WIDE Technical-Report in 2013

IPv6 deployment activites in WIDE Project

wide-tr-live-with-ipv6-wg-ipv6-depolyment-in-japan-01.pdf

WIDE Project : http://www.wide.ad.jp/

Title:	IPv6 deployment activites in WIDE Project
Author(s):	Hiroaki Hazeyama, Akira Kato, Osamu Nakamura, Satoshi Uda
Date:	2013-03-06

IPv6 deployment activites in WIDE Project

Hiroaki Hazeyama, Akira Kato, Osamu Nakamura, Satoshi Uda

March 6, 2013

Enclosed documentation is submitted as a part of joint research report of the WIDE Project and Korea Institute of Science and Technology Information (KISTI)[1], which is a research institute in Daejeon, Korea operating the Korean research and educatiion network named as KREONET in 2012. The report was written based on presentations given in the IPv6 workshop in conjunction with KREONET member meeting held in October 2012 in Busan, Korea.

WIDE Project acknowledges KISTI for the joint work on IPv6 deployment and for permission to publish the report for a reference to the global Internet community.

1 Japanese IPv6 Deployment Status

1.1 Organizations

There are several meeting groups and organizations related with IPv6 activities in Japan. Oldest and very active organization is IPv6 Promotion Council which was established at October 2000th, invited Prof. Jun Murai as President Chair. 18 companies and universities as initial members and MIAC (Ministry of Internal Affairs and Communications which was known as MIC, the Ministry of Information Communications) and JPNIC as observers, started IPv6 promotion activities. Current members of the council are 15 board of directors, 42 corporate members, over 50 individual members, and 4 Sponsors. The activities of IPv6 promotion council are done by working groups as shown below.

- Basic Strategy Steering Group

[1] `http://en.kisti.re.kr/`

- Certification WG
  Core, IPsec, MIPv6 MLDv2 SIP DHCPv6 Sub Working groups
- IPv4/IPv6 Coexistence WG
  Service Transition, IPv6 Home Router, v6fix, IPv6 application Sub Working groups
- FMC v6 Platform WG
- Digital Information Appliance v6 Platform WG
- Security WG
- Business Testbed WG
- Business Learning WG
- Business Exchange WG

Task Force on IPv4 Address Exhaustion, Japan is another active group. This task force was established at 2008, to make coordinated actions among all stakeholder groups from teleco and Internet industries and also governments, to overcome the crisis of the IPv4 address exhaustion in a coordinated manner in four aspects of solution of the issues (technique, operation and management), enlightenment and publicity, education and progress management.

Task Force members

- Internet Associations
  - IPv6 Promotion Council, Japan
  - Internet Association of Japan
  - Distributed IX Project
  - Japan Internet Providers Association
  - JPCERT Coordination Center
  - Japan Network Information Center (JPNIC)
  - Japan Network Operations Groups (JANOG)
  - Japan Network Security Association (JNSA)
  - Japan UNIX Users Society (jus)
  - Japan Registry Services (JPRS)

  - WIDE Project
- Telecommunications Associations
  - Telecom Service Association
  - Telecommunications Carriers Association
  - Japan Cable and Telecommunications Association
  - Japan Data Communications Association
  - Japan Approval Institute for Telecommunications Equipment (JATE)
- Vendors Associations
  - Communications and Information network Association of Japan (CIAJ)

Those groups were established for targeting about IPv4/v6. Other groups such as JANOG, JIPA, IAJ and so on, have several internal working groups for discussing about IPv6 related issues.

1.2 Strategy for IPv6 deployment

The strategy for IPv6 deployment has been discussing at IPv6 promotion council as open and multi-stake holder fashion. The consensus of the strategy is

1. enabling IPv6 at ISP backbone
2. development IPv6 enabling devices such as PC
3. deployment IPv6 services to the customers
4. enabling IPv6 services on CSP

We promoted and push 1) and 2) first since year of 2000. WIDE project developed IPv6 protocol stack as reference code on BSD UNIX as KAME project and Linux as USAGI project.

Those reference codes helped enabling IPv6 on many PC-based products. We also have been discussed with many network equipment vendors such as CISCO, Juniper, Foundry, Alaxala and so on, for IPv6 protocols in detail at IETF meetings as well as number of private discussions. We believe that those activities were successful.

We believe that basic preparation for IPv6 deployment has been completed by now. Microsoft Windows and Apple Mac OS supported IPv6.

Recently Android and IOS for iPhone also support IPv6 by default. Enabling IPv6 on ISP backbone was not so difficult. The equipments for backbone networks have been replaced in several years with regular replacement cycle. Those equipments support IPv6 by default. In early 2000's, major IPSs in Japan, such as NTT, IIJ, SONET, BIGLOBE, and so on, started experimental IPv6 operation. The serious issues are 3) and 4) above. How to deploy the IPv6 services to the customers? CSPs are hesitating to start IPv6 services as there are few IPv6 users.

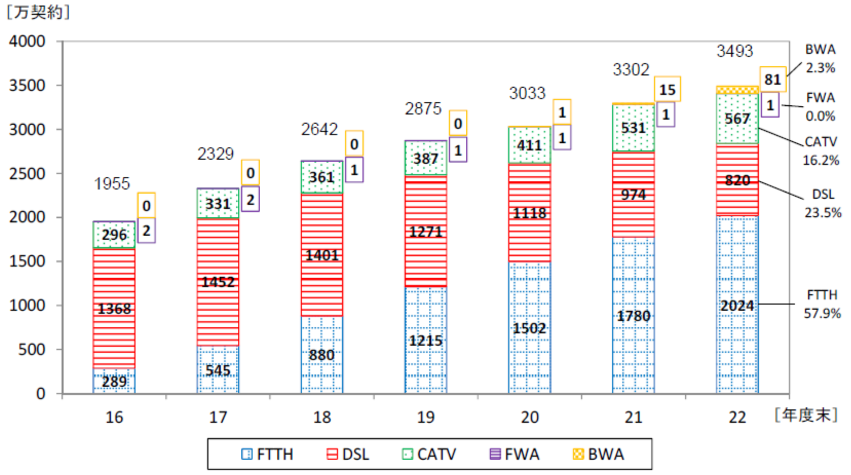


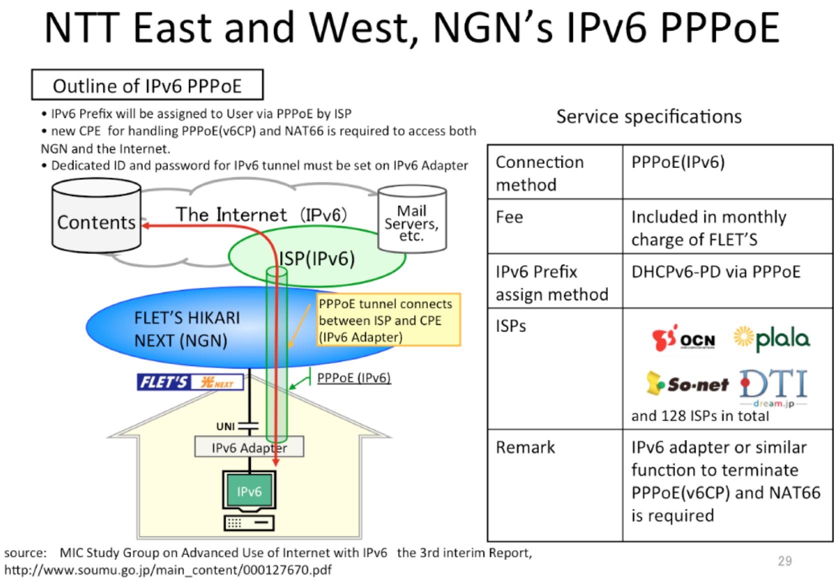
図 1-4　ブロードバンド加入者数の推移（総務省作成）

Figure 1: Broadband Subscribers Growth in Japan (by MIC)

The broadband subscriber growth in Japan is shown in Figure 1. Because a half of broadband subscribers use FTTH (Fiber To The Home). As the number of subscribers of ADSL is decreasing, and as the number of subscribers of CATV is increasing, we have been focusing on FTTH and CATV mainly.

In order to enable IPv6 to provide IPv4/v6 dual stack services on FTTH, there are two cases. One is that ISP provides fiber installation and Internet services as a single entity. Another is that the companies provide the fiber installation and other companies provide Internet service. First case is simple. It is possible to enables IPv6 services by corresponding company decision. In Japan, KDDI is this case. KDDI has been providing Internet services over their fiber system. The home routers are owned and operated by KDDI. In the last year, KDDI decided to start the IPv4/v6 dual stack services for every customer without additional fee and special operations by customer. KDDI updated firmware on the home routers remotely and started IPv6 services as dual stack operation. Now every KDDI customer has access to the IPv4 Internet as well as IPv6 Internet in a dual stack manner.

In Japan, over half of the FTTH market share is taken by NTT-East and NTT-West. They are not allowed to provide Internet service by Japanese law. They have been operating FTTH as data link for other ISPs. From 2010, NTT-East/West and many ISPs which are depending on the FTTH service provided by NTT-East/West have been discussing how to support IPv6 services on these infrastructure. In the last year, they defined two methods called as PPPoE and IPoE, as shown in Figure 2 and in Figure 3 respectively.



NTT East and West, NGN's IPv6 PPPoE

Outline of IPv6 PPPoE

- IPv6 Prefix will be assigned to User via PPPoE by ISP
- new CPE for handling PPPoE(v6CP) and NAT66 is required to access both NGN and the Internet.
- Dedicated ID and password for IPv6 tunnel must be set on IPv6 Adapter

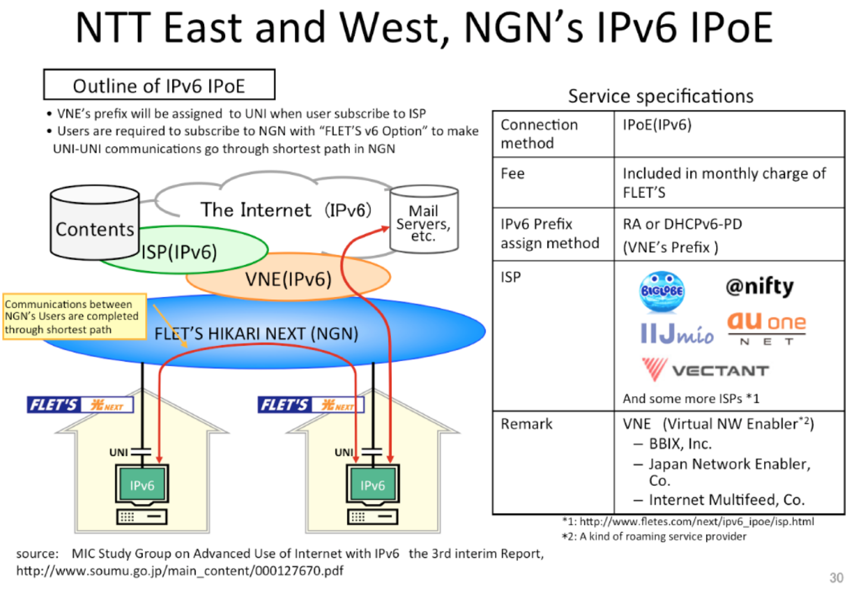
Service specifications

Connection method	PPPoE(IPv6)
Fee	Included in monthly charge of FLET'S
IPv6 Prefix assign method	DHCPv6-PD via PPPoE
ISPs	OCN plala So-net DTI dream.jp and 128 ISPs in total
Remark	IPv6 adapter or similar function to terminate PPPoE(v6CP) and NAT66 is required

source: MIC Study Group on Advanced Use of Internet with IPv6 the 3rd interim Report, http://www.soumu.go.jp/main_content/000127670.pdf

29

Figure 2: IPv6 Deployment using PPPoE



NTT East and West, NGN's IPv6 IPoE

Outline of IPv6 IPoE

- VNE's prefix will be assigned to UNI when user subscribe to ISP
- Users are required to subscribe to NGN with "FLET'S v6 Option" to make UNI-UNI communications go through shortest path in NGN

Communications between NGN's Users are completed through shortest path

Service specifications

Connection method	IPoE(IPv6)
Fee	Included in monthly charge of FLET'S
IPv6 Prefix assign method	RA or DHCPv6-PD (VNE's Prefix)
ISP	BIGLOBE @nifty IIJmio au one NET VECTANT And some more ISPs *1
Remark	VNE (Virtual NW Enabler*2) – BBIX, Inc. – Japan Network Enabler, Co. – Internet Multifeed, Co.

