	->	8.8.8.8:0.1	192.168.1.10:1	->	8.8.8.8:1	0
18:18:56.343	DELETE	ICMP	10.10.10.1:5	->	8.8.8.8:0.1	192.168.1.10:1	->	8.8.8.8:1	0
18:17:52.398	INVALID	ICMP	192.168.1.10:0	->	8.8.8.8:8.0	0.0.0.0:0	->	0.0.0.0:0	500
18:17:52.765	INVALID	ICMP	8.8.8.8:0	->	192.168.1.10:0.0	0.0.0.0:0	->	0.0.0.0:0	460

The previous examples of logging NAT events using NetFlow protocol can be used both for a classical NAT or CGN. Some CGN implementations offer another NetFlow extension – a bulk port allocation and bulk logging. These features allocates a block of ports for translation instead of allocating individual ports. The bulk port allocation works as follows:

1. Client initiates a connection through a CGN. The CGN device allocates multiple global ports of a single global IP address instead of a single global IP address and global port. The default number is 512 global ports.

2. CGN exports information about the allocated block using NetFlow protocol.

3. A new connection initiated from the same client use a free port from the previous bulk allocation. CGN does not log any information about the connection.

4. Based on the volume of translations, additional blocks of ports can be allocated.

The whole process is illustrated in Figure 4.7. A similar approach is described in RFC 7422 [128]. The difference is that RFC uses an approach where ports are presets in CGN – thus CGN does not export any logging information.

```
▽ FlowSet 1
    FlowSet Id: (Data) (263)
    FlowSet Length: 32
  ▽ Flow 1
      SrcAddr: 100.64.10.4 (100.64.10.4)
      Post NAT Source IPv4 Address: 147.229.64.100 (147.229.64.100)
      Ingress VRFID: 0
      Protocol: 6
      Nat Event: 1
      Observation Time Milliseconds: Jun 16, 2015 15:52:43.674000000 CEST
      Port block start: 1024
      Port block step size: 8
      Number of ports in block: 512
```

Figure 4.7: Bulk logging using NetFlow protocol exported from Cisco ASR router.

4.3 A new approach for NAT accounting

The previous section describes several issues that are introduced by CGN or NAT in a network. We described several approaches that can be used for logging all necessary events. However, several unresolved issues remains. Firstly, all approaches for logging of NAT events requires support directly on the NAT device. If the device does not support Syslog, NEL, NSEL or bulk port logging, there is currently no way how to obtain all the necessary information. Secondly, logging features are CPU sensitive. If there is a large number of session, the CPU can be overwhelmed. Thirdly, there is no easy way how to correlate exported data from the NAT with NetFlow data exported by a standalone NetFlow probe. We can configure to export both NAT logging and NetFlow to the same collector, but there is no easy way how to query the collected data. If there is a LEA request or a network administrator simply tries to backtrack a security incident of an IP address, NAT logs must be queried first to obtain information about the address translation. Equipped with the information, the administrator can query NetFlow records to obtain actual connections for the IP address.

We tried to solved these issues and this section presents a new approach for NAT accounting using standalone NetFlow probes. There is no need to change anything in network topology and there is no need for a NAT device to support logging as the logging is done outside of address translation process. The whole idea is based on a correlation traffic before and after the address translation. A NetFlow probe can be inserted in the network topology to monitor inner and outer traffic as depicted in Figure 4.8.

Using a probe that exports NetFlow records before and after the translation, we obtain two flows on the collector – `10.0.0.2 -> 198.51.100.1` exported before the translation and `192.0.2.1 -> 198.51.100.1` exported after the translation. The same is with the returning traffic.

Is it possible to create a monitoring system that is able to combine and correlate the flows? Firstly, let us create some properties that the solution should have:

- Inner and outer flows must be correlate to obtain only two pieces of information – one flow for the traffic going from the inner network to outer network and one flow for the returning traffic.
- Correlated flow information should contain all the necessary information – IP addresses and ports before and after translation, number of packets in the flow and number of transferred bytes.

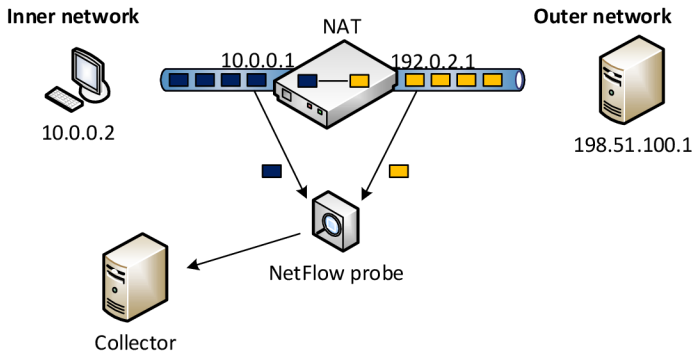


Figure 4.8: NetFlow probe can be inserted in a topology to monitor inner and outer traffic.

- There cannot be any heuristic or only 90% of probability that the correlation is correct as the data could not be used as an evidence during the data retention process in that case. The correlation simply must works in every circumstances and must be 100% correct – no ambiguity is allowed.
- The NetFlow probe should be able to handle traffic at reasonable speeds. The solution should work for 10 Gbps speed on a commodity hardware.
- TCP, UDP, ICMP must be supported. Other protocols should be supported as well.
- The network topology should not be limited only for a one probe monitoring both inner and outer link, but the solution should be flexible and allows more probes for better scaling.

4.3.1 Flows correlation

Before we start to discuss how we can correlate flows before and after translation, we have to take into consideration behavior of NetFlow probes. The first limitation is that the probes are stateless – every packet is analyzed separately. The probe computes a hash from key fields and stores the information to a flow cache. Another packet with the same hash updates the counters (number of packets/bytes) in the flow cache. Furthermore, the flow creation process allocates and fills all necessary data structures in the flow cache. All subsequent packets simply increase the appropriate counters in the flow cache. Thus, if we want to add additional information to a flow, we should do that during the creation of the flow.

