Evaluation of prerequisites for an IPv4 to IPv6 transition

Bachelor Thesis in Network Technology

Author: Elias Johansson
Supervisor: Svetlana Girs
Examiner: Moris Behnam

School of Innovation, Design and Technology
Mälardalen University
Västerås, Sweden
2016-06-09
Abstract

The increase in the number of internet capable devices has starved the Internet Protocol Version 4 (IPv4) address space. IPv6 is likely to replace IPv4 in the future because of an address space much greater than before. This thesis work focuses on how to transition from IPv4 to IPv6 for a specific network. This work includes a study of the most common transitioning techniques and an investigation of the characteristics of the network in question, Office IT-partner’s network. The main problem of the thesis is, based on the knowledge of existing transitioning techniques, to find the best fitting solution. The three best solutions were chosen for theoretical testing on the network. The solutions were tested with focus on network services, addressing, simplicity and future completion of the transition. The testing made it clear that no solution alone solves all problems, but dual stack together with translation satisfy the most requirements. The conclusion drawn from the literature study and the testing is that transitioning to IPv6 without the use of translation causes many complex problems. To successfully transition while providing IPv4 connectivity for WAN nodes, some kind of translation is almost required. A solution consisting of dual stack and Network Address Translation 64 (NAT64) was chosen because of its ability to make the transition easy and as user-friendly as possible. It could also be started right away and completed at a pace of own choice. The lack of ability to translate stateful from IPv4 to IPv6 together with the lack of availability for IPv4 nodes made up the decision of upgrading the internal network before upgrading the internet edge.
# Table of contents

1. **Introduction** .......................................................... 1
   1.1 Problem formulation .............................................. 1
   1.2 Method ............................................................. 1

2. **IPv4 and IPv6** ........................................................ 2
   2.1 IPv6 ...................................................................... 2
   2.2 Addressing .......................................................... 3
   2.3 Transitioning ......................................................... 4

3. **Transition techniques** ............................................... 5
   3.1 Dual Stack ............................................................ 5
   3.2 Tunneling techniques .............................................. 5
   3.3 Translation techniques ............................................ 8
   3.4 Experimental transition techniques ......................... 12

4. **Office IT-partner network** .......................................... 13
   4.1 Structure and characteristics .................................. 13
   4.2 Equipment .......................................................... 14
   4.3 Service Providers .................................................. 15

5. **Selection of a transition method** ................................ 15
   5.1 Dual Stack ............................................................ 15
   5.2 NAT64 .................................................................. 16
   5.3 BIH ...................................................................... 17
   5.4 Conclusion & protocol selection ............................... 17

6. **Discussion** ................................................................ 18

7. **Conclusion** .............................................................. 19

References ................................................................. 20
1 Introduction
In today’s world a lot of things rely on the internet, the whole society would crumble without it. Everything from bank errands to controlling a smart home is done via the internet. Every device connected to the internet needs some kind of identification, an address. In the current Internet Protocol (IP) version 4 [1] the address size is limited to four binary octets. As the internet has grown bigger with more connected devices, the address space of ~4.3 billion addresses is now crowded and there is a small amount of addresses left unassigned. This is stopping further development and expansion of the internet in terms of smart units. The solution to this is IPv6, that has an address space of $2^{128}$ addresses in comparison with $2^{32}$ in IPv4 [1]. This thesis focuses on how to do the transition from IPv4 to IPv6, studies existing transition methods and the ways they should be used, chooses the best one for a specific network.

The network considered in this thesis is the network of a Swedish company named Office IT-partner. Office IT-partner is planning on a transition to IPv6 in order to avoid addressing problems and securing future operation of their network. Office IT-partner delivers IT-solutions to other companies in terms of hosting different services, administrating their IT-departments and providing client-solutions like printers and workstations including technical support.

The thesis includes a comparison between known transitioning techniques in terms of situations they can be used in, their scalability and other characteristics. The comparison is a base for the evaluation of Office IT-partner’s network. Different techniques are theoretically tested on the network to determine which method is the best suited for the network to continue operating and making the transition transparent to the users.

1.1 Problem formulation
The goal of the work is to investigate the prerequisites for a transition from an IPv4 network to an IPv6 network, to find the most suitable transitioning scheme for a particular network. The thesis is done together with Office IT-Partner and their network is the subject for investigation. The following questions need to be answered throughout the work:

- Which transition methods already exist and what are their pros and cons?
- What are the characteristics of Office IT-Partner’s network?
- Taking its specific characteristics into account, what is the best way to do a migration in Office IT-Partner’s network?
- How should this migration be done? All at once? Step-by-step?

1.2 Method
The work on the thesis is divided into three main parts. The first part consists of a literature study of all the common existing transition techniques and a study of the Office IT-partner network. The techniques are compared to get an overview of their best use cases. Studying the network means analyzing the capabilities of the equipment, evaluating the addressing for a future addressing plan and studying the size and design. All of these parts make up to enough knowledge to eliminate some of the techniques.

The second stage is the actual application of the studies. This means theoretically “testing” the appropriate techniques to get some cases to compare and evaluate. Before testing, the techniques
will be narrowed down using the study of the current network. Testing too many techniques would be time consuming. If a test does not show appealing results, another technique is tested and so on. All the narrowed techniques are tested to be able to provide options in the recommendation.

The third part is the evaluation part. By using the tests in the stage before the best method for the network is chosen. The best method is evaluated to create the recommendations for the company. A deployment strategy combined with a proposed addressing plan is presented including recommendations for future development.

2 IPv4 and IPv6

As internet development proceeds, the need for a transition to IPv6 is getting more imminent. IPv6 does not only bring the advantage of a bigger address space, but also many more benefits and features [2, pp. 692-693]. The biggest issues with IPv4 are the lack of available addresses and the big global routing tables. With more and more nodes connecting to the internet the routing tables just continue to grow, which makes devices very dependent on powerful hardware. This powerful hardware is expensive and makes a downside in the long run. Another problem is the wide use of Network Address Translation (NAT), which goes against the aim of having an end-to-end model on the internet. End-to-end means that two end nodes can communicate without any impact from intermediate nodes like address translation. This creates problems for some applications that cannot handle a data flow where addresses are being translated along the way. Clients behind a NAT become sort of anonymous, which is good from a security point of view but does not fulfill the vision of an end-to-end internet.

