# Survey of IPv6 Transition Technologies for Security Analysis

Gábor LENCSE [†, ‡], and Youki KADOBAYASHI [†]

[†] Laboratory for Cyber Resilience, Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara, 630-0192, JAPAN

[‡] Department of Telecommunications, Széchenyi István University
Egyetem tér 1, Győr, H-9026, HUNGARY

E-mail: [†] {gabor-l,youki-k}@is.naist.jp, [‡] lencse@sze.hu

**Abstract** Due to the depletion of the public IPv4 address pool, the transition to IPv6 became inevitable. However, the transition will take a long time, while the two incompatible versions of the Internet Protocol coexist. Different IPv6 transition technologies were developed, which can be used to enable the communication in various scenarios, but they also involve additional security issues. In this paper, first, we develop a priority classification method for the ranking of different IPv6 transition technologies and their most important implementations, so that the vulnerabilities of the most crucial ones may be examined first. Then we survey the most important IPv6 transition technologies by describing their application scenario and the basics of their operation and we also determine the importance of their security analysis according to our ranking system.

**Key words** IPv6 transition technologies, Security

## 1. Introduction

Although IPv6, the new version of the Internet Protocol, was standardized in 1998 [1], its deployment was very slow at the beginning and it started to accelerate only in the latest years for several reasons [2]. Unfortunately, the old version, IPv4, and the new version, IPv6, are incompatible with each other. Several IPv6 transition technologies [3] were developed, which address various communication scenarios. In this paper, we survey the IPv6 transition technologies and identify those of them which would be worth submitting of a detailed security analysis. To achieve this goal, first we develop a priority classification method for both the technologies and their most important implementations, and then we present a near exhaustive overview of the existing IPv6 transition technologies together with their priority classification.

This paper aims to be the first step of a research, which targets to identify and mitigate the security vulnerabilities of those IPv6 transition technologies that will play the most important role in the upcoming transition to IPv6, which we are headed with for several years or perhaps decades.

## 2. Our Priority Classification Method

There are several IPv6 transition technologies, and some have many implementations, thus the thorough analysis of all of them would require a huge amount of resources. Therefore, we developed a simple method for their priority classification both at IPv6 transition technology level and at implementation level. We are aware that such ranking systems have their limits. E.g. considering too few factors, we may oversimplify the problem, whereas considering too many factors, we may make the problem too complex. The choice of the examined factors, the determination of their relative priority (or using a weighting system) are also subjective decisions.

First, we define some expressions, which will be used in the classification as conditions. We call a solution *standard*, if it is defined by a valid (non-obsoleted) IETF RFC of at least *Proposed Standard* state. We call the solution *obsolete* if the defining RFC was obsoleted by another RFC (and there was no new version defined). In all other cases we call it *non-standard.* As for communication scenarios (that is the problem to be solved by a given IPv6 technology), we call a scenario *relevant*, if the scenario is *common* (there are or there will be a high number of users) and *unavoidable* (its usage is not based on someone's unwise selection). Of course, both being common and being unavoidable are questionable, but this is the nature of the beast, as they refer to real life situations.

We classify an IPv6 transition technology as *important*, if it is *standard* and the communication scenario is *relevant.* An *important* technology is also *essential*, if the technology is the only known standard solution (to the given scenario), otherwise it is *replaceable.* The security analysis of essential technologies will have the very first priority. They can be followed by the replaceable ones, probably one carefully chosen technology from each communication scenarios first, and the others may follow later on. If a solution is *non-standard* or the communication scenario is not considered *relevant,* we classify the security analysis of the IPv6 technology as *optional*, unless we have good reasons to classify it *important*. The *optional* classification means the lowest priority and the *obsolete* class is trivial: they are not to be dealt with. For a more fine grain classification, we will also use the secondary term *aging*, to express that the communication scenario is expected to be no more common in the near future.

As for the implementations, we only deal with those that are free software [4] (also called open source [5]) for multiple reasons:

- The licenses of certain vendors (e.g. [6] and [7]) do not allow reverse engineering and sometimes even the publication of benchmarking results is prohibited.
- Free software can be used by anyone for any purposes thus our results can be helpful for anyone.
- Free software is available free of charge for us, too.

Within the category of the free software implementations, we give further priority to those, which are used widespread and/or are known to be stable and high performance (if such information is available).

## 3. Survey of IPv6 Transition Technologies

We give a comprehensive survey of all known IPv6 transition technologies, presenting their purpose and the basics of their operation. We classify them, and if they are considered essential, we give some implementations, too.