How is it possible to correlate inner and outer flows? First approach we tried was based on an assumption that it is possible to use two FIFO queues for flows correlation. We though that first packet captured on the inner interface could be matched with the first packet captured on the outer interface. Unfortunately, this approach can be used only if there is a small amount of traffic or if we have a strict control over Network Interface Card (NIC). The problem is that at higher speeds, a NIC does not use an interrupt to signal the kernel that there is a packet, but pushes several packets to an operating system in one batch. This lowers the amount of interrupts and increases the networking performance, but

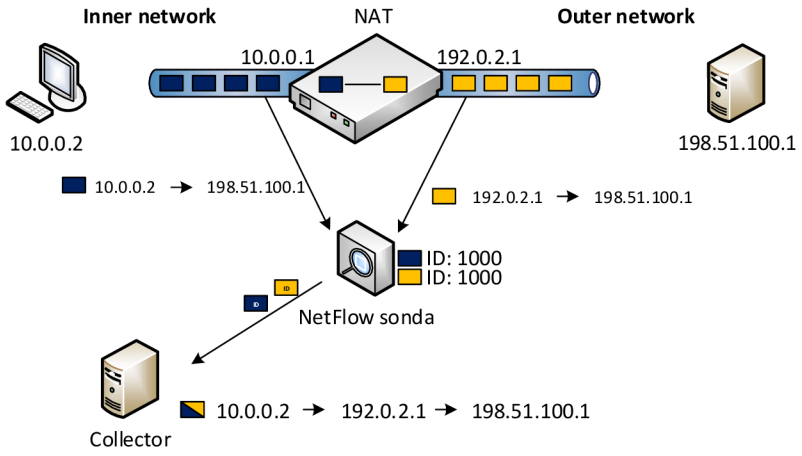


Figure 4.9: Flow pairing using computation of a unique value both for inner and outer traffic

```
Internet Protocol Version 4, Src: 192.168.1.4, Dst: 147.229.9.14
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0x8153 (33107)
    Flags: 0x02 (Don't Fragment)
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0x5ac9 [validation disabled]
    Source: 192.168.1.4
    Destination: 147.229.9.14
Transmission Control Protocol
    Source Port: 37663
    Destination Port: 3128
    Sequence number: 1378765392    (relative sequence number)
    Acknowledgment number: 0
    Header Length: 40 bytes
    Flags: 0x002 (SYN)
    Window size value: 5840
    Checksum: 0xcb9f [validation disabled]
    Urgent pointer: 0
    Options: (20 bytes), Maximum segment size, SACK permitted,
                         Timestamps, No-Operation (NOP), Window scale
```

Figure 4.10: IP and TCP headers before NAT translation

```
Internet Protocol Version 4, Src: 10.10.10.220, Dst: 147.229.9.14
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0x8153 (33107)
    Flags: 0x02 (Don't Fragment)
    Fragment offset: 0
    Time to live: 63
    Protocol: TCP (6)
    Header checksum: 0x0890 [validation disabled]
    Source: 10.10.10.220
    Destination: 147.229.9.14
Transmission Control Protocol
    Source Port: 37663
    Destination Port: 3128
    Sequence number: 1378765392    (relative sequence number)
    Acknowledgment number: 0
    Header Length: 40 bytes
    Flags: 0x002 (SYN)
    Window size value: 5840
    Checksum: 0x7866 [validation disabled]
    Urgent pointer: 0
    Options: (20 bytes), Maximum segment size, SACK permitted,
                         Timestamps, No-Operation (NOP), Window scale
```

Figure 4.11: IP and TCP headers after NAT translation

first packet on inner interface does not have to correspond with the first packet captured on the outer interface.

Thus, another approach must be used. We could extract information that remains the same before and after the translation and use the information to compute a unique identifier. Figure 4.9 shows a high level overview of such approach. The topology and traffic are the same as in previous Figure 4.8. The probe computes a unique value for the flow captured on probe's inner interface (`10.0.0.2 -> 198.51.100.1`). The same process is repeated for the flow captured on probe's outer interface `192.0.2.1 -> 198.51.100.1`.

As the computation uses fields that do not change, we should be able to compute the same unique identifier before and after the translation. Does the approach work in practice? Let us consider TCP protocol first. Figures 4.10 and 4.11 describe IP and TCP headers before and after translation.

Considering the IP header in Figure 4.11, destination IP address together with protocol remain the same, thus these fields can be used to compute the unique identifier. Other fields in the IP header either change during the translation or they are not stable enough

to be used, e.g., TTL field is decremented, and Identification remains the same, but not every operating systems fill the value.

Considering the TCP header in the the same figure, only the checksum field is changed. However, NAT can rewrite the source port so we have to exclude this field as well. As the first packet initiating connection is a SYN packet, we cannot used any application payload as SYN packets does not have any payload[1].

Initially, we used only TCP sequence number as unique identifier as the field is same before and after translation, it is randomly generated and has a reasonable size (32 bits). Using the TCP sequence number has also an advantage that the unique value is not tight with IP addresses, thus the probe does not have to distinguish inner and outer direction. There could be, however, a problem if only TCP sequence number is used – collision. Although the sequence number is randomly generated on most operating systems, there could be a situation that two host generate the same value. More traffic means higher probability of collision. How much higher? It can be computed, as it is the same problem as Birthday problem [129]. The probabilities of collisions for 1000 and 10 000 concurrent TCP session, thus, can be calculated same as in the Birthday problem with the following equation.

$$p(1000) = 1 - \binom{2^{32}}{1000} \times \frac{1000!}{(2^{32})^{1000}} \tag{4.1}$$

$$p(1000) = 0.0001162921... \tag{4.2}$$

$$p(10000) = 1 - \binom{2^{32}}{10000} \times \frac{10000!}{(2^{32})^{10000}} \tag{4.3}$$

$$p(10000) = 0.0115728899... \tag{4.4}$$

For 1000 simultaneous session, the probability is approximately 0.1 ‰. For 10 000 connection, the probability of collision increase to 1%. Large networks can easily reach more than 10 000 flows/s, thus the probability of collision is quite high. As we want to be sure that there is no ambiguity for flows correlation, we extended the number of fields to compute the unique value. Currently, we use combination of destination IP together with TCP sequence number to compute the unique value. Using this approach, there could be 10 000 simultaneous connection to one server. If it is still not enough and there is more than 10 000 simultaneous connection to the same server, another options can be used – IP Identification (if set), or TCP options.

UDP and ICMP traffic is rather easier as these protocols carry application payload. As CGN does not change the payload, we can hash the payload to obtain a unique value that is the same before and after the translation. Similarly to TCP, another fields can be used as well – IP Identification (if set) and destination port number in case of UDP protocol,

4.3.2 Implementation

We implemented the extension for NetFlow probe and tested it in a production environment at BUT network. The architecture of the system is as follows. We used Flowmon probe for exporting NetFlow records extended with ID value. The Flowmon probe allows to use an API for a development of an own plugin. Several functions (hooks) are provided to alter

[1] Although there is an attempt from Google and Apple to change this and use payload even with the first SYN packet.

parsing and processing of packets. We used an approach where all the necessary information is computed when a flow record is created. It means that computation of unique ID value is done only once for a flow. This leads to decreasing CPU load on the probe and increasing performance.