2.1 IPv6

IPv6 comes with many improvements to IPv4 beyond the larger address space [2, pp. 693-694]. The absence of broadcasts eliminates a lot of excessive traffic to nodes that have no interest in receiving it. IPv6 uses unicast and multicast addresses just like IPv4, but introduces the anycast address. The anycast address is a one-to-nearest type, e.g. a company with web servers in locations can have the same address to all of them and IPv6 makes sure that the closest one of the available is the one that is contacted. Anycast is a great feature when it comes to high availability since it can eliminate use of protocols for managing virtual addresses.

As shown in Figure 1 the IPv6 header is less complex and brings more efficient routing with e.g. the absence of checksum calculation [2, p. 693]. The flow label field eliminates the need for examination of higher layers in order to know if received packets belong to the same flow.
IPv6 also introduces the use of Mobile IP which has been around for a while and is supported by IPv4, but is not built-in IPv4 like it is in IPv6 [2, p. 693]. The main function of Mobile IP is to have the same address on a device regardless of its geographical location [4]. Native IPsec support is a feature that all nodes with an IPv6 stack can utilize without the need for any extensions since it is protocol mandatory [2, p. 693].

2.2 Addressing

As mentioned earlier the IPv6 address space consists of $2^{128}$ addresses which is a lot more than the $2^{32}$ that is provided by IPv4. The addresses are represented by 8 “hextets”, which means 8 fields consisting of four hexadecimal digits [2, pp. 700-701]. An example of an address is: 0001:ABCD:0501:ADDD:0000:0000:0000:10F3. The addresses can be shortened with two rules. At first, leading 0s in every hextet can be removed since it will be interpreted as 0s before if a hextet is less than 4 digits. Coherent hextets including only 0s can be shortened by using (::). It would look like this: 1:ABCD:501:ADDD::10F3 with both rules applied. The second rule has the constraint that it can only be used once in an address, since it would be impossible to know how many hextets of zeroes there are behind each (::) in that case.

The address is separated into two parts, where the last 64 bits is the Interface Identifier (ID), comparable to the host portion of a subneted IPv4 address [2, pp. 701-702]. These can be assigned completely manually, or, like in the case where Ethernet is the data-link protocol, the Media Access Control (MAC) address can be used for this. This is achieved by taking the 48 bit Mac address and inserting FFFE in the middle, between the Organizationally Unique Identifier (OUI) and the “serial number”. When using MAC addresses to assign the interface identifier, the MAC address is expected to be globally unique. This is called Extended Unique Identifier (EUI)-64. The first 64 bits consist of the global routing prefix and the subnet portion.
2.2.1 Address types
As mentioned before, the main address types adopted in IPv6 are unicast, multicast and anycast [2, pp. 704-707]. Unicast addresses can be divided into two types, global and link-local. A global unicast address is comparable to a public IPv4 address: globally unique, routable and is the main portion of the address space. The link local addresses are limited to the link and are used for automatic address assignment, neighbor discovery and by most routing protocols. The link-local addresses use the FE10::/10 prefix and a 64 bit interface identifier.

The multicast addresses work similar to their IPv4 equivalents and are assigned from with the FF00::/8 prefix. An example is FF02::1 which is a multicast to all nodes on the link [2, pp. 708-709].

2.2.2 Address assignment methods
IPv6 offers some more methods for addressing devices in comparison to IPv4. This makes the addressing more customizable.

Static Address Assignment
Assigning addresses statically is as simple as it sounds; an administrator configures each device with a global unicast address [2, pp. 719-721]. The address can be configured completely manually or by using the EUI-64 method for the interface identifier. IPv6 interfaces can be assigned with multiple addresses, for different purposes. These addresses can be used simultaneously.

Stateless Address Autoconfiguration
Stateless Address Autoconfiguration (SLAAC) is new to IPv6 and is a bit like Dynamic Host Configuration Protocol (DHCP) without a server [2, pp. 724-728]. Nodes can automatically configure themselves by appending an EUI-64 interface identifier to the 64 bit prefix of the local link. At first, nodes discover the data link layer address of each other by performing neighbor discovery. During neighbor discovery, the nodes use a 64 bit prefix announced on the link to append the EUI-64 identifier and automatically configure themselves with an address. For this to work, a device on the link needs to have the capability of providing stateless autoconfiguration to other devices. A router is typically performing this function. Each device also needs an address on the appropriate interface, so that the other devices can discover the prefix.

Dynamic Host Configuration Protocol Version 6
Dynamic Host Configuration Protocol Version 6 (DHCPv6) is much alike the IPv4 version [5]. The major difference is the absence of broadcasts which implies that the client and server use the neighbor discovery protocol to communicate at first. DHCPv6 is not capable of providing a default-gateway to clients, this is learned as next-hop during the neighbor discovery process.

2.3 Transitioning
A transition from IPv4 to IPv6 can be performed in three main ways: Dual Stack, Tunneling and Translation [2, p. 826]. Dual stack means that devices run both an IPv4 and an IPv6 stack at the same time, implying that two separate networks run at the same time. Tunneling can be used in two major cases, with isolated IPv4 networks or isolated IPv6 networks. The traffic from an isolated network to another is tunneled over a backbone running the other protocol without any impact on the backbone. Translation means to translate packets between IPv4 and IPv6 at different points in the network. Two networks running different protocols can be connected with a translating device in between.
3 Transition techniques
The following sections will describe different types of existing transition techniques and their characteristics. The techniques will also be compared to each other and the best ones are chosen for testing on the Office IT-partner network.

3.1 Dual Stack
Dual stack [2, p. 826] implies that a node has both IPv4 and IPv6 connectivity at the same time. This allows for the “old” IPv4 applications to be run on IPv4 and provides time to upgrade those applications with IPv6 support. During this time programmers can implement a new Application Programming Interface API to make use of both IPv4 and IPv6. This is a long term solution to a transition.

Dual-stack has the big advantage that it does not have any impact on the existing IPv4 network when used. At the same time, it brings extra overhead and is more resource intensive than running only one protocol. Running dual stack could imply that some hardware needs to be upgraded depending on the size and the load of the network. Consequently, dual stack is a technique for internal networks more than for e.g. provider networks since it is not scalable to have a “dual stack internet”.

3.2 Tunneling techniques
The purpose of tunneling is to encapsulate IPv4 in IPv6 or vice versa [2, p. 828]. This allows for IPv4 devices to communicate over an upgraded IPv6 network or IPv6 nodes to communicate over a legacy IPv4 network. Tunneling is often performed router-router or router-host but can also be host-host. The protocol encapsulated is called the passenger protocol and the protocol that the tunnel is created over is called the transport protocol. The most commonly used protocols for tunneling IPv6 traffic over an IPv4 network are 6rd and ISATAP.