As there are a high number of IPv6 transition technologies, we follow the categories presented in [8].

### 3.1. Single Translation Type Solutions and DNS64

The aim of these transition mechanisms is to enable a client, which can use only IPvX to communicate with a server, which can use only IPvY, where X and Y are from {4, 6} and X≠Y. They translate the IP data packets arriving from the client from IPvX to IPvY and also do the reverse translation for the packets arriving from the server. Although DNS64 [9] does not belong to them, we shall discuss it among them together with NAT64 [10] because these two technologies are used together.

#### *3.1.1. DNS64 and Stateful NAT64*

Both DNS64 [9] and stateful NAT64 [10] are *standard* solutions, and can be used together for enabling IPv6-only clients to communicate with IPv4-only servers. This communication scenario is expressly *relevant* because the ISPs (Internet Service Providers) cannot distribute public IPv4 addresses to their high number of new customers due the depletion of the global IPv4 address pool, and we consider it a commendable practice if they go ahead and deploy IPv6 instead of using CGN (Carrier Grade NAT) or any other solutions, which would keep their customers in the IPv4 world and thus make the transition period longer. However, still there are, and there will be servers, which can use only IPv4. Thus we consider the analysis of DNS64 and NAT64 *important* and also *essential*, because no other standard IPv6 transition technology exists for this scenario since NAT-PT was moved to historic status [11].

Now, we summarize the operation of DNS64 and NAT64 in a nutshell.

The DNS64 server acts as a proxy: when in receives a request for an IPv6 address (AAAA record) for a given domain name, it asks the normal DNS system about it. If the DNS64 server receives a valid answer, then it simply returns the answer. If it does not receive a valid answer, then it asks the normal DNS system about the IPv4 address (A record) of the given domain name. The DNS64 server uses the received IPv4 address to synthesize a so-called *IPv4-embedded IPv6 address* [12], which contains the IPv4 address at a well defined position. Finally, the DNS64 server returns the resulted IPv6 address (or an error message, if it had not received an IPv4 address).

When a stateful NAT64 gateway receives an IPv6 packet, which belong to a new communication session, the NAT64 gateway constructs an IPv4 packet with the destination IPv4 address taken from the appropriate position of the destination IPv6 address, and with its own public IPv4 address as source address, and it registers the new session into its connection tracking table. When it receives a reply packet, it identifies the communication session, which the IPv4 packet belongs to and constructs an IPv6 packet. It is an important restriction of stateful NAT64 that a communication session may be initiated only from the IPv6 side.

Most client-server applications can work well with the DNS64 plus NAT64 solution, for more information see [13].

There are several free software DNS64 implementations. We have examined the stability and performance of BIND [14], TOTD [15], Unbound [16] and PowerDNS [17], and they all proved to be viable solutions but they showed different performances [18]. We have developed an experimental DNS64 server, mtd64-ng [19], which is not ready to be used in production systems, but as for its performance, it seems to be a good candidate.

As “BIND is far the most widely used DNS software on the Internet” [14] and it supports DNS64, the security analysis of BIND is a must. We plan to select further candidate(s) after examining how the performances of the above mentioned DNS64 implementations scales up in a function of the number of CPU cores. We have both the benchmarking methodology [20] and the performance measurement tool [21] ready for the analysis.

As for free software stateful NAT64 implementations, we have experience with PF (Packet Filter) [22] of OpenBSD [23], which supports NAT64 since version 5.1, and the combination of the stateless TAYGA [24] and the Netfilter [25] of Linux (also called iptables after name of its user interface tool). We have examined and compared the stability and performance [26]. Ecdysis [27] and Jool [28] are two other free software stateful NAT64 implementations, which we did not test yet. We plan to compare the performance of these four NAT64 implementations, before selecting some of them for detailed security analysis, however, presently we do not have a stateful NAT64 benchmarking tool, which complies with the relevant Internet Draft [29]. There is one NAT64 benchmarking tool, which complies with the draft, but it implements only the sateless NAT64 tests [30].

#### *3.1.2. NAT-PT*

Basic NAT-PT and NAPT-PT were defined in a standard track RFC [31] in 2000. This rather complex solution addressed bidirectional translation between the IPv4 realm and the IPv6 realm but they were moved to historic status for several reasons in 2007 [11].

