# IPv4/IPv6 Transition Using DNS64/NAT64: Deployment Issues

Enis Hodzic
BH Telecom.d.o.o
Sarajevo, Bosnia & Herzegovina
enis.hodzic@bhtelecom.ba

Sasa Mrdovic
Faculty of Electrical Engineering
University of Sarajevo
Sarajevo, Bosnia & Herzegovina
sasa.mrdovic@etf.unsa.ba

***Abstract*—IPv4 address space is almost exhausted. Usage of IPv6 address by client end hosts is limited due to small percentage of domain names that have IPv6 address. This paper presents practical testing in ISP that gives its users IPv6 addresses and provides them transparent access to both IPv4 and IPv6 Internet locations. DNS64/NAT64 translation mechanism is used for this purpose. Tests measure resource requirements on ISP side and effects on client experience. Results show that additional DNS64 processing causes no visible impact on DNS server CPU load. There is requirement for NAT64 device at ISP on path between IPv6 users and IPv4 Internet. Test results show that memory requirements for this device are small and achievable with standard hardware devices used by ISPs. Measured increase in RTT from IPv6 clients to IPv4 Internet is less than 2%. Conclusion is that DNS64/NAT64 translation system is viable solution for ISP.**

***Index Terms*—IPv6; IPv4/v6 transition; DNS64; NAT64, ISP implementation**

## I. Introduction

Internet Service Providers (ISP) are running out of IPv4 addresses to give to their client. ISPs have enough IPv6 addresses but they are not very useful. IPv6 clients can only visit IPv6 sites. Percentage of all domain names that have IPv6 address records is only around 2% [1]. This means that the rest of the Internet is out of reach for client that have IPv6 addresses only. It would be very useful for ISPs if they could provide their IPv6 clients with access to all Internet locations IPv4 and IPv6. Providing such access requires additional processing of packets at ISP. This additional processing increases operating costs. ISPs should be able to evaluate this cost. The cost depends on number of factors. Number of users is one of them. The other important factor is IPv4/v6 translation mechanism used. Additional processing per packet or session depends directly on this mechanism.

This paper presents test implementation of ISP with IPv6 only clients that have access to IPv4 as well as IPv6 sites. The implementation is based on DNS64/NAT64 combination. All tests are performed with a real client of a real ISP. The tests are designed to evaluate impact that this setup has on ISP and clients. Required ISP resources for additional processing and increase in response time for clients are measured.

The rest of the paper is organized as follows. Related work is addressed in section 2. Section 3 explains DNS64/NAT64 translation. Experiment environment is presented in section 4. Section 5 presents conducted tests and analyzes their results. Conclusion and discussion on directions for future research work are in section 6.

## II. Related Work

Basic transition mechanisms for IPv4/IPv6 have been proposed by IETF [2]. Two mechanisms are specified: Dual IP layer and IPv6 over IPv4 tunneling. The idea was to enable IPv6 on networks and use dual stack routers to enable tunneling between IPv6 islands. This left open a question of communication between IPv6 and IPv4 hosts and networks. One approach to this issue was NAT-PT [3] based on stateless IP/ICMP translator (SIIT) [4]. Due to various issues this approach was abandoned [5]. Last year, IETF readdressed this issue with RFC 6144 [6]. It defines new framework for IPv4/IPv6 translation. Various scenarios of connecting IPv4 and IPv6 networks are considered in this RFC. The first scenario, an IPv6 network to the IPv4 Internet, is the one discussed in this paper. There are several components that enable implemnetation of this scenario. One component is DNS64 [7]. It provides mechanism for synthesizing AAAA records (IPv6 addresses) from A records (IPv4 addresses). Other components deal with various aspects of translation: address [8], IP and ICMP [9], maintaining state [10] and application level gateways for other application layer protocols.

Practical implementations based on above approach, usually called DNS64/NAT64, have been introduced recently. There are proprietary implementations that come from big companies. Microsoft implements DNS64/NAT64 with Forefront Unified Access Gateway (UAG) DirectAccess [11]. Recent versions of Cisco IOS, from XE 3.4S for ASR 1000 series routers and XR 4.1.2 for Carrier Routing System (CRS-1) support stateful NAT64 [12]. Juniper Networks operating system Junos implements stateful NAT64 in Services Physical Interface Card (PIC) i Services Dense Port Concentrator (DPC) [13]. Another popular DNS64/NAT64 implementation Brocade ServerIron ADX and Secure64 DNS Cache Platforms comes from Brocade Communications Systems [14].

