Research report
Amsterdam Internet Exchange

Project: IPv4 shortage AMS-IX

How can new routing devices participate in the sharing of BGP IPv4 routes on the main AMS-IX Internet Peering VLAN when the current IPv4 range is depleted?

Author: Stefan Plug
E-mail: stefan.plug@ams-ix.net
Version: 1.0
State: Final
Date: 29-05-2012
## Version Management

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>02-01-12</td>
<td>Start of document.</td>
</tr>
<tr>
<td>0.2</td>
<td>05-01-12</td>
<td>Added chapter 2 ‘Project Overview’.</td>
</tr>
<tr>
<td>0.3</td>
<td>13-01-12</td>
<td>Renamed and revised chapter 2 ”Research Overview”</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added chapter 3 ’IP Usage’</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added raw data into chapter 4</td>
</tr>
<tr>
<td>0.4</td>
<td>03-02-12</td>
<td>Renumbered chapter 3 ’IP usage’ into chapter 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added chapter 3 ’BGP’</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added chapter 5.1 ’Overview’,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added chapter 5.2 ’Route Server’</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added chapter 5.3 ’Physical Network’</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added chapter 5.4 ’Logical Network’</td>
</tr>
<tr>
<td>0.5</td>
<td>07-02-12</td>
<td>Changed the reason for MPLS/VPLS in chapter 5.4 ’Logical Network’</td>
</tr>
<tr>
<td>0.6</td>
<td>17-04-12</td>
<td>Added chapter 6.2 ’RFC 5549’</td>
</tr>
<tr>
<td>0.7</td>
<td>19-05-12</td>
<td>Added remaining chapters, ready for preliminary review</td>
</tr>
<tr>
<td>0.8</td>
<td>25-05-12</td>
<td>Spelling check</td>
</tr>
<tr>
<td>1.0</td>
<td>29-05-12</td>
<td>Ready to press!</td>
</tr>
</tbody>
</table>
Preface

This research report is part of my bachelors graduation project at the Hogeschool van Amsterdam, HvA, where I have been studying Computer Science: System and Network Engineering, SNE, for the past three and a half years. Just like every other Bachelor student at the HvA I had to find a final project where I could prove that I am ready to become a professional system and network engineer.

During my time at AMS-IX I have encountered a unique network with solutions for problems of which I had never heard of before. What SNE student has ever heard of a route server or an ARP sponge? Because of the unique network at the AMS-IX I might have spent too much time researching the current network situation leaving less time for me to research alternative solutions to RFC 5549. Although I do regret my decision to spend more time on the current situation in regard to this research report, I do not regret it on a personal level because I did learn a lot from trying to figure out, for example, what VPLS or the Internet Routing Registry database are.

Because of the AMS-IX's quite unique problem this project has been one of my favorites to work on. It had everything a System and Network student could ask for. It was a real honor for me to be allowed to work with the people who manage one of the biggest Internet exchanges on the planet. I would really like to thank Romeo Zwart who was the manager of the SNE study at the HvA for introducing me to Ariën Vijn when I was searching for my final project. Without his help I'm sure my final project wouldn't have been half as interesting as this one.

I already mentioned him, but I can’t thank Ariën Vijn enough for his help during my project. He gave me the choice out of two different projects the first time I met him when I visited the AMS-IX to talk about my final project. The first project was about a monitoring system for 100 Gigabit Ethernet. The second one was about a strange RFC he recently found which would allow BGP to send IPv6 next hops for IPv4 traffic. That last project immediately got my attention. It was such a weird idea, but when I thought about it I didn’t really see why it shouldn't be possible. I am happy that I chose the second one because it turned out to be a really cool project. I could stop here and just Thank Ariën, but that wouldn't give him enough credit. For example when I was trying to alter the Linux kernel itself and got stuck it was Ariën who asked me during a lunch break if I couldn't just use some kind of translation table. He basically saved my entire project with one sentence. So thanks again Ariën! (And the 240.0.0.0/16 range is way better than 169.254.0.0/16! :D)

Next I would like to thank Martin Pels for his general input in my project like reviewing my report and especially the idea about maybe using a class E range for the translation table.

This project wouldn't have been possible without the writers of RFC 5549 in the first place. That is why I would like to especially thank F. Le Faucheur and E. Rosen of Cisco Systems.
Also a big thanks to Klaas Wierenga with whom I worked with in a previous project for getting me in contact with the Cisco's BGP department.

A big thank you to Nathalie Trenaman who helped me when I had questions about the possibility for the AMS-IX to increase the IPv4 range by becoming a LIR.

My good friends Mike Walsteijn and Andrea Francis must both be thanked for reviewing my report. Without them this document would no doubt contain a lot more spelling error.

I sadly cannot mention everyone in this preface who helped me with this project personally. I do want to thank everyone of the AMS-IX's Network Operations Center, NOC, and everyone at the AMS-IX for giving me the best graduation project I could wish for!
# Index

Preface ......................................................................................................................... 3
1 Summary .................................................................................................................... 7
2 Introduction ................................................................................................................ 8
  2.1 AMS-IX .............................................................................................................. 8
  2.2 Research Questions ............................................................................................ 9
3 BGP .......................................................................................................................... 11
  3.1 Autonomous Systems ....................................................................................... 11
  3.2 BGP Neighbors .................................................................................................. 12
  3.3 BGP Message Types ......................................................................................... 13
    3.3.1 BGP Message Header ................................................................................. 13
    3.3.2 Open Message ........................................................................................... 14
    3.3.3 Update Message ........................................................................................ 14
    3.3.4 Notification Message ................................................................................. 15
    3.3.5 Keep-alive Message ................................................................................... 16
  3.4 MP-BGP ............................................................................................................. 16
    3.4.1 MP_REACH_NLRI Path Attribute ............................................................... 17
    3.4.2 MP_UNREACH_NLRI Path Attribute ......................................................... 18
4 IP Usage ................................................................................................................... 19
  4.1 The Internet Peering VLAN (ISP VLAN) ......................................................... 19
  4.2 IPv6 .................................................................................................................... 19
  4.3 IPv4 .................................................................................................................... 20
  4.4 Border Gateway Protocol BGP ......................................................................... 22
5 Current Network Situation ...................................................................................... 23
  5.1 Overview ........................................................................................................... 23
  5.2 Route Server .................................................................................................... 24
  5.3 ARP Sponge ..................................................................................................... 26
  5.4 Physical Network .............................................................................................. 28
  5.5 Logical Network ............................................................................................... 31
  5.6 Customer Equipment ....................................................................................... 34
6 Solutions .................................................................................................................. 37
  6.1 Increase IPv4 Range ......................................................................................... 37
    6.1.1 Finding IPv4 Address Space ...................................................................... 37
    6.1.2 Broadcasts ................................................................................................. 38
1 Summary

The AMS-IX will probably run out of its /22 IPv4 address range on the Internet Peering VLAN in 2016. When this happens no new routing devices can join the IPv4 AMS-IX Internet Peering VLAN anymore.

To double the current /22 to a /21 IPv4 range the AMS-IX will have to become a LIR by becoming a member of RIPE NCC. If the AMS-IX should decide to do this it would have to act quick as the RIPE NCC will no longer give out /21s to LIRs when the final /8 IPv4 range will start to be handed out. This will probably start to happen in the coming few months. This solution should be used to buy more time so better and more permanent solutions can be developed.

A promising more permanent solution is a new technique described in RFC 5549 which enables MP-BGP to advertise IPv6 next hop addresses for IPv4 destinations. This would eliminate the need for IPv4 entirely on the AMS-IX Internet Peering network.

In this document I describe 3 ways how RFC 5549 could be implemented. These are:

• native inter-protocol forwarding
• a translation table managed by the forwarding table
• a translation table managed by the routing daemon

There is now a working proof of concept which utilizes a translation table managed by the routing daemon. This proof of concept shows that RFC 5549 has potential as a permanent solution to the IPv4 shortage on the AMS-IX Internet Peering VLAN.

Further research on RFC 5549 is needed. For example there is a problem with traceroute in the current proof of concept. This document describes a theoretical way of solving this problem.

The AMS-IX could use this proof of concept and this research report to create a demand for this solution with its clients. The clients of the AMS-IX should then proceed to ask their routing vendors to implement RFC 5549 in their routing devices.

To see my presentation video please go to http://www.ams-ix.net/downloads/RFC5549/

---

1 This is a term I made up for describing the direct forwarding of IPv4 packets to IPv6 next hop addresses
2 Introduction

This document will explore the possibilities of how new routing devices can exchange IPv4 routes after the current AMS-IX IPv4 range is depleted on the Internet Peering VLAN. The reader of this document should have a basic understanding of computer network technology. This includes basic knowledge of the TCP/IP protocol stack, VLANs, and routing protocols in general.

This research report uses many sources pointing to IETF RFCs. The Internet Engineering Task Force, IETF, describes itself as follows on its website: “The mission of the IETF is to make the Internet work better by producing high quality, relevant technical documents that influence the way people design, use, and manage the Internet.”. The technical documents referred to in the previous quote are the many Request For Comments, RFCs. Many protocols used to make the Internet work are described in these RFCs. I recommend anyone reading this report to also read the RFCs mentioned in this report.

2.1 AMS-IX

The Amsterdam Internet Exchange, or simply AMS-IX, is one of the largest Internet exchanges in the world with peak traffic of over 1.5 Tb/s traversing their network. As an Internet exchange the AMS-IX's main job is to provide an independent switching platform where external networks can be interconnected with each other.

The name AMS-IX was first used in 1994 when the shared L2 infrastructure between academic organizations in The Netherlands was joined by international parties such as CERN in Switzerland. In 1997 the AMS-IX association was officially founded by twenty founding members including Surfnet, KPN, RIPE NCC, and AT&T. In 2000 the AMS-IX association formed the AMS-IX B.V.

The AMS-IX employs an international team of highly trained professionals consisting out of 37 people with 12 different nationalities. Because of the international nature of the company the lingua franca is English within the company.

The technical staff is managed by the Chief Technical Officer Henk Steenman and can be divided into a supporting branch and an operations branch. The supporting branch is responsible for managing the internal AMS-IX network including systems management, website management, and systems development. The Operations branch consists largely out of the Network Operations Center or NOC which is a team of highly trained network professionals who are responsible for managing and developing the AMS-IX peering network. Next to the NOC the operations branch also has a dedicated Implementation Manager and Project Manager.

Today the AMS-IX network interconnects almost 500 networks which together connect to the AMS-IX network with over 900 switch ports.
2.2 Research Questions

The main research question for this research report is: **How can new routing devices participate in the sharing of BGP IPv4 routes on the main AMS-IX Internet Peering network when the current IPv4 range is depleted?**

For this research project a good understanding of BGP is needed. To help me understand how BGP shares routes and how the BGP neighbor forming process works I will ask the following questions.

- How does a BGP enabled router exchange routes with other BGP routers?
- What kind of packets does BGP use?
- How does BGP handle IPv6?

The reason to research the main research question is that the AMS-IX has noticed that its available IPv4 addresses are being depleted rapidly. On the 17th of January 2010 the amount of available IPv4 addresses had to be doubled from 512 to 1024. In recent years the amount of available IPv4 addresses seems to have been depleted more rapidly than a few years ago. To find out if this in fact is true I will ask the following questions.

- How many IPv4 addresses have been assigned in the last five years?
- What was the yearly growth of used IPv4 addresses in the last five years?
- When will the current IPv4 address pool probably run out?
- What will the consequences be when the current available IPv4 addresses should run out?

Only when a clear problem has been diagnosed, then can we search for a cure. A cure can be very specific though, what might be an acceptable solution for one environment might not be acceptable for another. Because of this it will be important to take a good look at the current Internet Peering network. To help me get a clearer view on the Internet Peering network I will ask the following questions.

- What does the current physical Internet Peering network look like?
- What does the current logical Internet Peering network look like?
- What hardware and software devices are used by the AMS-IX in the Internet Peering network?
- What hardware and software are used by the customers connected to the Internet Peering network?

When we know what we have, then can we start researching how to solve our main problem. To solve the main problem we need to have more than one potential solution in case the other solutions prove not to be viable for the AMS-IX's situation. I will research the following questions.

- Can the current IPv4 range be increased?
- Can the IPv4 routes be shared with a IPv6 peer relationship described in IETF RFC 5549?
When all the sub-questions have been researched and answered I will advise the AMS-IX about my findings and about possible solutions for the IPv4 depletion in the Internet Peering network.
3 BGP

On the Internet BGP is used as the de facto standard routing protocol. I chose to start my research report with a research on BGP rather than the main IPv4 shortage problem because the chapters following this one will assume the reader has a good understanding of BGP and in particular how it forms neighbor relationships.

BGP stands for Border Gateway Protocol. BGP is the only Exterior Gateway Protocol, EGP, used on the Internet today. An EGP's job is to route between Autonomous Systems, ASs, as opposed to an Interior Gateway Protocol, IGP, like for example OSPF which job it is to route inside an AS.

BGP-4 was originally only able to support IPv4 and is described in RFC 4271. The Multi-Protocol extensions for BGP-4 were created so that BGP-4 would be able to support multiple routed protocols including IPv6. BGP-4 with the Multi-Protocol extensions is commonly called MP-BGP and is described in RFC 4760.

To help me get a better understanding of BGP I will answer the following question in this chapter.

- How does a BGP enabled router share routes with other BGP routers?
- What kind of packets does BGP use?
- How does BGP handle IPv6?

3.1 Autonomous Systems

BGP is meant to be used for inter-AS routing. An AS is described in RFC 1930 as: "a connected group of one or more IP prefixes run by one or more network operators which has a SINGLE and CLEARLY DEFINED routing policy." Some organizations like ISPs lets traffic traverse their network to reach other networks. To reach places connected to other ISPs, the ISPs have to connect their networks together using BGP.