#### *3.1.3. SIIT*

The stateless IP/ICMP translation algorithm can be used to translate between the IPv4 and the IPv6 headers (including ICMP headers) in both directions. Although its previously defining standard track RFCs (RFC 2765, RFC 6154) have been obsoleted, it is considered as *standard* (defined by a proposed standard state RFC) [32]. Being stateless, it cannot be used as a solution for the IPv4 address shortage problem, but we still consider it *relevant*, because it can be used as a building element of more complex technologies, thus we classify its security analysis as *important*. As it is used in multiple technologies, we consider the *essential* vs. *replaceable* question in the case of those technologies.

We note that the above mentioned Jool [28] implements also SIIT.

#### *3.1.4. SIIT-DC*

The *non-standard* SIIT-DC [33] is an application of SIIT in IPv6 data centers (DC). Its goal is the enable DC operators to use IPv6-only servers, while their system is also available for IPv4-only clients.

#### *3.1.5. IVI*

IVI [34] (the name is the contraction of the Roman numbers IV and VI) is a *non-standard* stateless translation solution between IPv4 and IPv6, where the translation may be initiated from both directions. (It is similar to the standard SIIT [32].)

#### *3.1.6. SA46T-AT*

The aim of the *non-standard* SA46T-AT [35] was to enable an IPv6-only host to access to an IPv4-only host. The scenario is similar to that of DNS64+NAT64, but this technology would have been worked also with private IPv4 addresses, which added significant complexity to the solution. Its Internet Draft expired and it did not became an RFC.

#### *3.1.7. TRT*

TRT [36] is an old *non-standard* solution (or rather concept) aimed to enable IPv6-only hosts to exchange TCP or UDP traffic with IPv4-only hosts. The concept was good and it was later realized as stateful NAT64.

#### *3.1.8. DNS46 + NAT46*

Although it is not typical now, later on it may be a realistic scenario that some old IPv4-only clients will need help in accessing IPv6-only servers. The *non-standard* DNS46+NAT46 [37] solution addresses this problem. Unfortunately, the logic of the DNS64 plus NAT64 solution can not be followed, because IPv6 addresses cannot be embedded into IPv4 address. Therefore, dynamic mappings are made between some elements of the IPv4 and of the IPv6 address range, which implies that the DNS46 server and the NAT46 gateway have to use a common database. Regrettably, the Internet Draft has never became an RFC, thus we are waiting for a standard solution, as we do not know any other workable solutions for this scenario.

### 3.2. Double Translation Type Solutions

The aim of these transition mechanisms is to carry IPv4 packets thorough IPv6 networks. They translate the IPv4 data packets to IPv6 data packets when they enter into the IPv6 network, and back to IPv4 when they leave the IPv6 network.

We note that double translation can not be used for carrying IPv6 packets thorough IPv4 networks, because the longer IPv6 addresses cannot be stored in an IPv4 packet.

#### *3.2.1. 464XLAT*

464XLAT [38] is a *non-standard* solution, which allows clients on IPv6-only networks to access IPv4-only Internet services, such as Skype. This scenario is significantly differ from the application scenario of DNS64+NAT64, because here there are some IPv4-only clients. It can be a legitimate decision of the ISPs that they use only IPv6 in their network, because of both the higher operational costs and more security vulnerabilities of a dual stack network. However, they need to satisfy their users' demand for the operability of their legacy IPv4-only applications. Therefore, although the security analysis of 464XLT has to be classified formally as *optional*, because 464XLAT is defined in an informational state RFC, we contend that it is worth doing so and therefore classify it *important* but *replaceable* and we give the basics of its operation.

464XLAT performs two translations. The CLAT device operates on the client side: it translates the IPv4 packets of the IPv4-only client software to IPv6 and also performs the translation of the reply packets in the other direction. (It actually performs SIIT.) The PLAT devices operates at the ISP side, and it actually performs stateful NAT64.

We note that CLAT acts as a router for IPv6 traffic. Thus 464XLAT can be used together with DNS64+NAT64 as follows: IPv6 capable clients receive IPv6 addresses, and they can reach IPv6 servers natively, whereas they can reach IPv4-only servers using DNS64+NAT64. Only the traffic of the legacy IPv4-only clients undergoes the double translation. As for DNS traffic, CLAT acts as a DNS proxy.

As for CLAT implementations, clatd [39] exists for Linux, and there is an implementation for Android, but we have no experience with them. Unfortunately, different CLATs will have to be tested for each mobile platform (e.g. Android, Windows Phone, iPhone) because the CLAT runs on the user's device.