Open source community has several DNS64 and NAT64 implementations. ISC Bind supports DNS64 from recent version 9.8 [15]. For previous Bind versions, synthetization of AAAA records from A records was possible with support of trick-or-treat-daemon [16]. Another DNS resolver Unbound also supports DNS64 in recent versions [17]. Stateless NAT64 implementation for Linux is Tayga [18]. Stateful NAT64 implementation is provided by Linuxnat64 project [19]. Ecdysis project developed open source implementations of DNS64 and stateful NAT64 [20].

There were several practical experiments with DNS64/NAT64 as translation mechanism to support IPv6 only networks access to IPv4 Internet. IETF RFC6586 [21] describes experiences from experiment with IPv6 only network, with access to the IPv4 only parts of the Internet via a NAT64 device with DNS64 support. Three IETF drafts also present experiences and issues with IPv6 only networks and describe issues that were resolved, that can and those that cannot be resolved [22], [23], [24]. Paper [25] describes similar experience. None of the mentioned documents considers performance issues.

Only two papers were found that deal with NAT64 performance issues. Paper [26] tested NAT64 performance versus NAT44. Their testing showed that NAT64 has lower CPU and memory utilization. Authors of [27] compared round trip times of three open source implementation of IPv4/IPv6 translators: NAT-PT, NAT64 and HTTP proxy. Their results suggest that NAT64 is better overall solution, but that dual stack HTTP proxy might be reasonable alternative for HTTP only traffic.

## III. DNS64/NAT64

DNS64 and NAT64 are explained in detail in their respective RFCs [7], [10]. Here, the protocols will be briefly explained with accent on performance issues that might affect ISPs using them to provide access to IPv4 Internet for their IPv6 only clients.

DNS64/NAT64 system enables IPv6 only hosts to access IPv4 servers in the following way. IPv6 host asks DNS, in this case DNS64, for IP address for the domain name of server it is trying to connect to. Since the server is IPv4 only and does not have IPv6 address, DNS64 embeds server's IPv4 address into an IPv6 address by using predefined prefix and the actual IPv4 address. IPv6 host uses this IPv6 address to initialize connection to the server. The packet, host sends, goes through NAT64. NAT64 recognizes the predefined prefix, extracts servers IPv4 address from destination IPv6 address in packet. From there it behaves like any NAT. It contacts destination server using extracted IPv4 address and creates translation state that maps this connection to IPv4 server with connection from IPv6 client that initiated it.

### A. DNS64

DNS64 performs synthetization of AAAA records based on an A record. DNS64 functionality works on all DNS servers based on DNS RFCs [28], [29]. DNS64 also supports the translation of multiple IPv6 prefixes. This allows that different ranges of IPv4 addresses can be mapped to different IPv6 prefixes. These separated mappings can be used to achieve load balancing between multiple NAT64 devices.

There are two possible options for the use of prefixes in translation, Well-Known Prefix (WKP) and Network-Specific Prefix (NSP). WKP is always 64:ff9b::/96. NSP, on the other hand, can have a variable length prefix, and prefix depends on the organization that uses this type of translation.

DNS64 can initiate two queries simultaneously, one for the A record and one for the AAAA record. These parallel queries can reduce delays in cases where the corresponding AAAA record is not found. Normally, DNS64 first sends a query for an AAAA records and in the case when that record is not found DNS64 than sends another query for an A record, which is then used for the synthetization of an AAAA record. If the required record is available locally, as in the case of the authoritative DNS, parallel queries are not required. Theoretically, the use of parallel queries can half the time delays compared to the sequential queries.

Additional processing that DNS64 has to perform, compared to standard IPv4 only DNS, is creation of AAAA records from A records returned by query. This processing requires additional resources. These resources are measured in this paper.

### B. Stateful NAT64

NAT64 works like NAT44 with one important exception. It performs address translation of IPv6 addresses into IPv4 addresses and vice versa. Because it is extremely similar to the NAT44, NAT64 also has the same problems, like problems with end-to-end communication. NAT64 is defined in two RFC's. RFC6145 [9] defines the Stateless IP / ICMP Translation (SIIT) algorithm, which in fact, represents a stateless NAT64, and RFC6146 [10] defines a stateful NAT64.