*1: http://www.fletes.com/next/ipv6_ipoe/isp.html
*2: A kind of roaming service provider

source: MIC Study Group on Advanced Use of Internet with IPv6 the 3rd interim Report, http://www.soumu.go.jp/main_content/000127670.pdf

30

Figure 3: IPv6 Deployment using IPoE

1.3 Current status

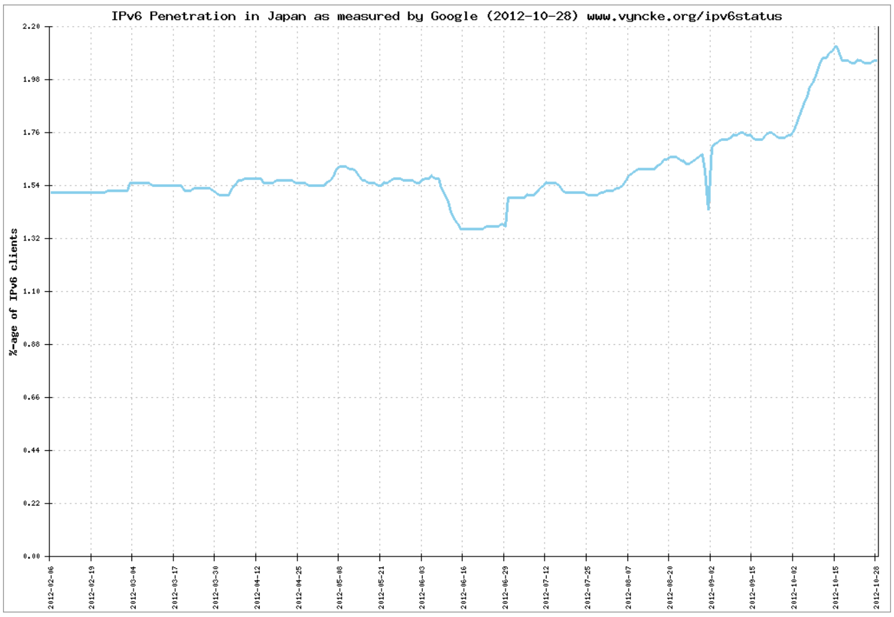
As of writing, it is estimated that about 20% of FTTH subscribers in Japan are IPv6 enabled. By Google statistics, over 2% access from Japan is IPv6. Major ISPs in Japan have agreed to move forward IPv6 deployment. They have a plan to start the IPv6 service by default. It means that no additional fee is required and that no special order request from user is required. Softbank started a new service since October 1st 2012. This service provides IPv6 by default as well as IPv4 by tunneling mechanism over IPv6 network. This service is provided by the IPoE method over NTT-East/West NGN network. As this service name is "1Gbps Internet service", it draws users' attention.

Major CATV companies are planning to start IPv6 service in year 2013.

There are a lot of statistical information about IPv6 deployment. Eric Vyncke's site[2] is one of the good sites. In this site, IPv6 penetration information measured at Google is provided. Figure 4 illustrates IPv6 penetration

[2] http://www.vyncke.org/

in Japan.



by `http://www.vyecke.org/`

Figure 4: JP IPv6 Penetration measured at Google

1.4 Fallback Problem

Fallback problem occurs in an incomplete dual stack environment where there is IPv4 connectivity to a destination while there is no IPv6 reachability to the same destination (and vice versa). In this case, if a TCP connection tries with its IPv6 address, it fails after a timeout, then tries with IPv4 address as seen in Figure 5.

This situation is not very special case. In a dual stack environment, when the IPv6 connectivity to a destination is lost, it takes significant moment to fallback to IPv4. But in Japan, fallback problems could occur everywhere because NTT-East/West provides a data link services to ISPs over a closed IPv6 network. This closed IPv6 network is used for their internal services such as telephone service and TV broadcasting service. It has not known widely when there were small numbers of IPv6 sites.

The detail of fallback problem is shown in Figure 6. Home network is

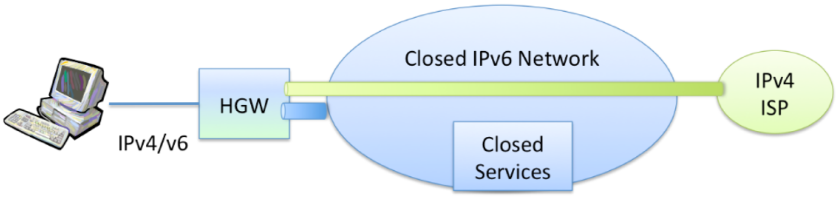


Figure 5: Fallback timeout happens with a closed IPv6 service

connected the IPv4 Internet and also IPv6 closed (no global connectivity) network.

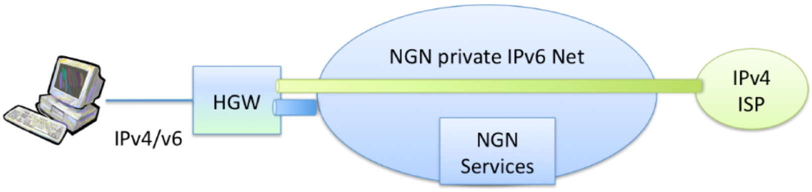


Figure 6: Fallback timeout happens with a NGN service

When a home PC connects one of the dual-stack destinations, which announces AAAA and A records in DNS, it tries in IPv6 first. This connection request results in a timeout, because there is no global reachability to the site in IPv6. Then the home PC tries in IPv4. This timeout is called fallback delay.

Table 1 was reported by BBIX. The fallback delay time depends on OS and its version, and application software such as Web browser.

Each party is required to have effective measures to mitigate the problem. A fundamental solution is to provide IPv6 global connectivity to every home. While ISPs and NTT-East/West will apply this method to the global IPv6 services over NTT-East/West network, it may take extra moment. Other possible mitigations in place are as follows:

- "Reset-er" : NTT-East/West installed a "reset-er". When it receives a TCP connection request packet to any unreachable site, it responses TCP reset packet to the sender to expedite the fallback.

Table 1: Measured Fallback Time

OS	Browser	Dualstack	IPv4 only	Delay (T1-T2)
WindowsXP	IE7	2.57s	0.54s	2.03s
WindowsXP	IE8	1.78s	0.49s	1.29s
WindowsXP	Firefox12	1.33s	0.66s	0.67s
WindowsXP	Chrome1.9	1.33s	0.66s	0.67s
Windows7	IE9	4.64s	0.48s	4.16s
Windows7	Firefox12	2.76s	0.49s	2.27s
Windows7	Chrome1.9	1.32s	0.66s	0.66s

Reported by Yahoo!Japan in June 2012

- AAAA filter on ISP: ISP provides a special DNS service for its IPv4 customers. This special DNS service filtered out AAAA records.
- AAAA filter on CSP: Google and some CSPs provide a special DNS service which control the DNS responses. It checks the origin of a DNS query, and gives a response without AAAA record if the origin is known to cause the fallback problem.
- Mac OS X: Apple provides a happy-eye-ball control in the recent release of the OS and the applications where IPv4 and IPv6 connection trial happens in parallel, and accept one which establishes first.
- Windows 8: Microsoft implemented a new control mechanism in Windows 8. At bootstrap time, the OS checks the network environment to see if it has global IPv6 connectivity by sending an IPv6 packet to a special destination.

2 Case Study

It may not intuitive to introduce IPv6 to the real environment is not that easy. Essentially, theoretical difficulties can be considered at the same level as in IPv6. However, the major difference was a point that there was no Internet before when we tried to connect to the Internet (or even intranets). But the IPv4 Internet connectivity is used for daily production purpose, and it must not give serious impact on the quality of IPv4 connectivity when IPv6 introduction is in place. In this section, we'd introduce a few examples as case study.

2.1 WIDE Project

WIDE Project has been operating its own network as a live testbed. It is an excellent playground to test new technologies in the production environment to tell various issues otherwise not found in the lab test. IPv6 was one of the nice examples what we have done. Very first operational IPv6 connectivity depended on tunnel over IPv4, we tried to integrate IPv6 as native as possible. When we introduced carrier-provided long-haul ATM circuits, IPv6 link can be integrated into IPv4 as a separate PVC, sharing the bandwidth. The routers had been upgraded to support IPv6 to operate the link in dual-stack manner. When layer-3 switch became practical due to their routing capability such as supported routing protocols, VLANs was used extensively to make IPv6-only link before merged with IPv4.

We have worked not only the layer-3 issues but also the applications rather than telnet/ftp/ssh. Our project web page has been IPv6 accessible in many years ago along with our project mail server. When we held a distance learning class between University of Wisconsin, and a couple of Japanese Universities (Keio and NAIST), we used DVTS (Digital Video TranSmission - which transfers IEEE 1394 AV stream over the Internet, and its implementation was developed by WIDE Project) technology over IPv6 to give minimum AV latency to make realistic discussions between the remote professor and the students possible. While high-performance commercial routers at that moment were not IPv6-capable, we used TransPAC ATM international circuit by obtaining a separate PVC from Madison, WI to Tokyo. So in WIDE Project, IPv6 have been used more than a decade as our production purpose. It can be said that we have completed our IPv6 Launch in more than 10 years ago.

2.2 DNS Servers

As IPv6 address is 128bits which is too long to remember, DNS is a must to publish IPv6 address. RFC1886 [1] defines `AAAA` record just to represent an IPv6 address in flat format. However, in order to simplify the renumbering process, alternate format of `A6` was proposed [2]. It was a nice proposal but was not accepted in the operational Internet due to a fact that overhead reduction on renumbering is not just DNS and a fast that it may need extra several query-response transactions to obtain a single IPv6 address.

Considering the referral information specifying child zone name server IPv6 addresses require multiple query-response process, the overhead can be significant in the worst case. So IETF decided to use AAAA rather than A6 in August 2001.

As soon as the implementation supported AAAA record in bind-8 implementation by Internet Software Consortium (current Internet Systems Consortium), we used it. As additional AAAA record occupies 28byte while corresponding A record consumes 16byte, it was expected that DNS response could be large enough to exceed 512byte DNS's packet size limitation in UDP. ISC released bind-8 version with EDNS0 [3] support to relax this restriction as bind-8.3.0 in November 2001. Versions of bind-8.3.0 or later, and all versions of bind-9 support EDNS0 to allow larger responses.

We requested JPNIC to publish our name servers' IPv6 address. It took a while to modify the zone generation system at JPNIC, but introduction of AAAA records to .JP zones was not very difficult including .JP DNS servers' IPv6 addresses. However, in order to make these IPv6 addresses public, they must be registered in the root zone and we, along with JPNIC and JPRS, negotiated with IANA to support AAAA record in the root zone. It was July 2004 when .JP's IPv6 addresses appeared in the root zone.