#### *3.2.2. MAP-T*

MAP-T [40] is a *standard* solution for the same problem solved by 464XLAT: its aim is to carry the IPv4 traffic of the users thorough the IPv6-only network of an ISP. The operation of the solution is rather complex, we give only some highlights. First, the MAP-T CE (Customer Edge) device performs a NAT44 operation to restrict the available TCP/UDP port numbers for the user.[1] Then the CE performs a special stateless translation from IPv4 to IPv6, where the source IPv4 address and the selected port bits are encoded into the source IPv6 address according the MAP-T rules. The IPv6 packets can be destined to other users, where similar CEs perform the necessary transformations, or to the outside IPv4

[1] At this point we have to mention that we have serious doubts with this design. A proper upper bound for the port number need of a user may be much higher than the average. Thus the statistical multiplexing of stateful NAT64 could be more advantageous. For the consequences of the lack of port situation, see [41], and for the port number requirements of web browsing see [42] and its references.

Internet, in which case the MAP-T Border Relay performs the necessary transformations.

The scenario is deliberately *relevant*, thus the security analysis of MAP-T is classified *important*, but also *replaceable*, because other solutions exists. And because of the complexity of the solution, we plan to prioritize other solutions. In addition to the security considerations provided in Section 13 of the RFC [40], we would like to mention two things: being a complex solution, there are a lot of room for security holes and it would be a non-trivial problem for firewalls to decode and interpret the IPv6 traffic containing the packets translated from IPv4 packets.

#### *3.2.3. dIVI*

The scenario of the *non-standard* dIVI [43] is the same as that of the standard MAP-T, and it also uses similar solution of encoding the port range into the IPv6 address.

## 3.3. Encapsulation Type Solutions

These solutions carry the packets of either IP version encapsulated into the packets of the other IP version. The tunnel may be explicitly created or automatic.

#### *3.3.1. 6in4*

The aim of the *standard* 6in4 [44] solution is to carry IPv6 packets using IPv4 networks. (The idea behind is to connect the IPv6 "islands" using the IPv4 Internet, until the IPv6 infrastructure is completely built.) It is done by using static tunnels. Between the endpoints of the tunnels, the IPv6 packets are encapsulated in IPv4 packets using the 41 protocol identifier for IPv6 in the IPv4 header. Due to the slow deployment of the IPv6 protocol, the scenario is still common and sometimes unavoidable, thus we classify the security analysis of 6in4 as *important*. Although different other tunneling technologies exist, 6in4 is so widely used (also as a component of other technologies) that we classify its security analysis also *essential*.

As for implementations, all major network operating systems support it. E.g. a 6in4 tunnel endpoint may be statically configured under Linux by using the `ip` command and the `sit` tunnel interface. (Note: SIT stands for Simple Internet Transition.)

#### *3.3.2. 4in6*

Similarly, the *standard* 4in6 solution is a tunnel which carries IPv4 datagrams over IPv6 networks. It can be said that is was defined in [45], though this RFC defines a general encapsulation scheme, where packets of various protocols can be encapsulated into IPv6, e.g. IPv4, IPX, etc.

As it is widely used (also as a building block of other technologies) its security analysis is *important*, but formally *replaceable*, because there are alternative solutions, e.g. double translation may be used instead.

#### *3.3.3. 6to4*

The *standard* 6to4 [46] solution aims to enable IPv6 sites, which have only IPv4 Internet connection, to communicate with other IPv6 sites being in the same situation or with the native IPv6 Internet. The only prerequisite is that the sites must have a public IPv4 address. The solution provides globally routable IPv6 addresses for the IPv6 sites using the 2002::/16 prefix and the public IPv4 address. The sites are made available through the node that has the public IPv4 address, functioning as a 6to4 router. The IPv6 packets are carried as encapsulated into IPv4 packets (using 6in4) between two 6to4 routers, or between a 6to4 router and a 6to4 relay, if the other party is a native IPv6 node. The solution has a very important advantage over using configured tunnels that here the tunnels are created automatically and no action form the site's administrator is needed.

As there are still many parts of the world, where the ISPs do not provide IPv6 Internet access, 6to4 is still in use and can be the most convenient way of easily getting IPv6 Internet access, thus we formally classify its analysis *important*, but also *aging*. Of course it is *replaceable* by explicit tunnels (from tunnel brokers).