Simple setting of NAT64 device can be visualized as a network device (router) with at least two interfaces. One interface is connected to the IPv4 network, while the other is connected to the IPv6 network. The network is configured in the manner that the packets from the IPv6 network go to the IPv4 network through the router. Router performs all necessary translations needed for the transfer of packets from the IPv6 network to IPv4 network and vice versa.

In stateless NAT64 translation state is not kept, which means that every IPv6 user requires a unique IPv4 address. Since there is a lack of IPv4 addresses, this NAT64 mode is not a good solution for ISPs that want to solve it by giving clients IPv6 address.

Stateful NAT64 multiplexes many IPv6 devices into a single IPv4 address. It can be assumed that this technology will be used mainly where IPv6-only networks and clients (ie. mobile handsets, IPv6 only wireless, etc...) need access to the IPv4 internet and its services.

The main difference between stateful and stateless NAT64 is the elimination of the algorithmic binding between the IPv6 address and the IPv4 address. Instead, stateful NAT64 creates translation state for each session. NAT64 only supports IPv6 initiated sessions. Unlike stateless NAT64, stateful NAT64 does not consume an IPv4 address for each IPv6 node that wants to communicate to the IPv4 Internet. In practice, it means that a number of IPv6-only users consume only one IPv4 address in a similar manner to the way IPv4-to-IPv4

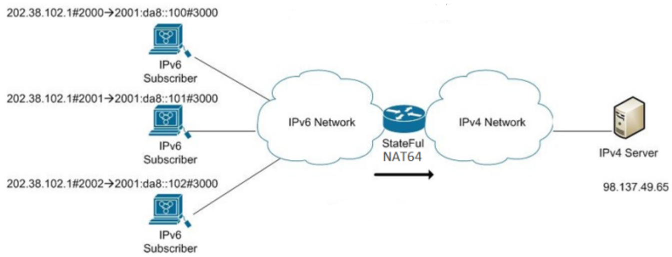


Figure 1. Stateful NAT64

network address and port translation works. This works well if the request for a connection is initiated from IPv6 towards the IPv4 Internet. If an IPv4-only device wants to speak to an IPv6-only server, manual configuration of the translation slot will be required, making this mechanism less attractive to provide IPv6 services towards the IPv4 Internet. Fig. 1 presents the implementation of stateful NAT64.

Traditional ISPs that provide their IPv4 clients with public IP addresses access to IPv4 Internet do not need to have NAT device between clients and IPv4 Internet. This means that providing IPv6 only clients access to IPv4 Internet would require additional NAT64 device and additional processing. This paper tests resource requirements for such a device and measures its influence on round trip time from clients to servers.

## IV. Experiment Environment

Block diagram, as well as IP addressing of DNS64/NAT64 test platform, are shown in Fig. 2. The experimental environment was mostly virtualized and consisted of a NAT64 translator, DNS64 server and DHCPv6 servers – Linux Ubuntu 10.04 LTS with Ecdysis, BIND9 and wide-dhcpv6 servers, and few IPv6-only or dual-stack clients with Ubuntu Linux and Windows operating systems. Cisco Networks equipment was used for routing purposes.

Software used for the implementation and testing of DNS64/NA64:

- VMware Workstation v8.0 for creation of virtual machines [30],
- Ecdysis NAT64 for NAT64 [31],
- BIND9 version 9.8.1 for DNS64 [15],
- wide-dhcpv6-server for DHCPv6 [32].

Linux server is a virtual machine (named dns64nat64) which is assigned one CPU core, 1.5 GB RAM and 8 GB of hard disk space. The amount of resources allocated to the virtual machine dns64nat64 was more than enough. In this way, virtual machines were not limiting the functionality of DNS64/NAT64. IPv6 WKP prefix was used as a DNS64 prefix. WKP prefix is defined in [8] as 64: ff9b:: / 96.

```
root@dns64nat64:/etc# cat named.conf
options {
directory "/etc/named";
dns64 64:ff9b::/96 {
clients { any; };
};
listen-on-v6 { any; };
};
```

NAT64 prefix used the same IPv6 prefix WKP 64: ff9b :: / 96. It should be noted that DNS64/NAT64 system can work only if NAT64 and DNS64 use the same prefix.

```
root@dns64nat64:/# cat nat64-config.sh
#!/bin/bash
IPV4_ADDR="10.100.13.65"
PREFIX_ADDR="64:ff9b::"
PREFIX_LEN="96"
```