BGP is a path vector routing protocol. This means that it uses path information sent by its neighbors and doesn't keep a 'road map' of the network like link state routing protocols such as OSPF or IS-IS. Path information in the case of BGP consists of the ASs which need to be traversed to reach the final destination network. In the image below there are two paths for a client in the 11.0.0.0/24 network to reach 12.0.0.0/24. Path 1 is through AS 3 directly and path 2 is to first go through AS 1, and then through AS 3. When only the AS path is used for calculating a route, then the router in AS 2 sees the route to AS 3 has a shorter path and will choose that path instead of the longer route through AS 1.
The AS path also prevents routing loops from occurring. When a router sees its own AS in a route advertised by a neighbor from another AS it will silently drop that route because that route will probably contain a loop. For this reason the AS numbers used in a BGP network must be unique. The Internet Assigned Numbers Authority, IANA, manages the distribution of all globally used AS numbers.

When we look inside an AS we can see that there are two types of BGP neighbors that a BGP enabled routing device can have, one with the same AS number, and one with a different AS number. A BGP connection between two routers within the same AS uses Internal BGP, iBGP, and a BGP connection between two routers with different AS numbers uses External BGP, eBGP.

### 3.2 BGP Neighbors

Like any routing protocol BGP needs neighbors to exchange routes with. BGP has a very different way to form neighbor relationships than IGPs like OSPF or RIP. The first major difference is that the network engineer must manually configure the router to form a neighbor relationship with a specific neighbor. The second major difference is that BGP uses TCP with port number 179 to send its messages. This means that theoretically BGP routers can be in different broadcast domains and still form a neighbor relationship and share routes with each other, just as long as they have established a working TCP connection between each other.
3.3 BGP Message Types

BGP enabled routing devices use four different types of messages between each other according to RFC 4271. These are:

- open message
- update message
- notification message
- keep-alive message

As stated before BGP uses TCP to send these messages and thus all BGP messages are encapsulated within a TCP segment.

3.3.1 BGP Message Header

Each of the BGP messages share a common three field header which is 19 bytes in length. This header consists of the following fields:

**Marker:**
The marker field is a 16 bytes field which is included for compatibility. This field is always set to all ones.

**Length:**
The length field is a 2 bytes field which indicates the total length of the message in bytes.

**Type:**
The type field is a 1 byte field which indicates what type of BGP message will follow. The following codes are defined by IANA:

<table>
<thead>
<tr>
<th>Description</th>
<th>Decimal</th>
<th>Binary, 1 byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td>00000000</td>
</tr>
<tr>
<td>Open message</td>
<td>1</td>
<td>00000001</td>
</tr>
<tr>
<td>Update message</td>
<td>2</td>
<td>00000010</td>
</tr>
<tr>
<td>Notification message</td>
<td>3</td>
<td>00000011</td>
</tr>
<tr>
<td>Keep-alive message</td>
<td>4</td>
<td>00000100</td>
</tr>
<tr>
<td>Route-refresh message, described in RFC 2918</td>
<td>5</td>
<td>00000101</td>
</tr>
</tbody>
</table>

Table 1: IANA BGP message codes
3.3.2 Open Message

An open message is the first message which is sent when a TCP connection is established between two BGP routers. This message will try to establish a neighbor relationship between the two routers. After the BGP connection is established this connection will then be kept open by subsequently timed Keep-alive messages. The Open message consists of the following fields.

<table>
<thead>
<tr>
<th>Marker</th>
<th>Length</th>
<th>Type</th>
<th>Version</th>
<th>AS</th>
<th>Hold Time</th>
<th>BGP ID</th>
<th>OPL</th>
<th>OP</th>
</tr>
</thead>
</table>

**Version:**
The version field is a 1 byte field indicating what BGP version is used, in the case of BGP-4 this value will be a 4.

**AS:**
The AS field is a 2 bytes field containing the AS number of which the sending router is a part of.

**Hold Time:**
The Hold Time field is a 2 bytes field containing the number of seconds the sending router will use to consider the receiving router to be offline. If the receiving router does not send a Keep-alive message back before this timer runs out, then the sending router will consider the receiving router offline and will then terminate the connection. Every time when a Keep-alive message is received the timer will be reset.

**BGP ID:**
The BGP ID field is a 4 bytes field containing the router ID of the sender. This router ID will be the same on every interface of the sending router.

**OPL:**
The Optional Parameters Length field is a 1 byte field which indicates the total length in bytes of the following optional parameters.

**OP:**
The Optional Parameters field is a variable length field which contains optional parameters.

3.3.3 Update Message

An update message is used to share actual routing information between BGP neighbors. The update message consists of the following fields.

<table>
<thead>
<tr>
<th>Marker</th>
<th>Length</th>
<th>Type</th>
<th>WRL</th>
<th>WR</th>
<th>TPAL</th>
<th>PA</th>
<th>NLRI</th>
</tr>
</thead>
<tbody>
<tr>
<td>16 B</td>
<td>2 B</td>
<td>1 B</td>
<td>2 B</td>
<td>VAR</td>
<td>2 B</td>
<td>VAR</td>
<td>VAR</td>
</tr>
</tbody>
</table>
WRL:
The Withdrawn Routes Length field is a 2 bytes field indicating the length in bytes of the following WR field. If the WRL is 0 then that means that there will be no following WR field.

WR:
The Withdrawn Routes field is a variable length sequence field which contains destinations which are no longer reachable via the sending router. This field consists out of one or more 2-tuples consisting out of <Length, Prefix>. Length is a 1 byte field which indicates the length of the following prefix in bits. Prefix is a variable length field which indicates the network address of the withdrawn route. For example, if the withdrawn route would be '192.0.2.0 /24' the length would be '24', and the prefix would be '192.0.2'.

TPAL:
The Total Path Attributes Length field is a 2 bytes field indicating the length in bytes of the following PA field

PA:
The Path Attributes field is a variable length sequence field which contains path attributes for the following NLRI which I further describe below. These attributes are used by the receiving router to determine the best routes though the network. An attribute is a triple consisting out of <attribute type, attribute length, attribute value>.

BGP has many well-known attributes defined by IANA. Vendors may also use proprietary attributes if they wish, these are called optional attributes. Three well-known attributes must always be present, these are called well-known mandatory attributes. The three well-known mandatory attributes are as follows.

- ORIGIN
- AS_PATH
- NEXT_HOP

ORIGIN defines if the route is learned via an IGP, EGP, or if the origin of this route is unknown. AS_PATH is the sequence of AS numbers that the route passes through to reach its destination network. NEXT_HOP is the next hop IPv4 address for this route.

NLRI:
The Network Layer Resource Information field is a variable length field which contains route destination network addresses. The NLRI resembles the WR field. It also contains a variable length field containing a sequence of 2-tuples consisting of <Length, Prefix>. 
3.3.4 Notification Message
A notification message is sent when an error is detected by the sending router. The BGP connection is closed immediately after it is sent. The notification message contains the following fields.

![Image 6: BGP notification message]

**Error code:**
The Error code field is a 1 byte field containing the main error code. The following codes have been defined by IANA:

<table>
<thead>
<tr>
<th>Description</th>
<th>Decimal</th>
<th>Binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td>00000000</td>
</tr>
<tr>
<td>Message header error</td>
<td>1</td>
<td>00000001</td>
</tr>
<tr>
<td>OPEN message error</td>
<td>2</td>
<td>00000010</td>
</tr>
<tr>
<td>UPDATE message error</td>
<td>3</td>
<td>00000100</td>
</tr>
<tr>
<td>Hold timer expired</td>
<td>4</td>
<td>00000100</td>
</tr>
<tr>
<td>Finite state machine error</td>
<td>5</td>
<td>00000101</td>
</tr>
<tr>
<td>Cease</td>
<td>6</td>
<td>00000110</td>
</tr>
</tbody>
</table>

Table 2: IANA BGP error codes

**Error subcode:**
The Error subcode field is a 1 byte field which contains a more specific code for the error.

**Error Data:**
The Error data field is a variable length field which contains additional information for the cause of this notification message.

3.3.5 Keep-alive Message
Keep-alive messages are sent to keep the connection between two routers open. These messages should be sent before the hold time indicated in the open message runs out. The keep-alive message itself is nothing more than a BGP header with a type field containing a '4' indicating this message is a keep-alive message.

![Image 7: BGP keep-alive message]
3.4 MP-BGP

MP-BGP is described in RFC 4760 and adds several extensions to BGP-4. MP-BGP is backward compatible to BGP-4 and adds the functionality to send and receive routing information for other routed protocols than IPv4 such as IPv6 and IPX.

MP-BGP adds two new path attributes to the BGP update message. These are the Multiprotocol Reachable Network Layer Resource Information (MP_REACH_NLRI), and the Multiprotocol Unreachable Network Layer Resource Information (MP_UNREACH_NLRI). Both of these new path attributes are optional and non-transitive, this means that routers that do not use the Multiprotocol extensions may ignore and drop these attributes.

The MP_REACH_NLRI path attribute is used to hold the final route destination instead of the NLRI field in the original update message. The MP_UNREACH_NLRI path attribute is used to send information about unreachable routes instead of the Withdrawn Routes field in the original update message.

When a BGP router wants to send an IPv6 route update it would set the Withdrawn Route Length to 0 indicating there will be no Withdrawn Routes field following and it would omit the NLRI field. Instead it will convey it's routing information in its Path Attributes field via the MP_REACH_NLRI and the MP_UNREACH_NLRI.

One further change is the use of the mandatory non-transitive NEXT_HOP path attribute, this should originally be an IPv4 address. In MP-BGP this attribute is ignored. The next hop information is incorporated into the MP_REACH_NLRI.

3.4.1 MP_REACH_NLRI Path Attribute

The MP_REACH_NLRI is used to convey new reachable routes to the receiving router and contains the following fields.

<table>
<thead>
<tr>
<th>AFI</th>
<th>SAFI</th>
<th>LNH</th>
<th>NANH</th>
<th>Reserved</th>
<th>NLRI</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 B</td>
<td>1 B</td>
<td>1 B</td>
<td>VAR</td>
<td>1 B</td>
<td>VAR</td>
</tr>
</tbody>
</table>

**AFI:**
The Address Family Identifier field is a 2 bytes field which together with the SAFI field identifies what address family is used in the NANH and in the NLRI fields. For example, a 1 indicates IPv4 and a 2 indicates IPv6. The AFI codes are managed by IANA and can be found on their website.

**SAFI:**
The Subsequent Address Family Identifier field is a 1 byte field which together with the AFI field identifies what address family is used in the NANH and in the NLRI fields. For example, a 1 indicates a unicast address and a 2 indicates a multicast address. The SAFI codes are managed by IANA and can be found on their website.
LNHA:
The Length of Next Hop Address field is a 1 byte field which indicates the length of the coming NHNA field in bytes.

NANH:
The Network Address of Next Hop field is a variable length field which indicates what the next hop address is for packets going to the destination networks contained in the NLRI field. The address family associated with the address in this field is identified by the combination of the AFI and SAFI fields.

Reserved:
The reserved field is a 1 byte field. This field is reserved and should be ignored.

NLRI:
The Network Layer Reachability Information field is a variable length field which contains 2-tuples <Length, Prefix> of destination network addresses for this advertised route. Like the NANH field the address family for this address is defined by the combination of the AFI and SAFI fields.

3.4.2 MP_UNREACH_NLRI Path Attribute
The MP_UNREACH_NLRI is used to advertise routes which are no longer reachable through the sending router. This path attribute is used instead of the Withdrawn Routes field in the original BGP update message. When a BGP Update message is sent with a MP_UNREACH_NLRI path attribute it is permitted to let this be the only path attribute. The MP_UNREACH_NLRI path attribute contains the following fields:

```
AFI|SAFI|Withdrawn Routes
2B|1B|VAR
```

Image 9: MP-BGP MP_REACH_NLRI path attributes

AFI & SAFI:
The AFI and SAFI fields are the same in the MP_UNREACH_NLRI as in the MP_UNREACH_NLRI.

Withdrawn Routes:
The Withdrawn Routes field is a variable length field containing 2-tuples <Length, Prefix> of the routes which are no longer reachable through this router.
4 IP Usage

According to the AMS-IX the available IPv4 addresses are running out on their Internet Peering VLAN. I will research the current and historical IP usage on the Internet Peering VLAN so I can answer the following research questions.

- How many IPv4 addresses have been assigned in the last five years?
- What was the yearly growth of used IPv4 addresses in the last five years?
- When will the current IPv4 address pool probably run out?
- What will the consequences be when the current available IPv4 addresses should run out?

### 4.1 The Internet Peering VLAN (ISP VLAN)

The AMS-IX uses several VLANs on their network to separate the network into different sub-networks, each with a specific function. The biggest VLAN by far is the Internet Peering VLAN. According to the AMS-IX's annual report of 2010 the Internet Peering VLAN is described as follows: “The Internet Peering VLAN is the most widely used exchange service. All of the new peers in 2010 connected to this VLAN except 2. it is the common service on which the Internet Exchange is based and where public peering is done. The Internet Peering VLAN is an Ethernet based non-blocking Unicast service, enabled to support both IPv4 and IPv6 natively”.

The current problem of insufficient IPv4 addresses is not new. On the 17th of January 2010 the amount of IPv4 addresses on the Internet Peering VLAN had to be doubled from 512 to 1024 because there were almost no more available IPv4 addresses left at that time. This was done by changing the network prefix from 23 to 22 bits. In theory we could again double the available IPv4 addresses to 2048 by using a 21 bit network prefix. It is very probable that this is not actually possible. Because of the worldwide shortage of IPv4 addresses the Réseaux IP Européens, RIPE, who allocates IP addresses in Europe and the Middle East are considering a policy proposal which would prohibit Internet Exchanges to request IPv4 ranges greater than /22.

### 4.2 IPv6

Customers on the AMS-IX Internet Peering VLAN can use both IPv6 and IPv4 addresses on the Internet Peering VLAN. When a customer uses IPv6, they should use a IPv6 address according to the AMS-IX IPv6 numbering scheme described on the AMS-IX website. This scheme tells the customers of the AMS-IX to incorporate their BGP AS number into their IPv6 address. Because of the 64 bit prefix used in the Internet Peering VLAN the number of 32-bit AS numbers will run out before the number of IPv6 addresses which are available.

