ERNW WHITEPAPER 57

IPV6 SOURCE ADDRESS SELECTION

Version:	1.0
Date:	10/26/2016
Classification:	Public

Author(s):	Jan-Pascal Kwiotek
	Christopher Werny
	Omar Eissa
	Christian Tanck

Nichole Vonderwell

Digitally signed by Nichole Vonderwell
Date: 2016.10.26 09:52:05 +02'00'

ERNW
providing security.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

After several IPv4 address exhaustion milestones we passed the last years, it is clear that there is a need to adopt to the next generation internet protocol IPv6. An IPv6 enabled host can have multiple addresses and the underlying paradigm of address configuration has changed. To select a source address in IPv6, the operating system implements a source address selection mechanism that compares multiple source address candidates and selects the best candidate. Criteria for this selection are defined in RFC6724. In this paper, we analyze the behavior of the IPv6 source address selection mechanism of different operating systems and check if they work compliant to RFC6724. As the source address selection might differ in different networks with distinct configurations, we set up multiple scenarios with different network configurations for our experiments. Our lab setup consists of state-of-the-art operating system at the latest patch level.

1 Introduction

With the ever increasing number of users and smart devices, the IPv4 address exhaustion became a major issue over the last few years. Thus, a growing number of networks around the world is already IPv6-enabled (IPv6 Enabled Networks, 2016). In IPv6, hosts may have several IPv6 addresses configured on a network interface. This intended property of IPv6 leads to a problem regarding the decision which source address a host or application shall use when establishing a connection. To address this problem a source address selection algorithm was initially specified in RFC 3484 (R. Draves, Microsoft Research, 2003) and later obsoleted by RFC 6724 (D. Thaler, 2012). To select the best source address - if an application on the host does not select it – the operating system implements the mentioned source address selection mechanism.

1.1 Motivation

All modern computer systems may have multiple interfaces as well as several addresses in the IPv6 world. There are different possibilities for IPv6 addresses that play a role when we talk about source address selection. Every major IPv6 protocol stack includes *source IPv6 address selection* functionality specified in RFC6724. These rules select source IPv6 address of an outgoing TCP or UDP packet in instances when the application doesn't specify it. First of all, every node has a link-local address but additionally there can be Global Unicast Addresses (GUAs) (Hinden R. e., 2003) which can be either statically configured, obtained via DHCPv6 or IPv6 Stateless Address Autoconfiguration (SLAAC) (Thomson, 2007). Furthermore, there might be a Unique Local IPv6 Unicast Address (ULA) (Hinden R. H., 2005) in place.

Generally, the network devices can only hint the hosts which networks are available but the decision for the source address selection is always up to the host operating system. Although the mechanisms for address selection is specified in the RFC, different operating systems show different behavior. Some operating systems like Linux tend to choose the first available mechanism. Other operating systems use multiple different mechanisms at a time. Windows for example asks for a DHCP address if the M-flag is set regardless if there are already assigned addresses from a static configuration or SLAAC (Pepelnjak, 2016). If systems have GUA and ULA addresses and they are both in the DNS system, most operating systems would prefer a GUA address. In an ideal world, every operating system should follow the same rules for source selection by implementing RFC 6724. From our experience, that is the expectation many have in regards to the source address selection. Our goal is to describe and quantify to what extend the specification is followed.

1.2 Contribution

In this study, we examine the behavior of the IPv6 source address selection mechanism of the state-of-the-art operating systems on a broader base as previous research. We further aim at mimicking real word setups with our lab experiments. The results of the experiments give an indication which operating systems implement the RFC correctly and select the source address as expected. This information is essential for real world network setups, as for example firewalls typically filter traffic based on white- or blacklisting of source addresses of different hosts. If the hosts use an unexpected source address, this might become a problem for large networks.

1.3 Related Work

In general, the IPv6 protocol is defined in the main specification in RFC2460 (S. Deering R. H., 1998) and the algorithm for source address selection in RFC6724 (D. Thaler, 2012). The paper (Sander Degen, 2014) of Sander Degen et. Al discusses various general security issues of IPv6 implementations.

To the best of our knowledge, there is not much research about this IPv6 source address selection mechanisms yet. Different blogs on the internet discuss some aspects of the source address selection (Auer, IPv6 Source Address Selection – what, why, how, 2012) and how to control it under different operating systems (Auer, Controlling IPv6 source address selection, 2012). Additionally, in the article (Schaub, 2014) of Markus Schaub the source address selection mechanism is explained in detail.

1.4 Overview

The remainder of this paper is structured as follows: In Section 2, we revisit the RFC standard and explain the relevant details of the IPv6 source address selection algorithm. In Section 3, we describe our test setup, as well as the test operating systems and present the results of our tests in Section 4. In Section 5, we give a summary of our paper and discuss some further work as well as the limitations of our experiments and finally conclude our work.

2 Prerequisites / Revisiting RFC

There are multiple ways how the source address can be selected in IPv6. First of all, applications can always choose the source address and override operating system mechanisms for source address selection. Typically, various server software makes use of this and defines an outgoing source address for the packets. An example for this is the bind daemon for name resolution, which always responds to DNS queries with the source address that was used for the corresponding incoming packet. It should be noted that this behavior might not apply for all applications.