3.2.1 6to4
6to4 tunneling [6] is a technique for tunneling IPv6 over the IPv4 internet or any other IPv4 network. An example is to connect two IPv6 company sites over the IPv4 internet. This is designed for connecting sites, but could as well be applied to a single host as long as it has a public IPv4 address. The sites connected like this can use any IPv6 addresses with the correct prefix; they do not have to embed IPv4 addresses. The actual mechanism is implemented at border routers and is transparent to the network users.

The specific prefix used for addressing the sites that are using 6to4 is 2002::/48. For example, for a company with the public IPv4 address 50.5.5.5 the 6to4 IPv6 prefix would be 2002:50.5.5.5::/48 (50.5.5.5 in hexadecimal) [6].

The actual tunneling is made through the outbound IPv4 connection [6]. The IPv6 packets are encapsulated in IPv4 packets with the protocol type field set to 41. If the Maximum Transition Unit (MTU) size of the IPv6 packets is larger than the MTU size for the IPv4 connection, fragmentation will occur. The 6to4 router knows if it receives an IPv6 packet destined for an address with e.g. the prefix 2002:192.168.0.1::/48 (192.168.0.1 in hexadecimal) it should be encapsulated in IPv4 and sent to the IPv4 address 192.168.0.1. DNS has to be used for the hosts to acquire the correct 6to4 addresses pointing to the node which they attempt to reach.
3.2.2 **ISATAP**

*Intra-Site Automatic Tunnel Addressing Protocol* (ISATAP) [7] is a protocol for tunneling IPv6 traffic over an IPv4 network. ISATAP requires the nodes to run dual stack. ISATAP uses a different address type from the other tunneling techniques, a 64 bit prefix followed by a 64 bit EUI-64 ID. The first 32 bits of the ID are the IANA OUI (00 00 5E) and the indication of an embedded IPv4 address (FE). The last 32 bits is the IPv4 address written in hexadecimal. An ISATAP tunnel is often created between a host and a router over an IPv4 network. When data is sent to an ISATAP node, the IPv4 address is extracted from the last 32 bits of the destination address. When it is received the receiver checks if the IPv4 address corresponds to the embedded IPv4 address in the IPv6 address of the encapsulated packet.

3.2.3 **6rd**

*IPv6 Rapid Deployment on IPv4 Infrastructures* (6rd) [8] is an automatic tunneling technique for implementing IPv6 to end nodes over an IPv4 provider network. 6rd uses 6to4 functionality with a slight difference that the providers’ IPv6 address is used instead of the well-known prefix. The 6rd domain is a combination of *Customer Edge* (CE) routers and *6rd Border Relays* (BRs). Traffic passes through BRs only when it is destined outside the 6rd domain. Beyond the CE’s IPv6 is implemented natively and the IPv6 data is encapsulated and decapsulated in the same way as in 6to4. The major difference between 6rd and 6to4 is the administration and scalability since the providers’ own IPv6 prefix can be used. If the CE acquires its IPv4 address dynamically with DHCP, a special option should be used that provides the CE node with information about the 6rd prefix and the IPv4 address to one or more BRs.

3.2.4 **6over4**

6over4 [9] describes a format for sending IPv6 packets over an IPv4 multicast domain between hosts. The underlying IPv4 network will be seen as a “virtual LAN” for the IPv6 hosts. This will make isolated IPv6 hosts able to communicate with each other over an IPv4 network. Additionally, this technique does not require IPv6 addresses that are “compatible” with IPv4. Similar to the techniques described above, fragmentation will occur if the IPv6 packets are larger than the IPv4 MTU size. Like the mechanism 6to4, the IPv6 packets/fragments are sent in IPv4 packets with the protocol field set to 41. 6over4 supports multicast traffic because the IPv4 network is required to have multicast functionality and IPv6 multicast addresses are mapped to IPv4 multicast addresses that start with 239.192; the two last bytes in the IPv6 address are put in the last two octets. To participate, a host needs a virtual interface with a link-local address, fe80::IPv4 address, which is an IPv4 address in hex with the fe80::/64 prefix. The fact that multicast is required in the IPv4 network makes 6over4 not quite useful when it comes to provider networks.

3.2.5 **Softwire Mesh Framework**

An IP backbone is often traversed by traffic that is initiated from outside of the actual backbone. For this to work, typically all routers are given information about external routes via *Border Gateway Protocol* (BGP). Another solution to this is to tunnel the traffic from one edge router to another, which will remove the need for BGP in the core of the backbone. Softwire mesh [10] is a technique for addressing the issues of having an IPv6-only or an IPv4-only backbone when traffic using the other protocol needs to traverse the backbone. This can be solved by using only a dual stack configuration at the edge routers in combination with a tunneling technique. Those routers are called
Address Family Border Routers (AFBRs). The edge routers communicate the routes via BGP, and for the Softwire to work, an IPv4 route needs an IPv6 next-hop address and vice versa.

3.2.6 Tunneling IPv6 over UDP through NATs
Tunneling IPv6 over UDP through NATs (Teredo) [11] makes IPv6 hosts behind IPv4 NATs able to communicate. These hosts receive a global address from a Teredo server and send data through a Teredo relay. There is a specific Teredo IPv6 prefix and an UDP port: 2001:0000:/32, 3544. A Teredo client contacts a Teredo server to learn the UDP port to use and the IPv4 address of the NAT that is the furthest away from the client. The prefix combined with the servers’ IPv4 address and the information the client has just learned makes up the Teredo client address. After that, two clients can communicate as long as the prefix is announced on the internet. The server only helps with “starting” the session, which means that it informs the client of which relay it should send upcoming data to. The traversal of potentially several NATs makes the technique a bit complex, but it works well for smaller implementations.

3.2.7 Dual-Stack Lite
Dual-Stack Lite [12] is a tunneling technique that allows ISPs to upgrade to an IPv6 backbone and still provide IPv4 internet connectivity to the customers without them replacing their equipment. Two existing mechanisms are used in Dual-Stack Lite, IP in IP and NAT. At the customer end, NAT is not performed like it usually is. Private addresses should be used and routed to the ISP as the NAT operation will be performed there. The device at the customer end is connected to the ISP with IPv6 and all the IPv4 traffic is encapsulated within a tunnel, called the B4 element of the IPv6 interface. The tunnel is terminated at an AFTR in the backbone that connects to the IPv4 internet. The AFTR is discovered using DHCPv6 with a specific option. IPv6 traffic from the customer is not encapsulated, it is sent natively over the backbone. The B4 interface is responsible for fragmenting the IPv6 packets containing IPv4 packets when they exceed the MTU of the link.