It is, however, not enough sufficient to compute the unique value before and after translation and export the NetFlow record to a collector. Why? The collector typically stores incoming NetFlow records to a temporary file and rotates the file every five minutes. This greatly increase the probability of ID collision as there are many more flows in the five minute file. To overcame the situation, we use an external in-memory cache on Flowmon probe. Each time the incoming packet is seen on probe's inner interface, we compute the unique value from the fields described previously and use the value as a key for the cache. The value for the key is a 64 bit random number. If the packet crosses the NAT, probe sees the packet on the outer interface. The probe computes the same unique value as for inner flow and search the in-memory cache. If the key is found, the NetFlow exporter running on probe's outer interface obtains the random value from the cache, stores it to flow's internal data structure and invalidates the record in the cache. This leads to a situation that NetFlow probe exports two flows – before and after the translation and both flows contains the same unique 64 random number. The 64-bit random value provides enough entropy that collision is very unlikely. An advantage of this approach is also a fact that only traffic that was actually translated by CGN can be tight together on a collector. There is an expiration timeout set to the key the cache that invalidates the key automatically after a certain period of time – so far a second, but the timeout is configurable. The timeout ensures that the cache does not growth endlessly.

Date first seen	Src IP Addr:Port		Dst IP Addr:Port	Bytes	Pkts
2016-06-04 12:25:13.840	8.8.8.8:53	->	192.168.1.4:36220	192	2
2016-06-04 12:25:13.844	10.10.10.220:37663	->	147.229.9.14:3128	487	8
2016-06-04 12:25:13.869	192.168.1.4:37664	->	147.229.9.14:3128	2247	41
2016-06-04 12:25:13.844	147.229.9.14:3128	->	192.168.1.4:37663	781	5
2016-06-04 12:25:13.870	147.229.9.14:3128	->	192.168.1.4:37664	57123	49
2016-06-04 12:25:13.870	10.10.10.220:37664	->	147.229.9.14:3128	2247	41
2016-06-04 12:25:13.870	147.229.9.14:3128	->	10.10.10.220:37664	57123	49
2016-06-04 12:25:13.844	192.168.1.4:37663	->	147.229.9.14:3128	487	8
2016-06-04 12:25:13.844	147.229.9.14:3128	->	10.10.10.220:37663	781	5
2016-06-04 12:25:13.826	10.10.10.220:36220	->	8.8.8.8:53	128	2
2016-06-04 12:25:13.825	192.168.1.4:36220	->	8.8.8.8:53	128	2
2016-06-04 12:25:13.839	8.8.8.8:53	->	10.10.10.220:36220	192	2

Src IP Addr:Port		Dst IP Addr:Port	X-late Src IP:Port		X-lateDst IP:Port	Bytes	Pkts
192.168.1.4:37664	->	147.229.9.14:3128	10.10.10.220:37644	->	147.229.9.143128	2247	41
147.229.9.14:3128	->	10.10.10.220:37664	147.229.9.14:3128	->	192.168.1.4:37664	2247	41
192.168.1.4:37663	->	147.229.9.14:3128	10.10.10.220:37663	->	147.229.9.14:3128	57123	49
147.229.9.14:3128	->	10.10.10.220:37663	147.229.9.14:3128	->	192.168.1.4:37663	487	8
192.168.1.4:36220	->	8.8.8.8:53	10.10.10.220:36220	->	8.8.8.8:53	128	2
8.8.8.8:53	->	10.10.10.220:36220	8.8.8.8:53	->	192.168.1.4:36220	128	2

Figure 4.12: Example of merging inner and outer flows to one flow that contains all the necessary information

Collector processes NetFlow data and extract 64-bit random numbers. These numbers are compared with each other and if there is a match, flows are merged together. One of the biggest advantages is that there is only one record that describes the whole translation process and also includes statistics such as number of packets and bytes. Figure 4.12 shows an example of NetFlow data before and after the merge on a collector.

4.4 Summary

This chapter describes Network address translation process and issues that the mechanism presents for user accounting. Several possibilities for NAT logging were discussed – Syslog, NEL, NSEL and bulk port allocation and bulk logging. These methods can be used, but have some disadvantages. NAT device must support the logging mechanism, CPU processor on a NAT device must be powerful enough to handle logging even if there are a lot of sessions and these logging methods must be combined with traditional NetFlow data to obtain the whole picture about user's activities.

We introduced a novel approach for NAT logging that eliminates these issues. Logging is performed from an external, standalone NetFlow probe that intercepts traffic before and after the translation. The benefit of this approach is that NAT does not have to support logging which save NAT's CPU. Furthermore, NetFlow probe extends NetFlow records with an unique identifier that is used on a collector for correlation and merging flows. There can be only one record with all the necessary information, thus, we can save large amount of disk space.

The approach for NAT logging were discussed in our paper – [8]. Implementation of the proposed solution was supported by CESNET grant 546R1/2014. Feasibility of the approach was discussed in GÉANT Best Practice workgroup. Currently, the implementation runs in production at BUT dormitory campus network.

5

Conclusion

The thesis deals with the issues of user monitoring and accounting especially in next generation networks. We focused primary on IPv6 protocol as other proposals of new networking protocols or architectures are still in a development stage and not widely deployed. The thesis is devided into three main chapters. The chapter 2 presents analysis and statistics about the transition from IPv4 to IPv6 and provides the necessary knowledge about the IPv6 transition progress for the rest of the thesis. Chapter 3 describes challenges for user accounting presented by dual stack networks and networks where some kind of transition mechanism between IPv4 and IPv6 is deployed. The chapter introduces a central system that is able to account user even in these networks. Chapter 4 extends the acounting system with a support for network address translation mechanism. All these chapters together solve issues with user accounting created by the transition between two incompatbile networking protocols. Recursive InterNetwork Architecture (RINA), Content Centric Networking (CCN) or Named Data Networking (NDN) are other examples of new networking architectures that are currently proposed. All these new architectures are, however, incompatible with IPv4/IPv6. Hence, we can use the same approach presented in this thesis for the future transition as well. The following parts of this chapter summarize the main observations and contributions presented by the thesis.

Analysis of the IPv4-IPv6 transition

The thesis closely describes the transition to IPv6 in chapter 2. The chapter is divided to three main section where each of the sections analyses the transition process from a different angle.

The first section of the chapter analyses routing infrastructure. The global BGP table is examined for current trends in IPv6 adoption. Several interesting facts emerged. The support for IPv6 is slightly lower among new companies (32-bit ASNs). The growth of transit only ASes supporting IPv6 drops significantly in 2014 – 2015 period. Slower growth of IPv6 support in the routing core means smaller path diversity and robustness. We did a correlation of BGP analysis with NetFlow data from BUT and CESNET networks. It is a novel approach, as BGP analysis is usually presented without any relationship with the real network traffic. We found out that the the ratio between the number of unique `/64` prefixes in one `/48` prefix is around 1.5 – 2. The ratio should be much higher in developed IPv6 networks – we see the ratio around 10 for developed networks (Facebook, Google, O2 Czech, etc.).

We present several figures showing the IPv6 support among the web, mail and DNS

services since 2012. We compared our results with other projects measuring the IPv6 adoption and found out, that these projects overestimate the IPv6 content penetration. The main reason is that other projects use much smaller dataset compare to ours. The section also discuss a quality of IPv6 connection to dual stack websites. The conclusion is that IPv4 and IPv6 perform similarly in most cases. We also found out that occurrences where one protocol performs better than other are mainly caused by differences in routing paths. Our measurements show that there is a substantial number of sites (around 5 %) we are not able to connect to.