If applications or other mechanisms do not set the IPv6 source address, the source address selection algorithm of the operating system decides which source address is used. Therefore, all the addresses that are available to the operating system are sorted into a list in the first step, including IPv4 addresses. On this list, up to 8 different rules are applied that reorder the sequence of the list. This algorithm always compares two elements (A, B) of the list and put either A or B on top. When the algorithm terminates, the topmost element is returned and used as source address for an outgoing packet. The rules will be processed sequentially with a tie breaker rule (Rule 8) at the end of the list.

These are the rules for sorting the source addresses according to RFC 6724 that are applied to the source address candidates:

- Rule 1: Prefer same address.
- Rule 2: Prefer appropriate scope.
- Rule 3: Avoid deprecated addresses
- Rule 4: Prefer home addresses.
- Rule 5: Prefer outgoing interface.
- Rule 6: Prefer matching label.
- Rule 7: Prefer temporary addresses.
- Rule 8: Use longest matching prefix.

To understand those rules, more detail for each is given in the following sections.

2.1 Rule 1: Prefer Same Address

The first rule says that if one of the candidate (source) addresses actually *is* the destination address, this one is preferred over all other candidates. This rule is especially for the case when the host is using connections to itself. This is equal to using a connection to *localhost* in the IPv4 world.

2.2 Rule 2: Prefer Appropriate Scope

The second rule should prefer a similar scope of addresses. This means for example, if we have a global destination address, the global source address is preferred or if we have a link-local destination address, the link-local source address is preferred over another type of addresses. If we have candidates of the same scope, this is decided within another rule.

2.3 Rule 3: Avoid Deprecated Addresses

If we have two addresses (a non-deprecated one and a deprecated one[1]), the non-deprecated will always be preferred. An example of a deprecated address (prefix) is the site-local prefix fec0::/10 which specifies that the address is valid only within the site network of an organization. It was part of the original addressing architecture in December 1995, but its use was deprecated in September 2004 (C. Huitema, 2004).

2.4 Rule 4: Prefer Home Addresses

The fourth rule is preferring home addresses over other addresses according to the IPv6 mobile standard, which is defined in RFC 6275 (C. Perkins, 2011). This rule only has relevance if the host is using mobility support in IPv6. Although this is standardized in the RFC, it is used very rarely since today.

2.5 Rule 5: Prefer Outgoing Interface

The fifth rule is preferring addresses if they are on the interface that the packet will go out from over addresses of other interfaces. This basically means that if we already decided over which interface we send a packet, the addresses of this interface is preferred over addresses from other interfaces.

[1] *Deprecated is defined here in terms of RFC 4862 (Thomson, 2007).*

2.5.1 Rule 5.5: Prefer Addresses in a Prefix Advertised by the Next-Hop

This special rule is preferring a source address from the PIO of the next-hop router. If source address "A" is assigned by the PIO of the next-hop that will be used to reach Destination "D", and source address "B" is assigned by the PIO of another next-hop, source address "A" will be preferred.

2.6 Rule 6: Prefer Matching Label

The sixth rule is comparing the matching label of an address which means that it can distinguish between the IPv4 and the IPv6 addresses. Additionally, it is looking up the prefix and label in a table that is pairing them together. This is also done for each possible source address and if they match with the label of the destination address they are preferred in the sorted table.

2.7 Rule 7: Prefer Temporary Addresses

Rule seven is preferring addresses that are configured according to the privacy extensions feature in IPv6, which is defined in RFC 4941 (T. Narten, 2007). They are preferred because if a host has configured privacy addresses it is very likely that they should be used, which is also the understanding of the RFC.

2.8 Rule 8: Use Longest Matching Prefix

Rule eight is applying a longest prefix matching on the candidate source addresses and compares them with the destination address. Again here are the most of contiguous leading bits compared to a destination address preferred over an address with less contiguous leading bits.

Those eight rules build the core of the IPv6 source address selection mechanism. Ideally, every operating system would implement this exactly. As a result, we would expect the same behavior of every operating system regarding the source address selection. Unfortunately, in the past we have seen that many operating systems tend to have a different behavior, which is unpredictable regarding only the rules of the RFC. We aim at analyzing this problem in Section 3 and give some examples for the correct implementation and which result we expect in the following section.

2.9 Expected Behavior Examples

To clarify those eight rules here are some examples that could be seen in many IPv6 deployments. The source address selection rules from RFC6724 (R. Draves, Microsoft Research, 2003), in conjunction with the default policy table, produce the following behavior

2.9.1 Example 1

We have the destination *2001:db8:1::1* and the Candidate Source Addresses: *2001:db8:3::1* or *fe80::1.*

The resulting choice would be:

- *2001:db8:3::1* because of Rule 2 (prefer appropriate scope).

This address is selected because the scope of candidate fe80::1 does not match 2001:db8:1::1 so it is sorted below 2001:db8:3::1. The scope of an IPv6 address can have 8 different values that are defined in RFC4007, in this case 2001:db8:1::1 and 2001:db8:3::3 have the scope global and fe80::1 has the scope link-local (S. Deering B. H., 2005).

2.9.2 Example 2

We have the destination *2001:db8:1::1* and the Candidate Source Addresses: *2001:db8:1::2* or *2001:db8:3::2*. The resulting choice would be:

- *2001:db8:1::2* because of Rule 8 (longest matching prefix).

As we can see the prefix of the destination *2001:db8:1::1* shares the first 48 bit with *2001:db8:1::2*, so this source address is preferred over *2001:db8:3::2*, which shares fewer bits with the destination address.