Tunneling summary & comparison
Tunneling is a good alternative when a part of a network, e.g., a backbone, is upgraded. It is best used by service providers and the customers of these providers to connect their sites over a backbone running another protocol than the sites. Looking at the router-to-router protocols, 6to4 is a simple technique but has the constraint of using the pre-defined prefix. It also relies on DNS to provide the correct 6to4-address to hosts when they are about to send data. The transparency for the users is an advantage, but the drawback here is the scalability issue that a pre-defined prefix brings. Additionally, 6to4 can only be used for IPv6 over an IPv4 network. 6rd takes the good parts from 6to4 and eliminates the scalability issue by using the provider’s prefix instead of a pre-defined one. The Softwire Mesh framework is a good technique for solving the problem with traffic using another protocol traversing a backbone network. The framework itself has no obvious drawbacks but the tunneling technique that is chosen will bring its own downsides.

Moving to the host-to-host techniques, ISATAP is a bit complicated but fully functional technique for hosts to use when e.g., a new IPv6 application is used on an “old” IPv4 network. Apart from ISATAP, 6over4 satisfies the same need for IPv6 connectivity for hosts on an IPv4 network. The downside for 6over4 is the requirement for a multicast IPv4 network. This could be a problem in provider networks. At last, the Teredo tunneling technique also works only for IPv6 hosts on an IPv4 network,
but provides the support for traversing even multiple NATs along the way. The big downside of Teredo is very complex operation with servers and relays which limits the size of the implementation.

3.3 Translation techniques

The main task of translation techniques is to convert IPv4 packets to IPv6 packets and vice versa, to allow IPv4-only nodes to communicate with IPv6-only nodes [2, p. 826]. Translation can be resource intensive and is recommended for long-term use when there is legacy equipment that is not planned to be upgraded to IPv6.

3.3.1 Stateless IP/ICMP Translation

Stateless IP/ICMP Translation (SIIT) [13] is a transition technique built on the idea that every IPv6 host owns an IPv4 address. The IPv6 addresses are created by combining the IPv6 prefix 0::ffff:0:0/96 with a following IPv4 address, e.g. 0::fff:0:192.168.0.1 (The decimal IPv4 address is converted to hexadecimal). This address is called IPv4-translated address. Those are assigned to IPv6 nodes and are possibly matching an IPv4 address. Another type of address is the IPv4-mapped address, which is an IPv6 address that refers to an IPv4 node that is not IPv6 compliant. The IPv4-mapped address is similar to the IPv4-translated address seeing that the form of the address is 0::ffff:192.168.0.1. When data is sent to this address, the SIIT translator (potentially a router) will translate the IPv6 address to the IPv4 address 192.168.0.1. Also the header is translated and possibly fragmented to fit the IPv4 standard. The standard does not specify how the hosts on different sides of this “SIIT translator” learn about these addresses and how routing should be performed [11]. The translation can be done by adding the prefixes to the destination and source addresses at the translator and vice versa with traffic in the opposite direction. The translation is stateless in the meaning that each packet is translated individually and no tables of sessions or similar are created. Despite the lack of specifications on how the translation should be done, there are not many downsides. The major one is the fixed /96 prefix, which could cause scalability issues. Also this method uses a big amount of IPv4 addresses since it is a static translation. The positive aspects are that there is no need for additional configuration of the nodes (specified by the standard), and that it is bidirectional. A potential real-world application of this is an internet connected server farm that runs IPv6 only [14]. The translation would be performed at the edge and a user in the IPv4 internet that connects to a server via a public IPv4 address (that gets translated) will not see any difference. The translation is transparent to the user.

3.3.2 IVI

IVI is an improved SIIT [11]. The big difference is the prefixes; IVI uses a variable, network specific prefix. The mapped and the translated addresses use the same prefix, which implies that the IPv6 addresses and the IPv4-translated addresses have the same prefix. That makes the routing easier in comparison to SIIT.

3.3.3 DNS64/ALG

DNS64 is an extension to the traditional DNS, created to work together with translation mechanisms and help with distribution of translated addresses [11]. The goal here is that when an IPv4 client sends an A query to the DNS server, it converts the A query to an AAAA query. The server then derives the IPv6 address from the AAAA post, converts it back to an A record by removing the used prefix and creates a NAT-mapping between (when used together with NAT64) the IPv6 address and the IPv4 address. This IPv4 address is the one sent as a response to the querying client.
The other case, when an IPv6 client makes an AAAA query is similar in operation [11]. The DNS server converts the AAAA query to an A query, the response is converted back to an AAAA post by adding a prefix and sent to the client as an IPv4-mapped address.

The ALG part is used when it comes to applications such as FTP, which embeds IP addresses in the payload [15]. The ALG performs a deep packet inspection that results in the extraction of these addresses and port numbers that are needed for a proper translation of packets including this kind of applications. DNS64 and DNS ALG functionalities are not automatically combined, the decision is left to the administrator.

### 3.3.4 Network Address Translation – Protocol Translation

**NAT-PT** (Network Address Translation – Protocol Translation) [16] is a mechanism similar to SIIT in what it achieves; possibility for IPv4 nodes to talk to IPv6 nodes without any additional client configuration. The protocol is similar to SIIT with a centralized translator. The main usage of NAT-PT is between IPv4-only and IPv6-only, it is not recommended to be used like a tunnel because it would cause a double translation.

There a several ways of implementing NAT-PT: static, dynamic and *Port Address Translation* (PAT) [16]. Static translation is similar to traditional NAT i.e., one IPv6 address is translated to one IPv4 address and vice versa. These translations are statically configured at the translating device, meaning that a network with many hosts that need translation implies a lot of work for an administrator.

Dynamic translation means that a pool of both IPv4 and IPv6 addresses is configured and when a packet comes to the NAT-PT device from IPv4, an IPv6 address is selected from the dynamic pool and assigned as source address for the egress packet [16]. The need for double pools, one for IPv4 and one for IPv6, depends on the need of bidirectional translation. The translating device keeps a table of sessions, so that the same client will use the same address from the pool every time it accesses something beyond the translator, within some kind of timeframe.