DHCPv6 server was configured to operate as a stateless, and its only function was to send IPv6 address of DNS64 server to the hosts.

```
root@dns64nat64:/etc/wide-dhcpv6# cat
dhcp6s.conf
option domain-name-servers
2a02:27b0:2:13::65;
```

The complete functionality of DNS64/NAT64 system can be presented using the case where an IPv6-only host wants to access a Web site that is located on IPv4 HTTP server. Fig. 3 shows such an IPv6 host with IPv4 Facebook Web server communication over DNS64/NAT64 test platform with the following steps:

1) DNS query from IPv6-only client to DNS64 server: AAAA record for www.facebook.com.
2) DNS query from DNS64 server to authoritative DNS server: AAAA record for www.facebook.com.
3) DNS response from an authoritative DNS server to DNS64 server: no AAAA records for www.facebook.com.
4) DNS query from DNS64 server to authoritative DNS server: A record for www.facebook.com.
5) DNS response from an authoritative DNS server to DNS64 server: A record of www.facebook.com is 69.171.228.11.
6) DNS64 server synthesizes AAAA records 64:ff9b::45ab:e40b and sends the DNS64 answer: IPv6 address of www.facebook.com is 64: ff9b :: 45ab: e40b.
7) HTTP GET packet from an IPv6 client to the HTTP server facebook.com routed to NAT64
8) NAT64 translates the header of HTTP GET packet and forwards it to the HTTP server using the IPv4 address.
9) HTTP server sends the page content using IPv4 address.
10) NAT64 translates the content of HTTP packet header and forwards it to the IPv6 client.

## V. Experiment and Results

Using described test platform three test were run. Aim of two tests was to help evaluation of additional resources required at ISP side if it wants to have IPv6 only clients. The third test compared round trip times for IPv4 clients and IPv6 clients accessing the same IPv4 locations through DNS64/NAT64 system.