#### *3.3.4. Teredo*

The *standard* Teredo [47] can be used instead of 6to4, if no public IPv4 address is available for the site. It was designed to be a last resort if no others solutions available. We classify its security analysis *important* and *replaceable* (tunnel brokers) but also *aging*.

#### *3.3.5. 6rd*

The aim of the *standard* 6rd [48] is to provide an easy and fast method for ISPs to provide IPv6 Internet access for its customers using the IPv4 infrastructure of the ISP. The solution operates similarly to 6to4 with the important difference that it does not use the 2002::/16 prefix, but rather the own IPv6 prefix of the ISP and it eliminates all the operational and QoS issues, which arose from the broken reverse path relays in the case of 6to4 [49].

Although we admit that 6rd may be still useful for some ISPs, we recommend the use of native IPv6 and therefore we classify the security analysis of 6rd as *optional*.

#### *3.3.6. ISATAP*

The *non-standard* ISATAP [50] aims to connect dual stack nodes over IPv4 networks. Being non-standard, we classify it as *optional*.

#### *3.3.7. MAP-E*

The *standard* MAP-E [51] aims to address the same scenario as MAP-T, and solutions are also similar, but MAP-E uses encapsulation and decapsulation instead of double translation.

Similarly to MAP-T, we classify the security analysis of MAP-E *important*, but *replaceable*, and prefer other solutions.

#### *3.3.8. DS-Lite*

The *standard* DS-Lite [52] aims to address the same scenario as 464XLAT, and the solution is somewhat similar, but DS-Lite use encapsulation and decapsulation and then CGN (carrier grade NAT) for the IPv4 traffic. (It carries the IPv6 traffic of the user unmodified, and its CPE also provides a DNS proxy for the IPv4 applications as the CPE of 464XLAT does.)

We classify the security analysis of DS Lite *important*, but *replaceable*, and prefer other solutions due to the problems described in [53].

#### *3.3.9. Public 4over6*

The *non-standard* Public 4over6 [54] aims to provide IPv4 Internet connectivity over native IPv6 network using global IPv4 addresses. The defining informational RFC [54] recommends

Lightweight 4over6 for new deployments, thus we mention this solution only for completeness, and we do not deal with it.

#### *3.3.10. Lightweight 4over6*

The *standard* Lightweight 4over6 [55] addresses the same scenario as DS-Lite, and the solution itself is an extension of DS-Lite. We classify its security analysis *replaceable*, and prefer other solutions.

#### *3.3.11. SA46T*

The *non-standard* SA46T [56] is another technology aims to provide a way to carry IPv4 packets over the single-stack IPv6 backbone of ISPs. Its Internet Draft expired and was not published as an RFC, thus we do not deal with it.

#### *3.3.12. AYIYA*

The *non-standard* AYIYA [57] makes it possible to use tunnels, which carries any version IP packets in any version IP packet even over several NAT devices. The solution is deployed and used by tunnel brokers, but the Interned Draft expired long time ago.

#### *3.3.13. MPT*

The *non-standard* MPT [58] is a novel network layer multipath communication technology, which can be used as a tunnel solution, which supports both IPv4 or IPv6 tunnel over single or multiple IPv4 or IPv6 paths. It has one implementation [59], which has been successfully applied for different tasks, e.g. path throughput capacity aggregation [60], fast connection recovery [61] or elimination of the stalling events on YouTube video playback [62], but is has not been standardized yet.

## 4. Discussion and Future Work

By ranking the high number of existing IPv6 transition technologies into a few number of priority classes, we put the first step towards the security analysis of the most important ones.

Our next step is the development of a methodology for the identification of potential security issues of different IPv6 transition technologies. This methodology will be based on STRIDE, which is the abbreviation of Spoofing, Tampering, Repudiation, Information disclosure, and Elevation of privilege. This method was developed for software design, and uses a systematic approach to help uncovering potential vulnerabilities [63]. Marius Georgescu recommended a possible way of the application of the STRIDE method to the security analysis of IPv6 transition technologies [8]. That paper used the STRIDE method for examining the possible vulnerabilities of the following four categories of IPv6 transition technologies: dual stack, single translation, double translation, and encapsulation. We found that approach very promising, and we would like to go further using a more fine grain analysis by using the STRIDE method for the security analysis of the selected IPv6 transition technologies and their most important implementations. Currently we are working on a paper in, which we define the methodology and demonstrate its operability on the example of DNS64 and NAT64 [64].

## 5. Conclusion

We have developed a priority classification method for the ranking of different IPv6 transition technologies and their most important implementations, so that the vulnerabilities of the most crucial ones may be examined first.