PAT is also called overload and is the most common type of IPv4 NAT [16]. In this application one or more IPv6 addresses are translated into one IPv4 address and separated by port numbers or vice versa. PAT can be used with a pool, but the most common usage is with one interface.

The translator is responsible for announcing the IPv4-mapped address when traffic is initiated from the IPv4 side because the source host does not have the ability to figure out which address and port to send data to [11]. When traffic is initiated from the IPv6 side the mapped addresses look like the ones in SIIT i.e. a /96 prefix plus the destination IPv4 address. The same operation of removing the prefix is performed to get the destination IPv4 address. The opposite is performed for the return traffic. The solution for making this easier is DNS. The NAT-PT standard is deprecated and succeeded by NAT64.

### 3.3.5 NAT64

NAT64 is an “upgraded” and refined NAT-PT [17]. NAT64 is NAT-PT’s successor due to the DNS requirements of NAT-PT. NAT64 supports translation for both IPv4->IPv6 and IPv6->IPv4. NAT64 is separated from any DNS operation which makes it less complex. Translations can be both stateless and stateful, with similar operation as traditional IPv4 NAT. The stateless method is statically addressed, needs IPv6 addresses that are translatable into IPv4 and the use of IPv6 addresses configured by *Stateless address autoconfiguration* (SLAAC) is not supported.
Stateful translation is preferred over stateless since it follows the standards for translating IP headers and IP addresses [17]. NAT64 uses either a Well-known prefix (WKP) or a Network specific prefix (NSP). The WKP 64:ff9b::/96 is not globally routable and is therefore used for translation between private networks. NSP is derived from the organization prefix, and is a public prefix, which makes it usable for translating at the internet edge. The NSP is also a /96 prefix. These prefixes are used to represent IPv4 addresses as IPv6 addresses. For example, if an IPv6 host knows the IPv4 address represented as an IPv6 address to a service on the IPv4 internet (NSP+IPv4 address), it sends data with this IPv6 address in the IPv6 header. When the data reaches the NAT64 translator, it is recognized with the prefix. The translator strips the prefix and creates a binding with a public IPv4 address from a pool or with a port number for overload. The data gets sent over the IPv4 internet, and when the return traffic reaches the translator, the translation is made backwards from the binding table and the communication is successful. In a real world example this would be integrated with a DNS64 server that provides the hosts with IPv4-represented IPv6 addresses.

3.3.6 Bump-in-the-stack

Bump-in-the-stack (BIS) [18] is a technique to make IPv6 nodes able to use IPv4 applications together with other IPv6 nodes. Hosts that implement BIS need a dual stack setup. BIS defines a module that works between the NIC and the TCP/IP stack. This module assigns internal IPv4 addresses to the real IPv6 hosts, so that the IPv4 applications can send data. The internal IPv4 addresses translate to IPv6 addresses and the data is sent on the medium. This means that IPv4-only applications can coexist with a new IPv6 network during a transition phase. The actual translation of the header is made with the same method as in SIIT. When an application needs to send a DNS query for IPv4, a similar method to DNS64 is used where the AAAA record is converted to an A record that points to one of the locally generated IPv4 addresses. Downsides with this technique are that it only supports unicast applications and those applications that send addresses in the payload may not work because the translation of those packets is difficult.

3.3.7 Bump-in-the-host

Bump-in-the-host (BIH) [19] is a similar mechanism to BIS since it is also entirely performed on each host. It is more of a combination of BIS and Bump-In-The-API (BIA), therefore considered a successor to those techniques. BIH can be implemented in two different ways, either a protocol translator between the protocol stacks on a host or a translator between the IPv4 socket API and the TCP/IP module. With a translation between the stacks, the SIIT mechanism is used. The major difference to BIS is that the locally generated addresses in BIH are included in the IPv4 private address space (RFC1918). The technique does not support applications that send IP addresses in payloads. The major improvements are additional to the private address space, support for reverse DNS queries and that BIH is a proposed standard.

3.3.8 464XLAT

464XLAT [20] is a combination of stateful and stateless approaches to transition. It is combining existing techniques to a whole solution. It allows full IPv6 connectivity plus the ability to use IPv4 applications that use the client-server model with a server that has a public IPv4 address. 464XLAT uses SIIT and NAT64 for translation and does not require DNS64 to function. There are two main components, customer-side translator (CLAT) and provider-side translator (PLAT). The CLAT is either a router or a client such as a workstation. Its task is to translate global IPv6 addresses to private IPv4 addresses and vice versa using the SIIT method. The PLAT translates global IPv6 addresses to public
IPv4 addresses and vice versa using the stateful NAT64 method. For this to work as intended, the CLAT needs to be assigned three different /64 prefixes, one for uplink, downlink and a dedicated prefix for transmitting and receiving packets that are translated stateless. The CLAT must also know about the prefix that the PLAT uses to translate packet destined for the IPv4 internet. The CLAT will use this prefix every time data is being sent to the IPv4 internet. Beyond the CLAT, dual stack is used to support both IPv4 and IPv6 hosts. This implies that IPv4 traffic will be translated twice, first at the CLAT and then at the PLAT as seen in Figure 2 Description of 464XLAT operation. The upside with this is that there is no need for DNS64, because IPv4-only destinations can be resolved with regular IPv4 DNS. If an IPv6 host initiates communication to an IPv6 host on the IPv6 internet, no translation has to be done.

![Diagram showing IPv4 and IPv6 connections with CLAT and PLAT]

**Figure 2 [21] Description of 464XLAT operation**

**Translation summary and comparison**

At first, SIIT and NAT-PT are techniques that have been replaced, but whose basic functionality is still used by other techniques. IVI is the upgraded SIIT and is a simple method for statelessly translating packets between IPv4 and IPv6 networks. The downside to it is the IPv4 address consumption since each IPv6 host creates an IPv4 address. NAT64 is the replacement for NAT-PT and can be used in many situations. It is commonly used for translation at the internet edge between the IPv4 internet and an IPv6 network or vice versa. It conserves the IPv4 address space by having the opportunity to use overload, to translate many IPv6 addresses into one IPv4 address by using port numbers. The statefulness is the only downside since it is vulnerable to different kinds of DoS attacks. BIS and BIH are the techniques for making IPv4 applications usable over an IPv6 network. There are no actual downsides to this except the ALG problem where some applications embed addresses in the payload which can cause problems for the translation mechanism. At last, the 464XLAT is the most suitable for provider networks and is a composition of different techniques put together into a whole solution. It allows IPv4 sites with a private address space to connect to both the IPv4 and the IPv6 internet by double translating using NAT64. The downside of 464XLAT is the requirement that the hosts on the IPv4 network need to run dual stack to be able to communicate with both IPv6 and IPv4.
nodes on the internet. The approach is made to phase out IPv4 on long term by having an IPv6 infrastructure that still supports the use of IPv4.