2.9.3 Example 3

We have the destination *2001:db8:1:d5e3:0:0:1* and the Candidate Source Addresses: *2001:db8:1::2* (public) or *2001:db8:1::d5e3:7953:13eb:22e8* (temporary). The resulting choice would be:

- *2001:db8:1:d5e3:7953:13eb:22e8* (temporary)

because of Rule 7 (prefer temporary address). The temporary gets preferred over the other non-temporary address and sorted to the top of the resulting list.

2.10 RFC 6724 Source Address Selection Flowchart

The following RFC 6724 flowchart made by Edward Horley[2] outlines the rules explained above:

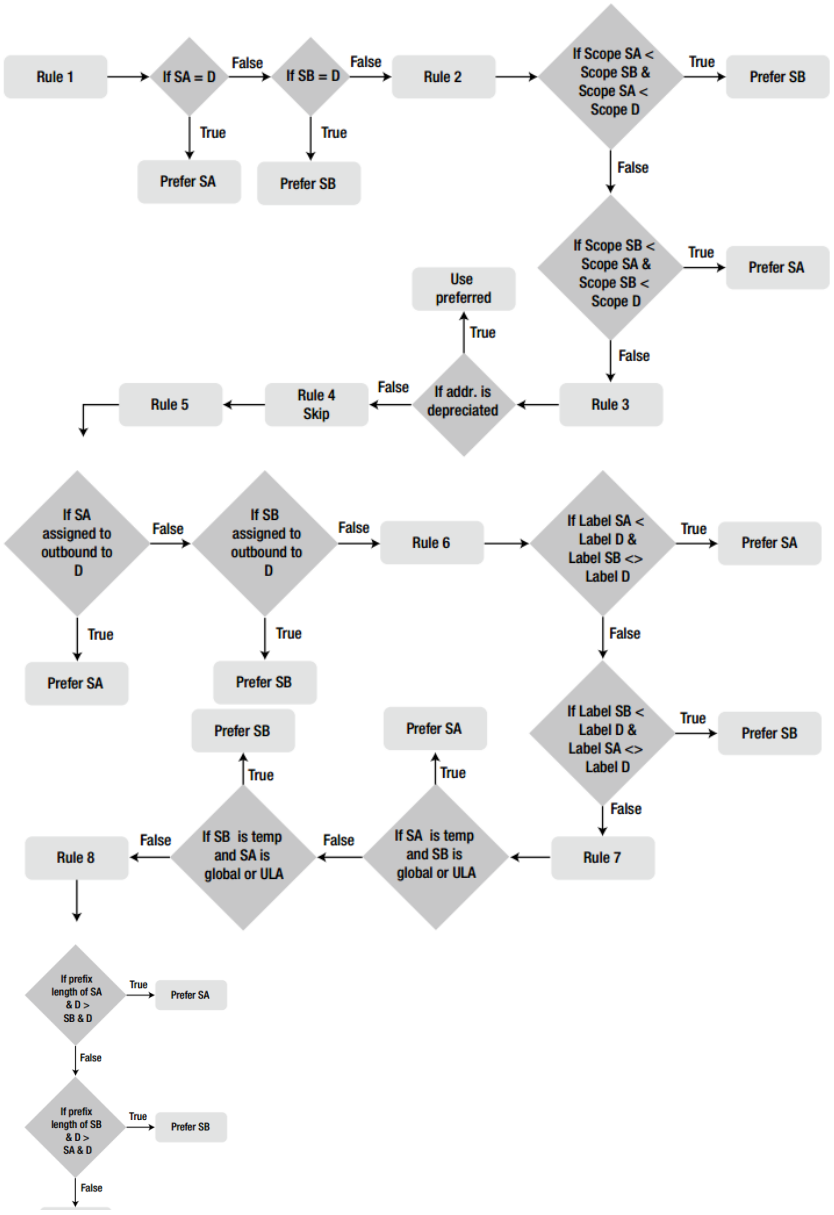

Figure 1: RFC 6724 Flowchart

[2] *From "Practical IPv6 for Windows Administrators" book – ISBN-10: 1430263709. Many thanks to Ed for the permission to use the flowchart!*

3 Study

In this section, we describe the test setup with the different tested operating systems in our experiment. We further describe the test cases that we use to test the source address selection algorithm of the systems and show the results of those tests.

3.1 Test Setup

For testing the source address selection in IPv6 for different operating systems we decided to build a small lab setup which guarantees an identical environment for each test. The basic setup consists of a webserver and clients that connect to the server and are connected either in the same subnet or a routed subnet. The IPv6 information are distributed via Router Advertisements to the clients which perform SLAAC. Unix and Linux based systems can configure the DNS settings via a Router Advertisement Option for DNS (J. Jeong, 2010). For the Windows based operating systems additionally the o-flag is set with a stateless DHCP server only for distributing the DNS server address.

