Stream: Internet Engineering Task Force (IETF)
RFC: 8948
Category: Standards Track
Published: December 2020
ISSN: 2070-1721
Authors: CJ. Bernardos *UC3M* A. Mourad *InterDigital*

RFC 8948
Structured Local Address Plan (SLAP) Quadrant Selection Option for DHCPv6

Abstract

The IEEE originally structured the 48-bit Media Access Control (MAC) address space in such a way that half of it was reserved for local use. In 2017, the IEEE published a new standard (IEEE Std 802c) with a new optional Structured Local Address Plan (SLAP). It specifies different assignment approaches in four specified regions of the local MAC address space.

The IEEE is developing protocols to assign addresses (IEEE P802.1CQ). There is also work in the IETF on specifying a new mechanism that extends DHCPv6 operation to handle the local MAC address assignments.

This document proposes extensions to DHCPv6 protocols to enable a DHCPv6 client or a DHCPv6 relay to indicate a preferred SLAP quadrant to the server so that the server may allocate MAC addresses in the quadrant requested by the relay or client. A new DHCPv6 option (QUAD) is defined for this purpose.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc8948.

Copyright Notice

Table of Contents

1. Introduction

The IEEE structures the 48-bit MAC address space in such a way that half of it is reserved for local use (where the Universal/Local (U/L) bit is set to 1). In 2017, the IEEE published a new standard [IEEEStd802c] that defines a new optional Structured Local Address Plan (SLAP) that specifies different assignment approaches in four specified regions of the local MAC address space. These four regions, called SLAP quadrants, are briefly described below (see Figure 1 and Table 1 for details):

- In SLAP Quadrant 01, Extended Local Identifier (ELI) MAC addresses are assigned based on a 24-bit Company ID (CID), which is assigned by the IEEE Registration Authority (RA). The remaining bits are specified as an extension by the CID assignee or by a protocol designated by the CID assignee.
- In SLAP Quadrant 11, Standard Assigned Identifier (SAI) MAC addresses are assigned based on a protocol specified in an IEEE 802 standard. For 48-bit MAC addresses, 44 bits are available. Multiple protocols for assigning SAIs may be specified in IEEE standards. Coexistence of multiple protocols may be supported by limiting the subspace available for assignment by each protocol.
- In SLAP Quadrant 00, Administratively Assigned Identifier (AAI) MAC addresses are assigned locally by an administrator. Multicast IPv6 packets use a destination address starting in 33-33, so AAI addresses in that range should not be assigned. For 48-bit MAC addresses, 44 bits are available.
- SLAP Quadrant 10 is "Reserved for future use" where MAC addresses may be assigned using new methods yet to be defined or until then by an administrator as in the AAI quadrant. For 48-bit MAC addresses, 44 bits would be available.

```
LSB                 MSB
M  X  Y  Z  -  -  -  -
|  |  |  |
|  |  |  +------------ SLAP Z-bit
|  |  +--------------- SLAP Y-bit
|  +------------------ X-bit (U/L) = 1 for locally assigned
+--------------------- M-bit (I/G) (unicast/group)

Legend:
LSB: Least Significant Bit
MSB: Most Significant Bit
```

Figure 1: IEEE 48-Bit MAC Address Structure (in IEEE Representation)

Quadrant	Y-bit	Z-bit	Local Identifier Type	Local Identifier
01	0	1	Extended Local	ELI
11	1	1	Standard Assigned	SAI

Quadrant	Y-bit	Z-bit	Local Identifier Type	Local Identifier
00	0	0	Administratively Assigned	AAI
10	1	0	Reserved	Reserved

Table 1: SLAP Quadrants

1.1. Problem Statement

The IEEE is developing mechanisms to assign addresses [IEEE-P802.1CQ-Project]. And [RFC8947] specifies a new mechanism that extends DHCPv6 operation to handle the local MAC address assignments. This document proposes extensions to DHCPv6 protocols to enable a DHCPv6 client or a DHCPv6 relay to indicate a preferred SLAP quadrant to the server so that the server may allocate the MAC addresses in the quadrant requested by the relay or client.

In the following, we describe two application scenarios in which a need arises to assign local MAC addresses according to preferred SLAP quadrants.

1.1.1. Wi-Fi (IEEE 802.11) Devices

Today, most Wi-Fi devices come with interfaces that have a "burned-in" MAC address, allocated from the universal address space using a 24-bit Organizationally Unique Identifier (OUI) (assigned to IEEE 802 interface vendors). However, recently, the need to assign local (instead of universal) MAC addresses has emerged particularly in the following two scenarios:

- IoT (Internet of Things): In general, composed of constrained devices [RFC7228] with constraints such as available power and energy, memory, and processing resources. Examples of this include sensors and actuators for health or home automation applications. Given the increasingly high number of these devices, manufacturers might prefer to equip devices with temporary MAC addresses used only at first boot. These temporary MAC addresses would just be used to send initial DHCP packets to available DHCP servers. IoT devices typically need a single MAC address for each available network interface. Once the server assigns a MAC address, the device would abandon its temporary MAC address. Home automation IoT devices typically do not move (however, note that there is an increase of mobile smart health monitoring devices). For this type of device, in general, any type of SLAP quadrant would be good for assigning addresses, but ELI/SAI quadrants might be more suitable in some scenarios. For example, the device might need to use an address from the CID assigned to the IoT communication device vendor, thus preferring the ELI quadrant.
- Privacy: Today, MAC addresses allow the exposure of user locations making it relatively easy to track user movements. One of the mechanisms considered to mitigate this problem is the use of local random MAC addresses: changing them every time the user connects to a different network. In this scenario, devices are typically mobile. Here, AAI is probably the best SLAP quadrant from which to assign addresses; it is the best fit for randomization of addresses, and it is not required for the addresses to survive when changing networks.