---

2 IETF RFC 3513 'Ipv6 Addressing Architecture' recommends that all IPv6 interface addresses use a 64-bit interface ID giving 2^64 available IPv6 addresses.
4.3 IPv4

At the moment of writing, 3 January 2012, there are still 424 available IPv4 addresses left in the AMS-IX Internet Peering VLAN. This number includes the 2 non usable network and broadcast addresses.

The data I found about the historical IPv4 usage were XML files starting in 2006. These XML files document all the used IPv4 addresses in the Internet Peering VLAN at that time. I used the first XML file for every year to gather my data.

When we look at the data we can see that the growth of used IPv4 addresses has been relatively stable with an average growth of 33 in 2006 until 2010. In the following years the growth increased to 49 in 2011, and to 84 in 2012.

<table>
<thead>
<tr>
<th>Date</th>
<th>Used</th>
<th>Reserved</th>
<th>Total used</th>
<th>Available</th>
<th>Free</th>
<th>Growth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan 4, 2006</td>
<td>289</td>
<td>64</td>
<td>353</td>
<td>512</td>
<td>159</td>
<td></td>
</tr>
<tr>
<td>Jan 3, 2007</td>
<td>318</td>
<td>64</td>
<td>382</td>
<td>512</td>
<td>130</td>
<td>29</td>
</tr>
<tr>
<td>Jan 2, 2008</td>
<td>352</td>
<td>64</td>
<td>416</td>
<td>512</td>
<td>96</td>
<td>34</td>
</tr>
<tr>
<td>Jan 5, 2009</td>
<td>481</td>
<td>64</td>
<td>545</td>
<td>512</td>
<td>61</td>
<td>33</td>
</tr>
<tr>
<td>Jan 4, 2010</td>
<td>419</td>
<td>48</td>
<td>467</td>
<td>512</td>
<td>45</td>
<td>32</td>
</tr>
<tr>
<td>Jan 4, 2011</td>
<td>468</td>
<td>48</td>
<td>516</td>
<td>1024</td>
<td>508</td>
<td>49</td>
</tr>
<tr>
<td>Jan 2, 2012</td>
<td>552</td>
<td>48</td>
<td>600</td>
<td>1024</td>
<td>424</td>
<td>84</td>
</tr>
</tbody>
</table>

Table 3: Historic IPv4 allocation in the Internet Peering VLAN

When the data is plotted in a table we can see that the total use IPv4 addresses is increasing rapidly and that the growth of used addresses has increased in 2010 and 2011.
The growth for 2011 is 48 which is almost 1.5 times the previous average of 33. For 2012 it was 84 which is more than 2.5 times the previous average.

The reason for this sudden increase of used IPv4 addresses is probably a combination of the introduction of the ability for customers to split their connection into virtual ports and sell them on to third-party customers and the introduction of a dedicated sales team in September 2009.

The AMS-IX also reserves a range of IPv4 addresses for itself to use. Which addresses are actually reserved is not documented, but in the data I used some IPv4 addresses were labeled with a state of “reserved”. While asking around about which IPv4 address should be reserved I found out that there is a lot of confusion about this. For my report I will use the scheme also used by Ariën Vijn. In his scheme there were 64 addresses reserved until 2010 when 16 extra addresses were released for use making the current total of 48 reserved IPv4 addresses. The reserved addresses could be released in the future should the need for them arise.

When the total of used IPv4 addresses is plotted in a new graph along with an average growth of 100 new used IPv4 address for the near future, then we can see that the IPv4 address pool in the Internet Peering VLAN will be depleted in just over 4 years, in the year 2016.
The main reason for the existence of the Internet Peering VLAN is for customers to share routes with each other. To do this they use the BGP routing protocol, in chapter 3 'BGP' I discuss BGP in greater detail.

To be able to reach other routing devices and start with the peering process each router needs to have its own IP address so it is reachable for other routers. The problem for the IPv4 network is that, as stated in chapter 4.3 "IPv4", the IPv4 addresses for the Internet Peering VLAN are running out. When there are no more IPv4 addresses to give to new routing devices on the Internet Peering VLAN, then new routing devices will not be able to join the IPv4 network and share their IPv4 routes.
5 Current Network Situation

An understanding of the current network situation is needed if we want to be able to find a good solution to the problem of IPv4 addresses running out on the Internet Peering network. To help me get a better understanding of how the Internet Peering network operates I will research the following questions.

- What does the current physical Internet Peering network look like?
- What does the current logical Internet Peering network look like?
- What hardware and software devices are used by the AMS-IX in the Internet Peering network?
- What hardware and software are used by the customers connected to the Internet Peering network?

5.1 Overview

When we look at the AMS-IX peering network at a very simplistic level it looks like the following image.

The AMS-IX provides a common place where companies can connect their routing devices and share BGP routes with each other. At the time of writing, 06 Jan 2012, the AMS-IX has 470 connected ASs\textsuperscript{xiv}. For the customers who want to create a neighbor relationship with as many other routing devices as possible it would be an extremely labor intensive job to maintain all of their peerings. This is why the AMS-IX provides 2 route servers for their customers to establish a neighbor relationship with.
5.2 Route Server

A route server is described in the IETF Internet-Drafts 'Internet Exchange Route Server' and 'Internet Exchange Route Server Operations'. Because these are both still IETF drafts I have to cite them as being work in progress and they may be updated, replaced or obsoleted at any time. Regardless, I will use them as a reference for my research into route servers in general at this time because they are still a good, general source.

A BGP route server acts like a central point for BGP routing devices to exchange reachability information with. In essence it works like a BGP route reflector described in IETF RFC 4456, but instead of using iBGP it uses eBGP. The problem a route server tries to solve is that of scalability. Without a central route server the routing devices would have to form neighbor relationships with every other routing device they would want to exchange reachability information with. On smaller networks this is not a big problem, but if a new company joins the AMS-IX Internet Peering network the technician on the new routing device would need to establish and manage 470 unique neighbor relationships by hand.

It is very important to note that the route server itself will never be a part of the route taken by the traffic through the network. The route server only concentrates all the possible routes so the other routing devices have a single place to get their routing information from. The route server itself will never be the next hop address for any route and it will never inject its own AS number in the AS_PATH attribute.

---

3 iBGP and eBGP are two different 'flavors' of BGP. iBGP is used to route within the same AS, and eBGP is used to route between AS's.
When a routing device sends its routes to a route server it cannot filter its outgoing routes going to other routing devices connected to the route server. The route server itself is responsible for the routes it sends to its connected routing devices. When the route server receives a route to a destination from several connected routing devices it will calculate the best path and only propagate this path to the other connected routing devices.

A problem called “path hiding” can occur when the route server (RS) is configured to filter a route for one routing device (R1) through another specific connected routing device (R2). When the route server (RS) gets several routes to a single destination (AS3) it will only use the best route. If that best route happens to go through the filtered connected routing device (R2) the route server (RS) will filter that route for routing device (R1) without sending another viable route. Routing device (R1) will now think there are no routes to destination (AS3) although there might be another route possible (through R3).

Path hiding can only occur when per-client policies are used on the route server. When no per-client policies are used to filter routes, then every routing device will be able to use all the best routes calculated by the route server.

To solve the problem of path hiding with the ability to use per-client policies the Internet Exchange Route Server draft mentions two different strategies. First the route server could keep multiple Routing Information Bases or RIB’s. In this situation a separate RIB is maintained for every connected routing device on the route server. The second strategy is for the route server to advertise multiple paths instead of just the best path. Both strategies are not mutually exclusive and could be implemented simultaneously.

The AMS-IX route servers use OpenBGPd 5.0 as their route server software which uses the multiple RIB technique to make sure that no path hiding can occur. This enables the use of
Research report
IPv4 shortage AMS-IX

per-client policies without path hiding. The route server derives its per-client policies from the IRRdb⁴ aut-num object defined by the customer.

The AMS-IX route servers are a service for its customers, this means that the customers may choose not to use the route servers if they do not want to. When a customer does not want to make use of the AMS-IX route servers they will have to set up and manage their neighbor relationships with other customers directly.

At the time of writing, 17 January 2012, the AMS-IX uses two route servers called Prefix and Radix. Both servers run on the same hardware platform which is a Dell PowerEdge R310 with the following specifications.

- Quad Core Xeon X3480 (3.06GHz, 8MB Cache, HT, DDR3-1333MHz)
- 24GB (6x4 Dual Rank RDIMMs) 800MHz
- 2x 300GB SAS 6Gbps 15K HD Hot Plug
- PERC H700A RAID Controller w/512MB Cache
- 16x SATA DVD-ROM
- Redundant PSU (400W)
- iDRAC6 w/VFLASH (8GB SD card)

Both servers run the OpenBSD 5.0 operating system using the OpenBGPd v4.6 software as routeserver.

In the following table some server specific information is listed.

<table>
<thead>
<tr>
<th>Physical location</th>
<th>Prefix</th>
<th>Radix</th>
</tr>
</thead>
<tbody>
<tr>
<td>FQDN</td>
<td>rs1.ams-ix.net</td>
<td>rs2.ams-ix.net</td>
</tr>
<tr>
<td>IPv4 address</td>
<td>195.69.144.255</td>
<td>195.69.145.0</td>
</tr>
<tr>
<td>Ipv6 address</td>
<td>2001:7f8:1::a500:6777:1</td>
<td>2001:7f8:1::a500:6777:2</td>
</tr>
</tbody>
</table>

Table 4: AMS-IX route servers

5.3 ARP Sponge

As stated in chapter 5.1 the AMS-IX Internet Peering VLAN is one big broadcast domain. IPv4 depends on broadcast messages such as ARP requests which are used to map IPv4 addresses to L2 MAC addresses. These broadcasts messages are indiscriminately sent to every other host in the broadcast domain and need to be processed by all receiving devices. In 'normal' sized IPv4 broadcast domains this behavior does not cause problems, but the AMS-IX Internet Peering VLAN is not a normal sized broadcast domain. When the amount of

---

⁴ The Internet Routing Registry database, IRRdb, is a registry on the Internet where routing policies are stored. This information is public and can be retrieved with a whois command. For more information on the IRRdb consult IETF RFC 2650, 'Using RPSL in Practise'.
clients increased over the years some clients started dropping BGP sessions because they were too busy processing ARP requests.

To remedy this problem the AMS-IX created a tool called the ARP Sponge. The ARP sponge will listen to all ARP requests and will reply to ARP requests when no other host replies. This will prevent hosts from sending multiple ARP requests when no-one answers. When a host comes up in the network it will usually send a gratuitous ARP message telling everyone its IP and MAC address. When the ARP sponge hears this it will stop ARP sponging for that host.

In 2009 a study was done by Marco Wessel and Niels Sijm as part of their masters study at the Universiteit van Amsterdam. This study asks the question “What differences are there between IPv4 and IPv6 as relating to the ARP Sponge and infrastructure, and is an IPv6 implementation of the ARP Sponge necessary?”. The conclusion of this study was that although IPv6 does not need an ARP sponge, the IPv4 broadcast domain benefits greatly by its implementation. A good example of the ARP Sponge’s effectiveness is shown in the following quote in chapter 1.4.1:

“The ARP sponge results in a significant decrease in ARP requests on the network. For instance, on May 12th, 2008 the ARP sponge was down for more than one hour due to a request from a customer. The average number of ARP packets per minute in that month was about 1,450. During the outage, a peak of 13,902 ARP packets per minute was measured, almost ten times as much as normal. Fifteen minutes after the ARP Sponge was restarted, the amount ARP traffic was reduced to its normal rate of about 1,450 packets per minute.”

The ARP Sponge v3.11 is hosted in the EU networks data-center in the Amsterdam Amstel business park on the mon-eun-014 machine.

<table>
<thead>
<tr>
<th></th>
<th>mon-eun-014</th>
</tr>
</thead>
<tbody>
<tr>
<td>Physical location</td>
<td>EU Networks</td>
</tr>
<tr>
<td>FQDN</td>
<td>Mon-eun-014.noc.ams-ix.net</td>
</tr>
<tr>
<td>IPv4 address</td>
<td>91.200.16.148</td>
</tr>
<tr>
<td>Ipv6 address</td>
<td>2001:67c:1a8:125::54</td>
</tr>
</tbody>
</table>

Table 5: ARP Sponge
5.4 Physical Network

At the time of writing, 18 January 2012, the AMS-IX's network is spread out over 8 independent data centers in the Amsterdam Metropolitan Area. The following locations are host to part of the AMS-IX network:

- SARA Amsterdam Science Park
- NIKHEF Amsterdam Science Park
- GlobalSwitch Amsterdam Slotervaart
- Equinix Amsterdam Zuid-Oost
- Telecity 2 Amsterdam Zuid-Oost
- Telecity 4 Amsterdam Amstel Business Park
- euNetworks Amsterdam Amstel Business Park
- Interxion AMS5 Schiphol-rijck

In 2012 the AMS-IX plans to spread out to 4 more locations.

- Terremark, Schiphol Noord
- EvoSwitch Haarlem
- Telecity 5 Amsterdam Zuid-Oost
- Equinix 3 Science Park

![Image 13: Geographical location the the AMS-IX data centers](image-url)
When we look at the actual topology we see the AMS-IX uses the following basic design:

The AMS-IX Internet Peering network is designed as a ‘collapsed backbone’ topology if we use Cisco’s hierarchical network design terminology. In a collapsed backbone the distribution and core layers are combined into one layer. In the case of the AMS-IX the core/distribution layer consists of 4 Brocade MLXe-32 routers.

One could argue that the Photonic eXchange Connection or PXC devices should be seen as access layer devices, but I do not agree with this because the PXC devices are no switches but rather automated patch panels, although the PXC devices do need to be powered unlike regular patch panels.

The following image shows the AMS-IX Internet Peering network topology at the time of writing, 20 Jan 2012:

In the second quarter of 2012 the AMS-IX is planning to add several routers between the access and core layer routers creating a separate distribution layer. This will split the 10-100 Ge connections into 2 separate router blocks. By introducing a separate distribution layer
traffic which is destined for another location within the same router block will not go through the core needlessly. When the separate distribution layer is added the topology will change to a quadruple core topology.

The AMS-IX uses the following networking equipment in the Internet Peering network.