3.4 Experimental transition techniques

Also, there is a set of experimental techniques that have not yet reached standard status. They are not yet supported, but could be some future solutions.

3.4.1 IPv4 Residual Deployment via IPv6

IPv4 Residual Deployment via IPv6 (4rd) [22] is the reverse of 6rd, a technique for tunneling IPv4 packets over an IPv6 backbone in a provider network. The mapping of headers is easier with IPv4-IPv6 since the header is bigger and can fit all the necessary information, plus the TCP and UDP checksums can be used to be sure that packets are intact at the destination. The components are similar to 6rd, the major ones are the CE and the BR. The IPv4-only traffic is tunneled across the IPv6 backbone between the CE and the BR at the edge of the IPv4 internet by transforming the IPv4 packets into IPv6 packets. There is no need for encapsulation of the whole IPv4 packet, the header is transformed and the upper layer payload is inserted in the IPv6 packet. IPv4 addressing for the customers can be either public or private with NAT64 operation at the BR.

3.4.2 Mapping of Address and Port with Encapsulation

Mapping of Address and Port with Encapsulation (MAP-E) [23] is a technique for allowing sites to communicate over a backbone running another protocol than the sites. It combines A+P port address translation with tunneling. A+P is a technique for multiplexing IPv4 addresses using port numbers. It would theoretically allow 65535 customers to use a single IPv4 address. It preserves the end-to-end model by not using translation. The technique works as a NAT by translating between two sides. The translation is made between an internal side that consists of the customers and an external side that is the public IPv4 address space. The translating unit multiplexes many IPv4 addresses to one using fixed port ranges for each customer [24]. An automatic tunneling approach means that a device could tunnel packets by only knowing the destination address. The IPv4 addresses are embedded in IPv6 for extraction. Packets from a private IPv4 network are translated in terms of TCP/UDP ports to the assigned port range. The packets are then encapsulated in IPv6 packets and sent over the IPv6 network to the border relay that decapsulates the packets and sends them out on the internet.

3.4.3 Mapping of Address and Port using Translation

Mapping of Address and Port using Translation (MAP-T) [25] is an extended NAT64 solution with double stateless translation. The difference between MAP-T and MAP-E is that MAP-T uses translation instead of encapsulation. The translation is preferred in cases where encapsulation prevents the network operation, e.g., when DSCP field has to be used in the IPv4 header for Quality of Service (QoS). The actual operation of MAP-T is similar to MAP-E. The CE first translates a private address to the public IPv4 address with a port inside the assigned port range, and then translates that into an IPv6 address. The packet is sent to the border relay at the IPv4 internet for translation. If IPv6-only nodes in the IPv6 backbone have an IPv4-embedded address, the IPv4 nodes can communicate without any extra configuration.

Transition summary

To summarize, many transition techniques exist, but some of them stand out with no obvious flaws. The choice of a method depends much on the specific network. The main candidates for usage in long-term solutions are Dual Stack and NAT64 where it is possible due to equipment support. These
two bring transparency and ease of transition as long as the equipment has the required resources. NAT64 is good where translation is possible together with an implementation of ALG. When it comes to more short term fixes, tunneling could be used to solve transitioning problems. Tunneling makes the network much more complex and susceptible to errors but is very useful when there are tricky topologies. Before a customer is completely ready to fully transition to IPv6, tunneling is perfect for having an internal IPv6 network and still be able to reach the IPv4 internet through a provider network. When it comes to single nodes that need to use IPv4 applications on an IPv6 network, BIH translation and ISATAP tunneling are preferred.

4 Office IT-partner network

This section describes the characteristics of Office IT-partners network, the equipment included and the IPv6 capabilities.

Figure 3 Office IT-partner network topology

4.1 Structure and characteristics

The company Office IT-partner is located in an own building with conference rooms, a front desk, lunch area and an office area.

Figure 3 Office IT-partner network topology shows how the equipment in the network is connected and where it is located. The network is switched with the use of multilayer switches and layer 2 switches. The internal network is addressed with one /24 network for the internal servers and clients at the office. Management, VMware and storage are addressed in a different /24 network. At last
there is a third /24 network in a shared environment with mail servers, SQL-servers and other services for the customers. Except for the firewall no routing is performed, only switching. It means that IPv6 support on internal switches is not required other than IPv6 management.

All of the company’s customers have their own VLAN in the network for connecting to provided services that is trunked from the firewall to the IBM blade switch, where each of the customers have their own virtual switch inside VMware. This provides isolation for each customer within the network. The network is monitored by Mikrotik’s The Dude, which has no support for IPv6 as we speak. The servers and other critical devices are addressed statically, while workstations and other devices are dynamically addressed. All traffic passing through the firewall is being translated using NAT, both traffic from the office and customer traffic.

4.2 Equipment
The Office IT-partner network consists of a WatchGuard XTM850 firewall, layer 2/3 switches, FiberChannel switches and end nodes. The switches are IBM CN4096, HP Procurve 2810 and 2910, where 2810 has only layer 2 capabilities and 2910 has such layer 3 features as routing. There are wireless controllers and access-points from Meru and Watchguard.

Apart from network equipment there are client devices such as workstations, servers and cameras. The servers that provide services for the network are Windows Server 2012 R2 with Windows Deployment Service (WDS), DHCP and DNS.

IPv6 capabilities
HP Procurve 2810 [26]
- Layer 2 switch
- No support for an IPv6 management address

HP Procurve 2910 [27]
- Multilayer switch
- Supports management from IPv6 hosts
- Supports Dual Stack technology
- IPv6 multicast flooding protection

WatchGuard XTM850 [28]
- IPv6 routing (static, OSPF, BGP)
- IPv6 filtering & policies
- DHCPv6
- 6in4
- IPv6 DNS

IBM CN4096 [29]
- IPv6 routing
- IPv6 management
- IPv6 ACLs
Wireless controllers & AP’s

- IPv6 support

4.3 Service Providers

The main internet service provider is Gästabudstaden (GBS) which handles most of the internet traffic. GBS offers native IPv6 through static addressing, DHCPv6 and DHCPv6-PD for dynamic address assignment will be provided in the future. Since there is no support for tunneling or translation services, these types of solutions are eliminated. Translation could be done by Office IT-partner itself at the internet edge with a device that supports it.