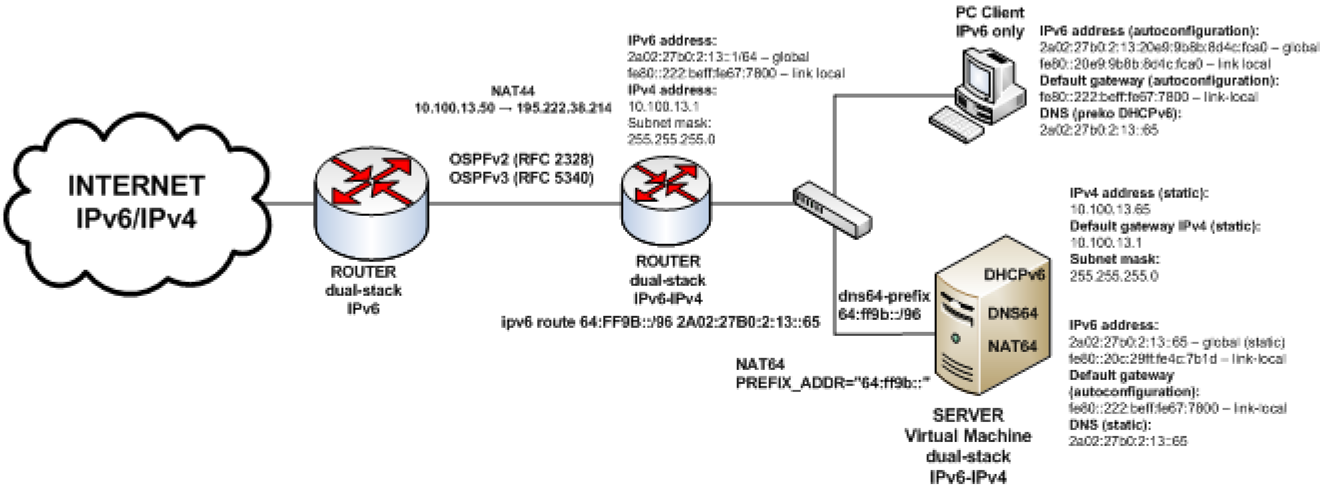


Figure 2. DNS64/NAT64 test platform

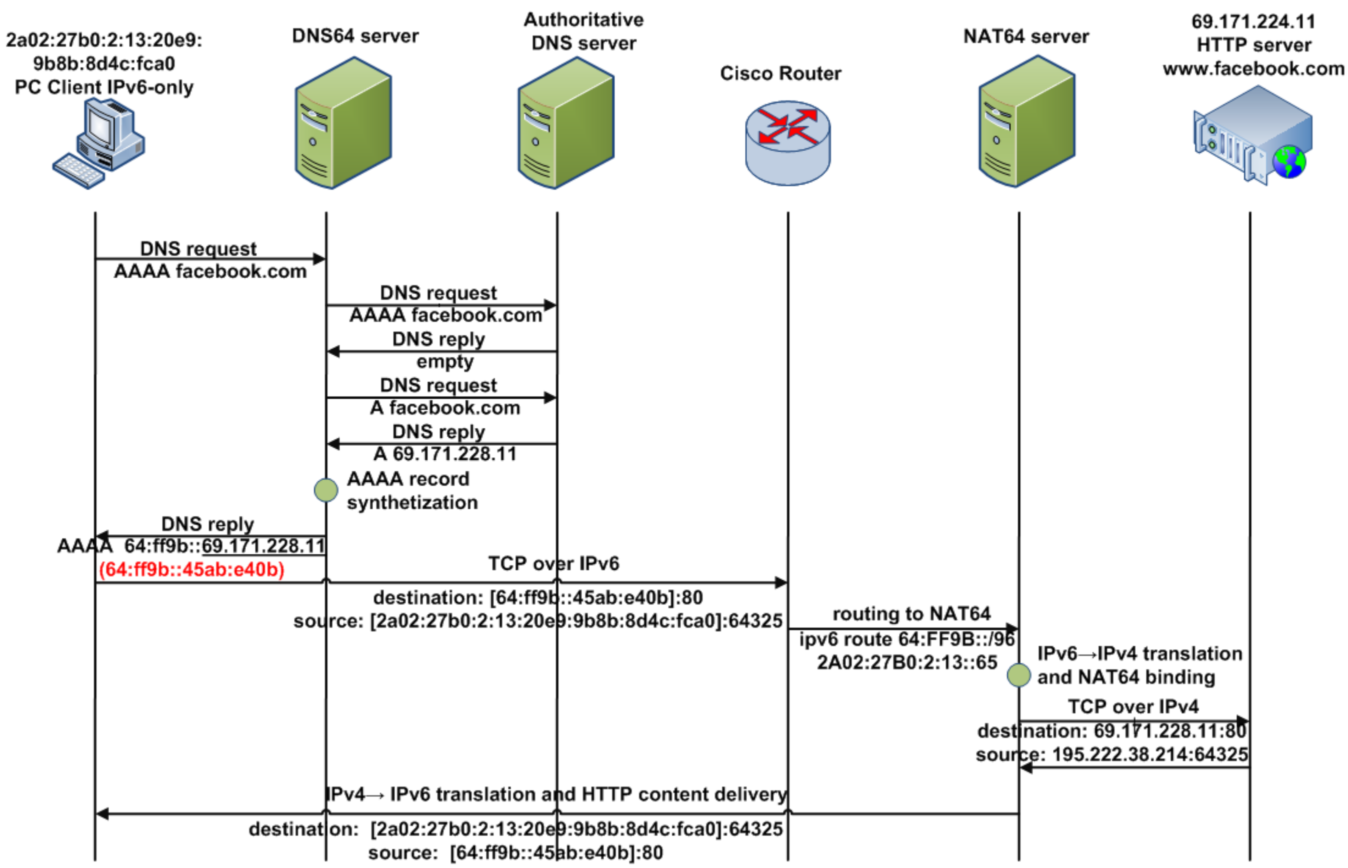


Figure 3. IPv6 host - IPv4 server communication over DNS64/NAT64 system

Table I
COMPARISON OF DNS64 CPU LOAD

| DNS requests [requests/s] | CPU Load [%] | |
|---|---|---|
| | A record | AAAA record |
| 200 | 7 | 8 |
| 800 | 21 | 20 |
| 1200 | 30 | 34 |
| 2000 | 49 | 45 |
| 2300 | 59 | 55 |
| 3200 | 78 | 75 |
| 3800 | 85 | 87 |

### *A. DNS64 CPU test*

This test measured CPU usage for DNS server. In the first test server was queried for A record for domain name “ekupon.ba”. This domain does not have IPv6 address and therefore no AAAA record. In the second test the same DNS64 server was queried for AAAA record for the same domain name. In this case DNS64 server had to perform additional processing in order to synthesize AAAA record from A record. During both tests number of requests per second was increased until CPU usage was 90%. This test was designed to measure additional CPU load caused by creation of AAAA records from A records. This would be a main difference on DNS server in case it has to serve IPv6 only clients. Test results are given in Table I.