- GlimmerGlass Intelligent Optical System 100
- GlimmerGlass Intelligent Optical System 600
- Brocade MLX-8
- Brocade MLX-16
- Brocade MLX-32
- Brocade MLXe-32

The GlimmerGlass Intelligent Optical System 100 and 600 are best to be described as automated fiber patch panels. Small mirrors in the devices redirect the optical signal to the correct egress port. When triggered, for example when the switch on that egress port fails, the mirrors will automatically switch to a backup egress port connected to a backup switch. According to GlimmerGlass the switching process can take up to 20 milliseconds\textsuperscript{xiii}. The difference between the 100 and 600 systems is the number of ports available.

The Brocade MLX series routers are classed by Brocade as carrier-class routers. Brocade lists the following specifications in the MLX series routers data sheet:\textsuperscript{xiii}

<table>
<thead>
<tr>
<th>Features</th>
<th>MLX-8</th>
<th>MLX-16</th>
<th>MLX-32</th>
<th>MLXe-32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface slots</td>
<td>8</td>
<td>16</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>Switch fabric capacity</td>
<td>1.92 Tbps</td>
<td>3.84 Tbps</td>
<td>7.68 Tbps</td>
<td>15.36 Tbps</td>
</tr>
<tr>
<td>Data forwarding capacity</td>
<td>1.28 Tbps</td>
<td>2.56 Tbps</td>
<td>5.12 Tbps</td>
<td>6.4 Tbps</td>
</tr>
<tr>
<td>Maximum 100 GbE ports</td>
<td>4</td>
<td>8</td>
<td>16</td>
<td>32</td>
</tr>
<tr>
<td>Maximum 10 GbE ports</td>
<td>64</td>
<td>128</td>
<td>256</td>
<td>256</td>
</tr>
<tr>
<td>Maximum 1 GbE ports</td>
<td>384</td>
<td>768</td>
<td>1536</td>
<td>1536</td>
</tr>
</tbody>
</table>

Table 6: Brocade MLX specifications
5.5 Logical Network

The customer routing devices use the eBGP routing protocol to exchange routes with each other. eBGP needs a TCP connection between two hosts to be able to form a neighbor relationship. In the past the AMS-IX provided a single physical switch per data center for customers to connect to. A problem with a single switch is that it does not have an infinite amount of interfaces. At some point the AMS-IX had to find a way to add more switch ports in their network for customers to connect to. The obvious solution was to add more switches.

A problem with Ethernet switching is the occurrence of switching loops in a redundant network. The rule for a switch to forward broadcast traffic is to send it out of every other port except back out into the one where the switch received the frame on. This rule can quickly lead to 'broadcast storms' when a redundant topology is used.

In the example above an infinite loop is created, (1, 2, 3, 4, 2, 3, 4, 2, 3, 4, etc.). The broadcast will forever loop the network and any device it passes will have to dedicate precious processor time and memory handling the frames unless the loop is broken somehow.

To still be able to use redundant links as a backup the Spanning Tree Protocol, STP (802.1D), was developed by the Institute of Electrical and Electronics Engineers, IEEE. STP seeks out and disables all redundant links for all traffic except STP’s own Bridge Protocol Data Units, BPDUs. If the active link should fail, then STP will recalculate a new path and unblock one of the backup links ensuring that no switching loop will form.
The problem with STP is that it entirely blocks a viable route to all traffic except its own. This wastes a lot of load balancing potential. Also convergence in the case of a link failure is relatively slow. Even with updated technologies like Rapid Spanning Tree Protocol, RSTP (802.1w), or Multiple Spanning Tree Protocol, MSTP (802.1s) convergence is not as fast as a routing protocol like Open Shortest Path First, OSPF. The following quote was taken from a research paper created by Cisco called ‘Cisco’s High Availability Campus Network Design – Routed Access Layer using EIGRP or OSPF’xxiv: “Comparing the convergence times for an optimal Layer 2 access design (either with a spanning tree loop or without a loop) against that of the Layer 3 access design, you can obtain a four-fold improvement in convergence times, from 800–900msec for the Layer 2 design to less than 200 msec for the Layer 3 access.”

A routed network is not ideal either. The problem with a routed network is that it routes between broadcast domains. In other words, a routed network would split the current network into multiple broadcast domains. This is unacceptable because the routing protocol used by the customers, eBGP, should not form neighbor relationships across multiple hops.

The solution AMS-IX uses to increase port capacity by adding multiple redundant switches without the drawbacks of Spanning Tree Protocol while still keeping only one broadcast domain is a combination of MultiProtocol Label Switching, MPLS, together with Virtual Private LAN Service, VPLS.

First we will look at MPLS which creates multiple, load balancing paths through the network without forming switching loops. MPLS works with labels, every IP packet is assigned an MPLS label by the first router it passes. This router checks which path the packet should take. Each route or Label Switched Path, LSP, has a different label assigned to it. When the label is assigned to the packet then the router passes it on to the next router along the LSP. The next router only checks the label on the packet and sends it on its way along the LSP. When the packet reaches the end of the LSP then the router ‘pops’ the label and delivers it to its destination.

Although LSPs can be automatically created, AMS-IX maintains their LSPs manually. It was found that when this process was done automatically sometimes undesirable paths were chosen instead of more direct paths.
VPLS creates a Virtual Private Network, VPN, by encapsulating packets. When a Customer Edge, CE, router sends a packet into the broadcast domain it first passes a Provider Edge, PE, device which encapsulates the packet and then sends it through the network. When the packet arrives at the destination PE device then the PE device strips the VPLS encapsulation and sends it to the receiving CE router. The VPLS infrastructure is transparent for both CE routers.
5.6 Customer Equipment

The AMS-IX does not manage any aspect of the customer equipment connected to its network. Because of this a large range of routing equipment from different vendors can be expected to be connected to the AMS-IX network. I would like to know what vendors are most commonly used on the AMS-IX network so we know what vendors we should speak to if we want them to implement a feature in their routing software in the future.

We can gather data about which vendors have routing equipments on the AMS-IX network by looking at the MAC addresses traversing the network. Each routing device uses a unique 48-bit MAC address of which the first 24 bits identify the routing vendor.

The following list was given to me by Arien Vijn on the 15\textsuperscript{th} of may 2012 and shows what vendors and how many routing devices per vendor are used on the AMS-IX Internet Peering VLAN.
<table>
<thead>
<tr>
<th>Vendor</th>
<th>Number of routing devices</th>
</tr>
</thead>
<tbody>
<tr>
<td>3COM</td>
<td>1</td>
</tr>
<tr>
<td>AIRESPIDER</td>
<td>1</td>
</tr>
<tr>
<td>ALCATEL</td>
<td>3</td>
</tr>
<tr>
<td>ASUSTEK</td>
<td>1</td>
</tr>
<tr>
<td>AXIOM</td>
<td>1</td>
</tr>
<tr>
<td>Belden</td>
<td>1</td>
</tr>
<tr>
<td>BROCADE</td>
<td>60</td>
</tr>
<tr>
<td>CAPORIS</td>
<td>2</td>
</tr>
<tr>
<td>CISCO</td>
<td>254</td>
</tr>
<tr>
<td>COMDA</td>
<td>2</td>
</tr>
<tr>
<td>COMMIL</td>
<td>2</td>
</tr>
<tr>
<td>CONNECT</td>
<td>1</td>
</tr>
<tr>
<td>DELL</td>
<td>4</td>
</tr>
<tr>
<td>DITECH</td>
<td>1</td>
</tr>
<tr>
<td>ECOPILT</td>
<td>2</td>
</tr>
<tr>
<td>ERICSSON</td>
<td>1</td>
</tr>
<tr>
<td>EXTREME</td>
<td>3</td>
</tr>
<tr>
<td>FERRAN</td>
<td>1</td>
</tr>
<tr>
<td>FORCE10</td>
<td>1</td>
</tr>
<tr>
<td>HITRON</td>
<td>2</td>
</tr>
<tr>
<td>HUAWEI</td>
<td>1</td>
</tr>
<tr>
<td>INTEL</td>
<td>4</td>
</tr>
<tr>
<td>JETCELL</td>
<td>1</td>
</tr>
<tr>
<td>JUNIPER</td>
<td>177</td>
</tr>
<tr>
<td>LANNER</td>
<td>1</td>
</tr>
<tr>
<td>NEWPORT</td>
<td>1</td>
</tr>
<tr>
<td>ON</td>
<td>1</td>
</tr>
<tr>
<td>PENTACOM</td>
<td>1</td>
</tr>
<tr>
<td>PITNEY</td>
<td>1</td>
</tr>
<tr>
<td>RACKABLE</td>
<td>1</td>
</tr>
<tr>
<td>SPLICECOM</td>
<td>2</td>
</tr>
</tbody>
</table>
In the previous table we can see that the top 3 vendors used on the AMS-IX Internet Peering VLAN are:

1. Cisco 254
2. Juniper 177
3. Brocade 60

These three vendors are clearly the most widely used vendors on the AMS-IX Internet Peering VLAN.
6 Solutions

With the information in the previous chapters we can start researching possible solutions to the main research question, 'How can new routing devices participate in the sharing of BGP IPv4 routes on the main AMS-IX Internet Peering VLAN when the current IPv4 range is depleted?'. To help me find possible answers for the main question I will research the following questions.

- Can the current IPv4 range be increased?
- Can the IPv4 routes be shared with a IPv6 peer relationship described in IETF RFC 5549?

6.1 Increase IPv4 Range

At the time of writing, 02 March 2012, the AMS-IX uses a /22 IPv4 range in their Internet Peering VLAN. This means that there are 1024 available IPv4 addresses. As explained in chapter 4.3 this will not be enough. The most obvious solution is to increase the current IPv4 range to accommodate for new clients. In this chapter I will research this possibility.

How many IP addresses would be enough for future use? This question cannot be answered in a definitive way. We can make an educated guess however. In chapter 4.3 I assumed that the IPv4 address usage would increase every year with 100. I stopped counting in 2016 because in that year the limit of 1024 will be reached this way.

If we could change the IPv4 range to a /21 we would have doubled our available addresses to 2048. Starting with 600 IPv4 addresses in 2012 and adding 100 per year will give us 14.5 years before the /21 range will run out, this would be some time in 2026. This increase should give AMS-IX enough time to prepare for other more permanent solutions.

At first glance this seems like a simple solution. However it does have two major problems which have to be overcome. First we need to make sure we can actually obtain a /21 IPv4 range. And second such a big broadcast domain will have a lot of broadcast traffic. How does this affect hosts on the broadcast domain?

6.1.1 Finding IPv4 Address Space

IPv4 addresses have become a very scarce resource on the Internet. To create a bigger IPv4 range the AMS-IX would have to use a /21 network which would allow for 2048 IPv4 addresses. AMS-IX has a unique relationship with RIPE NCC in that it gets its IPv4 range directly from RIPE NCC instead of from a LIR. RIPE NCC manages all IP addresses in Europe, the Middle East, and Central Asia and only assigns /21 IPv4 ranges to Local Internet Registries, LIRs, or if there is a clear need for them within 12 months.

The number 1024 includes 2 non-usable network and broadcast addresses which would make the actual available range 1022.
To become a LIR the AMS-IX has to become a member of RIPE NCC. All LIRs have to implement RIPE NCC’s policies described in the document “IPv4 Address Allocation and Assignment Policies for the RIPE NCC Service Region”, or ripe-530xxv.

Important rules regarding the AMS-IX situation are as follows:

**Chapter 5.0:** “The RIPE NCC allocates enough address space to LIRs to meet their needs for a period of up to 12 months.”

**Chapter 5.1:** “The RIPE NCC’s minimum allocation size is /21.”

**Chapter 5.1:** “Members can receive an initial allocation when they have demonstrated a need for IPv4 address space.”

**Chapter 5.6:** “Allocations for LIRs from the last /8
1.a. LIRs may only receive one allocation from this /8. The size of the allocation made under this policy will be exactly one /22.”

I have had contact about the possibilities for AMS-IX to become a LIR with Nathalie Trenaman who is works at RIPE NCC. She gave me the following reply in Dutch:

“De minimum allocation size is een /21 (Chapter 5.1: “The RIPE NCC’s minimum allocation size is /21.”) Om daarvoor in aanmerking te komen moet je 1 IP kunnen verantwoorden. Daarbij kijken we niet naar wat je al hebt (de /22) en die mag je ten alle tijden houden en hoeft niet omgenummerd te worden.”

Which translates to:

The minimum allocation size is a /21 (Chapter 5.1: “The RIPE NCC’s minimum allocation size is /21.”). To be eligible for this range you will have to be able to justify 1 IP address. We do not look at what has already been assigned to you (the /22). You can keep your original range which does not have to be renumbered.

According to Nathalie Trenaman it is possible to become a LIR and get an extra /21 IPv4 range.

If the AMS-IX should decide to become a LIR it should do this as soon as possible because when the RIPE NCC starts to allocate its last /8 range even LIRs will not be able to get a / 21xxvi.

### 6.1.2 Broadcasts

IPv4 is heavily dependent on broadcast messages which are directed to every other host in the broadcast domain. For example the ARP mechanism uses broadcast messages to map IPv4 addresses to L2 addresses which are used for actual transmission over the line.

As discussed in chapter 5.3 “ARP Sponge" the AMS-IX already experienced trouble with routers succumbing to the high number of ARP traffic on the broadcast domain without the use of the special ARP sponge software. As the number of client routers using IPv4 increase the number of ARP messages will logically also increase making the event of a ARP sponge failure even more profound.
6.1.3 Recommendation
It would be best for AMS-IX to become a LIR and get the /21 range now before the RIPE NCC starts allocating its last /8 IPv4 range. This will give the AMS-IX more time to find a better and more permanent solution.

6.2 RFC 5549
IETF RFC 5549 ‘Advertising IPv4 Network Layer Reachability Information with an IPv6 Next Hop’ was written by F. Le Faucher and E. Rosen from Cisco systems. I have added a copy of RFC 5549 in this document as Appendix 1.

RFC 5549 proposes a change in MP-BGP that would allow IPv4 destinations to be reached through an IPv6 next hop address.