The third section describes support for IPv6 protocol among users' devices and IPv6 traffic volume. The section presents long-term statistics of user IPv6 support in BUT campus network and finds out that IPv6 support is very high – around 80 % of devices (laptops, PCs and mobiles) actively use IPv6. Furthemore, the section discusses IPv6 traffic volume and flow ratio. The traffic volume oscillates around 20 % and flow ratio around 10 %. There is no big difference between traffic volume in 2016 and three years ago. This is caused by the fact, that main content providers enabled IPv6 in 2012. There was not any bigger movement in IPv6 support for small websites that comprises the rest of the network traffic in recent years.

The main contributions of chapter 2 are the presented observations and statistics about the IPv4-IPv6 transition. All these analysis and statistics use large and unique datasets. Content analysis uses our own dataset much larger then others. All the statistics are publicly available online and are regularly updated.

User accounting and transition technologies

Chapter 3 describes the process of address assignment and user accounting in today's IPv4 networks. Address assignment techniques for the IPv6 protocol are described as well, together with transition techniques that are used in today's network to overcome incompatibility between IPv4 and IPv6 protocols. The chapter discusses issues and challenges that must be addressed if we want to have a robust accounting system both for IPv4 and IPv6. It introduces a central accounting system that is able to account users even in dual-stacked networks or network with a transition technology deployed. The system uses hardware or software probes that are able to detect IPv6 transition techniques and account the traffic inside the tunnel. Data from these probes are collected on a central data store and extended with additional information. This information is gathered from various sources, e.g., L3 gateways, server logs or by passive monitoring techniques.

The main contributions of chapter 3 are the following. *i* Detailed description of challenges for users accounting process. *ii* Observations and practical experience with behavior of operating systems in IPv6 networks, how many IPv6 addresses can the network administrator expect, etc. *iii* Specialized probes that are able to cope with different transition techniques. The standard NetFlow probes or routers do not provide such functionality. *iv* An accounting system that can process and store all the necessary information. The system is application-aware, meaning there is an API that can be used for future applications. The system is built using open source technologies and it is freely available to everyone. Furthermore, it is proven that the system can run in rather large and complex network as BUT campus. The system was used several times in practice to track IPv6 security incidents and malevolent users.

User accounting and address translation technologies

Chapter 4 describes the network address translation process and issues that the mechanism introduces for user accounting. The chapter focuses on different types of NAT logging that could be used for extending the central accounting system presented in Chapter 3, e.g., Syslog, NEL, NSEL, bulk port allocation and bulk logging. A novel approach for NAT logging that eliminates issues presented by address translation is introduced. The NAT logging process is performed from an external, standalone NetFlow probe that intercepts traffic before and after the translation. The benefit of this approach is that NAT does not have to support logging which save NAT's CPU. Furthermore, NetFlow probe extends NetFlow records with an unique identifier that is used on a collector for correlation and merging flows.

The main contributions of chapter 4 are the following. *i* In-depth description of different address translation approaches. *ii* A novel aproach for NAT logging. The approach saves NAT's CPU and allows NAT accounting even if the NAT box does not have support for it. Furthemore, it is possible to obtain a complete view on the translation process in one flow. Hence, it is possible to easily trace back security incidents and create statistics for user accounting.

5.1 Future work

The thesis solves several problems of user accounting in IPv6 networks and networks with address translation. Furthemore, it presented several statistics and different views on the IPv6 transition progress. However, the job is not yet done. In the nearest future, we would like to solve a few remaining issues with the measurement of IPv6 transition. Mainly, NetFlow data correlation with BGP in cases where an ISP uses `/48` prefix per customer. We plan to run more tests for NAT logging approach introduced in chapter 4. Even though the approach works, we would like to test it with more users and with different NetFlow probes.

Bibliography

[1] D. C. Mowery and T. Simcoe, "Is the Internet a US invention? — an economic and technological history of computer networking," Research Policy, vol. 31, no. 8–9, 2002, pp. 1369 – 1387. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0048733302000690

[2] J. Postel, "NCP/TCP transition plan," RFC 801, Internet Engineering Task Force, Nov. 1981. [Online]. Available: http://www.ietf.org/rfc/rfc801.txt

[3] A. Malis, "ARPANET 1822L Host Access Protocol," RFC 851, Internet Engineering Task Force, Apr. 1983, obsoleted by RFC 878. [Online]. Available: http://www.ietf.org/rfc/rfc851.txt

[4] G. Huston, "Is the Transition to IPv6 a „Market Failure?"," September 2009, [cited 19.4.2013]. [Online]. Available: http://www.potaroo.net/ispcol/2009-09/v6trans.html

[5] I. Livadariu, A. Elmokashfi, A. Dhamdhere, and K. Claffy, "A first look at IPv4 transfer markets," in Proceedings of the ninth ACM conference on Emerging networking experiments and technologies. ACM, 2013, pp. 7–12.

[6] J. Curran, "An Internet Transition Plan," RFC 5211 (Informational), Internet Engineering Task Force, Jul. 2008. [Online]. Available: http://www.ietf.org/rfc/rfc5211.txt

[7] M. Grégr, T. Podermański, and M. Švéda, "Measuring Quality and Penetration of IPv6 Services," ICNS 2014, 2014.

[8] M. Grégr and M. Švéda, "Challenges with Transition and User Accounting in Next Generation Networks," in ICNP 2014. Raleigh, NC, USA, US: Institute of Electrical and Electronics Engineers, 2014.

[9] G. Huston, "Testing IPv6 for World IPv6 Day," May 2011, [cited 19.4.2013]. [Online]. Available: http://www.potaroo.net/ispcol/2011-05/ip6test.html

[10] L. Colitti, S. Gunderson, E. Kline, and T. Refice, "Evaluating IPv6 Adoption in the Internet," in Passive and Active Measurement. Springer Berlin / Heidelberg, 2010.

[11] G. Huston, "Measuring IPv6 - Country by Country," July 2012, [cited 19.4.2013]. [Online]. Available: http://www.potaroo.net/ispcol/2012-07/v6report.html

[12] M. Karir, G. Huston, G. Michaelson, and M. Bailey, "Understanding ipv6 populations in the wild," in Passive and Active Measurement, ser. Lecture Notes in

Computer Science. Springer Berlin Heidelberg, 2013, vol. 7799, pp. 256–259. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-36516-4˙27

[13] E. Karpilovski, A. Gerber, D. Pei, J. Rexford, and A. Shaikh, "Quantifying the Extent of IPv6 Deployment," in Network Measurement. Springer Berlin, 2009.

[14] G. Huston, "BGP Reports," [cited 12.11.2014]. [Online]. Available: http://bgp.potaroo.net

[15] X. Zhou and P. Van Mieghem, "Hopcount and E2E Delay: IPv6 Versus IPv4," in Passive and Active Measurement. Springer Berlin / Heidelberg, 2005.