The last challenge was deployment of IPv6 access to the Root DNS servers. As WIDE Project has been operating one of the 13 Root DNS servers since 1997, it was one of the serious issues. There were long discussion among the Root DNS operators to see what consequence could result if AAAA records were added to the root-servers.net zone. In the worst case scenario, if some DNS servers failed to receive/interpret the responses from Root DNS servers, they would virtually fail to access any resources on the Internet. We were certain that the DNS implementations works well if some of them just ignore AAAA information. But the point was the bigger response packets with EDNS0. It was reported that one of the firewall implementations filtered bigger response packets regardless of the existence of EDNS pseudo resource record. After the vendor issued a notification with bug fixed version software, IANA decided to accept AAAA record to the root-servers.net zone.

In February 2008, AAAA records for 6 Root Servers, A, F, H, J, K, and M, added as these servers have been ready to accept queries in IPv6 as well as in IPv4. Since then, D, I, and L have been enabled their IPv6 service and published their IPv6 addresses as of writing. Other letters are in preparation phase. For the up-to-date information, please refer to the Root DNS servers' page at http://www.root-servers.ORG/.

2.3 A case of Japan Advanced Institute of Science and Technology

Japan Advanced Institute of Science and Technology (JAIST) is a national graduate institute in Japan. It has three graduate schools, no undergraduate students, 900+ masters' and doctoral course students and 300+ faculty and administrative staffs. In JAIST, research center for advanced computing infrastructure is managing and operating computing and networking infrastructure in centralized manner. JAIST is also known for trying to install and operate leading-edge computing and networking infrastructure as their information environment.

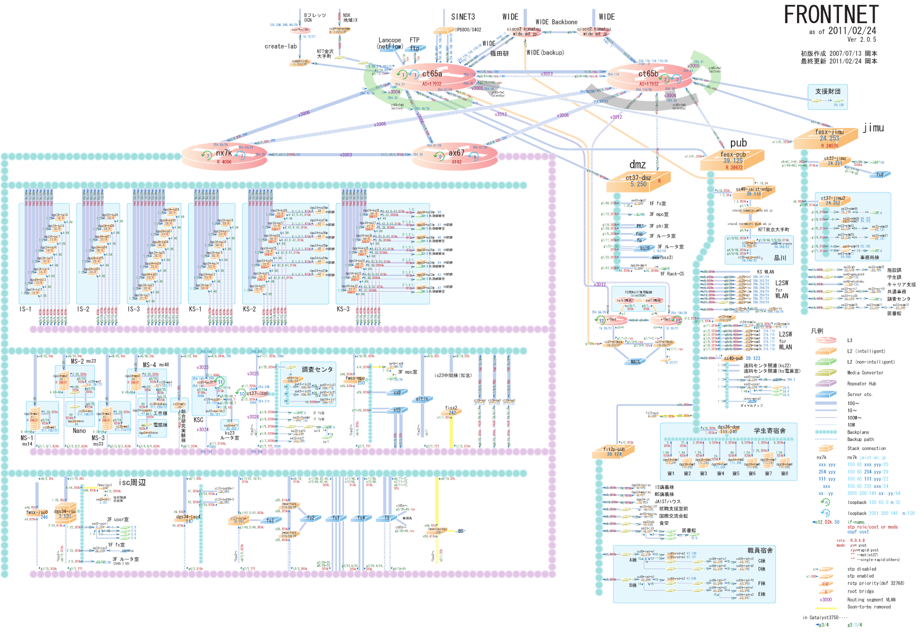


Figure 7: Configuration of FRONTNET

The campus network, named FRONTNET, is consists of 10GbE based links as shown in Figure 7 and designed on central routing architecture. It has two core routers, Cisco Nexus 7018 and Alaxala AX6700S, and all floor switches are connected to both routers for redundancy. It has a pair of Fortinet FortiGate-3950B as firewall, Juniper MX80 and Brocade XMR4000 as border gateway routers. These core network also supports IPv4/v6 unicast/multicast features.

The history of IPv6 operation on FRONTNET was began in 2004. JAIST got a IPv6 address block assignment, `2001:200:141::/48`, from WIDE. FRONTNET started dual-stack operation on its backbone network, and started providing IPv6 connectivity service for limited experimental users. In those days, it was based on some special links and routing hacks to bypass IPv6 non-capable equipments. After that, IPv6 trial was remaining, that include replacement old IPv6 non-capable equipments with IPv6 capable new one, verification of side effect for end-users on enabling IPv6, etc.. Finally IPv6 had been enabled on almost all user-side (client-side) segments and backbone had been operating dual-stack completely including firewalls in 2009.

It is necessary to examine addressing policy on developing IPv6 to the campus. In the case of FRONTNET, there is a reflection point that IPv4 Firewall ACLs has been growing up endlessly, there are multiple thousands lines now, as a result of repeated renumbering. So we tried satisfying both of the visibility to an operator and simplification of the ACL. Therefore we adopt mapping to security zone and physical place; that is dividing all space based on Security Zone, and assigning based on the physical place in each block. For example, the IPv6 address `2001:200:141:4XYZ::/64` is assigned to Subnet Z, Floor Y, Building X, School of Information Science.

We think about whether should we start deploying IPv6 from client-side or server side. In this consideration, we need understand pros and cons of each. If we start from client-side, PCs which support IPv6 can use IPv6 services outside of JAIST initially. If we start from server-side, only outside users can use our IPv6 services initially, but this can be a public relation agenda of our effort in IPv6 to outside. We chose starting from client-side because there are some expert users to want to use the IPv6 as soon as possible. For other reasons, it was relatively easy to deploy IPv6 to client-side because we were not using web-based client authentication for connecting our wireless network. If we were using these features, it is difficult for us to deploy from client-side because supporting IPv6 on these appliances was slowish.

On the other hand, we also need to go forward supporting IPv6 on the systems other than network equipments including super computers, storage systems, enterprise appliances/software and embedded systems. We decided to write down "IPv6 is required" in the procurement specifications of almost all computer systems in 2005 as our policy. Supporting status of IPv6 was not good for these systems at that moment. However, standing on the view of the system replacement interval (every 4+ years in JAIST), it was important to began it early due to avoiding the risk of missing deadline. We had some systems which we give up, however, almost all systems support IPv6 well now.

Recently we face a trouble related to IPv6 non-capable system. The major reason for the trouble was our SSL-VPN system had not supported IPv6 yet. An our user say “I have rejected to connect to internal service server from home network via SSL-VPN today”. After investigating logs, it was found out that the user’s access was on IPv6. Some ISPs in Japan has been starting enabling IPv6 without any special notice to their users. In addition, as for many new OS’s, IPv6 has been enabled by default now. That is his ISP has been enabled IPv6 on his home network, it make effect to connect the internal server with IPv6, the access to server had been denied because the access was not via VPN tunnel (the access is came via an IPv6 native network) because the SSL-VPN system cannot handle any IPv6. It mean this has big problems on the security. On using the Internet, the users don’t care which protocol version they are using due to there is automatic fallback feature. The users without any special knowledge on the Internet protocol believe they are using safe-channel. However they use VPN outside non-safe channel without a thought.

In JAIST, there is a private cloud environment based on Virtual Machine (VM) technology. On this environment, we are troubled with the IPv4 address lack of the segment due to growth of the environment. We need to make the re-numbering of the IPv4 address continues in the case of IPv4 address lack on any subnet because we cannot prepare large space preliminarily. However in the IPv6 world we will get away from limitation of maximum number of IP addresses on each subnet. So we have a high expectation for IPv6, and get start to trying to shift to use the IPv6 mainly as backend network of our cloud environment from IPv4.

2.4 A case of the University of Tokyo

The University of Tokyo is the largest research university in Japan, and than 14,000 undergraduate students, 14,000 graduate students, and 10,000 faculty members and other staffs. It has three major campus, Hongo Campus (Tokyo), Komaba Campus (Tokyo), and Kashiwa Campus (Chiba), along with Shirogane and other 50 remote research laboratories. The campus network system is called as UTnet and its information technology center is in charge of the development and the operation of UTnet. The responsibility of UTnet is to deliver connectivity to each building. It means it is up to the people to provide connectivity within the building.

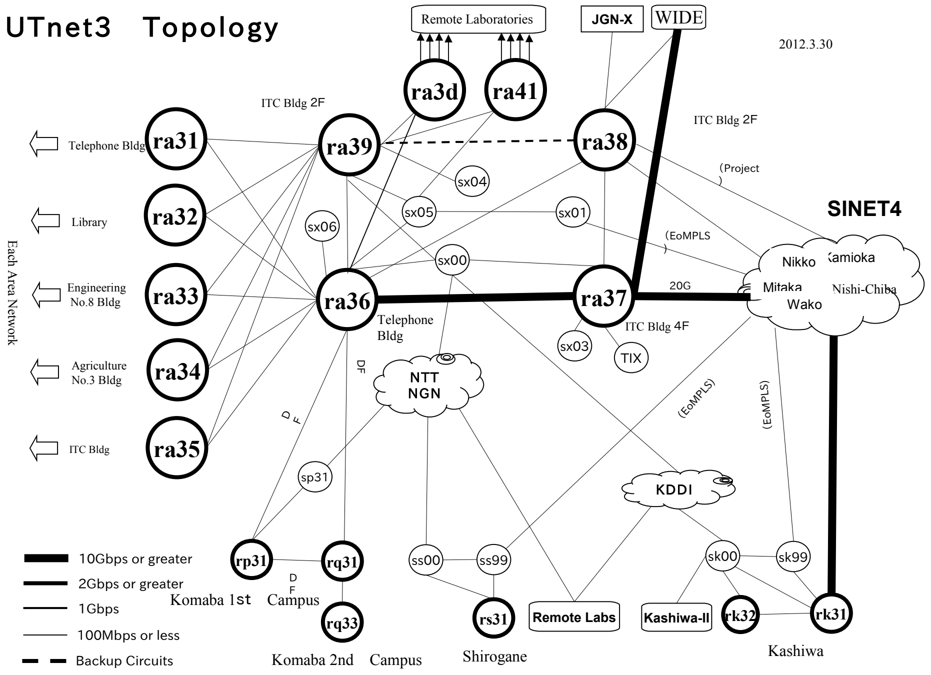


Figure 8: Configuration of the University of Tokyo Campus Network

The campus network is consists of 10GE (10Gigabit Ethernet) based links as shown in Figure 8[3] with high-performance layer-3 switches, currently Cisco Catalyst6500 series with SUP720 supervisor engines. By the nature of the layer-3 switches, 802.1Q VLAN technology has been used everywhere, and in the UTnet, each subnet is identified unique VLAN ID. There are many subnets which are beyond the security devices provided by each faculty and they don't have unique UTnet VLAN IDs. More than 650 UTnet VLANs including backbone portion are configured as of writing.

As each subnet is associated with its VLAN ID, VLAN ID is used for IPv6 address assignment. If a subnet is identified by VLAN ID 298, the IPv6 address of the subnet is `2001:200:180:298::/64` where `2001:200:180::/48` address block was delegated from WIDE Project owned space `201:200::/32`. Note that VLAN ID is in decimal where IPv6 address is in hexadecimal. But for convenience, decimal representation is embedded in the IPv6 address.

[3] `https://www.nc.u-tokyo.ac.jp/image/backbone.pdf`

Since around 2003 UTnet started route IPv6 and IPv6 is enabled upon request. UTnet didn't force dual-stack operation. Currently 28 user subnet directly attached one of the UTnet L3 switches (which means the number excludes backbone subnet) are IPv6 ready. But there are non-UTnet operating 68 subnets are IPv6 enabled. In this case subnet field of their IPv6 address would be includes at least one of A-F or more than "4096".

2.5 A case of Keio University, Graduate School of Media Design

Keio University, one of the research private universities, opened a couple of brand new graduate programs celebrating its 150th anniversary, and one of them is Graduate School of Media Design. It is a small graduate school including 180 masters' and doctoral student with about 20 faculty members. The student background has a large diversity; from medical school to art school or music school. Only a few students have computer science background, but more than half of the student use Macintoshs and iPhones.

The routers and switches to provide connectivity to the subnets of the graduate school are operated by ITC upon request by the graduate school. An email server, a DNS server, and a web server of the graduate school are, however, operated by the graduate school.