This method would solve the IPv4 shortage problem in the Internet Peering VLAN because no IPv4 addresses would be needed at all. All routing devices would form neighbor relationships with each other over IPv6 and use those addresses as next hop for both IPv4 and IPv6 destinations.

At first glance this might seem to be impossible. How can a packet change from an IPv4 packet into an IPv6 packet? When we look closer, however, we can see that the routing devices never actually change the L3 packets at all. The routing devices only change the source, SRC, address and the destination, DST, address of the L2 Frame which encapsulates the L3 packet.
As we can see the L3 packet itself never changes. The first router knows that to reach the 101.0.0.0/24 network it should send the packet to the IPv6 next hop address 2001::2. It then checks which L2 address corresponds with IPv6 address 2001::2, changes the SRC and DST L2 frame addresses and sends the L3 packet unchanged but with a new L2 frame encapsulation on its way to the next router. One important thing the sending router must do is to set the Ethertype field in the frame to IPv4. Although the packet might be sent to an IPv6 next hop address, the packet is still an IPv4 packet and the receiving router must process it as such.

This means that as long as the routers in between have some way of knowing what L2 address corresponds with what L3 next hop address it should not matter what L3 protocol is used in between hops. The original L3 packet keeps the original IPv4 SRC and DST addresses.

### 6.2.1 RFC 5549 Details

MP-BGP uses the MP_REACH_NLRI path attribute to advertise routes, consult chapter 3.3 and 3.4 for more information on BGP packets. To identify which address family is used in both the Network Address of Next Hop, NANH, and the Network Layer Reachability Identifier, NLRI, the MP_REACH_NLRI uses the Address Family Identifier, AFI, and Subsequent Address Family Identifier, SAFI, fields.

---

6 Except for the packet’s TTL and checksum fields
For example, when the AFI is 1 (IPv4) then both the NANH and NLRI must be encoded as IPv4 addresses.

RFC 5549 suggest that if the AFI is set to 1 (IPv4) and the SAFI is either 1, 2, 4, or 128 then the NANH must be interpreted as a IPv6 address when the Length of Next Hop Address, LNHA, field is set to either 16 or 32.

The LNHA field contains the length of the following NANH field in bytes. An IPv6 address is 128 bits, or 16 bytes in length. So RFC 5549 tells us that whenever you encounter a MP_REACH_NLRI with a AFI of 1 (IPv4) but with a LNHA of 16 or 32, then the NANH must be interpreted as one IPv6 address in the case of 16, and as two IPv6 addresses in the case of 32⁷.

### 6.2.2 RFC 5549 Capability Advertisement

RFC 5492 specifies a way for BGP routing devices to advertise to potential neighbors what BGP capabilities they support by including them in the BGP OPEN message.

The BGP capabilities are added in the Optional Parameters, OP, field. The potential neighbor can examine the sent capability list and determine what capabilities they both support. When both routing devices support a capability they may use it in their BGP connection in the future.

The OP field as described in RFC 4271 ‘BGP-4’ consists of one or more parameters which are made up of triples <Parameter Type, Parameter Length, Parameter Value>. Each parameter triple is encoded as follows:

<table>
<thead>
<tr>
<th>Parameter Code</th>
<th>Parameter Length</th>
<th>Parameter Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 B</td>
<td>1 B</td>
<td>VAR</td>
</tr>
</tbody>
</table>

**Parameter Code:**
A 1 byte field which identifies what kind of optional parameter this is. IANA is responsible for the list of Parameter codes.

---

⁷ The second IPv6 address could be a link local address
### Parameter Length:
A 1 byte field which indicated the length in bytes of the following Parameter Value field.

### Parameter Value:
A 1 byte field which contains the actual information of the optional parameter. The actual Capability Parameter Value field may be one or more triples <Capability code, Capability Length, Capability Value>. Each capability triple is encoded as follows:

<table>
<thead>
<tr>
<th>Capability Code</th>
<th>Capability Length</th>
<th>Capability Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 B</td>
<td>1 B</td>
<td>VAR</td>
</tr>
</tbody>
</table>

**Image 25: Capability parameter field**

### Capability Code:
A 1 byte field which identifies the capability advertised. IANA is responsible for maintaining the list of capability codes.

### Capability Length:
A 1 byte field which indicated the length in bytes of the following Capability Value field.

### Capability Value:
a variable length field which contains additional information about the advertised capability.

RFC 5549 uses Capability Code 5 'Extended Next Hop Encoding' to advertise that a routing device supports RFC 5549. the Capability Value for RFC 5549 is made up of one or more triples <NLRI AFI, NLRI SAFI, Nexthop AFI> which indicate combinations of NLRI address families support what other next hop address family. Each triple is encoded as follows:

<table>
<thead>
<tr>
<th>NLRI AFI</th>
<th>NLRI SAFI</th>
<th>Nexthop AFI</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 B</td>
<td>2 B</td>
<td>2 B</td>
</tr>
</tbody>
</table>

**Image 26: RFC 5549 Compatibility Value**

### NLRI AFI:
A 2 bytes field which together with the NLRI SAFI field indicates the address family of the destination NLRI. The only NLRI AFI which concerns RFC 5549 is 1 (IPv4).

### NLRI SAFI:
A 2 byte field which together with the NLRI AFI field indicate the address family of the destination NLRI. The only NLRI SAFIs which concerns RFC 5549 are 1 (Unicast), 2 (Multicast), 4 (NLRIIs with MPLS labels), and 128 (MPLS labeled VPN address).

### Nexthop AFI:
A 2 byte field which indicates the address family of the nexthop address. The only NLRI which concerns RFC 5549 is 2 (IPv6).

We can use the capability advertisements in BGP OPEN messages to check if a routing device supports RFC 5549. For our research we can determine if a routing device is compatible with RFC 5549 if it sends the following information in its BGP OPEN message:

- Parameter Code 2 Capability
- Capability Code 5 RFC 5549, Extended Next Hop Encoding
- NLRI AFI 1 IPv4
- NLRI SAFI 1 Unicast
- Nexthop AFI 2 Ipv6

This combination would allow the routing devices to use an IPv6 next hop address for an IPv4 destination.

### 6.2.3 RFC 5549 Testing

During my research I searched for software implementations of RFC 5549 so I could test this solution. My search yielded no implementations anywhere. I have tried to contact the authors F. Le Faucher and E. Rosen via e-mail on Friday the 02\textsuperscript{nd} of March 2012 to ask if they might know of any implementations. On Tuesday the 13\textsuperscript{th} of March 2012 I received the following reply from F. Le Faucher:
“Hello Stefan,

To our knowledge it is not supported yet. We have requested confirmation and will get back to you know if it turned out to be supported.
If you have a firm requirement for this, as usual, I'd recommend you make that clear to your account team.
Thank you and sorry for the delay in responding.

Francois”

On Friday 16-03-2012 I also received the following e-mail from Bertrand Duvivier, the BGP Product Manager at Cisco:

“All,

This is a radar item, not yet in any roadmap, …
nor in IOS classic, nor in NX-OS or nor in IOS-XR.

BRGDS Bertrand
BGP Product Manager”

These e-mails confirm that at least Cisco has no implementations yet and is not planning on implementing this solution in the near future. Cisco and other routing vendors need a good reason to spend their resources on implementing new features. The best justification to implement a new feature is a demand for it. If my research shows that RFC 5549 is a technically sound solution for the AMS-IX then I should build my own proof of concept. This proof of concept can then be used to demonstrate RFC 5549 in action and get enough clients of the AMS-IX and other Internet exchanges interested so they will ask for this feature to be implemented into their routing software by their vendors.

6.2.3.1 RFC 5549 Forwarding

As stated above in chapter 6.2.1, RFC 5549 assumes that a router is able to send an IPv4 packet to an IPv6 next hop address. To test if this is possible I will try to set up a static IPv4 route with an IPv6 next hop associated with it on a Linux system. I choose to do this test on a Linux system because of the open-source nature and popularity of the Linux kernel.

My test setup consists out of two Ubuntu v11.10 systems running on top of Linux kernel v3.0.0-16-generic which is based on the Linux kernel v3.0.4 upstream stable kernel. It should be noted that these two systems are Virtual Machines (VMs) running inside VirtualBox on a Windows 7 laptop.

In test_#1 I want to prove that we in fact only need to know the correct L2 address of the next hop to forward a packet. In the following example I have added static routes which forwards packets to a non-existent IPv4 next hop.
The non-existent IPv4 address does point to the correct next hop L2 address in the ARP tables of both systems because I manually added these entries.

As expected the result of a ping from Ubuntu-01 to 192.168.2.1 succeeds without any problems.

It is important to note that Linux does not allow us to configure the next hop to be outside of one of our directly connected subnets, for example 192.168.100.10. When configuring such a route the Linux RTnetlink messenger system, which is used for messaging routing information between Linux user and kernel space, will report the error "RTNETLINK answers: No such process" indicating that this is not allowed. In theory there is no reason why, when using the method shown in test_#1, a bogus next hop IPv4 address outside of a directly attached subnet should not work.

Linux also allows you to configure a static route without any next hop IP address at all, only the outgoing interface is specified. This method relies on the ability of other routers to perform Proxy ARP.

When we use this method on Ubuntu-01 it will think and act like network 192.168.2.0/24 is directly connected to one of its own interfaces, eth2 in this case. When a packet destined for 192.168.2.1 passes Ubuntu-01, it will send out an ARP request directly asking for the MAC.
address for 192.168.2.1. Of course 192.168.2.1 cannot respond itself, but a router holding a route to the 192.168.2.0/24 network can act like a proxy. It pretends it is 192.168.2.1 by sending its own MAC address back as an ARP reply. Ubuntu-01 will now send all packets destined for 192.168.2.1 to directly to the other router's MAC address, because it believes that the other router actually is 192.168.2.1.

In theory we do not need any IP address for our next hop, we only need the L2 MAC address. The only reason we still need to give the interfaces in the intermediary hop an IPv4 address at all is because Linux, does a sanity check to protect against faulty configurations. Linux does not process any IPv4 packets coming in or wanting to go out of an interface which does not have an IPv4 address assigned to it.

Test_#1 suggests that using an IPv6 next hop for an IPv4 destination should indeed be possible if the system knows what L2 address is associated with that IPv6 address. Usually IPv6 addresses are stored in a different neighbor table than IPv4. When the routing table contains an IPv6 next hop the system must know to look in the IPv6 Neighbor Discovery, ND, table.

Now we can try to send an IPv4 packet to an IPv6 next hop address.

A simple test shows that Linux does not support an IPv4 route with an IPv6 next hop.

Ubuntu-01: sudo route add -net 192.168.2.0 netmask 255.255.255.0 gw 2001::2 dev eth2

2001::2: Unknown host

Ubuntu-01: sudo ip route add 192.168.2.0/24 via 2001::2

Error: an inet address is expected rather than “2001::2”.

It is no big surprise that Linux does not support a IPv6 address as a next hop for an IPv4 route. The Linux kernel was never designed to handle this kind of scenario. The implementation of "native inter-protocol forwarding" is beyond the scope of this research report.

6.2.3.2 RFC 5549 IPv6 to IPv4 Translation Table

To create a proof of concept for RFC 5549 without any major changes to the Linux kernel we could translate the received IPv6 next hop addresses into a non-existent IPv4 addresses which we can then insert into the main routing table without any problems. This solution relies on the fact that we can forward a packet to a bogus IPv4 next hop address like shown in test_#1. For this method to work we do need to know on which L2 address the bogus IPv4 address can be reached. In the translation from IPv6 to IPv4 address we can update the...
ARP table so that the new bogus IPv4 address has the same L2 address associated with it as the original IPv6 next hop.

The bogus IPv4 address is only important on the router itself, it has no meaning on other devices including its direct neighbors on the same subnet. Because of this a routing device does not have to advertise to other routing devices into which IPv4 address it translates an IPv6 address. We do have to keep in mind that we should not use a bogus IPv4 address which may indeed exist somewhere as this address will not be reachable anymore when we associate a different L2 address to it. Because of this we will need to use a “special-use” IPv4 address range for our translation table which will never be used by an actual host. There are several of these ranges in IPv4 which are described in RFC 5735.

<table>
<thead>
<tr>
<th>IPv4 address range</th>
<th>Description</th>
<th>Described in IETF RFC #</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0.0.0/8</td>
<td>This network</td>
<td>RFC 1122</td>
</tr>
<tr>
<td>10.0.0.0/8</td>
<td>Private IPv4 address range</td>
<td>RFC 1918</td>
</tr>
<tr>
<td>127.0.0.0/8</td>
<td>Localhost</td>
<td>RFC 1122</td>
</tr>
<tr>
<td>169.254.0.0/16</td>
<td>Link Local</td>
<td>RFC 3927</td>
</tr>
<tr>
<td>172.16.0.0/12</td>
<td>Private IPv4 address range</td>
<td>RFC 1918</td>
</tr>
<tr>
<td>192.0.0.0/24</td>
<td>IETF protocol assignments</td>
<td>RFC 5736</td>
</tr>
<tr>
<td>192.0.2.0/24</td>
<td>TEST-NET-1</td>
<td>RFC 5737</td>
</tr>
<tr>
<td>192.88.99.0/24</td>
<td>6to4 anycast relay</td>
<td>RFC 3068</td>
</tr>
<tr>
<td>192.168.0.0/16</td>
<td>Private IPv4 address range</td>
<td>RFC 1918</td>
</tr>
<tr>
<td>198.18.0.0/15</td>
<td>Benchmark tests</td>
<td>RFC 2544</td>
</tr>
<tr>
<td>198.51.100.0/24</td>
<td>TEST-NET-2</td>
<td>RFC 5737</td>
</tr>
<tr>
<td>203.0.113.0/24</td>
<td>TEST-NET-3</td>
<td>RFC 5737</td>
</tr>
<tr>
<td>224.0.0.0/4</td>
<td>IPv4 Multicast</td>
<td>RFC 3171</td>
</tr>
<tr>
<td>240.0.0.0/4</td>
<td>Reserved for future use, except for 255.255.255.255</td>
<td>RFC 1112, RFC 0919, RFC 0922</td>
</tr>
</tbody>
</table>

Table 10: Special-use IPv4 address ranges

For the use of our translation table we need to be absolutely sure that a bogus address can never take the place of a valid IPv4 address because we will assign a new L2 address to that IPv4 address ensuring that the original device assigned with that same IPv4 address is no longer reachable. This excludes the following IPv4 ranges:

- Private IP ranges 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16
- Local host IP range 127.0.0.0/8
- Link local IP range 169.254.0.0/16
- 6to4 anycast IP range 192.88.99.0/24
- benchmark tests IP range 198.18.0.0/15
The reason for my research is that the AMS-IX Internet Peering VLAN's /22 network is running out of IPv4 addresses. For this reason we need to use a bigger subnet. I propose using a /16 which would give us 65,536 available bogus IPv4 addresses. This should be more than enough for even the biggest Internet exchanges including the AMS-IX. This rule excludes the following IPv4 ranges:

- IETF protocol assignments 192.0.0.0/24
- TEST-NET-1 192.0.2.0/24
- TEST-NET-2 198.51.100.0/24
- TEST-NET-3 203.0.113.0/24

This only leaves us the 0.0.0.0/8 and 240.0.0.0/4 networks. The 240.0.0.0/4 network is reserved for future use. Although this means that it could be opened up in the future for valid unicast IPv4 addresses I do not believe that this will actually happen. Because the 240.0.0.0/4 range has been an invalid IPv4 range there are now a lot of software implementations who filter any packet which have a source IPv4 address in the 240.0.0.0/4 range.