We have surveyed the available IPv6 transition technologies by describing their application scenario and the basics of their operation and we also determined the importance of their security analysis according to our ranking system.

## Acknowledgment

This work was supported by International Exchange Program of National Institute of Information and Communications Technology (NICT).

## References

[1] S. Deering and R. Hinden, “Internet Protocol, version 6 (IPv6) specification”, IETF RFC 2460, 1998.
[2] M. Nikkhah and R. Guérin, “Migrating the internet to IPv6: An exploration of the when and why”, IEEE/ACM Transactions on Networking, vol. 24, no 4, pp. 2291–2304, Aug. 2016. DOI: 10.1109/TNET.2015.2453338
[3] M. Georgescu, H. Hazeyama, Y. Kadobayashi and S. Yamaguchi, “Empirical analysis of IPv6 transition technologies using the IPv6 Network Evaluation Testbed”, *EAI Endorsed Transactions on Industrial Networks and Intelligent Systems*, volt 15, no 2, e1, DOI: 10.4108/inis.2.2.e1
[4] Free Software Foundation, “The free software definition”, [Online]. Available: http://www.gnu.org/philosophy/free-sw.en.html
[5] Open Source Initiative, “The open source definition”, [Online]. Available: http://opensource.org/docs/osd
[6] Cisco, “End user license agreement”, [Online]. Available: http://www.cisco.com/c/en/us/products/end-user-license-agreement.html
[7] Juniper Networks, “End user license agreement”, [Online]. Available: http://www.juniper.net/support/eula/
[8] M. Georgescu, H. Hazeyama, T. Okuda, Y. Kadobayashi, and S. Yamaguchi, “The STRIDE towards IPv6: A comprehensive threat model for IPv6 transition technologies”, *Proc. 2nd International Conference on Information Systems Security and Privacy*, Rome, February 2016. DOI: 10.13140/RG.2.1.2781.6085
[9] M. Bagnulo, A Sullivan, P. Matthews and I. Beijnum, “DNS64: DNS extensions for network address translation from IPv6 clients to IPv4 servers”, RFC 6147, April 2011.
[10] M. Bagnulo, P. Matthews and I. Beijnum, “Stateful NAT64: Network address and protocol translation from IPv6 clients to IPv4 servers”, IETF RFC 6146, April 2011.
[11] C. Aoun and E. Davies, “Reasons to move the Network Address Translator - Protocol Translator (NAT-PT) to historic status”, IETF RFC 4966, July 2007.
[12] C. Bao, C. Huitema, M. Bagnulo, M Boucadair and X. Li, “IPv6 addressing of IPv4/IPv6 translators”, IETF RFC 6052, October 2010.
[13] S. Répás, T. Hajas and G. Lencse, “Application compatibility of the NAT64 IPv6 transition technology”, in *Proc. TSP 2014*, Berlin, Germany, Jul. 1-3, 2014, pp. 49–55. DOI: 10.1109/TSP.2015.7296383
[14] Internet Systems Consortium, “BIND: Versatile, Classic, Complete Name Server Software”, [Online]. Available: https://www.isc.org/downloads/bind
[15] The 6NET Consortium, “An IPv6 Deployment Guide”, Edited by Martin Dunmore, September, 2005. [Online]. Available: http://www.6net.org/book/deployment-guide.pdf
[16] NLnet Labs, *Unbound*, [Online]. Available: http://unbound.net