Generally network engineers may hesitate to turn on IPv6 on a subnet which has been used for users' production purpose already as it could interrupt the users' activities. So it would be a good question to see what happened dual-stack subnets existed prior to the non-technical people started to live on them.

As of writing no IPv6 specific trouble has been reported by the students. Our email server accessible in POP3 [4] also offers IPv6 service internally and externally. Most of the students on-campus check their email arrival in IPv6 even if the server's IPv6 address is not explicitly told to the students. Some (it could be many) graduate students do not understand what is IPv4 or IPv6, but they are using IPv6 whenever possible. When they are in the dormitory or their home, they usually use IPv4 as no IPv6 connectivity is provided in the home networks yet.

3 Toward IPv6-only Internet

Through the world IPv6 day in 2011 and the world IPv6 launch in 2012, IPv6 has become to be deployed into not only R&D networks, but also into in commodity networks. In the current situation, several IETF working groups explore technologies or solutions to move IPv6 only network and withdraw IPv6 only network without any significant impacts, especially in Softwires working group (softwire), Sunsetting IPv4 working group (sunset4), and IPv6 Operations working group (v6ops). Softwires working group (softwire) is specifying the standardization of discovery, control and encapsulation methods for connecting IPv4 networks across IPv6 networks and IPv6 networks across IPv4 networks in a way that will encourage multiple, inter-operable implementations. Sunsetting IPv4 working group (sunset4) standardizes technologies that facilitate the graceful sunsetting of the IPv4 Internet in the context of the exhaustion of IPv4 address space while IPv6 is deployed. IPv6 Operations working group (v6ops) develops guidelines for the operation of a shared IPv4/IPv6 Internet and provides operational guidance on how to deploy IPv6 into existing IPv4-only networks, as well as into new network installations.

To promote standardization activities on each IETF working group, and to store TIPS of design and operation on IPv4 across IPv6 networks and on IPv6-only networks, the WIDE project has conducted experiments in WIDE camps. WIDE camp is a bi-annual research workshop for WIDE project members held in March and September to promote extensive discussions. In a WIDE camp, an experimental network with various advanced technologies is deployed and provided as the access network on the workshop place for participants. From September 2011, three "Life with IPv6 experiments" have been conducted. "Life with IPv6 experiments" specify the evaluation on advanced technologies that are promoted or under discussion in IETF softwire, sunset4 and/or v6ops working groups. The evaluation result of each experiment has been reported to the IETF through a few versions of an internet draft. Its latest revision is attached in Appendix A. We introduce summary of each experiment in following subsections.

3.1 1st experiment

The 1st experiment was held in September 2011 at Matsushiro Royal hotel in Nagano. We set 3 evaluation items in this 1st experiment.

3.1.1 Evaluation items

The first evaluation item was achieving a VPN by a prototype L2TP v3 implementation through a commercial IPv6 only network. From May 2011, IIJ has provided IPv6 network services for home users through NTT NGNv6 backbone and access lines. The prototype L2TP v3 implementation was developed by Yukito Ueno from Keio University.

The second evaluation item was evaluating IPv6 capability of devices, OSes, and applications on an IPv6 only network through actual business usages of participants. The IPv6 only network was composed of stateful DHCP6, 6to4 translation through DNS64 and NAT64. In the 1st day of this camp, we provided only this IPv6 only network for participants.

The third evaluation item was testing IPv4 across IPv6 network technologies. We evaluated two IPv4 across IPv6 network technologies, one was SA46T and the other was murakami-4RD (murakami-4RD is now integrated into MAP of IETF softwire working group). SA46T was used to provide global IPv4 addresses by DHCP4 on the IPv6 only network of the second evaluation item. We employed a prototype SA46T implementation that was developed on a collaboration research between Keio university and Fujitsu. On the other hand, murakami-4RD provided a private IPv4 network. With core developers of IIJ SEIL routers, we evaluated the interoperability between vyatta implementation and IIJ SEIL implementation and grasped hidden issues on the specification or implementations on the murakami-4RD.

3.1.2 Lessons

We got following lessons from the 1st experiment.

- We can live in an IPv6 only network much more comfortable than we thought.
  20 participants lived only in the IPv6 only network.

- For users, DHCP6 function is required.
  Most older OSes and SmartPhone OSes did not support DHCP6 function. Manual setting of IPv6 was difficult due to the dependency on IPv4 of DNS name resolution or others.

- Long timeout / fallback routine occurred in the initial WiFi setting if a user use his/her device in dual stack mode.
  To live in an IPv6 only network with conformance, IPv4 property should be turned off, or some mitigation method should be developed.

- Major Internet Messaging applications did not support IPv6.
  Windows Live Messenger and Skype did not support IPv6. XMPP based IMs, for instance jabber, worked well.

- Failure of name resolution on DNS64 occurred due to the inappropriate implementation or configuration of authoritative DNS servers.

- Many VPN applications could not be available in IPv6 only network or in IPv4 across IPv6 network.
  Because of no affinity to network address translation, many VPN applications could not be available. MTU black hole problem was one of reasons not to use VPN applications through encapsulation techniques.

3.2 2nd experiment

The 2nd experiment was held in March 2012 at the same hotel. We set 3 evaluation items in this 2nd experiment.

3.2.1 Evaluation items

The first evaluation item was achieving IPv6 only networks with using native IPv6 addresses through commercial IPv6 services for emulating a home user environment. We prepared two native IPv6 networks, one was provided in IPoE method, the other was provided in PPPoE method. DHCP-PD routers on the hotel side were IIJ SEIL router. DNS server addresses was informed through stateless DHCP6.

The second evaluation item was a comparative test on three IPv4 across IPv6 network technologies from the point of view of commercial network games. This evaluation item was conducted with contributions from NTT East, Internet MultiFeed, IIJ, Fujitsu, NEC AccessTechnica, JPIX, and Konami Digital Entertainment. We evaluated murakami-4RD, 464XLAT and SA46T by STUN. The STUN test program was contributed by Konami Digital Entertainment.

The third evaluation item was evaluating the availability of each network through participants' usages.

3.2.2 Lessons

- At that moment, participants moved to networks where IPv4 was available because they wanted to use IPv6 incapable applications. However, around 60 participants lived in IPv6 only network.

- Long fallback / timeout problem might be occurred due to the on-link assumptions.

- MTU or fragmentation handling were different among OSes. In trouble shooting on an MTU black hole problem, multiple OSes should be prepared to compare the behaviors of each OS.

- Failures of DNS64 fallback are changed by DNS implementations on authoritative servers.

- Take care of hair-pinning support and address port mapping algorithms on NAPT implementations for consumer (P2P) network games.

3.3 3rd experiment

In the 3rd experiment performed in September 2012, we focused on exploring a work around to mitigate long timeout / fallback problems of dual stack nodes in an IPv6 only network.

3.3.1 Evaluation environment

As a base experiment environment, we setup a DHCP-PD router by a PC. DNS information to a DNS64 service was provided by stateless DHCP6. The DNS64 service was placed on the WIDE backbone network. In this network, dual stack nodes met following timeout/fallback problems or other troubles:

- Windows Vista, 7, 8
  IPv4 connectivity check on the initial wireless setting spent one or a few minutes.

- Windows XP, MacOS X (Snow Leopard or older)
  Uses had to do manual settings of DNS.

- MacOS X (Lion or Mountain Lion)
  Users had to wait for IPv4 connection timeout when they use Google Chrome or Firefox. However, Safari had no such a problem.

- iOS5
  Network settings were not completed and continuously failed due to the retry sequence of network settings.

- Android
  DNS was not usable. No manual configuration was available.

Through deep inspection on each behavior, we found that these timeout / fallback problems were mainly derived from three root causes as follows:

- Connectivity check by DHCP4 (Windows Vista or later, iOS5)
- Dependency on Name Resolution on IPv4 (Mac OS X Snow Leopard or older, Android, iOS5).
- On-link-assumption of IPv4 and HappyEyeBall-like address selection behavior (Mac OS X Lion or Mountain Lion)

3.3.2 Workaround

To figure out a work around to mitigate these timeout / fallback problems, we added several functions to the base experiment environment. As a result, we got following workaround:

- Install a dual-stack DNS proxy on the local subnet.
  The DNS proxy forwards all query to DNS64 except query type is `A` (IPv4 address). The DNS proxy should be on-link with clients. Since there is no IPv4 connectivity on the client, all queries to `A` should be filtered and returns `NO_DATA`, just like `AAAA` filtering. This filter should be enabled both on IPv4 and IPv6 transport.
- Configure a DHCP4 server and announce private IPv4 address, router, and DNS.
  There are no actual IPv4 connectivity to outside. But some client will be happy only when these 3 address can be obtained through DHCP4. The IPv4 router should drop all IPv4 packets and return ICMP unreach. The DNS announced by DHCP4 must indicate to the DNS proxy with `A` filter.

4 Summary

The Internet community has no more IPv4 free space. If you need additional address space, you may need to “buy” it. It could impact the routing table growth as well as the tariff of the service to provide the users. However, as described in this report, IPv6 has been considered to solve the addressing issue, however, it is not matured technology. As every technology in the Internet is still in development, IPv6 is as well as IPv4. In order to solve the address space issue, it is necessary that the Internet community work together, sharing their experiences to provide stable IPv6 connectivity.

It is a time to work for it seriously.

Acknowledgments

We'd like to thank KISTI to support part of the IPv6 deployment related activities of WIDE Project. We also thank the following organizations who contributed valuable resources including human resources (order is not significant): ALAXALA Networks Corporation, Anchor Technologies, Konami Digital Entertainment, Fujitsu Limited, Hitachi Limited, IBM Japan, Limited, Internet Initiative Japan Inc., Internet MultiFeed, IPv6 Forum, JPIX, Keio University, NEC AccessTechnica, NEC Corporation, Nippon Ericsson K.K., NTT Software Corporation, Sharp Corporation, Toshiba Corporation, The University of Tokyo, Yokogawa Electric Corporation, and number of other WIDE Project sponsors and partners.

References

[1] S. Thomson and C. Huitema. DNS Extensions to support IP version 6. RFC 1886 (Proposed Standard), December 1995. Obsoleted by RFC 3596, updated by RFCs 2874, 3152.

[2] M. Crawford and C. Huitema. DNS Extensions to Support IPv6 Address Aggregation and Renumbering. RFC 2874 (Historic), July 2000. Updated by RFCs 3152, 3226, 3363, 3364.

[3] P. Vixie. Extension Mechanisms for DNS (EDNS0). RFC 2671 (Proposed Standard), August 1999.

[4] J. Myers and M. Rose. Post Office Protocol - Version 3. RFC 1939 (Standard), May 1996. Updated by RFCs 1957, 2449, 6186.

A Internet Draft on WIDE Camp IPv6-only Network

This Internet Draft has been posted to the IETF as individual ID and presented in V6OPS working group sessions in 82nd IETF Meeting (Taipei, November 2011) and 83rd IETF Meeting (Paris, March 2012). Slides used for the presentations are available respectively[4][5].

[4] http://www.ietf.org/proceedings/82/slides/v6ops-7.pdf

[5] http://www.ietf.org/proceedings/83/slides/slides-83-v6ops-0.pdf

Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 6, 2013

H. Hazeyama
NAIST
R. Hiromi
Intec Inc.
T. Ishihara
Univ. of Tokyo
O. Nakamura
WIDE Project

October 3, 2012

Experiences from IPv6-Only Networks with Transition Technologies in the WIDE Camp Autumn 2012
draft-hazeyama-widecamp-ipv6-only-experience-02

Abstract

This document reports and discusses issues on IPv6 only networks and IPv4/IPv6 transition technologies through our experiences on the 3rd experiment on the WIDE camp. The 3rd experiment was held from September 3rd to September 6th, 2012. As well as past two experiments, we conducted face to face interview to participants for grasping IPv6 capability on users' devices, OSes, and applications. In addition to this, we explored solutions to mitigate timeout / fallback problems of IPv4/IPv6 dual stack clients on an IPv6 only network that is composed of DHCP6 and DNS64/NAT64.