Because the 240.0.0.0/4 range will probably never be used for actual hosts on the Internet I think the usage for an internal IPv6 to IPv4 translation table would make good use of these otherwise difficult to use IPv4 addresses. I propose using the first 240.0.0.0/16 IPv4 address range for this use. This means all addresses between 240.0.0.0 and 240.0.255.255 may be used in a routing devices internal IPv4 to IPv6 translation table.

When we implement an IPv6 to IPv4 translation table we have to think about where we implement this. There are 2 ways to do this. The first way is to let the BGP routing daemon itself manage the translation table. The second is to let the kernel do this. Both solutions have pros and cons. First lets look at what happens when the BGP daemon manages the IPv6 to IPv4 translation table. A RFC 5549 enabled routing device would process a IPv4 packet as follows:
In this method we let the BGP deamon directly update the ARP table, the kernel cannot access the translation table directly. This means that the routing deamon itself has to make sure that the ARP table contains the correct entries for the 240.0.0.0/16 translation addresses because the kernel cannot simply use ARP to resolve these addresses as they do not really exist.

One method to keep the ARP table valid would be to let the BGP deamon resolve the next hop IPv6 address itself after it has updated its translation table after receiving an update message. If the next-hop IPv6 address is the same as the machine sending the update then the L2 address for that IPv6 next hop should be known in the ND table and can be copied over into the ARP table. The AMS-IX network uses route servers introduced in chapter 5.2. These route servers never send their own IP address as a next hop. When the next hop is not the same as the update message sender then it is possible that the ND table does not contain an entry for the next hop. In this situation the BGP deamon should first use ND to resolve the correct L2 address for the next hop.

When we let the BGP deamon itself resolves the L2 address after it has updated the translation table it should do this in parallel to its other jobs. This is because when the IPv6 address can not be found in its ND table it has to wait for ND to resolve the IPv6 address. When processing this in serial the BGP deamon will be stuck waiting for ND to give an answer.
Because the kernel has no way to resolve a 240.0.0.0/16 address with ARP we need to make sure that the ARP table is pro-actively updated and kept up to date with the translation and ND tables. For this we should have a separate L2 validation process which runs parallel to the BGP deamon and which continually checks if the IPv6 addresses in the translation table are present in the ND table. If the IPv6 address is present then we can update the ARP table, if not we should try to resolve the IPv6 address using ND and then update the ARP table with the results.

When we let the kernel itself manage the IPv6 to IPv4 translation table we can create a forwarding rule saying that for every frame going out to a 240.0.0.0/16 address do not look into your ARP table but instead look into your IPv6 to IPv4 translation table. A RFC 5549 enabled routing device would process a IPv4 packet as follows:

Letting the kernel manage the IPv6 to IPv4 translation table is clearly a much more elegant solution. No extra process is needed to keep the L2 table updated. This method does require that the kernel itself is updated to handle forwarding to 240.0.0.0/16 addresses.

### 6.2.3.3 RFC 5549 Proof of Concept

To implement RFC 5549 we can choose between one of the following methods:

- Native inter-protocol forwarding
- IPv6 to IPv4 translation table managed by the forwarding engine
- IPv6 to IPv4 translation table managed by the BGP deamon

Both native inter-protocol forwarding and letting the IPv6 to IPv4 translation table be managed by the forwarding engine would require Linux kernel adaptations. I want the first proof of concept for RFC 5549 to not require any kernel changes yet because this is easier for me to implement and also anyone wanting to review this application can then install it on a regular Linux kernel. This is why I have chosen to implement RFC 5549 with a IPv6 to IPv4 translation table managed by the routing deamon.
When we let the BGP daemon manage the IPv6 to IPv4 translation table we still need to adapt the Linux kernel to allow the IPv4 routing table to be updated when the next hop is not in any directly connected subnet. In my proof of concept I will not yet use the 240.0.0.0/16 next hop range because this would require a small update to the Linux kernel. Instead I will use another non-existent IPv4 address in a directly connected subnet as shown in test_#1 in chapter 6.2.4.1.

For details on the code itself and information on how to set up this implementation for yourself I have written a separate Implementation guide document which I have added as Appendix 1 to this document.

To test my RFC 5549 implementation I have set up a test network like depicted in image 33 on the next page. In this network all three central Ubuntu-R* machines have been installed as routers running the adapted Quagga software. The 3 outer Ubuntu-H* machines are regular Ubuntu machines not running any special software. The central "AMS-IX" cloud is a VirtualBox Internal Network which acts as a switch between the 3 central Ubuntu-R* machines.

In the test setup I was able to get complete connectivity between all Ubuntu-H* hosts using only RFC 5549 BGP update messages between the Ubuntu-R* machines. This proof of concept shows that we can indeed advertise IPv6 next hop addresses for a IPv4 route which would eliminate the need for IPv4 addresses in an intermediate network segment such as internet exchanges.
My proof of concept does show that there is a problem with traceroute messages. When a traceroute message is sent from, for example, Ubuntu-H01 to Ubuntu-H02 no reply can be sent back by Ubuntu-R02 because Ubuntu-R02 will reply with its own IPv4 address 200.0.0.1. Because Ubuntu-R02 also uses 200.0.0.1 it will drop this packet thinking that someone else is using the wrong IP address. When either Ubuntu-R01 or Ubuntu-R03 is configured with another IPv4 address the tracerace succeeds.

Although assigning unique IPv4 addresses to all routers in the AMS-IX broadcast domain enables the use of traceroute I do not believe that this will be a good permanent solution. This is because when we allow this behavior then the translation table will contain IPv4 addresses which might actually be used on the broadcast domain. This could lead to a type of ARP poisoning where the ARP table is updated to send traffic to the wrong next hop. For example when in the translation table a host has the IPv4 address of 200.0.0.11 which corresponds to IPv6 address 2001::1 then when a traceroute TTL exceeded message is received from 200.0.0.11 but with a different MAC address than IPv6 2001::1. then the ARP table is updated to use the new 200.0.0.11 MAC address which would lead to the routing device sending all traffic to the new, but wrong 200.0.0.11 device.

A better solution would be to add a forwarding rule which preemptively checks if the TTL will be exceeded at the next hop. If that will happen than you can already send an ICMP packet back with a TTL exceeded message containing, for example, the BGP ID “IPv4 address”. We know that the TTL will be exceeded at the next hop because this technology will only be used in between hops, never at the end of a route. By doing this we may not be able to trace the route to the actual next hop, but we at least know what network the traceroute was going to use to get to that next hop and on which device the ‘TTL == 1’ rule was triggered, assuming that unique BGP IDs are used.

The next hop will never be the actual destination of the route so we can safely say that this packet will never reach its destination if the TTL is 1, even if we were to forward it.

Image 34: Forwarding rule to preempt a TTL expired message

This technique requires the new forwarding rule to be implemented in the forwarding engine after the initial rule which checks if the next hop is in the special 240.0.0.0/16 range. We can
only do this after the special forwarding range check because we should only use this TTL
preemption rule if we are absolutely sure that the next hop cannot be the final destination.

Because this technique requires the forwarding engine to be updated it will probably be a
good idea to also implement the translation table in the forwarding engine at the same time,
although it is possible to still let the routing daemon manage the translation table.

6.2.4 Recommendation
RFC 5549 has the potential to eliminate the need for IPv4 entirely for the AMS-IX. The proof
of concept shows that using an IPv6 next hop for an IPv4 destination is possible.

The proof of concept shows that although RFC 5549 can be implemented without adapting
the current Linux forwarding engine we then do get problems with traceroute.

When we choose not to use native inter-protocol forwarding we should add forwarding rules
which state that when the next hop falls in the special translation range than we should look
into the translation table to get our IPv6 next hop. After this forwarding rule should be another
rule which preemptively sends out a TTL unreachable if the TTL equals 1.

If RFC 5549 is to be implemented in the real world it should be done with either native inter-
protocol forwarding or by letting the forwarding engine manage the translation table both of
which are more streamlined than letting the routing daemon manage the translation table.

This is a new technique which should definitely be researched further. I recommend that the
AMS-IX should contact the three biggest routing vendors\(^8\) on the AMS-IX Internet Peering
VLAN which are Cisco, Juniper, and Brocade, and inform them about the need to research
and implement RFC 5549. The AMS-IX should also inform their customers about this
research report and ask them to also ask their own routing vendors to implement RFC 5549
in their routing equipment so their routing vendors can justify spending resources on
implementing RFC 5549.

Although the AMS-IX probably still has until 2016 before their IPv4 range runs out. It is
important to act now and get RFC 5549 implemented by all routing vendors. When the
routing vendors have implemented RFC 5549 in their routing devices it should of course also
be implemented by the clients of the AMS-IX themselves. The AMS-IX cannot force its
clients to use RFC 5549, but it can educate them about the IPv4 problem and about the need
to use RFC 5549 forwarding in the future.

\(^8\) As can be seen in table 7 on page 35 and page 36
7. Conclusion

The conclusion of this research report is that the AMS-IX will probably have until 2016 before their IPv4 range will be depleted. When this happens, no new IPv4 routing devices can be added to the Internet Peering VLAN which would effectively halt the growth of IPv4 peers on the AMS-IX's Internet Peering VLAN.

Increasing the current IPv4 range from a /22 to a /21 range is possible but then the AMS-IX will have to become a RIPE NCC LIR. I advice the AMS-IX to become a LIR so it has more time to find better alternatives, for example RFC 5549.

If the AMS-IX should decide to become a LIR to acquire a /21 IPv4 range it should act as soon as possible because as soon the RIPE NCC starts assigning its last /8 IPv4 range a new policy will start to take effect which will prevent even LIRs from acquiring a /21 IPv4 range.

If an enlargement of the current IPv4 range is possible then there is still the broadcast problem which can cripple a lot of routing devices if the ARP sponge should fail. It should also be noted that it is possible to double the current IPv4 range than this would only be a temporary solution to the AMS-IX's problem.

A better alternative to simply enlarging the IPv4 range would be for the AMS-IX's clients to forward their IPv4 traffic to IPv6 next hop address eliminating the need of IPv4 addresses on the AMS-IX Internet Peering VLAN.

This research document describes three different methods of how RFC 5549 can be implemented. These are:

• native inter-protocol forwarding
• a forwarding engine managed translation table
• a routing daemon managed translation table

The proof of concept created for my research shows that it is possible to create a route by only advertising IPv6 next hops as described in RFC 5549. The current proof of concept creates a translation table which is managed by the routing daemon itself and uses this to translate the IPv6 next hops advertised into IPv4 addresses which are only locally significant on a routing device.

Although the current proof of concept does not allow a traceroute to work, I have provided a theoretical solution to this problem by preemptively sending out a TTL expired message if the TTL of a packet equals 1 when we are about the send the packet to a "RFC 5549" next hop.

The original research question of this research report was: “How can new routing devices participate in the sharing of BGP IPv4 routes on the main AMS-IX Internet Peering VLAN
when the current IPv4 range is depleted”. We can now give the following answer to that question.

The AMS-IX should prepare for this problem by convincing its customers to ask their routing vendors to implement RFC 5549 in their routing devices so they may share IPv4 routes in the future with IPv6 next hop addresses making IPv4 obsolete on the AMS-IX’s Internet Peering VLAN, because this could last longer than 2016. The AMS-IX should also become a RIPE NCC LIR which will give it a larger IPv4 range. Because of that larger IPv4 range, the AMS-IX will have more time so that RFC 5549 can be implemented by all its clients.
8. Reflection

In the reflection I will reflect upon my research for the AMS-IX and use the set list of competence points as used by the Hogeschool van Amsterdam. I took the liberty to translate these competence points from Dutch into English.

The following competence points were selected by me at the start of this project:

1 Analyze

1.2 *I create functional specifications.*

When this project started I had to first analyze the problem itself. Did the problem exist in the first place? How long until the AMS-IX runs out of IPv4 addresses? After this initial problem analysis I could form a short list of functional specifications that the potential solution would need to have. Of course the main specification was that the solution would need to solve the problem of new routing devices wanting to share its IPv4 routes when there are no more IPv4 addresses to give out. In the end result of this project are included functional specifications of how to Implement RFC 5549.

2 Advise

2.1 *Using analysis and meetings with stakeholders I will create a supporting advise for the reconfiguration of processes and/or information streams and for the development or acquisition of a new ICT-system.*

This competence point was without a doubt successfully completed as this project was giving advice to the AMS-IX and the AMS-IX’s clients about the development of a new function in the MP-BGP routing protocol.

3 Design

3.1 *I design an ICT-system with an architectural description and specifications using an analysis.*

This competence was met by creating the first working implementation of RFC 5549. This was of course done after a careful analysis of the RFC itself.