The basic settings on the router that are not changed during the tests are the following:

- `ipv6 nd reachable-time 500`
- `ipv6 nd other-config-flag`
- `ipv6 nd ra lifetime 60`
- `ipv6 nd ra interval 10`
- `ipv6 nd ra dns server 2001:DB8:0:1::2`
- `ipv6 dhcp server test`

The network setup is shown in Figure 2. The main part of the experiment is the Cisco router that provides the Router Advertisements to our network and gets reconfigured for each test. In each test case the systems either operate in “routed” mode (left) or they are placed in the same subnet as the webserver. The webserver serves a static http web page, which shows the actual IPv6 address of the client. This server answers also the DNS requests from the clients and gets configured as DNS server on them via Router Advertisements or a stateless DHCP. The mobile devices are connected via WiFi over an Access Point that is bridged into the actual network segment of the tests, which has enables no services or IP addresses that could disturb our setup.

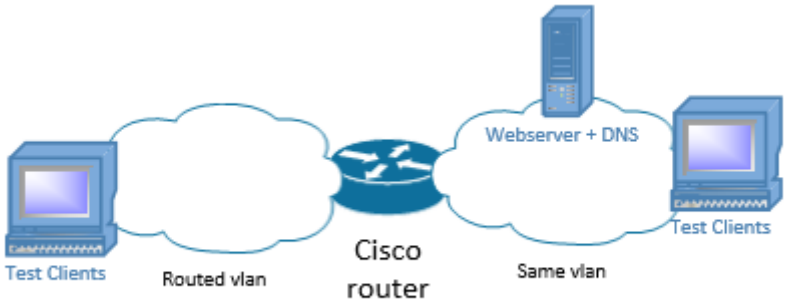

Figure 2: Lab Setup

In scope are the latest versions of the most popular operating systems as well as some selected mobile devices. The following devices/operating systems are in scope of the test:

Operating System / Device	Version
Windows 7 Pro / VM	SP1, update 27.06.2016
Windows 8.1 Pro / VM	update 27.06.2016
Windows 10 Pro / VM	update 27.06.2016
Ubuntu 16.4 LTS / VM	Kernel 4.4.0-21-generic
Debian 8 / VM	Kernel 3.16.0-4-amd64
iPhone 6s	IOS 9.3.2
Windows 10 Mobile / Lumia 550	update 25.06.2016
Google Nexus 9	Android Marshmallow 6.0.1
Apple MacBook Pro	OSX 10.11.5 (El Capitan)

In this setup, a webserver is in place which serves a website via HTTP and is responsible for DNS resolution of the clients. It has a static IPv6 GUA address and gateway configured and listens on all available IPv6 addresses. In the DNS record is for most of the tests only the GUA address of the webserver available. For the first part of the experiments the clients are in the same subnet as the webserver and for the second part they are in a routed subnet and reach the services using their default gateway. All operating systems except MAC OSX and the smartphone operating systems are in a virtualized environment that has a bridged network interface to the router. The software used during the experiments is the following:

- Webserver Debian 8 (2001:db8:0:1::2/64)
- Apache2 2.4.10
- Bind9 9.9.5
- Cisco 1921 router (15.4(3) M5)

3.2 Methodology

Each test is done individually with only one test operating system at a time. Each operating system is rebooted before every test case to ensure a fresh state. After this, each experiment is done once while recoding a PCAP. Each test scenario is repeated five times to verify the correct behavior. Additionally, the lifetime for the IPv6 addresses is set to a low value on the router to prevent caching of this information by the operating system. Each operating system is updated to the latest available updates before the initial test. To prevent influence from potential other systems, this lab setup has no connection to any other hosts or devices.

For each test the prepared website on the webserver is opened via the default web browser on the operating system. This is done twice, in the first step with the DNS record used and afterwards with only the IPv6 address directly. For each test the network traffic was captured to further analyze the behavior on the network stack. Additionally, the network interface configuration is observed to ensure the correct IPv6 addresses are set.

We took into account different scenarios of network configurations that might occur in large IPv6 deployments. They are described in the following sections.

3.3 Test Scenario 1

The clients are on the **same subnet** as the webserver and get only a link-local IPv6 address, the GUA prefix is not advertised via Router Advertisements but configured on the webserver as well as on the router.

Relevant router configuration of the client subnet:

- `ipv6 address FE80::1 link-local`
- `ipv6 address 2001:DB8:0:1::1/64`
- `2001:db8:0:1: :/64 no-advertise`

3.4 Test Scenario 2

The clients are on the **same subnet** as the webserver and get a GUA IPv6 address which is advertised via Router Advertisements.

Relevant router configuration of the client subnet:

- `ipv6 address 2001:DB8:0:1::1/64`
- `ipv6 nd prefix 2001:db8:0:1: :/64`

3.5 Test Scenario 3

The clients are on the **same subnet** as the webserver and get a GUA and a ULA address via Router Advertisements. The webserver is still statically configured to the GUA address and the DNS serves only the GUA address of the webserver.

Relevant router configuration of the client subnet:

- `ipv6 address 2001:DB8:0:1::1/64`
- `ipv6 address FC00:DEAD:BEEF:CAFE::1/64`
- `ipv6 nd prefix 2001:DB8:0:1::/64`
- `ipv6 nd prefix FC00:DEAD:BEEF:CAFE::/64`

3.6 Test Scenario 4

The clients are on the **same subnet** as the webserver and get a ULA address via Router Advertisements, while the GUA prefix is not advertised but present on the router. The webserver is still statically configured to the GUA address and the DNS serves only the GUA address of the webserver.