Results show that creation of AAAA records for domain names that have only A records does not have visible impact on CPU load of DNS server. CPU load for both tests was similar.

### *B. NAT64 RAM test*

This test measured RAM usage for NAT64 device. Since NAT64 device is a virtual machine in this setup it uses memory even without performing NAT64 translation. Test compared RAM usage when there were no NAT64 translations against period when there were 103 active NAT64 connections. NAT64 connections were created with HTTP GET request for default web page at site "source.ba". This site does not have IPv6 address and no DNS AAAA record. Requests were generated with rate of 100 per second. Test results are presented in Fig 4.

Results show that minimal memory load, when there were no NAT64 connections, was 433.25 MB. Maximal memory load for 103 NAT64 connections per second was 571.26 MB. Difference is 138 MB. If this difference is divided by 103 connections RAM cost per connection comes to approximately 13,5 MB. As it was concluded in [26] NAT64 memory load is comparable to that of NAT44.

### *C. Round trip time test*

For this test two measurements of RTT to two IPv4 hosts were made. The first measurement included comparison of ping to “bbc.com”. First IPv4 host pinged “bbc.com” on IP address 212.58.241.131. Then IPv6 only host pinged the same “bbc.com” through DNS64/NAT64 system on its synthesized IPv6 address 64:ff9b::d43a:f183. Average RTT for the first, IPv4, ping was 46.68 ms, and average time for the second ping was 47.53 ms. Standard deviations were 0.19 and 0.71 respectively. Average difference was 0.85 ms which is less than 2%.

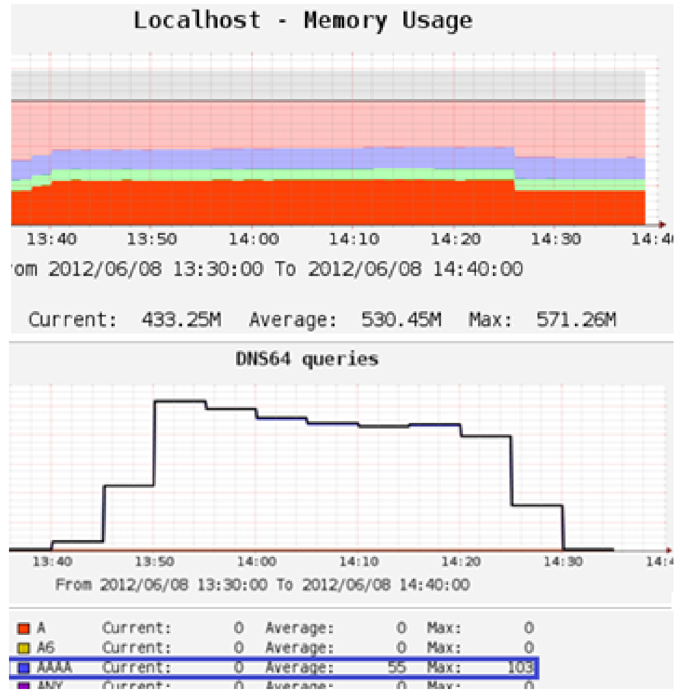


Figure 4. Stateful NAT64 RAM usage

The second measurement measured RTT to ”juniper.net” from IPv4 and IPv6 host. Again IPv4 ping went to IPv4 address 207.17.137.239 and IPv6 ping went through DNS64/NAT64 system to synthesized IPv6 address 64:ff9b::cf11:89ef. This time average RTT for IPv4 was 203.7 ms and for IPv6 was 205.9 ms. Standard deviations were 0.7 and 0.1 ms. Average difference was 2.2 ms which approximately 1%. Graph with results for this test is given in Fig. 5.