[16] X. Zhou, M. Jacobsson, H. Uijterwaal, and P. Van Mieghem, "IPv6 delay and loss performance evolution," International Journal of Communication Systems, vol. 21, no. 6, 2008, pp. 643–663.

[17] A. Berger, "Comparison of Performance over IPv6 versus IPv4," Akamai Technologies, 2011.

[18] A. Dhamdhere, M. Luckie, B. Huffaker, A. Elmokashfi, E. Aben et al., "Measuring the deployment of ipv6: topology, routing and performance," in Proceedings of the 2012 ACM conference on Internet measurement conference. ACM, 2012, pp. 537–550.

[19] M. Nikkhah, R. Guérin, Y. Lee, and R. Woundy, "Assessing IPv6 through web access a measurement study and its findings," in Proceedings of the Seventh COnference on emerging Networking EXperiments and Technologies. ACM, 2011, p. 26.

[20] J. Czyz, M. Allman, J. Zhang, S. Iekel-johnson, E. Osterweil, and M. Bailey, "Measuring IPv6 Adoption," in SIGCOMM 2014, Chicago, Illinois, USA, 2014, pp. 87–98.

[21] O. M. Crépin-Leblond, "IPv6 Matrix Project," [cited 22.12.2013]. [Online]. Available: http://www.ipv6matrix.org

[22] IPv6 Observatory, "Top-500 websites with AAAA records," [cited 22.12.2013]. [Online]. Available: http://www.ipv6observatory.eu/indicator/

[23] E. Vyncke, "IPv6 Deployment Aggregated Status," [cited 22.12.2013]. [Online]. Available: http://www.vyncke.org/ipv6status

[24] M. Leber, "Global IPv6 Deployment Progress Report," [cited 22.12.2013]. [Online]. Available: http://bgp.he.net/ipv6-progress-report.cgi

[25] A. Keranen and J. Arkko, "Some Measurements on World IPv6 Day from an End-User Perspective," RFC 6948 (Informational), Internet Engineering Task Force, Jul. 2013. [Online]. Available: http://www.ietf.org/rfc/rfc6948.txt

[26] H. Kaczmarek, "Internet IPv6 Adoption: Methodology, Measurement and Tools," July 2012, [cited 19.4.2013]. [Online]. Available: http://6lab.cisco.com/stats/information

[27] University of Oregon, “Route Views Project,” [cited 21.1.2015]. [Online]. Available: http://www.routeviews.org

[28] University of Oregon, “The State of the Internet,” [cited 21.1.2015].

[29] Q. Vohra and E. Chen, “BGP Support for Four-octet AS Number Space,” RFC 4893 (Proposed Standard), Internet Engineering Task Force, May 2007, obsoleted by RFC 6793. [Online]. Available: http://www.ietf.org/rfc/rfc4893.txt

[30] G. Huston, “BGP in 2014,” January 2015, [cited 23.2.2015]. [Online]. Available: http://www.potaroo.net/ispcol/2015-01/bgp2014.html

[31] R. Hinden and S. Deering, “IP Version 6 Addressing Architecture,” RFC 4291 (Draft Standard), Internet Engineering Task Force, Feb. 2006, updated by RFCs 5952, 6052, 7136, 7346, 7371. [Online]. Available: http://www.ietf.org/rfc/rfc4291.txt

[32] G. Huston, “Flailing IPv6,” December 2010, [cited 23.2.2015]. [Online]. Available: http://www.potaroo.net/ispcol/2010-12/6to4fail.html

[33] G. Huston, “Testing Teredo,” April 2011, [cited 23.2.2015]. [Online]. Available: http://www.potaroo.net/ispcol/2011-04/teredo.html

[34] B. Carpenter, “IPng White Paper on Transition and Other Considerations,” RFC 1671 (Informational), Internet Engineering Task Force, Aug. 1994. [Online]. Available: http://www.ietf.org/rfc/rfc1671.txt

[35] J. Livingood, “Considerations for Transitioning Content to IPv6,” RFC 6589 (Informational), Internet Engineering Task Force, Apr. 2012. [Online]. Available: http://www.ietf.org/rfc/rfc6589.txt

[36] D. Wing and A. Yourtchenko, “Happy Eyeballs: Success with Dual-Stack Hosts,” RFC 6555 (Proposed Standard), Internet Engineering Task Force, Apr. 2012. [Online]. Available: http://www.ietf.org/rfc/rfc6555.txt

[37] O. Bonaventure, “Happy eyeballs makes me unhappy...” December 2013, [cit. 27.2.2014]. [Online]. Available: http://perso.uclouvain.be/olivier.bonaventure/blog/html/2013/12/03/happy.html

[38] N. Sarrar, G. Maier, B. Ager, R. Sommer, and S. Uhlig, “Investigating IPv6 Traffic,” in Passive and Active Measurement. Springer Berlin / Heidelberg, 2012.

[39] J. Bitte, “HTTP logging and information retrieval tool,” [cit. 27.2.2014]. [Online]. Available: http://dumpsterventures.com/jason/httpry/

[40] T. Podermański and M. Grégr, “Worldwide online IPv6 penetration,” August 2013, [cit. 27.2.2014]. [Online]. Available: http://6lab.cz/live-statistics/data-source

[41] CZ.NIC, “.cz statistics,” [cited 22.12.2013]. [Online]. Available: http://stats.nic.cz

[42] D. Wing, “AAAA and IPv6 Connectivity statistics,” July 2009, [cit. 27.5.2015]. [Online]. Available: http://perso.uclouvain.be/olivier.bonaventure/blog/html/2013/12/03/happy.html

[43] E. Vyncke, "IPv6 Deployment Monitoring: Internet metrics," July 2013, [cit. 22.7.2015]. [Online]. Available: http://www.ipv6hackers.org/meetings/ipv6-hackers-1

[44] R. Zajíc, "Jak obejít nefunkční IPv6 na justice.cz," [cited 22.12.2013]. [Online]. Available: http://zajic.v.pytli.cz/2012/06/10/jak-obejit-nefunkcni-ipv6-na-justice-cz/

[45] NANOG discussion, "IPv6 Cogent vs Hurricane Electric," [cited 22.3.2016]. [Online]. Available: http://mailman.nanog.org/pipermail/nanog/2015-December/082669.html

[46] P. Saab, "Facebook IPv6 Strategy," March 2015, [cit. 06.10.2015]. [Online]. Available: https://youtu.be/An7s25FSK0U

[47] S. Greenstein, "Framing empirical work on the evolving structure of commercial Internet markets," Understanding the Digital Economy (Cambridge, MA: MIT Press, 2000a), 2000.

[48] ETSI, "Handover interface for the request and delivery of retained data," 2010, [version 1.7.1].