Relevant router configuration of the client subnet:

- `ipv6 address 2001:DB8:0:1::1/64`
- `ipv6 address FC00:DEAD:BEEF:CAFE::1/64`
- `ipv6 nd prefix 2001:DB8:0:1::/64 no-advertise`
- `ipv6 nd prefix FC00:DEAD:BEEF:CAFE::/64`

3.7 Test Scenario 5

The clients are on the **same subnet** as the webserver and get a GUA and a ULA address via Router Advertisements. The webserver is still statically configured to the GUA address but has additionally an ULA address configured. The DNS server holds in this case both addresses of the webserver as AAAA record and serves them in round robin mode.

Relevant router configuration of the client subnet:

- `ipv6 address 2001:DB8:0:1::1/64`
- `ipv6 address FC00:DEAD:BEEF:CAFE::1/64`
- `ipv6 nd prefix 2001:DB8:0:1::/64`
- `ipv6 nd prefix FC00:DEAD:BEEF:CAFE::/64`

3.8 Test Scenario 6

The clients are on the **same subnet** as the webserver and get two GUA addresses from different prefixes via Router Advertisements. The webserver is statically configured to the GUA address and the DNS server serves only the GUA address of the webserver.

Relevant router configuration of the client subnet:

- `ipv6 address 2001:DB7:0:1::1/64`
- `ipv6 address 2001:DB8:0:1::1/64`
- `ipv6 nd prefix 2001:DB7:0:1::/64`
- `ipv6 nd prefix 2001:DB8:0:1::/64`

3.9 Test Scenario 7

The clients are on a **different subnet** as the webserver and get a GUA address via Router Advertisements. The webserver is statically configured to the GUA address in the first subnet and the DNS serves only the GUA address of the webserver.

Relevant router configuration of the client subnet:

- `ipv6 address 2001:DB9:0:1::1/64`
- `ipv6 nd prefix 2001:DB9:0:1::/64`

3.10 Test Scenario 8

The clients are on a **different subnet** as the webserver and get a GUA as well as a ULA address via Router Advertisements. The webserver is statically configured to the GUA address in the first subnet but has an additional ULA address. The DNS server serves the GUA as well as the ULA address of the webserver.

Relevant router configuration of the client subnet:

- `ipv6 address 2001:DB9:0:1::1/64`
- `ipv6 address FC00:DEAD:BEEF:CCCC::1/64`
- `ipv6 nd prefix FC00:DEAD:BEEF:CCCC::/64`
- `ipv6 nd prefix 2001:DB9:0:1::/64`

3.11 Test Scenario 9

The clients are on a **different subnet** as the webserver and get two GUA addresses from different prefixes via Router Advertisements. The webserver is statically configured to the GUA address in the first subnet and the DNS server serves only the GUA address of the webserver.

Relevant router configuration of the client subnet:

- `ipv6 address 2001:DB9:0:1::1/64`
- `ipv6 address 2001:DB10:0:1::1/64`
- `ipv6 nd prefix 2001:DB9:0:1::/64`
- `ipv6 nd prefix 2001:DB10:0:1::/64`

3.12 Summary

The following table summarizes the configured IPv6 addresses on the clients for each test case:

Test case	Client IPv6 Address(es)	Server IPv6 Address(es)
1	LL-only	2001:db8:0:1::2
2	LL & GUA	2001:db8:0:1::2
3	LL & GUA & ULA	2001:db8:0:1::2
4	LL & ULA	2001:db8:0:1::2
5	LL & GUA & ULA	2001:db8:0:1::2 & fc00:dead:beaf:cafe::2
6	LL & 2 x GUA (different PIO)	2001:db8:0:1::2
7	LL & GUA	2001:db8:0:1::2
8	LL & GUA & ULA	2001:db8:0:1::2 & fc00:dead:beaf:cafe::2
9	LL & 2 x GUA (different PIO)	2001:db8:0:1::2

4 Results

The result section is split into two parts. The first part consists of the expected results under the assumption that all tested operating systems behave strictly according to the rules specified in RFC 6724. The second part consists of the results we observed during our evaluation in the lab environment.

4.1 Expected Results

4.1.1 Test Scenario 1

The clients are on the **same subnet** as the webserver and get only a link-local IPv6 address, the GUA prefix is not advertised via Router Advertisements but configured on the webserver as well as on the router.

4.1.1.1 Expected Result

Given that the client systems do only have Link Local IPv6 addresses, the source address selection algorithm might not apply in this case. Still, going through the rules of RFC 6724 the applicable rule would likely be **Rule 5: Prefer Outgoing Interface**

4.1.2 Test Scenario 2

The clients are on the **same subnet** as the webserver and get a GUA IPv6 address which is advertised via Router Advertisements.

Relevant router configuration of the client subnet:

- `ipv6 address 2001:DB8:0:1::1/64`
- `ipv6 nd prefix 2001:db8:0:1::/64`

4.1.2.1 Expected Result

In this scenario, the clients do have Global Unicast Addresses in addition to the Link Local Address. As the clients implement Privacy Extensions, the applicable rule as of RFC 6724 would be **Rule 7: Prefer Temporary Addresses.**

4.1.3 Test Scenario 3

The clients are on the **same subnet** as the webserver and get a GUA and a ULA address via Router Advertisements. The webserver is still statically configured to the GUA address and the DNS serves only the GUA address of the webserver.