Theses tests show that from client prospective round trip times increase less than 2%

## VI. CONCLUSION AND FUTURE WORK

IPv4/v6 transition using DNS64/NAT64 translation mechanism is not complicated or expensive. ISPs that do not have enough IPv4 addresses to give to all their client can give them IPv6 addresses and still provide the same service. DNS64/NAT64 translation mechanism enables IPv6 clients to access IPv4 Internet sites transparently. ISP needs to provide DNS64 and NAT64 services. DNS64 creates IPv6 address from IPv4 address for locations that do not have IPv6 address. This is the only additional work that DNS has to perform in this setup. Test results show that this additional work does not have visible impact on DNS server CPU load. NAT64 performs network address translation from IPv6 client address and port to IPv4 NAT64 outside address and port. It has been shown previously that NAT64 memory and CPU load is not

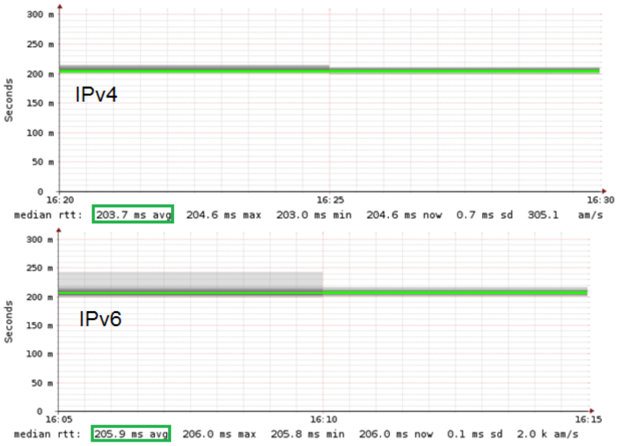


Figure 5. Ping RTT

bigger than NAT. Our test show NAT64 memory load of 13.5 MB per connection. From (IPv6) client perspective there is negligible increase in RTT time for requests to IPv4 locations that go through DNS64/NAT64 system when compared with direct request to IPv6 locations. This increase was measured to be less than 2%.

It must be stated that things will only get better. As more and more sites are reachable through IPv6 address there will be less need for traffic to go through DNS64/NAT64 translation. There will be decrease in resource requirements on ISP. IPv6 client traffic has to go through DNS64/NAT64 system only when it communicates with locations that do not have IPv6 address. Communication with IPv6 locations is direct, it does not have to go through NAT64 and there is no need to create IPv6 address for such locations. Although percentage of domain names that have IPv6 address is now little over 2% [1], it is increasing. It rose half percent during the period it took us to finish this paper. Since World IPv6 Launch Day, June 6, 2012, a number of sites have IPv6 address that did not have them previously. One of such sites is Facebook that was used in our presentation of DNS64/NAT64 functionality.

## References


[1] M. Leber, "Global IPv6 Deployment Progress Report.", Hurricane Electric, [Online]. Available: http://bgp.he.net/ipv6-progress-report.cgi. [Accessed: 18-Jun-2012].

[2] R. E. Gilligan and E. Nordmark, "Basic Transition Mechanisms for IPv6 Hosts and Routers." [Online]. Available: https://tools.ietf.org/html/rfc4213. [Accessed: 18-Jun-2012].

[3] G. T. <george.tsirtsis@bt.com>, "Network Address Translation - Protocol Translation (NAT-PT)." [Online]. Available: https://tools.ietf.org/html/rfc2766. [Accessed: 18-Jun-2012].

[4] E. N. <nordmark@sun.com>, "Stateless IP/ICMP Translation Algorithm (SIIT)." [Online]. Available: https://tools.ietf.org/html/rfc2765. [Accessed: 18-Jun-2012].

[5] E. B. Davies and C. Aoun, "Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status." [Online]. Available: https://tools.ietf.org/html/rfc4966. [Accessed: 18-Jun-2012].

[6] C. Bao, X. Li, K. Yin, and F. Baker, "Framework for IPv4/IPv6 Translation." [Online]. Available: https://tools.ietf.org/html/rfc6144. [Accessed: 18-Jun-2012].

[7] P. Matthews, M. Bagnulo, A. Sullivan, and I. van Beijnum, "DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers." [Online]. Available: https://tools.ietf.org/html/rfc6147. [Accessed: 11-Jun-2012].

[8] M. Boucadair, C. Bao, X. Li, C. Huitema, and M. Bagnulo, "IPv6 Addressing of IPv4/IPv6 Translators." [Online]. Available: https://tools.ietf.org/html/rfc6052. [Accessed: 18-Jun-2012].

[9] C. Bao, X. Li, and F. Baker, "IP/ICMP Translation Algorithm." [Online]. Available: https://tools.ietf.org/html/rfc6145. [Accessed: 18-Jun-2012].

[10] P. Matthews, M. Bagnulo, and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers." [Online]. Available: http://rsync.tools.ietf.org/html/rfc6146. [Accessed: 11-Jun-2012].