5 Selection of a transition method

The following section includes descriptions on how to implement different transition techniques on the Office IT-partner network. Techniques that are suitable only for provider networks are not interesting as a whole solution but the service could be of use in different approaches. Each solution description explains how to transition, which requirements have to be fulfilled, an addressing proposal and how to complete the transition in the future. The best method consisting of one or a combination of techniques is chosen in the end of the section.

5.1 Dual Stack

Dual stack is a method that fits the internal network during a transitioning phase. The existing network will not be modified, which secures the operation. An IPv6 network would be run together with the IPv4 network on the same equipment. All the HP switches are compatible with running dual stack which means that the internal network can be set up for IPv6. If there are any devices with services that are only used internally, they should be upgraded to IPv6 right away since all possible workstations will be running dual stack.

A dual stack approach makes the network able to transition in a slow pace which is good in terms of not disturbing operation of any services. Devices that can only use IPv4 can be kept while they are upgraded or replaced by IPv6 compliant devices. Since the firewall has no support for dual stack, all internet access would be via IPv4. Another possibility for this is to use separate interfaces at the firewall for IPv4 and IPv6 since separate service providers are available. This means there will be two interfaces - one interface inbound and outbound for IPv4 and one inbound and outbound for IPv6. With these interfaces connected to the dual stack switch, nodes could use both IPv4 and IPv6 to reach the internet.

5.1.1 Addressing

A thought for addressing could be to use the “private” address space fc00::/7 or obtain a prefix from a service provider. Regardless of the prefix, the interface IDs could either be the existing IPv4 addresses on the interfaces in hex preceded by zeroes or a new addressing plan could be created.

If the provider offers DHCPv6, it could be used with a relay to assign addresses to the internal network. Servers and other devices that provide services should be statically addressed to avoid issues with the firewall and access from the internet if an address change should occur.
5.1.2 Network services
A DNS server with AAAA records for all devices in the network should be used since remembering IPv6 addresses is quite hard. A DHCPv6 relay at the internet edge should be used for forwarding the DHCP requests to the service provider if that assignment method is chosen. If addresses are assigned locally within the network, the internal DHCP server should be upgraded to service DHCPv6. The current network monitoring system, The Dude, will need to be upgraded over time to support monitoring via IPv6. WDS is not affected and can work like before when run over IPv6, with the addition that only Windows 8 or newer could be distributed with IPv6. The Windows 7 machines would not be supported, but are probably not used the day the network runs IPv6 only.

5.1.3 Future
One of the future tasks is to part by part stop using dual stack and only use IPv6 on nodes and network equipment. This would be done after an upgrade of all the applications that are used but only support IPv4. The problem with the completion of the transition is that, when all devices in the network including the internet edge are upgraded to IPv6, IPv4 nodes can no longer communicate with the network. Long-term IPv4 connectivity for public services is therefore required until all customers have upgraded to IPv6 or a translation device is added to the network. A translation device could be a router or a Linux server which would require Linux expertise.

5.2 NAT64
Using NAT64 is another option that would work well with this network. The only real downside of this approach is that there is no existing device that supports it. It could also be resource intensive, but in this case with a separate device for translation it would not be a problem. The solution would be to invest in a router or some other device that supports NAT64. With this method the whole internal network can be upgraded to IPv6 as long as the devices support it since access to both the IPv6 and the IPv4 internet is available with only IPv6 addresses. The stateful application of NAT64 should be used since it removes the need for IPv6 addresses that are compatible with IPv4. Dual stack is recommended to use on critical services during the implementation to minimize the downtime. A NAT64 device requires DNS64 so the users are able to know which IPv4-represented addresses to send data to. It is most convenient for the DNS64 server to be placed on the same device as the NAT64 translator.

If there are devices in the network that do not support IPv6, these could still use IPv4 and the devices that they communicate with will use dual stack. If these devices need internet access, the NAT64 would have to run dual stack on the inbound interface. Since the routing tables are separated there is no problem in having both IPv4 and IPv6 hosts on the inside while using NAT64.

5.2.1 Addressing
The addressing could be either static or via SLAAC/DHCPv6. The easiest way is to have the addresses assigned is DHCPv6 provided by the service provider since the addresses should be globally unique and usable for communication on the IPv6 internet. The same as for the dual stack solution, servers should be addressed statically.

5.2.2 Network services
A DNS64 service is required for the NAT64 to operate properly, it will provide the internal network with IPv4-mapped addresses when needed. Regular DNS with AAAA records are also needed for
native IPv6 traffic. Network monitoring needs to be upgraded and WDS will continue to work like before.

5.2.3 Future
At first, the devices that can only use IPv4 should be replaced over time to remove the need for dual stack. Except from that, the thing left to do is to remove the translation and use only native IPv6. This is far ahead since it would make IPv4 nodes unable to contact the network. It requires all nodes that communicate with the network to run IPv6.

5.3 BIH
According to the BIH approach, the transition could be done by upgrading the network to IPv6 and using host-to-host translation to support the use of legacy IPv4 applications. The BIH approach should be used together with a technique like NAT64 because a full upgrade to IPv6 in the entire network without translation or dual stack would make it unreachable for IPv4 nodes on the internet. BIH does not involve other devices than the ones who need the service. The challenge with this solution is the support for the BIH technique on the devices, it is very likely that smaller devices with less advanced operating systems do not support BIH. Since BIH is not something that is built-in, third-party software is likely to be required when using this solution.

5.3.1 Addressing
The easiest approach is static addressing for servers and devices with services that need to be available at all times. All clients would be addressed using DHCPv6 to distribute globally unique addresses to the clients. The internal BIH addressing is performed automatically within each client’s protocol stack that uses it and is not visible for other devices.

5.3.2 Network services
A DNS server for IPv6 is needed alongside with DHCPv6. The network monitoring system could still be used with the devices that support BIH to monitor via IPv4, but the system needs an upgrade to work long-term. WDS is not affected here either, since it can be used with IPv6. For NAT64 to work, DNS64 is needed and is best used when placed on the same device.