4.1.3.1 Expected Result

In this scenario, the clients do have Unique Local Unicast and Global Unicast Addresses in addition to the Link Local Address. As the clients implement Privacy Extensions and the web server is reachable via the Global Unicast Address, the applicable rule as of RFC 6724 would be **Rule 7: Prefer Temporary Addresses.**

4.1.4 Test Scenario 4

The clients are on the **same subnet** as the webserver and get a ULA address via Router Advertisements, while the GUA prefix is not advertised but present on the router. The webserver is still statically configured to the GUA address and the DNS serves only the GUA address of the webserver.

4.1.4.1 Expected Result

In this scenario, the clients do have Unique Local Unicast Addresses in addition to the Link Local Address. As the clients implement Privacy Extensions and the web server is reachable via the Global Unicast Address, the applicable rule as of RFC 6724 would be **Rule 7: Prefer Temporary Addresses.**

4.1.5 Test Scenario 5

The clients are on the **same subnet** as the webserver and get a GUA and a ULA address via Router Advertisements. The webserver is still statically configured to the GUA address but has additionally an ULA address configured. The DNS server holds in this case both addresses of the webserver as AAAA record and serves them in round robin mode.

4.1.5.1 Expected Result

In this scenario, the clients do have Unique Local Unicast Addresses in addition to the Link Local Address. As the clients implement Privacy Extensions and the web server is reachable via both addresses, the applicable rules as of RFC 6724 would be **Rule 7: Prefer Temporary Addresses & Rule 8: Use Longest Matching Prefix.**

4.1.6 Test Scenario 6

The clients are on the **same subnet** as the webserver and get two GUA addresses from different prefixes via Router Advertisements. The webserver is statically configured to the GUA address and the DNS server serves only the GUA address of the webserver.

4.1.6.1 Expected Result

In this scenario, the clients do have Global Unicast Addresses (2001:db8 and 2001:db7) from different prefixes in addition to the Link Local Address. As the clients implement Privacy Extensions and the web server is reachable via the Global Unicast Address 2001:db8, the applicable rules as of RFC 6724 would be **Rule 7: Prefer Temporary Addresses & Rule 8: Use Longest Matching Prefix.**

4.1.7 Test Scenario 7

The clients are on a **different subnet** as the webserver and get a GUA address via Router Advertisements. The webserver is statically configured to the GUA address in the first subnet and the DNS serves only the GUA address of the webserver.

4.1.7.1 Expected Result

In this scenario, the clients do have Global Unicast Addresses (2001:db9) residing in a different segment in addition to the Link Local Address. As the clients implement Privacy Extensions and the web server is reachable via the Global Unicast Address (2001:db8), the applicable rules as of RFC 6724 would be **Rule 7: Prefer Temporary Addresses & Rule 8: Use Longest Matching Prefix.**

4.1.8 Test Scenario 8

The clients are on a **different subnet** as the webserver and get a GUA as well as a ULA address via Router Advertisements. The webserver is statically configured to the GUA address in the first subnet but has an additional ULA address. The DNS server serves the GUA as well as the ULA address of the webserver in a round robin fashion.

4.1.8.1 Expected Result

In this scenario, the clients do have Global Unicast Addresses (2001:db9) and Unique Local Addresses residing in a different segment in addition to the Link Local Address. As the clients implement Privacy

Extensions and the web server are reachable via either the Global Unicast or Unique Local Address, the applicable rules as of RFC 6724 would be **Rule 7: Prefer Temporary Addresses & Rule 8: Use Longest Matching Prefix.**

4.1.9 Test Scenario 9

The clients are on a **different subnet** as the webserver and get two GUA addresses from different prefixes via Router Advertisements. The webserver is statically configured to the GUA address in the first subnet and the DNS server serves only the GUA address of the webserver.

4.1.9.1 Expected Result

In this scenario, the clients do have Global Unicast Addresses (2001:db9 and 2001:db1) from different prefixes in addition to the Link Local Address. As the clients implement Privacy Extensions and the web server is reachable via the Global Unicast Address 2001:db8, the applicable rules as of RFC 6724 would be **Rule 7: Prefer Temporary Addresses & Rule 8: Use Longest Matching Prefix.**

4.1.10 Overview of Expected Results

The following table lists the expected results for the test cases defined under the assumption that all operating systems behave strictly according to the rules specified in RFC 6724.

Test case	Result	Applicable Rule(s)
1	Link-local	Prefer Outgoing Interface
2	GUA TMP	Longest Matching Prefix & Prefer Temporary Addresses
3	GUA TMP	Longest Matching Prefix & Prefer Temporary Addresses
4	ULA TMP	Prefer Temporary Addresses
5	GUA TMP	Longest Matching Prefix & Prefer Temporary Addresses
6	GUA TMP (db8)	Longest Matching Prefix & Prefer Temporary Addresses

7	GUA TMP	Prefer Temporary Addresses
8	GUA TMP	Longest Matching Prefix & Prefer Temporary Addresses
9	GUA TMP (db9)	Longest Matching Prefix & Prefer Temporary Addresses

Table 1: Expected Results

4.2 Observed Results

We discuss each test case in detail to examine the behavior of the different operating systems.

4.2.1 Results - Test Scenario 1

For the **first** test case, we see that only the two Linux systems are able to resolve the IPv6 address via DNS and reach the server correctly. Windows systems do not support DNS information provided via Router Advertisements, but can access the server directly via IP address while Mac OSX and iOS based systems were not able to connect to the server at all.