[11] T. Kopczynski, "DirectAccess and UAG DirectAccess - Deployment Guide." Microsoft, 2010.

[12] I. Kakonyi, "Cisco Knowledge Network: NAT64 Theory and Applications in Service Provider Networks." Cisco Systems, 2011.

[13] Juniper Networks, "Configuring Stateful NAT64 for Handling IPv4 Address Depletion." Juniper Networks, 2011.

[14] Brocade, "Deploying NAT64 and DNS 64 with the Brocade ServerIron ADX and Secure64 DNS Cache Platforms." Brocade Communications Systems, 2011.

[15] "BIND | Internet Systems Consortium." [Online]. Available: https://www.isc.org/software/bind. [Accessed: 19-Jun-2012].

[16] "Trick or Treat DNS proxy." [Online]. Available: http://www.dillema.net/software/totd.html. [Accessed: 19-Jun-2012].

[17] "Unbound." [Online]. Available: http://unbound.net/. [Accessed: 19-Jun-2012].

[18] "TAYGA - NAT64 for Linux." [Online]. Available: http://www.litech.org/tayga/. [Accessed: 19-Jun-2012].

[19] "linuxnat64 Project Top Page - SourceForge.JP." [Online]. Available: http://en.sourceforge.jp/projects/sfnet_linuxnat64/. [Accessed: 19-Jun-2012].

[20] S. Perreault, J.-P. Dionne, and M. Blanchet, "Ecdysis: Open-Source DNS64 and NAT64." Viagenie, 2010.

[21] J. Arkko and A. Keranen, "Experiences from an IPv6-Only Network." [Online]. Available: https://tools.ietf.org/html/rfc6586. [Accessed: 19-Jun-2012].

[22] G. Chen, Z. Cao, C. Byrne, and Q. Niu, "NAT64 Operational Experiences." [Online]. Available: http://tools.ietf.org/html/draft-chen-v6ops-nat64-experience-01. [Accessed: 19-Jun-2012].

[23] S. Sivakumar, M. Boucadair, R. Penno, and T. Saxena, "Analysis of Stateful 64 Translation." [Online]. Available: http://tools.ietf.org/html/draft-ietf-behave-64-analysis-07. [Accessed: 19-Jun-2012].

[24] R. Hiromi, O. Nakamura, H. Hazeyama, and T. Ishihara, "Experiences from IPv6-Only Networks with Transition Technologies in the WIDE Camp Spring 2012." [Online]. Available: http://tools.ietf.org/html/draft-hazeyama-widecamp-ipv6-only-experience-01. [Accessed: 19-Jun-2012].

[25] H. Hazeyama, Y. Yamagishi, Y. Ueno, T. Yokoishi, H. Sato, and H. Ishibashi, "How much can we survive on an IPv6 network?: experience on the IPv6 only connectivity with NAT64/DNS64 at WIDE camp 2011 Autumn," in Proceedings of the 7th Asian Internet Engineering Conference, New York, NY, USA, 2011, pp. 144–151.

[26] K. J. O. Llanto and W. E. S. Yu, "Performance of NAT64 versus NAT44 in the Context of IPv6 Migration," Proceedings of the International MultiConference of Engineers and Computer Scientists, vol. 1, 2012.

[27] S. Yu and B. Carpenter, "Measuring IPv4-IPv6 translation techniques," http://www.cs.auckland.ac.nz/~brian/IPv4-IPv6coexistenceTechnique-TR.pdf, 2012.

[28] P. V. Mockapetris, "Domain names - concepts and facilities." [Online]. Available: https://tools.ietf.org/html/rfc1034. [Accessed: 20-Jun-2012].

[29] P. V. Mockapetris, "Domain names - implementation and specification." [Online]. Available: https://tools.ietf.org/html/rfc1035. [Accessed: 20-Jun-2012].

[30] "Download VMware Workstation 8.0." [Online]. Available: http://www.vmware.com/go/downloadworkstation& [Accessed: 20-Jun-2012].

[31] "Ecdysis: open-source nat64." [Online]. Available: http://ecdysis.viagenie.ca/. [Accessed: 20-Jun-2012].

[32] "WIDE-DHCPv6 | Free System Administration software downloads at SourceForge.net." [Online]. Available: http://sourceforge.net/projects/wide-dhcpv6/. [Accessed: 20-Jun-2012].