[49] M. Patrick, "DHCP Relay Agent Information Option," RFC 3046 (Proposed Standard), Internet Engineering Task Force, Jan. 2001, updated by RFC 6607. [Online]. Available: http://www.ietf.org/rfc/rfc3046.txt

[50] INNET VŠB - Technical University of Ostrava, "New computer registration," 2013, [cited 07.8.2013]. [Online]. Available: http://idoc.vsb.cz/en/okruhy/cit/tuonet/pripojeni/studenti/koleje/registrace/

[51] B. Robenek, "Informační systém klubu Silicon Hill - správa sítě," 2013. [Online]. Available: http://installfest.cz/if13/files/slidy/robenek-is-sh.pdf

[52] A. Houdek, "Pravidla používání internetu na kolejích KaM (mimo kolej 17.listopadu)," 2013, czech, [cited 07.8.2013]. [Online]. Available: http://www.kam.cuni.cz/KAM-33.html

[53] B. Claise, B. Trammell, and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information," RFC 7011 (INTERNET STANDARD), Internet Engineering Task Force, Sep. 2013. [Online]. Available: http://www.ietf.org/rfc/rfc7011.txt

[54] B. Claise, "Cisco Systems NetFlow Services Export Version 9," RFC 3954 (Informational), Internet Engineering Task Force, Oct. 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3954.txt

[55] Cisco Systems, Inc., "Introducing to Cisco IOS NetFlow," [online], Cisco, Tech. Rep., May 2012, [cited 08.8.2013].

[56] S. Kawamura and M. Kawashima, "A Recommendation for IPv6 Address Text Representation," RFC 5952 (Proposed Standard), Internet Engineering Task Force, Aug. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5952.txt

[57] Internet Assigned Numbers Authority, "Internet Protocol Version 6 Address Space," 2013, [cited 09.8.2013]. [Online]. Available: http://www.iana.org/assignments/ipv6-address-space/ipv6-address-space.xhtml

[58] S. Thomson, T. Narten, and T. Jinmei, "IPv6 Stateless Address Autoconfiguration," RFC 4862 (Draft Standard), Internet Engineering Task Force, Sep. 2007, updated by RFC 7527. [Online]. Available: http://www.ietf.org/rfc/rfc4862.txt

[59] T. Narten, R. Draves, and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6," RFC 4941 (Draft Standard), Internet Engineering Task Force, Sep. 2007. [Online]. Available: http://www.ietf.org/rfc/rfc4941.txt

[60] T. Aura, "Cryptographically Generated Addresses (CGA)," RFC 3972 (Proposed Standard), Internet Engineering Task Force, Mar. 2005, updated by RFCs 4581, 4982. [Online]. Available: http://www.ietf.org/rfc/rfc3972.txt

[61] T. Narten, E. Nordmark, W. Simpson, and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)," RFC 4861 (Draft Standard), Internet Engineering Task Force, Sep. 2007, updated by RFCs 5942, 6980, 7048, 7527, 7559. [Online]. Available: http://www.ietf.org/rfc/rfc4861.txt

[62] B. Haberman and R. Hinden, "IPv6 Router Advertisement Flags Option," RFC 5175 (Proposed Standard), Internet Engineering Task Force, Mar. 2008. [Online]. Available: http://www.ietf.org/rfc/rfc5175.txt

[63] C. Perkins, D. Johnson, and J. Arkko, "Mobility Support in IPv6," RFC 6275 (Proposed Standard), Internet Engineering Task Force, Jul. 2011. [Online]. Available: http://www.ietf.org/rfc/rfc6275.txt

[64] J. Jeong, S. Park, L. Beloeil, and S. Madanapalli, "IPv6 Router Advertisement Options for DNS Configuration," RFC 6106 (Proposed Standard), Internet Engineering Task Force, Nov. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc6106.txt

[65] IANA: Thomas Huth, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)," October 2013, [cited 24.10.2013]. [Online]. Available: http://www.iana.org/assignments/dhcpv6-parameters/dhcpv6-parameters.txt

[66] R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins, and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)," RFC 3315 (Proposed Standard), Internet Engineering Task Force, Jul. 2003, updated by RFCs 4361, 5494, 6221, 6422, 6644, 7083, 7227, 7283, 7550. [Online]. Available: http://www.ietf.org/rfc/rfc3315.txt

[67] T. Narten and J. Johnson, "Definition of the UUID-Based DHCPv6 Unique Identifier (DUID-UUID)," RFC 6355 (Proposed Standard), Internet Engineering Task Force, Aug. 2011. [Online]. Available: http://www.ietf.org/rfc/rfc6355.txt

[68] A. Conta and S. Deering, "Generic Packet Tunneling in IPv6 Specification," RFC 2473 (Proposed Standard), Internet Engineering Task Force, Dec. 1998. [Online]. Available: http://www.ietf.org/rfc/rfc2473.txt

[69] E. Nordmark and R. Gilligan, "Basic Transition Mechanisms for IPv6 Hosts and Routers," RFC 4213 (Proposed Standard), Internet Engineering Task Force, Oct. 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4213.txt

[70] A. Durand, R. Droms, J. Woodyatt, and Y. Lee, "Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion," RFC 6333 (Proposed Standard), Internet Engineering Task Force, Aug. 2011, updated by RFC 7335. [Online]. Available: http://www.ietf.org/rfc/rfc6333.txt

[71] X. Li, C. Bao, W. Dec, O. Troan, S. Matsushima, and T. Murakami, "Mapping of Address and Port using Translation (MAP-T)," RFC 7599 (Proposed Standard), Internet Engineering Task Force, Jul. 2015. [Online]. Available: http://www.ietf.org/rfc/rfc7599.txt

[72] O. Troan, W. Dec, X. Li, C. Bao, S. Matsushima, T. Murakami, and T. Taylor, "Mapping of Address and Port with Encapsulation (MAP-E)," RFC 7597 (Proposed Standard), Internet Engineering Task Force, Jul. 2015. [Online]. Available: http://www.ietf.org/rfc/rfc7597.txt

[73] R. Bush, "The Address plus Port (A+P) Approach to the IPv4 Address Shortage," RFC 6346 (Experimental), Internet Engineering Task Force, Aug. 2011. [Online]. Available: http://www.ietf.org/rfc/rfc6346.txt

[74] R. Despres, S. Jiang, R. Penno, Y. Lee, G. Chen, and M. Chen, "IPv4 Residual Deployment via IPv6 - A Stateless Solution (4rd)," RFC 7600 (Experimental), Internet Engineering Task Force, Jul. 2015. [Online]. Available: http://www.ietf.org/rfc/rfc7600.txt

[75] B. Carpenter and K. Moore, "Connection of IPv6 Domains via IPv4 Clouds," RFC 3056 (Proposed Standard), Internet Engineering Task Force, Feb. 2001. [Online]. Available: http://www.ietf.org/rfc/rfc3056.txt