4.2.2 Results - Test Scenario 2

The **second** test case has an expected result for all windows systems, Ubuntu and Android. MAC based systems show a different behavior and tend to skip the usage of the temporary address in this scenario but still have them configured on the interface. While Debian does not have a temporary address by default as this is disabled in the operating system.

4.2.3 Results - Test Scenario 3

The result of the **third** test case is identical to the second. However, since the ULA address of the webserver is not distributed via DNS this is expected behavior.

4.2.4 Results - Test Scenario 4

The **fourth** test case shows the expected result for all operating systems. Here the Mac OSX and iOS based systems also use the temporary address. The only exception is again Debian due to disabled Privacy Extensions.

4.2.5 Results - Test Scenario 5

The **fifth** test case shows an interesting result for Windows 7, Linux and Android based systems. In the DNS both the ULA and GUA address of the server is published and those systems tend to use this information in a round-robin behavior. The other systems always preferred the GUA address, even if they get both via DNS information. Again the MAC OSX and iOS based systems do not use the temporary address.

4.2.6 Results - Test Scenario 6

The **sixth** test case shows again a result that is expected with respect to RFC 6724.

4.2.7 Results - Test Scenario 7

The **seventh** test case shows also the expected results. However, this test case is very similar to the second case with only one GUA address but in this case the Mac OSX based systems make use the temporary address.

4.2.8 Results - Test Scenario 8

The **eighth** test case can be compared to the fifth case and shows the very same results with respect to the changed subnet addresses.

4.2.9 Results - Test Scenario 9

The **ninth** test case shows again the expected results with respect to RFC 6724.

4.3 Overview of Observed Results

We tested each operating system according to our methodology and verified the results. The following table shows the results of our experiment after the tests were finished. As we can see quickly, there are differences between the operating systems for the source address selection and not all operating systems select the same source address in our test setup.

Same Subnet														
Test	LL	GUA	ULA	Client Prefix	Server IP	Win 7	Win 8.1	Win 10	Ubuntu 16.4 TS	Debian 8	Mac OSX	Win 10 Mobile	Android	iPhone
										priv ext. Default off				
1	x			2001:db8:0:1::/64 no-advertise	2001:db8:0:1::2	LL	LL	LL	LL	LL	Not reachable	LL	LL	Not reachable
2	x	x		2001:db8:0:1::/64	2001:db8:0:1::2	GUA TMP	GUA TMP	GUA TMP	GUA TMP	GUA	GUA	GUA TMP	GUA TMP	GUA
3	x	x	x	2001:db8:0:1::/64 FC00:DEAD:BEEF:CAFE::/64	2001:db8:0:1::2	GUA TMP	GUA TMP	GUA TMP	GUA TMP	GUA	GUA	GUA TMP	GUA TMP	GUA
4	x		x	2001:db8:0:1::/64 no-advertise FC00:DEAD:BEEF:CAFE::/64	2001:db8:0:1::2	ULA TMP	ULA TMP	ULA TMP	ULA TMP	ULA	ULA TMP	ULA TMP	ULA TMP	ULA TMP
5	x	x	x	2001:db8:0:1::/64 FC00:DEAD:BEEF:CAFE::/64	2001:db8:0:1::2 FC00:DEAD:BEEF :CAFE::2	TMP ULA OR GUA (based on DNS Answer)	GUA TMP	GUA TMP	TMP ULA OR GUA (based on DNS Answer)	ULA OR GUA (based on DNS Answer)	GUA	GUA TMP	ULA OR GUA (based on DNS Answer)	GUA
6	x	2		2001:db8:0:1::/64 2001:db7:0:1::/64	2001:db8:0:1::2	GUA TMP (db8)	GUA TMP (db8)	GUA TMP (db8)	GUA TMP (db8)	GUA TMP (db8)	GUA TMP (db8)	GUA TMP (db8)	GUA TMP (db8)	GUA TMP (db8)
Routed Subnet														
7	x	x		2001:db9:0:1::/64	2001:db8:0:1::2	GUA TMP	GUA TMP	GUA TMP	GUA TMP	GUA	GUA TMP	GUA TMP	GUA TMP	GUA TMP
8	x	x	x	2001:db9:0:1::/64 FC00:DEAD:BEEF:CCCC::/64	FC00:DEAD:BEEF :CAFE::2	DNS - ULA TMP IP (GUA) - GUA TMP	GUA TMP	GUA TMP	DNS - ULA TMP IP (GUA) - GUA TMP	DNS - ULA IP (GUA) - GUA	GUA TMP	GUA TMP	GUA TMP	GUA TMP
9	x	2		2001:db9:0:1::/64 2001:db10:0:1::/64	2001:db8:0:1::2	GUA TMP (db9)	GUA TMP (db9)	GUA TMP (db9)	GUA TMP (db9)	GUA (TMP)	GUA TMP (db9)	GUA TMP (db9)	GUA TMP (db9)	GUA TMP (db9)

Table 2: Overview of Observed Results

4.4 Further Interesting Observations

As we expected, some of the operating systems do not select the source address as we predicted them before. Interesting here is the fact, that Windows 7 as well as Ubuntu and Debian seem to use the two AAAA records in the DNS in round-robin mode and prefer sometimes the ULA address and another time the GUA address while newer versions of Windows use always the GUA address as well as Mac OSX and iOS.