[17] Powerdns.com BV, “PowerDNS”, [Online]. Available: http://www.powerdns.com
[18] G. Lencse and S. Répás, “Performance analysis and comparison of four DNS64 implementations under different free operating systems”, *Telecommunication Systems*, vol. 63, no. 4, pp. 557–577, Nov. 2016, DOI: 10.1007/s11235-016-0142-x
[19] G. Lencse and D. Bakai, “Design, implementation and performance estimation of mtd64-ng a new tiny DNS64 proxy”, *Journal of Computing and Information Technology*, vol. 25, no. 2, June 2017, pp. 91–102, DOI:10.20532/cit.2017.1003419
[20] G. Lencse, M. Georgescu, and Y. Kadobayashi, “Benchmarking Methodology for DNS64 Servers”, *Computer Communications*, vol. 109, no. 1, pp. 162–175, September 1, 2017, DOI: 10.1016/j.comcom.2017.06.004
[21] G. Lencse and D. Bakai, “Design and implementation of a test program for benchmarking DNS64 servers”, *IEICE Transactions on Communications*, vol. E100-B, no. 6. pp. 948–954, Jun. 2017. DOI: 10.1587/transcom.2016EBN0007
[22] P. N. M. Hansteen, *The Book of PF: A No-Nonsense Guide to the OpenBSD Firewall*, 2nd ed., San Francisco: No Starch Press, 2010. ISBN: 978-1593272746
[23] Theo de Raadt, “The OpenBSD 5.1 Release”, May 1, 2012, ISBN 978-0-9784475-9-5, [Online]. Available: http://www.openbsd.org/51.html
[24] “TAYGA: Simple, no-fuss NAT64 for Linux” [Online]. Available: http://www.litech.org/tayga/
[25] H. Welte, P.N. Ayuso, “The netfilter.org project”, [Online]. Available: http://www.netfilter.org/
[26] G. Lencse and S. Répás, “Performance analysis and comparison of the TAYGA and of the PF NAT64 Implementations”, in *Proc. TSP 2013*, Rome, Italy, Jul. 2013. pp. 71–76. DOI: 10.1109/TSP.2013.6613894
[27] S. Perreault, J.-P. Dionne, and M. Blanchet, “Ecdysis: Open-Source DNS64 and NAT64”, in *Proc. AsiaBSDCon*, Feb. 8, 2010, [Online]. Available: http://viagenie.ca/publications/2010-03-13-asiabsdcon-nat64.pdf
[28] Network Information Center Mexico, “Jool: SIIT & NAT64”, [Online]. Available: http://jool.mx/en/index.html
[29] M. Georgescu, L. Pislaru, and G. Lencse, “Benchmarking methodology for IPv6 transition technologies”, Internet Draft (RFC Editor Queue), https://tools.ietf.org/html/draft-ietf-bmwg-ipv6-tran-tech-benchmarkin g-08
[30] P. Bálint, “Test software design and implementation for benchmarking of stateless IPv4/IPv6 translation implementations”, in *Proc. TSP 2017*, Barcelona, Spain, July 5-7, 2017, pp. 74–78.
[31] G. Tsirtsis and P. Srisuresh, “Network Address Translation - Protocol Translation (NAT-PT)”, IETF RFC 2766, 2000.
[32] C. Bao, X. Li, F. Baker, T. Anderson, F. Gont, “IP/ICMP Translation Algorithm”, IETF RFC 7915, June 2016.
[33] T. Anderson, “SIIT-DC: Stateless IP/ICMP translation for IPv6 data center environments”, IETF RFC 7755, February 2016.
[34] X. Li, C. Bao, M. Chen, H. Zhang, J. Wu, “The China Education and Research Network (CERNET) IVI translation design and deployment for the IPv4/IPv6 coexistence and transition”, IETF RFC 6219, May 2011.
[35] N. Matsuhira, K. Horiba, Y Ueno, O. Nakamura, “SA46T address translator”, expired Internet Draft, https://tools.ietf.org/html/draft-matsuhira-sa46t-at-06
[36] J. Hagino, K. Yamamoto, “An IPv6-to-IPv4 transport relay translator”, IETF RFC 3142, June 2001.
[37] D. Liu, H. Deng, “NAT46 consideration”, expired Internet Draft, https://tools.ietf.org/html/draft-liu-behave-nat46-02
[38] M. Mawatari, M. Kawashima, C. Byrne, “464XLAT: Combination of stateful and stateless translation”, IETF RFC 6877, April 2013.
[39] T. Anderson, “Clatd - a CLAT / SIIT-DC edge relay implementation for Linux”, [Online]. Available: https://github.com/toreanderson/clatd
[40] X. Li, C. Bao, W. Dec (ed), O. Troan, S. Matsushima, T. Murakami, “Mapping of address and port using translation (MAP-T)”, IETF RFC 7599, July 2015.