[76] C. Huitema, "An Anycast Prefix for 6to4 Relay Routers," RFC 3068 (Historic), Internet Engineering Task Force, Jun. 2001, obsoleted by RFC 7526. [Online]. Available: http://www.ietf.org/rfc/rfc3068.txt

[77] G. Huston, "Flailing IPv6," December 2010, [cited 17.12.2015]. [Online]. Available: http://www.potaroo.net/ispcol/2010-12/6to4fail.html

[78] T. Anderson, "IPv6 dual-stack client loss in Norway," December 2010, [cited 17.12.2015]. [Online]. Available: https://fud.no/ipv6/

[79] E. Aben and T. Anderson, "6to4 - How Bad is it Really?" December 2010, [cited 17.12.2015]. [Online]. Available: https://labs.ripe.net/Members/emileaben/6to4-how-bad-is-it-really

[80] B. Carpenter, "Advisory Guidelines for 6to4 Deployment," RFC 6343 (Informational), Internet Engineering Task Force, Aug. 2011. [Online]. Available: http://www.ietf.org/rfc/rfc6343.txt

[81] O. Troan and B. Carpenter, "Deprecating the Anycast Prefix for 6to4 Relay Routers," RFC 7526 (Best Current Practice), Internet Engineering Task Force, May 2015. [Online]. Available: http://www.ietf.org/rfc/rfc7526.txt

[82] R. Despres, "IPv6 Rapid Deployment on IPv4 Infrastructures (6rd)," RFC 5569 (Informational), Internet Engineering Task Force, Jan. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5569.txt

[83] W. Townsley and O. Troan, "IPv6 Rapid Deployment on IPv4 Infrastructures (6rd) – Protocol Specification," RFC 5969 (Proposed Standard), Internet Engineering Task Force, Aug. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5969.txt

[84] B. Carpenter and C. Jung, "Transmission of IPv6 over IPv4 Domains without Explicit Tunnels," RFC 2529 (Proposed Standard), Internet Engineering Task Force, Mar. 1999. [Online]. Available: http://www.ietf.org/rfc/rfc2529.txt

[85] C. Huitema, "Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)," RFC 4380 (Proposed Standard), Internet Engineering Task Force, Feb. 2006, updated by RFCs 5991, 6081. [Online]. Available: http://www.ietf.org/rfc/rfc4380.txt

[86] J. Hoagland, "The Teredo Protocol: Tunneling Past Network Security and Other Security Implications," November 2006, [cit. 3.1.2016]. [Online]. Available: http://www.symantec.com/avcenter/reference/Teredo˙Security.pdf

[87] Microsoft, "Teredo Overview," January 2007, [cit. 5.1.2016]. [Online]. Available: https://technet.microsoft.com/en-us/library/bb457011.aspx

[88] D. Veit and C. Palmer, "Usage of Teredo and IPv6 for P2P on Windows 10 and Xbox One," May 2015, [cited 7.1.2016]. [Online]. Available: http://mailman.nanog.org/pipermail/nanog/2015-May/075244.html

[89] C. Palmer, "Usage of Teredo and IPv6 for P2P on Windows 10 and Xbox One," May 2015, [cited 7.1.2016]. [Online]. Available: http://mailman.nanog.org/pipermail/nanog/2015-May/075244.html

[90] A. Nakagawa, "Operational experience of map-e," Working Draft, IETF Secretariat, Internet-Draft draft-akira-v6ops-mape-experience-00, July 2015, http://www.ietf.org/internet-drafts/draft-akira-v6ops-mape-experience-00.txt. [Online]. Available: http://www.ietf.org/internet-drafts/draft-akira-v6ops-mape-experience-00.txt

[91] F. Gont, A. Cooper, D. Thaler, and W. Liu, "Recommendation on Stable IPv6 Interface Identifiers," October 2015, work-in-progress, [cit. 18.1.2016]. [Online]. Available: https://tools.ietf.org/html/draft-ietf-6man-default-iids-08

[92] E. Jankiewicz, J. Loughney, and T. Narten, "IPv6 Node Requirements," RFC 6434 (Informational), Internet Engineering Task Force, Dec. 2011. [Online]. Available: http://www.ietf.org/rfc/rfc6434.txt

[93] T. Anderson, "Issue 32621: Support for DHCPv6 (RFC 3315)," June 2012, [cit. 21.1.2016]. [Online]. Available: https://code.google.com/p/android/issues/detail?id=32621

[94] W. Kumari, O. Gudmundsson, P. Ebersman, and S. Sheng, "Captive-Portal Identification Using DHCP or Router Advertisements (RAs)," RFC 7710 (Proposed

Standard), Internet Engineering Task Force, Dec. 2015. [Online]. Available: http://www.ietf.org/rfc/rfc7710.txt

[95] G. Halwasia, S. Bhandari, and W. Dec, "Client Link-Layer Address Option in DHCPv6," RFC 6939 (Proposed Standard), Internet Engineering Task Force, May 2013. [Online]. Available: http://www.ietf.org/rfc/rfc6939.txt

[96] M. Elich, M. Grégr, and P. Čeleda, "," in Traffic Monitoring and Analysis, ser. Lecture Notes in Computer Science 6613. Springer Verlag, 2011, pp. 64–71. [Online]. Available: http://www.fit.vutbr.cz/research/view˙pub.php?id=9604

[97] M. Elich, P. Velan, T. Jirsik, and P. Celeda, "An investigation into teredo and 6to4 transition mechanisms: Traffic analysis," in Local Computer Networks Workshops (LCN Workshops), 2013 IEEE 38th Conference on, Oct 2013, pp. 1018–1024.

[98] M. Grégr, "IPv6 transition techniques monitoring tool," October 2012. [Online]. Available: http://www.fit.vutbr.cz/˜igregr/prods.php.en?id=268

[99] L. Polčák, M. Holkovič, and P. Matoušek, "A New Approach for Detection of Host Identity in IPv6 Networks," in Proceedings of the 4th International Conference on Data Communication Networking, 10th International Conference on e-Business and 4th International Conference on Optical Communication Systems. SciTePress - Science and Technology Publications, 2013, pp. 57–63. [Online]. Available: http://www.fit.vutbr.cz/research/view˙pub.php?id=10362

[100] N. Moore, "Optimistic Duplicate Address Detection (DAD) for IPv6," RFC 4429 (Proposed Standard), Internet Engineering Task Force, Apr. 2006, updated by RFC 7527. [Online]. Available: http://www.ietf.org/rfc/rfc4429.txt

[101] S. Routhier, "Management Information Base for the Internet Protocol (IP)," RFC 4293 (Proposed Standard), Internet Engineering Task Force, Apr. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4293.txt

[102] R. Hofstede, A. Sperotto, T. Fioreze, and A. Pras, Networked Services and Applications - Engineering, Control and Management: 16th EUNICE/IFIP WG 6.6 Workshop, EUNICE 2010, Trondheim, Norway, June 28-30, 2010. Proceedings. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, ch. The Network Data Handling War: MySQL vs. NfDump, pp. 167–176. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-13971-0˙16

[103] P. Velan and R. Krejčíř, "Flow information storage assessment using IPFIXcol," in Dependable Networks and Services. Springer, 2012, pp. 155–158.