Status of this Memo

This Internet-Draft will expire on April 6, 2013.

Copyright Notice

Table of Contents

1. Introduction

This document reports and discusses issues on IPv6 only networks and IPv4/IPv6 transition technologies through our experiences on the 3rd experiment on the WIDE camp. The 3rd experiment was held from September 3rd to September 6th in Matsushiro Royal Hotel, Nagano, Japan, where is the same hotel of the 1st and 2nd experiments.

1.1. History of ‘‘Live with IPv6 experiments’’ on the WIDE camp

"Live with IPv6 experiment" aims to evaluate commercial IPv6 network services, the availability of IPv6 networks with several IPv4 / IPv6 translation / encapsulation technologies by actual users' experiences, and to grasp issues on IPv4 exhaustion situation or IPv4 / IPv6 transition. These experiments are based on an assumption that ISP backbone networks will be constructed on IPv6 only and end customer will have to use an IPv6 network with 64 translators or an IPv4 network with 464 translators to keep current usage of the Internet services.

1.1.1. Summary of the 1st experiment

The 1st experiment was held in Matsushiro Royal Hotel from September 6th to September 9th, 2011 with 153 participants, and the experiment result was reported in the v6ops BoF on IETF 82 Taipei. In the 1st experiment, we constructed an IPv6 only network with stateless NAT64 and DNS64 as a part of the WIDE backbone through IPv6 L2TP over a commercial IPv6 network service. The commercial IPv6 network service was provided by NTT-East as an Access Carrier, Internet MultiFeed (MFeed) as a Virtual Network Enabler (VNE) and IIJ as an IPv6 Internet Service Provider (IPv6 ISP). In addition to an IPv6 connectivity with NAT64/DNS64, we also tested a SA46T [I-D.draft-matsuhira-sa46t-spec] based IPv4 global network service and a murakami-4RD [I-D.draft-murakami-softwire-4rd] based IPv4 private network service (murakami-4RD is now merged into MAP [I-D.draft-ietf-softwire-map-02]). With referring IETF's IPv6 only network experiences [RFC6586], we reported several new issues on an IPv6 only network with IPv4 / IPv6 transition technologies, especially on inappropriate DNS replies mentioned in [RFC4074], on MTU mismatch, on VPN protocols and applications through IPv4 / IPv6 translators.

1.1.2. Summary of the 2nd experiment

According to the experiences on the 1st experiment, the 2nd experiment was conducted from March 5th to March 8th, 2012 in Matsushiro Royal Hotel, the same hotel of the 1st experiment. 171 participants joined this 2nd experiment, most of them were engineers or academic people. The 2nd experiment result was reported in the v6ops BoF on IETF 83 Paris.

The settings of the core network in the 2nd experiment was same as the 1st experiment. In the 1st experiment, a commercial IPv6 network service was employed as a backbone network, in other word, we did evaluate the availability of commercial IPv6 network services from the view of home users. Therefore, the evaluation target of the 2nd experiment was planned as living in commercial IPv6 networks with IPv4 / IPv6 translation technologies or IPv4 / IPv6 translation services.

The user access networks of the 2nd experiment were achieved by two types of commercial IPv6 network services through the NTT NGNv6 access network, with four kinds of IPv4 / IPv6 translation technologies. One of the two commercial IPv6 network services was /48 prefix IPv6 network service through IPoE[RFC0894] on NTT NGNv6 (we name it "native IPoE" in this draft), the other was /56 prefix IPv6 network service through PPPoE[RFC2516] on NTT NGNv6 (we label it "native PPPoE" in this draft) [YasudaAPRICOT2011]. Both IPv6 networks were served from NTT-East, MFeed and IIJ as same as the 1st experiment.

Usually, IPv6 networks on both native IPoE and native PPPoE were provided with only DNS v6 proxy. We constructed DNS64/NAT64 service on the WIDE backbone and on the camp core network, and served it through stateless DHCP6 [RFC3736] both on native IPoE and on native PPPoE.

Along with the DNS64/NAT64 translation service, for aiming to evaluate more practical approaches on the current commercial environments, we tested three IPv4 services over IPv6 networks, murakami-4RD [I-D.draft-murakami-softwire-4rd], SA46T [I-D.draft-matsuhira-sa46t-spec] and 464XLAT [I-D.draft-ietf-v6ops-464xlat]. We mainly served seven IP networks to participants by combination of those networks and translation services, that is, native IPoE with DNS64/NAT64, native PPPoE with DNS64/NAT64, murakami-4RD on both IPoE and PPPoE, 464XLAT on both IPoE and PPPoE, SA46T on PPPoE.

Three evaluations were mainly conducted by the evaluation team, i) user survey about the availability of each network through face to face interview, ii) analysis of DNS behaviors to grasp inappropriate behaviors mentioned in [RFC4074], iii) availability test of VPN applications to analyze MTU problems For to grasp whether an unavailability of VPN applications was intentional one due to the specification of a translation technology or not. Also, Konami Digital Entertainment (KDE) joined in this experiment, and evaluated

NAT/Firewall traversal testing on each IPv6 network or each translator service from the view of commercial (P2P) Network Game services. KDE gave us the importance / requirements of hair-pinning functions and of MTU / packet fragmentation handling on NAT/NAPT for P2P based Multiplayer Online Games.

1.2. Abstract of the 3rd experiment

The 3rd experiment was conducted from September 3rd to September 6th, 2012 in Matsushiro Royal Hotel, the same hotel of the past two experiments. 136 participants joined this 3rd experiment, most of them were engineers or academic people.

The aims of 3rd experiments were 1) continuous user survey on IPv6 capability of devices, OSes and applications, 2) exploration of a practical solution to mitigate timeout / fallback problems of IPv4/ IPv6 dual stack clients on an IPv6 only network.

The first aim was conducted to grasp the IPv6 capability of users' devices, OSes, and applications and to collect users' experiences through face to face interview. From the 2nd experiments, several new OSes or new devices have been released. Through this continuous survey, we saw the current development / deployment strategy of IPv6 on commercial vendors or Telecom / Internet Service providers. This user survey was mainly carried on September 3rd and September 4th.

The second aim was derived from our experiences of an IPv6 only network with DHCP6/DNS64/NAT64 on past two experiments. In past two experiments, various OSes met several timeout / fallback problems, in the initial connection setting through Wi-Fi settings, in the name server selection, in the establishment of a TCP connection. Most OSes and applications, that met tedious timeout / fallback problems, preferred IPv4 to IPv6, or required IPv4 settings to enable IPv6 settings. These timeout / fallback problems were seemed to be derived from an assumption that there are no IPv6 only network on the current situation.

Toward the sunset of IPv4, we have to explore and achieve a practical solution to move from IPv4/IPv6 dual stack networks to IPv6 only networks without giving stress or difficulties to end users. In IPv4/IPv6 transition situation, end users will usually use IPv4/IPv6 dual stack mode, and they will leave all IPv4 / IPv6 network settings by OSes' auto configuration behaviors on their devices except for selecting Wi-Fi connections.

We focused on testing an IPv6 only network that was basically composed of DHCP6, DNS64 and NAT64. In this IPv6 only network, we sought a current practice of timeout / fallback mitigation among

IPv4/IPv6 dual stack networks and IPv6 only networks. According to results of the user survey, we added several functions to a basic DHCP6/DNS64/NAT64 network in step by step fashion, and we analyzed or revised mitigation methods for timeout / fallback problems.

This draft is composed of following sections. We explain the overview of the network settings in the 3rd experiment at first. Next, we report the result of the user survey. Then, we describe the experiment on timeout / fallback mitigation methods. Finally, we summarize our practical timeout / fallback mitigation method. We also mention about limitations our mitigation method and our recommendations on development / deployment of IPv6 capability on end clients.

1.3. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Technology and Terminology

In this document, the following terms are used. "NAT44" refers to any IPv4-to-IPv4 network address translation algorithm, both "Basic NAT" and "Network Address/Port Translator (NAPT)", as defined by [RFC2663].

"Dual Stack" refers to a technique for providing complete support for both Internet protocols -- IPv4 and IPv6 -- in hosts and routers [RFC4213].

"NAT64" refers to a Network Address Translator - Protocol Translator defined in [RFC6052], [RFC6144], [RFC6145], [RFC6146], [RFC6384].

"DNS64" refers DNS extensions to use NAT64 translation from IPv6 clients to IPv4 servers with name resolution mechanisms that is defined in [RFC6147].

"DHCP4" refers Dynamic Host Configuration Protocol for IPv4 that is defined in [RFC2131].

"DHCP6" refers Dynamic Host Configuration Protocol for IPv6. So called "Stateful DHCP6" is defined in [RFC3315] and "Stateless DHCP6" is defined in [RFC3736]. "DHCP-PD" or "DHCPv6 Prefix Delegation" refers IPv6 Prefix Options for DHCP6 that is initially defined in [RFC3633] and updated in [RFC6603].

"ND" refers Neighbor Discovery for IP version 6 (IPv6) that is defined in [RFC4861] and updated in [RFC5942].

3. Basic configuration of Network and Experiments

The WIDE Camp Autumn 2012 was held at Matsushiro Royal Hotel in Nagano Prefecture of Japan, the same place of the 1st and 2nd experiment, from September 3rd to September 6th, 2012. Figure 1 shows the overview of the whole network topology on the WIDE Camp Autumn 2012.

Besides our IPv6 only experiments, the camp NOC team set up a core network (camp-net-core) for preparing a backup plan of our IPv6 only network experiments and for conducting other experiments such as OLSR emulation, SA46T-AT [I-D.draft-matsuhira-sa46t-at-00] and NAT44 double translation, and measurement of a satellite link. All server instances and routing instances of the core network were built on StarBED that is a cloud / network emulation testbed in Japan. We constructed two layer 2 tunnels between StarBED and Matsushiro Royal hotel through IPv4 PPPoE. The layer 2 tunnels over IPv4 PPPoE were constructed by NEC IX2015. The OLSR network and the satellite link were served as IPv4 / IPv6 dual stack networks. The wireless Accesses to these networks were provided by CISCO Systems Mesh Wi-Fi Access Point and WLC (Wireless LAN Controller).

As well as our 2nd experiment, a commercial IPv6 service was employed to achieve our IPv6 only network experiments. The Access Carrier (AC), the Virtual Network Enabler (VNE) and the IPv6 Internet Service Provider (v6ISP) of this 3rd experiment were same combination of past experiments, that is, NTT-East as AC, MFeed as the VNE and IIJ Mio as v6ISP. We contracted two external FTTH lines by NTT NGNv6 IPoE method. We changed the IPv6 address allocation method on NTT NGNv6 IPoE during this camp.

From September 2th (the preparation day) to September 4th, we used the RA method for the external connectivity. Figure 2 represents details of the IPv6 only network by RA method. From September 5th to September 6th, we changed the external connectivity to the DHCP-PD method.

In the RA method, we tested the DHCP6 client behaviors when two stateless DHCP6 servers exist, one is placed by the VNE or ISP to indicate AAAA name servers, the other is located in the local subnet to lead clients to a DNS64 name server. On the other hand, we explored mitigation methods for timeout / fallback problems after we changed the external connectivity to the DHCP-PD method. We explain the experiment on the RA method in Section 4.1 and the experiments on the DHCP-PD method in Section 4.2, respectively.