[41] S. Miyakawa, “IPv4 to IPv6 transformation schemes”, *IEICE Transactions on Communications*, vol. E93-B, no. 5, pp. 1078–1084, May 2010. DOI:10.1587/transcom.E93.B.1078
[42] G. Lencse, “Estimation of the port number consumption of web browsing”, *IEICE Transactions on Communications*, vol. E98-B, no. 8. pp. 1580–1588. DOI: 10.1587/transcom.E98.B.1580
[43] C. Bao, X. Li, Y. Zhai, W. Shang, “dIVI: Dual-stateless IPv4/IPv6 translation”, (long time dead but currently active) Internet Draft, https://tools.ietf.org/html/draft-xli-behave-divi-07
[44] E. Nordmark, R. Gilligan, “Basic transition mechanisms for IPv6 hosts and routers”, IETF RFC 4213, October 2005.
[45] A. Conta, S. Deering, “Generic packet tunneling in IPv6 specification”, IETF RFC 2473, December 1998.
[46] B. Carpenter, K. Moore, “Connection of IPv6 domains via IPv4 clouds”, IETF RFC 3056, February 2001.
[47] C. Huitema, “Teredo: Tunneling IPv6 over UDP through network address translations (NATs)”, IETF RFC 4380, February 2006.
[48] W. Townsley, O. Troan, “IPv6 Rapid Deployment on IPv4 Infrastructures (6rd) -- Protocol Specification”, IETF RFC 5969, August 2010.
[49] E. Aben, “6to4 – How bad is it really?”, RIPE NCC, [Online]. Available: https://labs.ripe.net/Members/emileaben/6to4-how-bad-is-it-really
[50] F. Templin, T. Gleeson, D. Thaler, “Intra-site automatic tunnel addressing protocol (ISATAP)”, IETF RFC 5214, March 2008.
[51] O. Troan (ed), W. Dec, X. Li, C. Bao, S. Matsushima, T. Murakami, T. Taylor (ed), “Mapping of address and port with encapsulation (MAP-E)”, IETF RFC 7597, July 2015.
[52] A. Durand, R. Droms, J. Woodyatt, Y. Lee, “Dual-stack lite broadband deployments following IPv4 exhaustion”, IETF RFC 6333, August 2011.
[53] Y. Lee, R. Maglione, C. Williams, C. Jacquenet, M. Boucadair, “Deployment considerations for dual-stack lite”, IETF RFC 6908, March 2013.
[54] Y. Cui, J. Wu, P. Wu, O. Vautrin, Y. Lee, “Public IPv4-over-IPv6 access network”, IETF RFC 7040, November 2013.
[55] Y. Cui, Q. Sun, M. Boucadair, T. Tsou, Y. Lee, I. Farrer, “Lightweight 4over6: An extension to the dual-stack lite architecture”, IETF RFC 7596, July 2015.
[56] N. Matsuhira, “Stateless automatic IPv4 over IPv6 encapsulation / decapsulation technology: Specification”, expired Internet Draft, https://tools.ietf.org/html/draft-matsuhira-sa46t-spec-11
[57] J. Massar, “AYIYA: Anything in anything”, expired Internet Draft, https://tools.ietf.org/html/draft-massar-v6ops-ayiya-02
[58] G. Lencse, Sz. Szilágyi, F. Fejes, M. Georgescu, “MPT network layer multipath library”, active Internet Draft, June 30, 2017, https://tools.ietf.org/html/draft-lencse-tsvwg-mpt-00
[59] B. Almási, G. Lencse, Sz. Szilágyi, “Investigating the multipath extension of the GRE in UDP technology”, *Computer Communications*, vol. 103, no. 1, pp. 29–38, May 1, 2017, DOI: 10.1016/j.comcom.2017.02.002
[60] Á. Kovács, “Comparing the aggregation capability of the MPT communications library and multipath TCP”, in: *Proc. CogInfoCom 2016*, Wroclaw, Poland, Oct. 16-18, 2016, pp. 157–162, DOI: 10.1109/CogInfoCom.2016.7804542
[61] F. Fejes, R. Katona, and L. Püsök, “Multipath strategies and solutions in multihomed mobile environments”, in: *Proc. CogInfoCom 2016*, Wroclaw, Poland, Oct. 16-18, 2016, pp. 79–84, DOI: 10.1109/CogInfoCom.2016.7804529
[62] F. Fejes, S. Rácz, and G. Szabó, “Application agnostic QoE triggered multipath switching for Android devices”, In: *Proc. IEEE ICC 2017*, Paris, France, May 21-25, 2017, pp. 1585–1591.
[63] S. Hernan, S. Lambert, T. Ostwald, and A. Shostack, “Threat modeling: uncover security design flaws using the STRIDE approach” MSDN Magazine, Nov. 2006, pp. 68–75.
[64] G. Lencse, Y. Kadobayashi, “Methodology for the identification of potential security issues of different IPv6 transition technologies”, will be available from: http://www.hit.bme.hu/~lencse/publications/