[104] M. Grégr, P. Matoušek, T. Podermański, and M. Švéda, "Practical ipv6 monitoring - challenges and techniques," in Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011). IEEE Computer Society, 2011, pp. 660–663. [Online]. Available: http://www.fit.vutbr.cz/research/view˙pub.php?id=9639

[105] M. Grégr, T. Podermański, and M. Švéda, User identification in IPV6 network. Zilina University Publisher, 2012, pp. 5–8. [Online]. Available: http://www.fit.vutbr.cz/research/view˙pub.php?id=9960

[106] L. Polčák, M. Grégr, M. Kajan, P. Matoušek, and V. Veselý, "Designing lawful interception in ipv6 networks," in Security and Protection of Information. Brno University of Defence, 2011, pp. 114–126. [Online]. Available: http://www.fit.vutbr.cz/research/view˙pub.php?id=9620

[107] P. Richter, M. Allman, R. Bush, and V. Paxson, "A Primer on IPv4 Scarcity," ACM SIGCOMM Computer Communication Review, vol. 45, no. 2, 2015, pp. 21–31. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2766330.2766335

[108] M. Mueller, B. Kuerbis, and H. Asghari, "Dimensioning the elephant: an empirical analysis of the IPv4 number market," info, vol. 15, no. 6, 2013, pp. 6–18. [Online]. Available: http://dx.doi.org/10.1108/info-07-2013-0039

[109] Alfa Telecom s.r.o, "Registration of IP and Autonomous systems," 2016, [cited 13.6.2016]. [Online]. Available: http://ip-as.com/index.html

[110] G. Maier, F. Schneider, and A. Feldmann, NAT Usage in Residential Broadband Networks. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 32–41. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-19260-9˙4

[111] K. Egevang and P. Francis, "The IP Network Address Translator (NAT)," RFC 1631 (Informational), Internet Engineering Task Force, May 1994, obsoleted by RFC 3022. [Online]. Available: http://www.ietf.org/rfc/rfc1631.txt

[112] P. F. Tsuchiya and T. Eng, "Extending the ip internet through address reuse," SIGCOMM Comput. Commun. Rev., vol. 23, no. 1, Jan. 1993, pp. 16–33. [Online]. Available: http://doi.acm.org/10.1145/173942.173944

[113] P. Francis, "Selected publications," 2008, [cited 22.4.2016]. [Online]. Available: http://www.cs.cornell.edu/people/francis/publications.html

[114] D. Clark, L. Chapin, V. Cerf, R. Braden, and R. Hobby, "Towards the Future Internet Architecture," RFC 1287 (Informational), Internet Engineering Task Force, Dec. 1991. [Online]. Available: http://www.ietf.org/rfc/rfc1287.txt

[115] P. F. Tsuchiya, "The IP Network Address Translator (Nat): Preliminary Design," in Bell Communications Research, 1991.

[116] P. Srisuresh and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)," RFC 3022 (Informational), Internet Engineering Task Force, Jan. 2001. [Online]. Available: http://www.ietf.org/rfc/rfc3022.txt

[117] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, and E. Lear, "Address Allocation for Private Internets," RFC 1918 (Best Current Practice), Internet Engineering Task Force, Feb. 1996, updated by RFC 6761. [Online]. Available: http://www.ietf.org/rfc/rfc1918.txt

[118] J. Weil, V. Kuarsingh, C. Donley, C. Liljenstolpe, and M. Azinger, "IANA-Reserved IPv4 Prefix for Shared Address Space," RFC 6598 (Best Current Practice), Internet Engineering Task Force, Apr. 2012. [Online]. Available: http://www.ietf.org/rfc/rfc6598.txt

[119] J. Rosenberg, J. Weinberger, C. Huitema, and R. Mahy, "STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs)," RFC 3489 (Proposed Standard), Internet Engineering Task Force, Mar. 2003, obsoleted by RFC 5389. [Online]. Available: http://www.ietf.org/rfc/rfc3489.txt

[120] F. Audet and C. Jennings, "Network Address Translation (NAT) Behavioral Requirements for Unicast UDP," RFC 4787 (Best Current Practice), Internet Engineering Task Force, Jan. 2007, updated by RFCs 6888, 7857. [Online]. Available: http://www.ietf.org/rfc/rfc4787.txt

[121] S. Guha, K. Biswas, B. Ford, S. Sivakumar, and P. Srisuresh, "NAT Behavioral Requirements for TCP," RFC 5382 (Best Current Practice), Internet Engineering Task Force, Oct. 2008, updated by RFC 7857. [Online]. Available: http://www.ietf.org/rfc/rfc5382.txt

[122] R. Penno, S. Perreault, M. Boucadair, S. Sivakumar, and K. Naito, "Updates to Network Address Translation (NAT) Behavioral Requirements," RFC 7857 (Best Current Practice), Internet Engineering Task Force, Apr. 2016. [Online]. Available: http://www.ietf.org/rfc/rfc7857.txt

[123] G. Huston, "Anatomy: A look inside network address translators," The Internet Protocol Journal, vol. 7, no. 3, 2004, pp. 2–32.

[124] J. Roskind, "Quic: Design document and specification rationale," 2013, [cited 22.4.2016]. [Online]. Available: https://www.chromium.org/quic

[125] M. Kosters and G. Huston, "CGNs in IP. What are you going to do about it?" January 2013, [cited 11.11.2013]. [Online]. Available: http://www.potaroo.net/presentations/2013-01-16-cgn-kosters-joint-techs.pdf

[126] Z. Chen, C. Zhou, T. Tsou, and T. Taylor, "Syslog Format for NAT Logging," Working Draft, IETF Secretariat, Internet-Draft draft-ietf-behave-syslog-nat-logging-06, January 2014. [Online]. Available: http://www.ietf.org/internet-drafts/draft-ietf-behave-syslog-nat-logging-06.txt

[127] S. Sivakumar and R. Penno, "IPFIX Information Elements for logging NAT Events," Working Draft, IETF Secretariat, Internet-Draft draft-ietf-behave-ipfix-nat-logging-09, May 2016. [Online]. Available: http://www.ietf.org/internet-drafts/draft-ietf-behave-ipfix-nat-logging-09.txt

[128] C. Donley, C. Grundemann, V. Sarawat, K. Sundaresan, and O. Vautrin, "Deterministic Address Mapping to Reduce Logging in Carrier-Grade NAT Deployments," RFC 7422 (Informational), Internet Engineering Task Force, Dec. 2014. [Online]. Available: http://www.ietf.org/rfc/rfc7422.txt

[129] Wikipedia, "Birthday problem — Wikipedia, The Free Encyclopedia," 2016, [Online; accessed 1-July-2016]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Birthday˙problem&oldid=726414796