Another observation is, that Mac OSX and iOS do not use privacy extensions (T. Narten, 2007) in every test case. We do not have an explanation for the observed behavior. In some cases, they prefer the GUA address over the GUA temporary address, what is not expected if we look at the rules of RFC 6724. This might lead to privacy problems for those systems, even an attacker could manipulate or insert Router Advertisements and force those clients to skip using their temporary generated address.

5 Summary and Conclusion

In this section we give a summary of our paper, discuss the limitations of our experiments as well as further work and finally conclude our paper.

5.1 Summary

In this paper we give an introduction and a motivation into the topic of IPv6 source address selection in the first part. Then we explain the different rules that apply in the source address selection according to RFC6724 and give some examples for this selection algorithm. Furthermore, we describe our tested operating systems as well as the test setup in detail. Finally, we give the results of our experiments and conclude our work.

5.2 Limitations

A limitation of our experiment is the fact, that all experiments are conducted in a controlled lab environment. In real word scenarios, other conditions might play a role, for example other statements in the router advertisements or scenarios with more than one router. It would be subject of future work to re-evaluate our results in real world networks.

Further, our experiments only show the behavior of the actual up-to-date operating systems, while systems with a lower patch-level might behave differently or new changes might be applied by the vendors in future versions. To this end, we can only present a current snapshot of system behavior. On Linux systems, the kernel version might also play a role, as well as the version of the used network manager.

To underline the reliability of our results, it would further be necessary to repeat our experiments with a larger test series of multiple runs per single test case and operating system than we did.

5.3 Future Work

Besides the already mentioned evaluation of big real world network setups, there are more questions that could not be answered during this research but may be interesting for future work. One interesting point is to observe the behavior for the source address selection across different patch levels or kernel versions of the same operating system. It might be interesting how a mix of different operating systems with different patch levels behave and interplay with each other in real world scenarios. Additionally, not all common operating systems could be tested in our experiments. There are some more systems that could be taken

into account, such as Android in different versions, as well as FreeBSD and other Linux derivates like SuSe or Redhat.

5.4 Conclusion

We can conclude that our work provides an up to date snapshot of the behavior of different current operating systems and their compliance with the RFC. This work might serve as a basis for proper network configuration evaluation of the different operating system behavior, as well as a list of open working points for the operating system developers. Further, our experiments and test cases might be reused in order to repeat the experiments after some time for newer version of the common operating systems or to compare development of those with different versions or in larger network setups.

5.5 Errata

Any errors and omissions are not intentional. Please provide feedback and corrections to cwerny@ernw.de and we will do our best to get it fixed.

6 References

Antonios Atlasis, E. R. (03. March 2015). *MLD Considered Harmful*. https://www.troopers.de/media/filer_public/7c/35/7c35967a-d0d4-46fb-8a3b-4c16df37ce59/troopers15_ipv6secsummit_atlasis_rey_salazar_mld_considered_harmful_final.pdf

Auer, K. (21. June 2012). *Controlling IPv6 source address selection*. http://biplane.com.au/blog/?p=30

Auer, K. (21. June 2012). *IPv6 Source Address Selection – what, why, how*. http://biplane.com.au/blog/?p=22

C. Huitema, B. C. (September 2004). *Deprecating Site Local Addresses*. https://tools.ietf.org/html/rfc3879

C. Perkins, D. J. (July 2011). *Mobility Support in IPv6*. https://tools.ietf.org/html/rfc6275

D. Thaler, E. e. (September 2012). *Default Address Selection for Internet Protocol Version 6 (IPv6)*. https://tools.ietf.org/html/rfc6724

Hinden, R. e. (August 2003). *IPv6 Global Unicast Address Format*. https://tools.ietf.org/html/rfc3587

Hinden, R. H. (October 2005). *Unique Local IPv6 Unicast Addresses*. https://tools.ietf.org/html/rfc4193

IPv6 Enabled Networks. (14. July 2016). http://v6asns.ripe.net/v/6

J. Jeong, S. P. (November 2010). *IPv6 Router Advertisement Options for DNS Configuration*. https://tools.ietf.org/html/rfc6106

Pepelnjak, I. (13. July 2016). *IPv6 Address Allocation Is Operating System-Specific*. http://blog.ipspace.net/2016/01/ipv6-address-allocation-is-operating.html

R. Draves, Microsoft Research. (February 2003). *Default Address Selection for Internet Protocol version 6 (IPv6)*. https://www.ietf.org/rfc/rfc3484.txt

S. Deering, B. H. (March 2005). *IPv6 Scoped Address Architecture*. https://tools.ietf.org/html/rfc4007

S. Deering, R. H. (December 1998). *Internet Protocol, Version 6 (IPv6)* https://tools.ietf.org/html/rfc2460

Sander Degen, A. H. (March 2014). *Testing the security of IPv6 Implementations*. https://www.tno.nl/media/3274/testing_the_security_of_ipv6_implementations.pdf

Schaub, M. (08. Januar 2014). *IPv6 – welche Interface-Adresse wird genutzt?* http://www.comconsult-research.de/ipv6-interface-adresse/

T. Narten, R. D. (September 2007). *Privacy Extensions for Stateless Address Autoconfiguration in IPv6*. https://tools.ietf.org/html/rfc4941

Thomson, S. e. (September 2007). *IPv6 Stateless Address Autoconfiguration*. https://tools.ietf.org/html/rfc4862