5 Manage

5.2 *responsible for installation, testing, integrating, and usage of a new ICT-system.*

This competence sadly was not possible because during my project I found out that the solution described in this research report simply does not exist yet. I hope that because of my research future students can be responsible for the installation, testing, integration, and usage of RFC 5549.

This project was a bit different than other projects I have done for the HvA for several reasons. The first reason is that because the AMS-IX has no control over the routers configuration of its clients I could never actually change the network itself, I could only give advise for the future. The second reason is that at the beginning of this project it was assumed that I would have to test existing implementations of RFC 5549 to see if it would solve the IPv4 problem of the AMS-IX. In the end we found that no implementations existed.
yet and I had to create my own implementations. Although this made my project a lot more interesting for me, it did limit the scope of the final product. It wouldn't be possible to simply tell the clients to start using RFC 5549. RFC 5549 must first be implemented by the routing vendors. I hope the end result of this project is not the end result for the implementation of RFC 5549, but just the second step in its implementation in the real world.
Advertising IPv4 Network Layer Reachability Information with an IPv6 Next Hop

Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (c) 2009 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents in effect on the date of publication of this document (http://trustee.ietf.org/license-info). Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Abstract

Multiprotocol BGP (MP-BGP) specifies that the set of network-layer protocols to which the address carried in the Next Hop field may belong is determined by the Address Family Identifier (AFI) and the Subsequent Address Family Identifier (SAFI). The current AFI/SAFI definitions for the IPv4 address family only have provisions for advertising a Next Hop address that belongs to the IPv4 protocol when advertising IPv4 Network Layer Reachability Information (NLRI) or VPN-IPv4 NLRI. This document specifies the extensions necessary to allow advertising IPv4 NLRI or VPN-IPv4 NLRI with a Next Hop address that belongs to the IPv6 protocol. This comprises an extension of the AFI/SAFI definitions to allow the address of the Next Hop for IPv4 NLRI or VPN-IPv4 NLRI to also belong to the IPv6 protocol, the encoding of the Next Hop in order to determine which of the protocols the address actually belongs to, and a new BGP Capability allowing MP-BGP Peers to dynamically discover whether they can exchange IPv4 NLRI and VPN-IPv4 NLRI with an IPv6 Next Hop.
1. Introduction

Multiprotocol BGP (MP-BGP) [RFC4760] specifies that the set of network-layer protocols to which the address carried in the Next Hop field may belong is determined by the Address Family Identifier (AFI) and the Subsequent Address Family Identifier (SAFI). A number of existing AFI/SAFIs allow the Next Hop address to belong to a different address family than the Network Layer Reachability Information (NLRI). For example, the AFI/SAFI <25/65> used (as per [L2VPN-SIG]) in order to perform L2VPN auto-discovery, allows advertising NLRI that contains the identifier of a Virtual Private LAN Service (VPLS) instance or that identifies a particular pool of attachment circuits at a given Provider Edge (PE), while the Next Hop field contains the loopback address of a PE. Similarly, the AFI/SAFI <1/132> (defined in [RFC4684]) in order to advertise Route Target (RT) membership information, allows advertising NLRI that contains such RT membership information, while the Next Hop field contains the address of the advertising router.

Furthermore, a number of these existing AFI/SAFIs allow the Next Hop to belong to either the IPv4 Network Layer Protocol or the IPv6 Network Layer Protocol, and specify the encoding of the Next Hop information in order to determine which of the protocols the address actually belongs to. For example, [RFC4684] allows the Next Hop address to be either IPv4 or IPv6 and states that the Next Hop field address shall be interpreted as an IPv4 address whenever the length of Next Hop address is 4 octets, and as an IPv6 address whenever the length of the Next Hop address is 16 octets.

There are situations such as those described in [RFC4925] and in [MESH-FMWK] where carriers (or large enterprise networks acting as...
carrier for their internal resources) may be required to establish connectivity between 'islands' of networks of one address family type across a transit core of a differing address family type. This includes both the case of IPv6 islands across an IPv4 core and the case of IPv4 islands across an IPv6 core. Where Multiprotocol BGP (MP-BGP) is used to advertise the corresponding reachability information, this translates into the requirement for a BGP speaker to advertise Network Layer Reachability Information (NLRI) of a given address family via a Next Hop of a different address family (i.e., IPv6 NLRI with IPv4 Next Hop and IPv4 NLRI with IPv6 Next Hop).

The current AFI/SAFI definitions for the IPv6 address family assume that the Next Hop address belongs to the IPv6 address family type. Specifically, as per [RFC2545] and [RFC3107], when the <AFI/SAFI> is <2/1>, <2/2>, or <2/4>, the Next Hop address is assumed to be of IPv6 type. As per [RFC4659], when the <AFI/SAFI> is <2/128>, the Next Hop address is assumed to be of IPv6-VPN type.

However, [RFC4798] and [RFC4659] specify how an IPv4 address can be encoded inside the Next Hop IPv6 address field when IPv6 NLRI needs to be advertised with an IPv4 Next Hop. [RFC4798] defines how the IPv4-mapped IPv6 address format specified in the IPv6 addressing architecture ([RFC4291]) can be used for that purpose when the <AFI/SAFI> is <2/1>, <2/2>, or <2/4>. [RFC4659] defines how the IPv4-mapped IPv6 address format as well as a null Route Distinguisher can be used for that purpose when the <AFI/SAFI> is <2/128>. Thus, there are existing solutions for the advertisement of IPv6 NLRI with an IPv4 Next Hop.

Similarly, the current AFI/SAFI definitions for advertisement of IPv4 NLRI or VPN-IPv4 NLRI assume that the Next Hop address belongs to the IPv4 address family type. Specifically, as per [RFC4760] and [RFC3107], when the <AFI/SAFI> is <1/1>, <1/2>, or <1/4>, the Next Hop address is assumed to be of IPv4 type. As per [RFC4364], when the <AFI/SAFI> is <1/128>, the Next Hop address is assumed to be of VPN-IPv4 type. There is clearly no generally applicable method for encoding an IPv6 address inside the IPv4 address field of the Next Hop. Hence, there is currently no specified solution for advertising IPv4 or VPN-IPv4 NLRI with an IPv6 Next Hop.

This document specifies the extensions necessary to do so. This comprises an extension of the AFI/SAFI definitions to allow the address of the Next Hop for IPv4 NLRI or VPN-IPv4 NLRI to belong to either the IPv4 or the IPv6 protocol, the encoding of the Next Hop information in order to determine which of the protocols the address actually belongs to, and a new BGP Capability allowing MP-BGP peers to dynamically discover whether they can exchange IPv4 NLRI and VPN-IPv4 NLRI with an IPv6 Next Hop. The new BGP Capability allows...
gradual deployment of the new functionality of advertising IPv4 reachability via an IPv6 Next Hop, without any flag day nor any risk of traffic black-holing.

2. 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].

3. Extension of AFI/SAFI Definitions for the IPv4 Address Family

As mentioned earlier, MP-BGP specifies that the set of network-layer protocols to which the address carried in the Next Hop field may belong is determined by the Address Family Identifier (AFI) and the Subsequent Address Family Identifier (SAFI). The following current AFI/SAFI definitions for the IPv4 NLRI or VPN-IPv4 NLRI (<1/1>, <1/2>, <1/4>, and <1/128>) only have provisions for advertising a Next Hop address that belongs to the IPv4 protocol. This document extends the definition of the AFI/SAFI for advertisement of IPv4 NLRI and VPN-IPv4 NLRI to extend the set of network-layer protocols to which the Next Hop address can belong, to include IPv6 in addition to IPv4.

Specifically, this document allows advertising with [RFC4760] of an MP_REACH_NLRI with:

- AFI = 1
- SAFI = 1, 2, 4, or 128
- Length of Next Hop Address = 16 or 32
- Next Hop Address = IPv6 address of next hop (potentially followed by the link-local IPv6 address of the next hop). This field is to be constructed as per Section 3 of [RFC2545].
- NLRI= NLRI as per current AFI/SAFI definition

This is in addition to the current mode of operation allowing advertisement of NLRI for <AFI/SAFI> of <1/1>, <1/2> and <1/4> with a next hop address of IPv4 type and advertisement of NLRI for <AFI/SAFI> of <1/128> with a next hop address of VPN-IPv4 type.

The BGP speaker receiving the advertisement MUST use the Length of Next Hop Address field to determine which network-layer protocol the next hop address belongs to. When the Length of Next Hop Address field is equal to 16 or 32, the next hop address is of type IPv6.
Note that this method of using the Length of the Next Hop Address field to determine which network-layer protocol the next hop address belongs to (out of the set of protocols allowed by the AFI/SAFI definition) is the same as used in [RFC4684] and [L2VPN-SIG].

4. Use of BGP Capability Advertisement

[RFC5492] defines a mechanism to allow two BGP speakers to discover if a particular capability is supported by their BGP peer and thus whether it can be used with that peer. This document defines a new capability that can be advertised using [RFC5492] and that is referred to as the Extended Next Hop Encoding capability. This capability allows BGP speakers to discover whether, for a given NLRI <AFI/SAFI>, a peer supports advertisement with a next hop whose network protocol is determined by the value of the Length of Next Hop Address field, as specified in Section 3.

A BGP speaker that wishes to advertise to a BGP peer an IPv6 Next Hop for IPv4 NLRI or for VPN-IPv4 NLRI as per this specification MUST use the Capability Advertisement procedures defined in [RFC5492] with the Extended Next Hop Encoding Capability to establish whether its peer supports this for the NLRI AFI/SAFI pair(s) of interest. The fields in the Capabilities Optional Parameter MUST be set as follows:

- The Capability Code field MUST be set to 5 (which indicates the Extended Next Hop Encoding capability).
- The Capability Length field is set to a variable value that is the length of the Capability Value field (which follows).
- The Capability Value field has the following format:

```
+-----------------------------------------------------+
| NLRI AFI - 1 (2 octets)                             |
| NLRI SAFI - 1 (2 octets)                            |
+-----------------------------------------------------+
| Nexthop AFI - 1 (2 octets)                          |
+-----------------------------------------------------+
| .....                                               |
+-----------------------------------------------------+
| NLRI AFI - N (2 octets)                             |
| NLRI SAFI - N (2 octets)                            |
+-----------------------------------------------------+
| Nexthop AFI - N (2 octets)                          |
+-----------------------------------------------------+
```
where:

* each triple <NLRI AFI, NLRI SAFI, Nexthop AFI> indicates that NLRI of <NLRI AFI / NLRI SAFI> may be advertised with a Next Hop address belonging to the network-layer protocol of Nexthop AFI.

* the AFI and SAFI values are defined in the Address Family Identifier and Subsequent Address Family Identifier registries maintained by IANA.

Since this document only concerns itself with the advertisement of IPv4 NLRI and VPN-IPv4 NLRI with an IPv6 Next Hop, this specification only allows the following values in the Capability Value field of the Extended Next Hop Encoding capability:

- NLRI AFI = 1 (IPv4)
- NLRI SAFI = 1, 2, 4, or 128
- Nexthop AFI = 2 (IPv6)

This specification does not propose that the Extended Next Hop Encoding capability be used with any other combinations of <NLRI AFI, NLRI SAFI, Nexthop AFI>. In particular, this specification does not propose that the Extended Next Hop Encoding capability be used for NLRI AFI/SAFIs whose definition already allows use of both IPv4 and IPv6 next hops (e.g., AFI/SAFI = <1/132> as defined in [RFC4684]). Similarly, it does not propose that the Extended Next Hop Encoding capability be used for NLRI AFI/SAFIs for which there is already a solution for advertising a next hop of a different address family (e.g., AFI/SAFI = <2/1>, <2/2>, or <2/4> with IPv4 Next Hop as per [RFC4798] and AFI/SAFI = <2/128> with IPv4 Next Hop as per [RFC4659]).

It is expected that if new AFI/SAFIs are defined in the future, their definition will have provisions (where appropriate) for both IPv4 and IPv6 Next Hops from the onset, with determination based on Length of Next Hop Address field. Thus, new AFI/SAFIs are not expected to make use of the Extended Next Hop Encoding capability.

A BGP speaker MUST only advertise to a BGP peer the IPv4 or VPN-IPv4 NLRI with an IPv6 Next Hop if the BGP speaker has first ascertained via BGP Capability Advertisement that the BGP peer supports the Extended Next Hop Encoding capability for the relevant AFI/SAFI pair.

The Extended Next Hop Encoding capability provides information about next hop encoding for a given AFI/SAFI, assuming that AFI/SAFI is
allowed. It does not influence whether that AFI/SAFI is indeed allowed. Whether a AFI/SAFI can be used between the BGP peers is purely determined through the Multiprotocol Extensions capability defined in [RFC4760].

The Extended Next Hop Encoding capability MAY be dynamically updated through the use of the Dynamic Capability capability and associated mechanisms defined in [DYN-CAP].

5. Operations

By default, if a particular BGP session is running over IPvx (where IPvx is IPv4 or IPv6), and if the BGP speaker sending an update is putting its own address in as the next hop, then the next hop address SHOULD be specified as an IPvx address, using the encoding rules specified in the AFI/SAFI definition of the NLRI being updated. This default behavior may be overridden by policy.

When a next hop address needs to be passed along unchanged (e.g., as a Route Reflector (RR) would do), its encoding MUST NOT be changed. If a particular RR client cannot handle that encoding (as determined by the BGP Capability Advertisement), then the NLRI in question cannot be distributed to that client. For sound routing in certain scenarios, this will require that all the RR clients be able to handle whatever encodings any of them may generate.

6. Usage Examples

6.1. IPv4 over IPv6 Core

The extensions defined in this document may be used as discussed in [MESH-FMWK] for the interconnection of IPv4 islands over an IPv6 backbone. In this application, Address Family Border Routers (AFBRs; as defined in [RFC4925]) advertise IPv4 NLRI in the MP_REACH_NLRI along with an IPv6 Next Hop.