We employed following implementations for key components;

o DNS64 and recursive cache server: NLNet Labs Unbound 1.4.7 with DNS64 patch

o NAT64 : OpenBSD 5.1 PF (Packet Filter)

o DHCP-PD client : WIDE DHCP client (dhcp6c)

o Stateless DHCP6 server : Alaxala 3630

```
       +----------------------------------------------------+
       |               The Internet  (IPv4 / IPv6)          |
       +----------------------------------------------------+
              |     |                      |
              |     |                      |
     +-----------+  |               +--------------+
     | IIJ (ISP) |  |               |  WIDE (ISP)  |---------+
     +-----------+  |               +--------------+         |
              |     |                  |         |           |
              |     |                  |      (L2 Tunnel)    |
              |     |             (IPv4/IPv6)    |           |
              |     |                  |     +---------+     |
              |     |                  |     | IX 2015 |     |
           +-----------+               |     +---------+     |
           |MFeed (VNE)|               |         |           |
           +-----------+          +---------------+  +-------------+
                  |               | camp-net-core |  | SAT station |
                  |               +---------------+  +-------------+
                  |                                          |
      +--------------------------+                           |
      |  NTT NGNv6 (Access Line) |                  (satellite link)
      +--------------------------+                           |
           |      |                                  +-------------+
         (IPoE method)                               | SAT station |
           |      |                                  +-------------+
           |      +--------------+                           |
           |      |              |                           |
        (native IPv6)  (L2 tunnel over IPv4 PPPoE)       (L2 Tunnel)
           |      |              |                           |
           |      |        +---------+                  +---------+
           |      |        | IX 2015 |                  | IX 2015 |
           |      |        +---------+                  +---------+
           |      |              |                           |
           |      |           (vlan)                      (vlan)
           |      |              |                           |
    +------------------------------------------------------------+
    |      Wi-fi Access (CISCO  Layer 2 mesh, 11b/g/n Wi-fi)     |
    +------------------------------------------------------------+
```

Over view of the 2nd experiment topology

Figure 1

4. Experiments

4.1. An Experiment in RA method

4.1.1. Details of Network Configuration

The experiment conducted in RA method was overwriting client DNS information by a local stateless DHCP6 server. Figure 2 shows the test network topology. The RA method provided /64 prefix addresses and routing information through RA. The RA was set managed flag as zero (M flag == 0) and the other flag to one (O flag == 1) to let clients query to stateless DHCP6 servers. In this case, a stateless DHCP6 server was placed on the VNE network of MFeed and IIJ that advertised two AAAA name servers. Those two AAAA name servers returned only AAAA records to any queries.

We wanted to inform only the DNS64 IPv6 address to clients on this RA method while using address assignment and default route settings by the RA method. Of course, we could not control the DHCP6 server on the VNE network. Therefore, we tried to use the preference option of DHCP6.

The preference option of DHCP6 (section 22.8 of [RFC3315]) defines that "the Preference option is sent by a server to a client to affect the selection of a server by the client". Section 17.1.3 of [RFC3315] defines the criteria on the behavior of DHCP6 server selection by a client when the client has received two or more valid advertise messages;

- Those Advertise messages with the highest server preference value are preferred over all other Advertise messages.

- Within a group of Advertise messages with the same server preference value, a client MAY select those servers whose Advertise messages advertise information of interest to the client. For example, the client may choose a server that returned an advertisement with configuration options of interest to the client.

- The client MAY choose a less-preferred server if that server has a better set of advertised parameters, such as the available addresses advertised in IAs.

We assumed we could overwrite the name server information by sending advertise messages with highest preference value from a local stateless DHCP6 server. Thus, we placed a local stateless DHCP6 server shown in Figure 2.

This overwriting was partially succeeded as well as we assumed, however, several inconveniences were reported through face to face interview and inspection by the special observation team.

```
                                              +-------+ +------+
                                              | DNS64 | | NAT64|
                                              +-------+ +------+
                                                  |        |
                                                (-- StarBED --)
                                                  |        |
       +--------------- IPv6 Internet ----------------------+
             |
       +-------------+
       | IPv6 router |
       |   on ISP    |
       +-------------+
             |
       +---------+  +---------+  +----------+             +---------+
       | IPv6 GW |  | DHCP6   |  |AAAA name |             | DHCP6   |
       |         |  | to AAAA |  | servers  |             | to DNS64|
       +---------+  +---------+  +----------+             +---------+
             |         |              |                     |
           (----------  VNE  --------------)             (-- Hotel --)
             |         |              |                     |
       +----------------  /64 prefix segment -----------------------+
                                                         |
                                                     +---------------+
                                                     | users devices |
                                                     +---------------+
```

The Test Topology on RA method

Figure 2

4.1.2. User Survey

59 participants (42.8 %) replied our face to face interview. We show the client profile in Section 4.1.2.1 and reported troubles in Section 4.1.2.2 and Section 4.1.2.3.

4.1.2.1. Client Profile

94 unique devices were profiled. The distribution of the pair of device and OS were shown in Table 1.

Device Type	OS Type	# of devices (%)
PC/AT Note PC	Windows 7	16 (17.0 %)
PC/AT Note PC	NetBSD	2 (2.1 %)
PC/AT Note PC	Linux	4 (4.3 %)
Apple Note PC	Mountain Lion	15 (16.0 %)
Apple Note PC	Lion	18 (19.1%)
Apple Note PC	Snow Leopard	9 (9.6 %)
Apple Note PC	Windows 7 (Bootcamp)	3 (3.2 %)
iPhone / iPod	iOS 5	9 (9.6 %)
Android Phone	Android OS 4	3 (3.2 %)
Android Phone	Android OS 2	4 (4.3 %)
Android Phone	Android OS 1	1 (1.0 %)
iPad	iOS 6	1 (1.0 %)
iPad	iOS 5	6 (6.4 %)
Android Tablet	Android OS 4	2 (2.1 %)
Kindle	Kindle 3.3	1 (1.0 %)
Total		94

Table 1: The distributions of devices of participants

4.1.2.2. Behaviors of DHCP6 Clients

Many users reported inconveniences of DHCP6 client behaviors in the RA method. We focused on the analysis of DHCP6 client behavior of Windows 7 and of Mac OS X Lion / Mountain Lion. Both Windows 7 and Mac OS X usually stored DNS64 IPv6 address to their name server information, however, both of them sometime stored two AAAA name servers on the VNE network. Differences of their DHCP6 client behaviors were as follows;

- In most cases, Windows 7 preferred to the advertise message from the local DHCP6 server that indicated the DNS64 server, however, it often preferred the advertise message from the DHCP6 server on the VNE network at the RA refresh timing.

  * When the DHCP6 client preferred to the DHCP6 server on the VNE network, an user had to reset the Wi-Fi device of his/her PC and to reconnect to the Wi-Fi network. "ipconfig /renew" or simply reconnecting by Wi-Fi selection icon often failed to prefer the advertise message from the local DHCP6 server.

- On the other hand, Mac OS X Lion and Mountain Lion often failed to prefer the advertise message from the local DHCP6 server at the initial set up on Wi-Fi setting, however, "Renew DHCP lease" on the detail of network settings always preferred to the advertise message from the local DHCP6 server, that is, Mac OS X always changed the name server setting to only DNS64 IPv6 address by "Renew DHCP lease". At RA refresh timing, Mac OS X sometime preferred to the DHCP6 server on the VNE network, then, the user had to refresh DHCP configurations again.

4.1.2.3. Timeout / Fallback Problems

Many users reported inconveniences due to timeout / fallback problems. Root causes were roughly categorized into 1) troubles of DNS64, 2) incapability of IPv6 and of DNS64 on various servers and applications mentioned in [RFC4074] and [RFC6586], 3) incapability of DHCP6 client and / or IPv4 dependency on OSes. In Section 4.2, we explain the detail of timeout / fallback problems without effects by the selection of stateless DHCP6 servers.

4.2. Experiments in DHCP-PD method

On the contrary of the RA method mentioned in Section 4.1, the DHCP-PD method provided /56 prefix delegation by DHCP6 prefix delegation mechanism. We settled a DHCP-PD client PC router and set up static routes to two delegated /64 networks, one was labeled as "v6only-basic", the other was named as "v6only-fallback". The v6only-basic network was a basic IPv6 only network that was composed of stateless DHCP6, DNS64 and NAT64. On the other hand, we tested several timeout / fallback mitigation methods in "v6only-fallback". Figure 3 shows the basic network topology of experiments on DHCP-PD method.

4.2.1. Basic Network Configuration

Figure 3 shows the basic network topology of experiments on DHCP-PD method.

```
                                              +-------+ +------+
                                              | DNS64 | | NAT64|
                                              +-------+ +------+
                                                  |        |
                                                (-- StarBED --)
                                                  |        |
       +--------------- IPv6 Internet ----------------------+
                              |
                      +-------------+      +----------------+
                      | IPv6 router |      | DHCP PD server |
                      |   on ISP    |      |    on VNE      |
                      +-------------+      +----------------+
                              |                    |
    +-- (VNE network) ----------------+----------------------+
                                      |
                                      |(v6)
                                      |
                               (---- Hotel ----)
                                      |
                                +---------------+
                                | DHCP-PD Client|
                                |  PC router    |
                                +---------------+
                                      |
      +-----------------+             |
      | Stateless DHCP6 |             |
      +-----------------+             |
                 |                    |
                 |                    |
       +------------- each /64 prefix segment ---------------+
                                              |
                                      +---------------+
                                      | users devices |
                                      +---------------+
```

Basic Network Topology on DHCP-PD method (v6only-basic)

Figure 3

4.2.2. Experiment 0

In the experiment 0, we observed OSes behaviors again. Actually, the inconvenience on the selection of two stateless DHCP6 servers were resolved by DHCP-PD and placing one stateless DHCP6 server onto each /64 prefix subnet. However, we clearly recognized several timeout / fallback problems. In following sections, we explain timeout / fallback problems due to DHCP6 client incapability and IPv4 dependency of OSes.

4.2.2.1. Waiting timeout of DHCP4 in Windows 7

In Windows 7, timeout of DHCP4 queries spent a few minutes in the initial Wi-Fi connection setup. After fallback on the initial Wi-Fi connection, there were no problem on using IPv6 capable applications. DNS64 fallback failures due to the inappropriate authoritative servers still occurred, however, several authoritative servers, that returned inappropriate AAAA reply in past experiments, had been fixed to have appropriate fallback.

4.2.2.2. Long TCP fallback in Mac OS X Lion and Mountain Lion

Mac OS X implementations, such as Lion and Mountain Lion, had more serious timeout / fallback problems than Windows 7. After the timeout of DHCP4 queries with a few minutes as well as Windows 7, the interface that is allocated IPv4 link local address was inserted as IPv4 default route. This Mac OS X behavior may be along with IPv4 on-link assumption in Section 3.3 of [RFC3927]. Section 3.3 of RFC3927 mentions "Interaction with Hosts with Routable Addresses", which assumes all IPv4 address are on-link at Link-Local configuration.

Also, getaddrinfo implementation on Mac OS X did HappyEyeball like behavior. The getaddrinfo of Mac OS X returned an IP address list where IPv4 addresses were inserted the top of the list initially. Combining the on-link-assumption and the HappyEyeball like getaddrinfo caused long long TCP fallback from IPv4 to IPv6 in the initial TCP connection setup. Once the long long TCP fallback occurred, getaddrinfo of Mac OS X marked some flag that IPv4 is not available at the moment, then the getaddrinfo gave higher priority to IPv6 addresses than IPv4 addresses until ARP and / or ND tables were refreshed. When ARP and / or ND tables were refreshed, Mac OS X users face long long TCP fallback from IPv4 to IPv6 again.

4.2.2.3. Incompletion of network settings in iOS 5

In iOS 5, "Network Setting" were not completed, "Network Settings" will be completed only if IPv4 address, IPv4 router, and IPv4 DNS can be retrieved via DHCPv4 or manually configured all of these 3.