5.3.3 Future
IPv4-only applications should be upgraded for IPv6 usage to eliminate the need for client translations. The availability problem is present in this solution too, a network fully upgraded to IPv6 cannot be contacted from the IPv4 internet. Translation or dual stack at the internet edge would be needed long-term before all possible customers have upgraded to IPv6.

5.4 Conclusion & protocol selection
The conclusion of the results is that a dual stack solution with a future investment in a NAT64 translation device is the best fitting solution for the network. This is because it could be begun with right away, without having to invest in any new equipment before starting. It would also give time to upgrade parts of the network during the transition such as applications and services. When NAT64 is implemented, dual stack towards the service provider could be used to let native IPv6 traffic pass without translation and only translate traffic destined for IPv4 nodes.

With the proposed solution, the current addressing design could be reused to keep the transition as simple as possible. The three different /24 networks mentioned before should be granted an IPv6
equivalent. Since acquiring an IPv6 prefix from the service provider before having the opportunity to use IPv6 for internet connection is rather unnecessary, addresses from the private address space should be used. These are called Unique Local IPv6 Unicast Addresses [30] and start with fc00::/7.

Table 1 Subnet conversion

<table>
<thead>
<tr>
<th>IPv4 subnets</th>
<th>IPv6 subnets</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.0.11.0/24</td>
<td>FC00:0:0:A::/64</td>
</tr>
<tr>
<td>192.168.0.14.0/24</td>
<td>FC00:0:0:B::/64</td>
</tr>
<tr>
<td>192.168.0.15.0/24</td>
<td>FC00:0:0:C::/64</td>
</tr>
</tbody>
</table>

The subnetworks in Table 1 are bigger than needed, but when using private addresses no conservation of the address space is needed. In the future, with a native IPv6 network and an IPv6-only internet connection, a public prefix would be used and conservation of addresses would be more important. The subnetworks for the customers could be created for IPv6 in the same way as the three local networks. An example of a host address in the first subnet is FC00:0:0:A::1/64.

6 Discussion
The main concern with the different solutions is the existing support, at the moment the only solution that could be implemented is dual stack. NAT64 cannot be used since no device is capable of performing the translation. BIH also needs NAT64 for the network to function on the IPv4 internet. These constraints make the dual stack solution the only one that can be implemented without any new hardware. With the future in mind, the dual stack solution will also need a translator to make a full IPv6 transition possible since the IPv4 internet will remain for a long time. The upside is that all devices that need an internet connection could still use IPv4 which makes the need for translation less imminent. Dual stack also makes WDS and The Dude available during a transition period.

The downside to the dual stack solution is that smaller devices may not support it and that it could be resource intensive. The idea of one interface inbound and outbound for each protocol could mean a high load and a higher cost from the service provider for a second connection. The common challenge with all three solutions is the internet edge. All customers have to be able to connect to the offered services at the same time as an ongoing transition. Since no device at the moment can perform translation, all the customers would have to connect via IPv6 if the internal network is upgraded. The customers cannot be required to use IPv6 since there are 30 of them and therefore a translating device is required to successfully transition. The most convenient way of solving the internet edge problem is to add a NAT64 translator that works together with DNS64 on the same device. The DNS64 service could be placed somewhere else as well but it is more susceptible to errors. An issue with NAT overall is the lack of end-to-end connectivity, but a change from IPv4 NAT to NAT64 will not make a difference regarding end-to-end. The NAT64 provides flexibility because if it is used together with a dual stack internet connection, traffic to the IPv6 internet would not have to be translated and only traffic destined for nodes on the IPv4 internet would have to be translated. This will lessen the load on the translating device and reduce latency for passing traffic.

Another concern is how fast to make the transition. A transition made too fast could result in services and applications not working as intended and network operation could be disrupted. The transition should be done at a slow pace, where the first step is implementing the dual stack and making sure
the network can utilize the IPv6 features and services before moving on. The NAT64 translation should be tested thoroughly before implementing at the internet edge to not disrupt normal operation. Testing could be done internally.

The method used to find the answers to the research questions worked out well. After investigating existing transitioning techniques and taking the network characteristics into account, seeing the possible solutions was easy. The network and the possibilities of the service provider narrowed the choices down and these were theoretically applied to the network. The best solution without doubt was dual stack combined with NAT64. The BIH solution could also work but the issue is the support for the technique, it is not built-in. Small devices like cameras may not have the possibility of installing such a third-party application to support BIH.

7 Conclusion
The main goal of this thesis work was to find a solution for transitioning from IPv4 to IPv6 for Office IT-partner’s network. The current equipment and characteristics created some limitations that had to be followed to find a working solution.

The work started with a literature study of existing transition techniques, which has raised a couple of interesting points. Dual stack and translation are more of long-term solutions while tunneling is better used during a short period with specific circumstances. That does not apply to provider networks, where tunneling could be more long-term by tunneling IPv4 over an upgraded IPv6 backbone for example. There are also not many complete techniques since many of them are used together with others to create a working solution.

For a network with services accessed by others, NAT64 is in principle required unless the internet edge uses dual stack or multiple WAN connections with different protocols. Since the provider of a service cannot know which protocol all possible connecting clients use, NAT64 is the best way of providing connectivity with either one or both protocols independent of the internal network. The issue is when the internal network runs IPv4 and is being translated to IPv6 outbound, because that way NAT64 can only operate statelessly. This means that every IPv6 address needs to be represented with a unique IPv4 address, which is impossible when the internet is the network that is translated to. This would work internally if there are two internal networks with different protocols since private addresses could be used. The best scenario for NAT64 is IPv6 network -> IPv4 internet, since it could be stateful using only one IPv4 address, and static mappings IPv4->IPv6 could be created if multiple public IPv4 addresses are owned to redirect traffic to e.g. a server on the internal network.

Office IT-partner network came with some constraints, but in the end the dual stack solution was found to fit the network well. The main issue was the support for transitioning techniques on the existing equipment. At the moment no device can perform translation between IPv4 and IPv6. This is required since it is the easiest way of providing IPv4 connectivity to an IPv6 network in this scenario. The solution could be deployed without investing in any new equipment but it is recommended to do so. Network services will not be affected until IPv4 connectivity is scaled down except for IPv6 versions of existing network services that need to be added. Internally, the transition could start right away with implementing dual stack where possible and running IPv6 on devices only used internally. This makes it easier to decide the pace of the transition, when all of it does not need to be done at once.
References


0000=1463138868. [Accessed 13 May 2016].