The MP_REACH_NLRI is encoded with:

- AFI = 1
- SAFI = 1
- Length of Next Hop Network Address = 16 (or 32)
- Network Address of Next Hop = IPv6 address of Next Hop
- NLRI = IPv4 routes
During BGP Capability Advertisement, the PE routers would include the following fields in the Capabilities Optional Parameter:

- Capability Code set to "Extended Next Hop Encoding"
- Capability Value containing <NLRI AFI=1, NLRI SAFI=1, Nexthop AFI=2>

6.2. IPv4 VPN over IPv6 Core

The extensions defined in this document may be used for support of IPv4 VPNs over an IPv6 backbone. In this application, PE routers would advertise VPN-IPv4 NLRI in the MP_REACH_NLRI along with an IPv6 Next Hop.

The MP_REACH_NLRI is encoded with:

- AFI = 1
- SAFI = 128
- Length of Next Hop Network Address = 16 (or 32)
- Network Address of Next Hop = IPv6 address of Next Hop
- NLRI = IPv4-VPN routes

During BGP Capability Advertisement, the PE routers would include the following fields in the Capabilities Optional Parameter:

- Capability Code set to "Extended Next Hop Encoding"
- Capability Value containing <NLRI AFI=1, NLRI SAFI=128, Nexthop AFI=2>

7. IANA Considerations

This document defines, in Section 4, a new Capability Code to indicate the Extended Next Hop Encoding capability in the [RFC5492] Capabilities Optional Parameter. The value for this new Capability Code is 5, which is in the range set aside for allocation using the "IETF Review" policy defined in [RFC5226].

8. Security Considerations

This document does not raise any additional security issues beyond those of BGP-4 and the Multiprotocol extensions for BGP-4. The same security mechanisms are applicable.
Although not expected to be the typical case, the IPv6 address used as the BGP Next Hop Address could be an IPv4-mapped IPv6 address (as defined in [RFC4291]). Configuration of the security mechanisms potentially deployed by the network operator (such as security checks on next hop address) need to keep this case in mind also.

9. Acknowledgments

The authors would like to thank Yakov Rekhter, Pranav Mehta, and John Scudder for their contributions to the approach defined in this document.

10. References

10.1. Normative References


10.2. Informative References


Research report
IPv4 shortage AMS-IX

RFC 5549 v4 NLRI with v6 NH May 2009


Authors' Addresses

Francois Le Faucheur
Cisco Systems
Greenside, 400 Avenue de Roumanille
Sophia Antipolis 06410
France

EMail: flefauch@cisco.com

Eric Rosen
Cisco Systems
1414 Massachusetts Avenue
Boxborough, MA 01719
USA

EMail: erosen@cisco.com
Appendix 2: Implementation guide
Implementation guide for RFC 5549
Amsterdam Internet Exchange

Project: IPv4 shortage AMS-IX
How to setup the RFC 5549 Quagga proof of concept

Author: Stefan Plug
E-mail: stefan.plug@ams-ix.net
Version: 1.0
State: Final
Date: 29-05-2012
## Version Management

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>25-05-12</td>
<td>Start of document.</td>
</tr>
<tr>
<td>0.2</td>
<td>25-05-12</td>
<td>Spelling check</td>
</tr>
<tr>
<td>1.0</td>
<td>29-05-12</td>
<td>Ready to press!</td>
</tr>
</tbody>
</table>
Index

1 Introduction .......................................................................................................................... 4
2 Setup .................................................................................................................................... 5
  2.1 Installation .................................................................................................................... 6
  2.2 troubleshooting tips ....................................................................................................... 11
1 Introduction

This document will show the reader how to setup AMS-IX's proof of concept for IETF RFC 5549. This document is part of the project: 'IPv4 shortage AMS-IX'. For more background information about RFC 5549 please read chapter 6.2 of the research report for this project which will be made publicly available when this project has been graded by the Hogeschool van Amsterdam at http://www.ams-ix.net/downloads/RFC5549/. If you have any questions or remarks regarding either this implementation guide or the research report please email me at stefan.plug@ams-ix.net.
2 Setup

The setup of the proof of concept is generally quite straight forward. I will assume the reader uses a clean installation of Ubuntu Linux 11.04 desktop-i386. although any other Linux distribution and version should work fine in theory I cannot guarantee this is actually the case.

Before we begin I would like to introduce you to the following network topology which I will use to setup my routing device.

![Network Topology](image35.png)

In this document I will describe how to setup Ubuntu-R03 as seen in the network topology.

In this topology Ubuntu-R03 acts as the default gateway for the Ubuntu-H03 machine and connects to a central broadcast domain to which its future BGP peers Ubuntu-R01 and Ubuntu-R02 are also connected. We can also see that all routing devices use the IPv4 address **200.0.0.1** on their eth0 interface connecting to the central broadcast domain. This is intentional. The IPv4 address will only be used to enable Linux to process and send IPv4 packets on that interface. The actual routes will be created using the IPv6 addresses on the eth0 interfaces.
2.1 Installation
We should first make sure that we have everything to start installing our proof of concept. Make sure you have the following software:
- Ubuntu Linux 11.04 desktop-i386
- Proof of concept Quagga source
- ARPupdate source
- A working Internet connection to install some extra packages required for installing Quagga

The proof of concept and arpupdate source codes are available at the following link: http://www.ams-ix.net/downloads/RFC5549/

Step 1: Install and prepare Ubuntu Linux
Perform a standard installation of Ubuntu Linux 11.04 desktop-i386.

Ensure you have a working Internet connection and install the gawk and dia packages needed for the installation of the Quagga proof of concept. issue the following apt-get command:
sudo apt-get install gawk
sudo apt-get install dia

We don't need any Internet connection anymore so we can configure our networking devices as seen in image 1.
eth0, IPv4: 200.0.0.1/16
eth0, IPv6: 2001::3/64
eth1, IPv4: 192.168.3.1/24

Step 2: Configure Linux to do IP forwarding
Before Linux can be used as a router it needs to be told to actually forward packets. By default this feature is disabled.

To check if Linux is already in a forwarding state we can read the contents of the ip_forward file. A 0 indicates IP forwarding is disabled, and a 1 indicates it is enabled.
cat /proc/sys/net/ipv4/ip_forward

To enable IP forwarding we will issue the following command:
sudo sysctl -w net.ipv4.ip_forward=1

Step 2: Prepare Quagga
Before we can install the proof of concept we need to create a user who will run Quagga for us.
To create the quagga user issue the following command and go through the user creation wizard.

```
sudo adduser quagga
```

Next we want to create a folder where the quagga user can find the configuration files for Quagga. The quagga user needs to have the correct permissions to access the configuration files so we will also make the quagga user the owner of this new folder.

```
sudo mkdir /usr/local/quagga
sudo chown quagga:quagga /usr/local/quagga
```

**Step 3: Install Quagga**

First we must unpack the quagga .tar file we downloaded from the AMS-IX website.

```
sudo tar -xvzf quagga-0.99.20-RFC5549-0.1.tar.gz
```

Now we need to run the configure script with 2 extra options pointing the folder we have just created for the quagga user. Go to the main Quagga directory and issue the following command.

```
sudo ./configure --sysconfdir=/usr/local/quagga --localstatedir=/usr/local/quagga
```

When the configuration script has run we should make and make install Quagga.

```
sudo make
sudo make install
```

Finally we will want to update our shared libraries by issuing the following command.

```
sudo ldconfig
```

**Step 4: Configure zebra and bgpd**

Now Quagga is installed we will have to create some basic configurations for zebra and bgpd. Luckily Quagga provides some sample configuration files which we can use as templates. The configuration files are located in the folder we specified in the configuration script, /usr/local/quagga/ in our case. If you didn't specify a specific folder in the configuration script then the configuration files will probably be located in /usr/local/etc/

```
sudo cp /usr/local/quagga/zebra.conf.sample /usr/local/quagga/zebra.conf
sudo cp /usr/local/quagga/bgpd.conf.sample /usr/local/quagga/bgpd.conf
```

The zebra configuration is now good to go, but we will still have to edit the bgpd configuration file to use RFC 5549 forwarding. Use your favorite text editor and edit the bgpd.conf as described on the next page. Please note that any line beginning with a '!' is a comment and could be omitted.
hostname bgpd
password zebra
!enable password please-set-at-here
!
!bgp multiple-instance
!
! General IPv4 block.
! we could add traditional IPv4 neighbors and advertise IPv4 networks here if we
! wanted to. We put this router in AS3 and create IPv6 neighbors but don't activate
! them yet we will activate them in the IPv6 block.

router bgp 3
    bgp router-id 3.3.3.3
    neighbor 2001::1 remote-as 1
    neighbor 2001::2 remote-as 2
    no neighbor 2001::1 activate
    no neighbor 2001::2 activate

! IPv6 block.
! Activate our neighbors, use the manual next-hop route-maps, and add
! the IPv4 network we would like to advertise via our IPv6 next hop.

address-family ipv6
    neighbor 2001::1 activate
    neighbor 2001::2 activate
    neighbor 2001::1 route-map set-nexthop out
    neighbor 2001::2 route-map set-nexthop out
    network 192.168.3.0/24
exit-address-family

! for RFC 5549 forwarding we need to manually configure the next hops our
! neighbors should use to reach the advertised destination. use this routers own
! IPv6 address(es) here.
route-map set-nexthop permit 10
    set ipv6 next-hop global 2001::3
    set ipv6 next-hop local fe80::a00:27ff:fe44:d6a5

! log stdout
The previous bgpd.conf has 5 ‘blocks’ of code.

Block 1: Basic bgpd information.
\[
\begin{align*}
\text{hostname} & \quad \text{bgpd} \\
\text{password} & \quad \text{zebra}
\end{align*}
\]

Block 2: BGP initialization and optional IPv4 neighbor relationships.
\[
\begin{align*}
\text{router bgp} & \quad 3 \\
\text{bgp router-id} & \quad 3.3.3.3 \\
\text{neighbor} & \quad 2001::1 \text{ remote-as} 1 \\
\text{neighbor} & \quad 2001::2 \text{ remote-as} 2 \\
\text{no neighbor} & \quad 2001::1 \text{ activate} \\
\text{no neighbor} & \quad 2001::2 \text{ activate}
\end{align*}
\]

Block 3: IPv6 configuration
\[
\begin{align*}
\text{address-family ipv6} \\
\text{neighbor} & \quad 2001::1 \text{ activate} \\
\text{neighbor} & \quad 2001::2 \text{ activate} \\
\text{neighbor} & \quad 2001::1 \text{ route-map set-nexthop out} \\
\text{neighbor} & \quad 2001::2 \text{ route-map set-nexthop out} \\
\text{network} & \quad 192.168.3.0/24 \\
\text{exit-address-family}
\end{align*}
\]

Block 4: route-map set-nexthop
\[
\begin{align*}
\text{route-map set-nexthop permit 10} \\
\text{set ipv6 next-hop global} & \quad 2001::3 \\
\text{set ipv6 next-hop local} & \quad \text{fe80::a00:27ff:fe44:d6a5}
\end{align*}
\]

Block 5: logging
\[
\text{log stdout}
\]

Step 5: Create ip4to6 table
Now we have to manually create the IPv4 to IPv6 translation table. The translation table luckily is nothing more than a simple file in the /usr/local/sbin/ folder.
\[
\begin{align*}
\text{sudo touch /usr/local/sbin/ip4to6} \\
\text{sudo chmod 777 /usr/local/sbin/ip4to6}
\end{align*}
\]

Step 6: Prepare the arpupdate code
The current arpupdate program is a very simple and quite static C program which uses the Linux NETLINK interface to update the ARP table. arpupdate looks into the /usr/local/sbin/ip4to6 translation table file, gets an <IPv6, IPv4> combination and then searches the IPv6 ND table for a MAC address and injects this MAC address into the IPv4 ARP table with the correct 'bogus' IPv4 address.

In the arpupdate.c source code we will probably need to change line 72 in the injectmacipv4() function.
\[
\text{req2.r.ndm_ifindex = 2; // /sys/class/net/eth(X)/ifindex}
\]

Research report v1.0
Project: IPv4 shortage AMS-IX
AMS-IX 78
This line statically assigns the outgoing interface index number for the ARP update. This
should be the interface connected to the central broadcast domain.

If you are using the same network topology as in this document than this should be interface
eth0. To find the index number for your interface you can look in the
/sys/class/net/eth(X)/ifindex file, where eth(X) is the interface you would like to know the
ifindex number for.

In our case we have the following result

```
cat /sys/class/net/eth0/ifindex
2
```

so in my arpupdate.c source code I will have the following line 72

```
req2.r.ndm_ifindex = 2;
```

After the change we have to recompile the source code. You can use your own favorite C
compiler to do this, but because I use gcc for this I will issue the following command in the
arpupdate source folder to compile our arpudater.

```
gcc arpupdate.c /o arpupdate
```

If you would like to update the source code to be more flexible than please do not hesitate to
rewrite the source code. I could imagine that we could also get the outgoing interface from
the ND table when we search for the MAC address.

**Step 7: Run Quagga**

Your computer should now be ready to advertise and forward IPv4 routes with an IPv6 next
hop address.

Fist we should run the zebra deamon which is Quagga's interface between BGPD and Linux.

```
sudo /usr/local/sbin/zebra &
```

Next the BGPD deamon itself.

```
sudo /usr/local/sbin/bgpd &
```

And finally the arpudater.

```
sudo (arpupdate folder)/arpupdate &
```

There is a bug in the current arpudater that will return the following error:

**RFC 5549: Problem receiving from netlink**

I have found that when I try to run the arpudater again after a minute or so then sometimes
the arpudater just works. I have no idea why Netlink sometimes fails as of yet.
2.2 Troubleshooting tips

I cannot forward any packets?
Remember that Linux does not forward any packets by default! See step 2 on how to check if your Linux installation has forwarding enabled and how you can enable this.

Problems during ./configure of Quagga:
Make sure the gawk package is installed.
```
sudo apt-get install gawk
```

Problems during install:
```
sudo apt-get install
'error while loading shared libraries: libzebra.so.0: cannot open shared object file: No such file or directory'
```

The shared libraries might not have been updated yet, to update them issue the following command:
```
sudo ldconfig
```
Sources