4.2.2.4. Incapability of IPv6 DNS settings by DHCP6

Windows XP, older Mac OS X (Snow Leopard and older) and Android OS required an IPv4 address for an DNS server even when they can use IPv6. In an IPv6 only network, DNS information should be gotten via DHCP6, these OSes did not support DHCP6 client. Also, Android cannot be configured to use DNS over IPv6 even in manual configuration.

4.2.3. Experiment 1

4.2.3.1. Diff of network settings

In the Experiment 1, we added a DHCP4 server that provided only IPv4 private address to DHCP4 client without the default gateway IPv4 address nor IPv4 address of DNS. We employed ISC-DHCP for this DHCP4 server.

```
                                              +-------+ +------+
                                              | DNS64 | | NAT64|
                                              +-------+ +------+
                                                  |        |
                                                (-- StarBED --)
                                                  |        |
       +--------------- IPv6 Internet ----------------------+
                              |
                      +-------------+      +----------------+
                      | IPv6 router |      | DHCP PD server |
                      |   on ISP    |      |    on VNE      |
                      +-------------+      +----------------+
                              |                    |
    +-- (VNE network) ----------------+----------------------+
                                      |
                                      |(v6)
                                      |
                               (---- Hotel ----)
                                      |
                                +---------------+
                                | DHCP-PD Client|
                                |  PC router /  |
                                |    DHCP4      |
                                +---------------+
                                      |
      +-----------------+             |
      | Stateless DHCP6 |             |
      +-----------------+             |
                 |                    |
                 |                    |
       +------------- /64 prefix segment ---------------+
                                              |
                                      +---------------+
                                      | users devices |
                                      +---------------+
```

Test Topology on Experiment 1 (v6only-fallback)

Figure 4

4.2.3.2. Result

As the result of Experiment 1, only timeout of DHCP4 was solved, that is, only Windows 7 was working well without any fallback problems except for DNS64 name resolving. TCP fallback problem on MacOS X still occurred. iOS applications were sometimes working, but periodically failed due to retrying Wi-Fi connection setup.

4.2.4. Experiment 2

4.2.4.1. Diff of network settings

In the Experiment 2, we put BIND9 forwarder on-link and configured DHCP4/6 to use this DNS. We configured BIND9 forwarder with: * deny-answer-addresses { 0.0.0.0/0; }; * which directed that no IPv4 address answer should be trusted. It returned SERVFAIL to resolvers.

```
                                              +-------+ +------+
                                              | DNS64 | | NAT64|
                                              +-------+ +------+
                                                  |        |
                                                (-- StarBED --)
                                                  |        |
       +--------------- IPv6 Internet ----------------------+
                              |
                      +-------------+      +----------------+
                      | IPv6 router |      | DHCP PD server |
                      |   on ISP    |      |    on VNE      |
                      +-------------+      +----------------+
                              |                    |
    +-- (VNE network) ----------------+----------------------+
                                      |
                                      |(v6)
                                      |
                               (---- Hotel ----)
                                      |
                                +------------------+
                                | DHCP-PD Client   |
                                |  PC router /     |
                                |    DHCP4 /       |
                                | Bind 9 forwarder |
                                | which treats "A" |
                                | as SERVAFAIL     |
                                +------------------+
                                      |
      +-----------------+             |
      | Stateless DHCP6 |             |
      +-----------------+             |
                 |                    |
                 |                    |
       +------------- /64 prefix segment ---------------+
                                              |
                                      +---------------+
                                      | users devices |
                                      +---------------+
```

Test Topology on Experiment 2 (v6only-fallback)

Figure 5

4.2.4.2. Result

As result of Experiment 2, Android was working well. iOS was working, but periodically failed due to retrying to Wi-Fi connection setup. MacOS X variants were working, but timeout by TCP fallback still occurred. Windows XP was not working because all DNS queries failed due to SERVFAIL.

4.2.5. Experiment 3

4.2.5.1. Diff of network settings

In the Experiment 3, we hacked AAAA filtering code on BIND9 to filter "A records" instead of "AAAA records" both on IPv4/IPv6 transport. We put BIND9 above to the local link, which was configured to forward all queries to DNS64. We also configured DHCP4/DHCP6 to use the DNS proxy.

```
                                              +-------+ +------+
                                              | DNS64 | | NAT64|
                                              +-------+ +------+
                                                  |        |
                                                (-- StarBED --)
                                                  |        |
       +--------------- IPv6 Internet ----------------------+
                              |
                      +-------------+      +----------------+
                      | IPv6 router |      | DHCP PD server |
                      |   on ISP    |      |    on VNE      |
                      +-------------+      +----------------+
                              |                    |
    +-- (VNE network) ----------------+----------------------+
                                      |
                                      |(v6)
                                      |
                               (---- Hotel ----)
                                      |
                                +-------------------+
                                | DHCP-PD Client    |
                                |  PC router /      |
                                |    DHCP4 /        |
                                | Bind 9 DNS Proxy  |
                                |  with "A" filter  |
                                +-------------------+
                                      |
      +-----------------+             |
      | Stateless DHCP6 |             |
      +-----------------+             |
                 |                    |
                 |                    |
       +------------- /64 prefix segment ---------------+
                                              |
                                      +---------------+
                                      | users devices |
                                      +---------------+
```

Test Topology on Experiment 3 (v6only-fallback)

Figure 6

4.2.5.2. Result

As the result of Experiment 3, Windows XP, MacOS X variants, iOS, Android were working well. Some of applications still failed on IPv6 only due to the IPv6 incapability or DNS64 fallback problem, but many cases were fine: IE/Safari/Chrome/Firefox, Twitter, Facebook, Instagram, and so on.

Remaining issues were connection failures during a few minutes after Wi-Fi was connected. We guess the possible reason of this failures as follows: RS (Router Solicitation) was sent from kernel before Wi-Fi link was established. No IPv6 address was obtained until periodical RA (Router Advertisement) was received. The possible workaround to this connection failure is shortening RA interval to 5-10 seconds (though it disturb Wi-Fi ...) or detecting association through AP log and kicking RS or RA.

5. Conclusion

Timeout / fallback problems on IPv4/IPv6 dual stack clients in an IPv6 only network are caused by

1. timeout and fallback sequence on DHCP4 queries,

2. timeout and fallback sequence on the connectivity check to the IPv4 internet after the DHCP4 auto configuration,

3. connection retry sequence when the connectivity to the IPv4 internet was not given.

4. timeout and fallback sequence of a TCP connection on Mac OS X variants due to their HappyEyeball like behavior of getaddrinfo,

5. preference / dependency of IPv4 on name resolution,

6. connection failures during 1-2 minutes after Wi-Fi was connected.

To mitigate these timeout / fallback problems, our current practice is composed of following components;

o Configure a DNS64 and a NAT64 in somewhere.

o Configure a Dual-stack DNS proxy as follows

   * The DNS proxy forwards all queries to the DNS64 except "A" query type (IPv4 address). Since there is no IPv4 connectivity on the client, all queries to "A" should be filtered and the DNS proxy returns NO DATA, just like "AAAA" filtering.

   * This "A" filter should be enabled both on IPv4 and IPv6 transport.

  - This Dual-stack "A" filter DNS proxy should be "on-link" and reachable from IPv4/IPv6 dual stack mode clients.

- Configure a DHCP4 server to reply a private IPv4 address, an IPv4 gateway router, and IPv4 address of an "A" filter DNS proxy to DHCP4 client.

- Configure a DHCP6 server to indicate the IPv6 address of "A" filter DNS Proxy to DHCP6 client.

  - The IPv6 address of "A" filter DNS Proxy may be provided to IPv4/IPv6 dual stack mode clients by RDNSS [RFC6106]. However, from our experience on hot stage of Camp 1209 Autumn, Mac OS X Lion and Mountain Lion could handle RDNSS, but Windows 7 did not handle RDNSS.

  - Only one DHCP6 server should be placed in each /64 prefix segment or indicated by DHCP6 relay. According to our experience, we do not recommend overwriting DNS information by a local stateless DHCP6 server with highest preference value due to the differences of handling multiple DHCP6 replies among DHCP6 client implementations.

- Configure the IPv4 gateway router not to forward any IPv4 packets.

6. Security Considerations

As well as Arkko mentioned in [RFC6586], the use of IPv6 instead of IPv4 by itself does not make a big security difference. In our experience, we only set up following security functions; the access control list on routers / servers, accounting on the wireless network access.

7. IANA Considerations

This document has no IANA implications.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2663] Srisuresh, P. and M. Holdrege, "IP Network Address

Translator (NAT) Terminology and Considerations", RFC 2663, August 1999.

[RFC3736] Droms, R., "Stateless Dynamic Host Configuration Protocol (DHCP) Service for IPv6", RFC 3736, April 2004.

[RFC4213] Nordmark, E. and R. Gilligan, "Basic Transition Mechanisms for IPv6 Hosts and Routers", RFC 4213, October 2005.

8.2. Informative References

[I-D.draft-ietf-softwire-map-02] Troan, O., Bao, C., Matsushima, S., and T. Murakami, "Mapping of Address and Port with Encapsulation (MAP)", September 5, 2012, <draft-ietf-softwire-map-02 (work in progress)>.

[I-D.draft-ietf-v6ops-464xlat] Mawatari, M., Kawashima, M., and C. Byrne, "464XLAT: Combination of Stateful and Stateless Translation", September 2012, <draft-ietf-v6ops-464xlat-08 (work in progress)>.

[I-D.draft-matsuhira-sa46t-at-00] Matsuhira, N., Horiba, K., Ueno, Y., and O. Nakamura, "SA46T Address Translator", July 2011, <draft-matsuhira-sa46t-at-00 (work in progress)>.

[I-D.draft-matsuhira-sa46t-spec] Matsuhira, N., "Stateless Automatic IPv4 over IPv6 Tunneling: Specification", July 2012, <draft-matsuhira-sa46t-spec-05 (work in progress)>.

[I-D.draft-murakami-softwire-4rd] Murakami, T., Troan, O., and S. Matsushima, "Stateless Automatic IPv4 over IPv6 Tunneling: Specification", September 2011, <draft-murakami-softwire-4rd-01 (work in progress)>.

[RFC0894] Hornig, C., "Standard for the transmission of IP datagrams over Ethernet networks", STD 41, RFC 894, April 1984.

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.

[RFC2516] Mamakos, L., Lidl, K., Evarts, J., Carrel, D., Simone, D., and R. Wheeler, "A Method for Transmitting PPP Over Ethernet (PPPoE)", RFC 2516, February 1999.

[RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

[RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", RFC 3633, December 2003.

[RFC3927] Cheshire, S., Aboba, B., and E. Guttman, "Dynamic Configuration of IPv4 Link-Local Addresses", RFC 3927, May 2005.

[RFC4074] Morishita, Y. and T. Jinmei, "Common Misbehavior Against DNS Queries for IPv6 Addresses", RFC 4074, May 2005.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[RFC5942] Singh, H., Beebee, W., and E. Nordmark, "IPv6 Subnet Model: The Relationship between Links and Subnet Prefixes", RFC 5942, July 2010.

[RFC6052] Bao, C., Huitema, C., Bagnulo, M., Boucadair, M., and X. Li, "IPv6 Addressing of IPv4/IPv6 Translators", RFC 6052, October 2010.

[RFC6106] Jeong, J., Park, S., Beloeil, L., and S. Madanapalli, "IPv6 Router Advertisement Options for DNS Configuration", RFC 6106, November 2010.

[RFC6144] Baker, F., Li, X., Bao, C., and K. Yin, "Framework for IPv4/IPv6 Translation", RFC 6144, April 2011.

[RFC6145] Li, X., Bao, C., and F. Baker, "IP/ICMP Translation Algorithm", RFC 6145, April 2011.

[RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, April 2011.

[RFC6147] Bagnulo, M., Sullivan, A., Matthews, P., and I. van Beijnum, "DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers", RFC 6147, April 2011.

[RFC6384] van Beijnum, I., "An FTP Application Layer Gateway (ALG) for IPv6-to-IPv4 Translation", RFC 6384, October 2011.

[RFC6586] Arkko, J. and A. Keranen, "Experiences from an IPv6-Only Network", RFC 6586, April 2012.

[RFC6603] Korhonen, J., Savolainen, T., Krishnan, S., and O. Troan, "Prefix Exclude Option for DHCPv6-based Prefix Delegation", RFC 6603, May 2012.

[YasudaAPRICOT2011] Yasuda, A., "Building for IPv6 by IPv6 Promotion Council Japan.", February, 2011, <http://meetings.apnic.net/__data/assets/pdf_file/0003/30981/Ayumu-Yasuda-apricot.pdf>.

Appendix A. Acknowledgments

Here, we thank to all the participants of WIDE camp on the experiments. We also say thank you to whom serving implementations and services in the Matsushiro Royal Hotel.

R. Nakamura of Univ. of Tokyo, Y. Ueno of Keio Univ. and R. Shouhara of Univ. of Tokyo for helping us on the base settings of the IPv6 only experiments and merging into the camp-net.

O. Onoe of Sony Corporation for his deep inspection and testing of end node devices.

T. Jimei of Internet Systems Consortium for his quick hack on A filter of Bind 9.

Y. Atarashi of Alaxala Networks and R. Atarashi of IIJ Innovation Institute for designing the items of face to face interview and analyzing user survey data.

Authors' Addresses

Hiroaki Hazeyama
NAIST
Takayama 8916-5
Nara,
Japan

Phone: +81 743 72 5216
Email: hiroa-ha@is.naist.jp

Ruri Hiromi
Intec Inc.
1-3-3 Shin-Suna, Koutou
Tokyo,
Japan

Email: hiromi@inetcore.com

Tomohiro Ishihara
Univ. of Tokyo
3-8-1 Komaba, Meguro
Tokyo,
Japan

Email: sho@c.u-tokyo.ac.jp

Osamu Nakamura
WIDE Project
5322 Endo
Kanagawa,
Japan

Email: osamu@wide.ad.jp

B Bind 9 A filter Patch and Sample Configuration

B.1 Bind 9.9.2-P1 A filter patch

```
diff -ru bind-9.9.2-P1.org/bin/named/query.c bind-9.9.2-P1.a-filter/bin/named/query.c
--- bind-9.9.2-P1.org/bin/named/query.c 2012-10-26 13:50:34.000000000 +0900
+++ bind-9.9.2-P1.a-filter/bin/named/query.c 2013-02-23 19:14:08.000000000 +0900
@@ -1348,7 +1348,7 @@

  if (qtype == dns_rdatatype_a) {
 #ifdef ALLOW_FILTER_AAAA_ON_V4
-isc_boolean_t have_a = ISC_FALSE;
+ isc_boolean_t have_aaaa = ISC_FALSE;
 #endif

  /*
@@ -1390,7 +1390,7 @@
  if (result == ISC_R_SUCCESS) {
  mname = NULL;
 #ifdef ALLOW_FILTER_AAAA_ON_V4
-have_a = ISC_TRUE;
+ have_aaaa = ISC_TRUE;
 #endif
  if (!query_isduplicate(client, fname,
         dns_rdatatype_a, &mname)) {
@@ -1444,7 +1444,7 @@
   * There's an A; check whether we're filtering AAAA
   */
 #ifdef ALLOW_FILTER_AAAA_ON_V4
-if (have_a &&
+ if (have_aaaa &&
      (client->filter_aaaa == dns_v4_aaaa_break_dnssec ||
      (client->filter_aaaa == dns_v4_aaaa_filter &&
       (!WANTDNSSEC(client) || sigrdataset == NULL ||
@@ -5224,6 +5224,7 @@
 #ifdef ALLOW_FILTER_AAAA_ON_V4
 static isc_boolean_t
 is_v4_client(ns_client_t *client) {
+ return (ISC_TRUE);
  if (isc_sockaddr_pf(&client->peeraddr) == AF_INET)
  return (ISC_TRUE);
  if (isc_sockaddr_pf(&client->peeraddr) == AF_INET6 &&
@@ -6756,7 +6757,7 @@

  if (type == dns_rdatatype_any) {
 #ifdef ALLOW_FILTER_AAAA_ON_V4
-isc_boolean_t have_aaaa, have_a, have_sig;
+ isc_boolean_t have_a4, have_aaaa, have_sig;

  /*
   * The filter-aaaa-on-v4 option should
@@ -6765,8 +6766,8 @@
   * even in if it is not in our cache.  This assumption could
   * be wrong but it is a good bet.
   */
-have_aaaa = ISC_FALSE;
```

```
-have_a = !authoritative;
+ have_a4 = ISC_FALSE;
+ have_aaaa = !authoritative;
  have_sig = ISC_FALSE;
 #endif
  /*
@@ -6803,10 +6804,10 @@
   * that AAAAs can be hidden from IPv4 clients.
   */
  if (client->filter_aaaa != dns_v4_aaaa_ok) {
-if (rdataset->type == dns_rdatatype_aaaa)
+ if (rdataset->type == dns_rdatatype_a)
+ have_a4 = ISC_TRUE;
+ else if (rdataset->type == dns_rdatatype_aaaa)
  have_aaaa = ISC_TRUE;
-else if (rdataset->type == dns_rdatatype_a)
-have_a = ISC_TRUE;
  }
 #endif
  if (is_zone && qtype == dns_rdatatype_any &&
@@ -6866,7 +6867,7 @@
  if (client->filter_aaaa == dns_v4_aaaa_break_dnssec)
  client->attributes |= NS_CLIENTATTR_FILTER_AAAA;
  else if (client->filter_aaaa != dns_v4_aaaa_ok &&
- have_aaaa && have_a &&
+  have_a4 && have_aaaa &&
   (!have_sig || !WANTDNSSEC(client)))
    client->attributes |= NS_CLIENTATTR_FILTER_AAAA;
 #endif
@@ -6934,10 +6935,10 @@
       (!WANTDNSSEC(client) || sigrdataset == NULL ||
       !dns_rdataset_isassociated(sigrdataset))))
  {
-if (qtype == dns_rdatatype_aaaa) {
+ if (qtype == dns_rdatatype_a) {
  trdataset = query_newrdataset(client);
  result = dns_db_findrdataset(db, node, version,
-     dns_rdatatype_a, 0,
+      dns_rdatatype_aaaa, 0,
       client->now,
       trdataset, NULL);
  if (dns_rdataset_isassociated(trdataset))
@@ -6977,7 +6978,7 @@
   * if the recursion for the A succeeds.
   */
  result = query_recurse(client,
-dns_rdatatype_a,
+ dns_rdatatype_aaaa,
  client->query.qname,
  NULL, NULL, resuming);
  if (result == ISC_R_SUCCESS) {
@@ -6988,7 +6989,7 @@
  }
  }

-} else if (qtype == dns_rdatatype_a &&
+ } else if (qtype == dns_rdatatype_aaaa &&
     (client->attributes &
      NS_CLIENTATTR_FILTER_AAAA_RC) != 0) {
  client->attributes &=
Only in bind-9.9.2-P1.a-filter/bin/pkcs11: Makefile
Only in bind-9.9.2-P1.a-filter/bin/python: dnssec-checkds.py
Only in bind-9.9.2-P1.a-filter/bin/python: Makefile
Only in bind-9.9.2-P1.a-filter/bin/tests/system/dlz: prereq.sh
Only in bind-9.9.2-P1.a-filter/bin/tests/system/ecdsa: prereq.sh
Only in bind-9.9.2-P1.a-filter/bin/tests/system/gost: prereq.sh
Only in bind-9.9.2-P1.a-filter/bin/tests/virtual-time: conf.sh
Only in bind-9.9.2-P1.a-filter/bin/tests/virtual-time: Makefile
Only in bind-9.9.2-P1.a-filter/contrib: check-secure-delegation.pl
Only in bind-9.9.2-P1.a-filter/contrib: zone-edit.sh
diff -ru bind-9.9.2-P1.org/lib/dns/message.c bind-9.9.2-P1.a-filter/lib/dns/message.c
--- bind-9.9.2-P1.org/lib/dns/message.c 2012-10-26 13:50:34.000000000 +0900
+++ bind-9.9.2-P1.a-filter/lib/dns/message.c 2013-02-23 18:36:17.000000000 +0900
@@ -1811,14 +1811,14 @@
 norender_rdataset(const dns_rdataset_t *rdataset, unsigned int options)
 {
  switch (rdataset->type) {
-case dns_rdatatype_aaaa:
+ case dns_rdatatype_a:
  if ((options & DNS_MESSAGERENDER_FILTER_AAAA) == 0)
  return (ISC_FALSE);
  break;

  case dns_rdatatype_rrsig:
  if ((options & DNS_MESSAGERENDER_FILTER_AAAA) == 0 ||
-    rdataset->covers != dns_rdatatype_aaaa)
+     rdataset->covers != dns_rdatatype_a)
  return (ISC_FALSE);
  break;

Only in bind-9.9.2-P1.a-filter/lib/dns/tests: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/dns/include/dns: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/dns/include/dst: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/dns/include: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/dns: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/irs/include/irs: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/irs/include: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/irs: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/isc/include/isc: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/isc/include: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/isc: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/isc/nls: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/isc/nothreads/include/isc: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/isc/nothreads/include: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/isc/nothreads: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/isc/unix/include/isc: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/isc/unix/include: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/isc/unix: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/isccfg/include/isccfg: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/isccfg/include: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/isccfg: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/samples: Makefile
Only in bind-9.9.2-P1.a-filter/lib/export/samples: Makefile-postinstall
Only in bind-9.9.2-P1.a-filter/lib/irs/include/irs: Makefile
Only in bind-9.9.2-P1.a-filter/lib/irs/include/irs: netdb.h
Only in bind-9.9.2-P1.a-filter/lib/irs/include/irs: platform.h
Only in bind-9.9.2-P1.a-filter/lib/irs/include: Makefile
Only in bind-9.9.2-P1.a-filter/lib/irs: Makefile
Only in bind-9.9.2-P1.a-filter/lib/isc/tests: Makefile
```

B.2 named.conf for DNS forwarder by Bind 9.9.2-P1 A filter patch

```
key "rndc-key" {
        algorithm hmac-md5;
        secret "HXE6UugyLhLiRcHttJ2R7e7rEwnDt6x2pjE38aB3EH8=";
};

controls {
        inet 127.0.0.1 port 953
        allow { 127.0.0.1; } keys { "rndc-key"; };
};

logging {
        channel remote_log {
                severity info;
                print-category yes;
                print-severity yes;
                syslog local1;
        };
        category resolver { remote_log; };
        category security { remote_log; };
        category queries { remote_log; };
        category default { remote_log; };
};

options {
        directory "/etc/namedb";

allow-recursion { 203.178.136.0/26; 127.0.0.1; ::1; 10.0.0.0/8; 2409:12:6080:102::/64; };
        recursion yes;
        pid-file        "/var/run/named/pid";
        dump-file       "/var/dump/named_dump.db";
listen-on-v6 { any; };
// deny-answer-addresses { 0.0.0.0/0; };
forward only;
forwarders { 2001:200:0:ff10::4 port 53; };
filter-aaaa-on-v4 yes;
filter-aaaa { any; };
/*
dns64 2001:200:0:ff64::/96 {
clients {
::1;
};
mapped { !rfc1918; any; };
suffix ::;
recursive-only yes;
break-dnssec yes;
};
*/
};

acl rfc1918 { 10/8; 192.168/16; 172.16/12; };


zone "." {
type hint;
file "named.root";
};


zone "rpz.camp.wide.ad.jp" {
type master;
file "rpz.camp.wide";
allow-query { any; };
};
```