1.1.2. Hypervisor: Functions That Are and Are Not Migratable

In large-scale virtualization environments, thousands of virtual machines (VMs) are active. These VMs are typically managed by a hypervisor, which is in charge of spawning and stopping VMs as needed. The hypervisor is also typically in charge of assigning new MAC addresses to the VMs. If a DHCP solution is in place for that, the hypervisor acts as a DHCP client and requests that available DHCP servers assign one or more MAC addresses (an address block). The hypervisor does not use those addresses for itself, but rather it uses them to create new VMs with appropriate MAC addresses. If we assume very large data-center environments, such as the ones that are typically used nowadays, it is expected that the data center is divided in different network regions, each one managing its own local address space. In this scenario, there are two possible situations that need to be tackled:

- Migratable functions: If a VM (providing a given function) needs to be migrated to another region of the data center (e.g., for maintenance, resilience, end-user mobility, etc.), the VM's networking context needs to migrate with the VM. This includes the VM's MAC address(es). Since the assignments from the ELI/SAI SLAP quadrants are governed by rules per IEEE Std 802c, they are more appropriate for use to ensure MAC address uniqueness globally in the data center.
- Functions that are not migratable: If a VM will not be migrated to another region of the data center, there are no requirements associated with its MAC address. In this case, it is simpler to allocate it from the AAI SLAP quadrant, which does not need to be unique across multiple data centers (i.e., each region can manage its own MAC address assignment without checking for duplicates globally).

2. Terminology

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Where relevant, the DHCPv6 terminology from [RFC8415] also applies here. Additionally, the following definitions are updated for this document.

address
: Unless specified otherwise, a link-layer (or MAC) address, as specified in [IEEEStd802]. The address is 6 or 8 bytes long.

address block
: A number of consecutive link-layer addresses. An address block is expressed as a first address plus a number that designates the number of additional (extra) addresses. A single address can be represented by the address itself and zero extra addresses.

client
: A node that is interested in obtaining link-layer addresses. It implements the basic DHCP mechanisms needed by a DHCP client, as described in [RFC8415], and supports the options (IA_LL and LLADDR) specified in [RFC8947] as well as the new option (QUAD) specified in this document. The client may or may not support IPv6 address assignment and prefix delegation, as specified in [RFC8415].

IA_LL
: Identity Association for Link-Layer Address, an identity association (IA) used to request or assign link-layer addresses. Section 11.1 of [RFC8947] provides details on the IA_LL option.

LLADDR
: Link-layer address option that is used to request or assign a block of link-layer addresses. Section 11.2 of [RFC8947] provides details on the LLADDR option.

relay
: A node that acts as an intermediary to deliver DHCP messages between clients and servers. A relay, based on local knowledge and policies, may include the preferred SLAP quadrant in a QUAD option sent to the server. The relay implements basic DHCPv6 relay agent functionality, as described in [RFC8415].

server
: A node that manages link-layer address allocation and is able to respond to client queries. It implements basic DHCP server functionality, as described in [RFC8415], and supports the options (IA_LL and LLADDR) specified in [RFC8947] as well as the new option (QUAD) specified in this document. The server may or may not support IPv6 address assignment and prefix delegation, as specified in [RFC8415].

3. DHCPv6 Extensions

3.1. Address Assignment from the Preferred SLAP Quadrant Indicated by the Client

Next, we describe the protocol operations for a client to select a preferred SLAP quadrant using the DHCPv6 signaling procedures described in [RFC8947]. The signaling flow is shown in Figure 2.

```
 +--------+                             +--------+
 | DHCPv6 |                             | DHCPv6 |
 | client |                             | server |
 +--------+                             +--------+
     |                                       |
     +----1. Solicit(IA_LL(LLADDR,QUAD))--->|
     |                                       |
     |<--2. Advertise(IA_LL(LLADDR,QUAD))---+
     |                                       |
     +---3. Request(IA_LL(LLADDR,QUAD))---->|
     |                                       |
     |<------4. Reply(IA_LL(LLADDR))--------+
     |                                       |
     .                                       .
     .           (timer expiring)            .
     .                                       .
     |                                       |
     +---5. Renew(IA_LL(LLADDR,QUAD))------>|
     |                                       |
     |<-----6. Reply(IA_LL(LLADDR))---------+
     |                                       |
```

Figure 2: DHCPv6 Signaling Flow (Client-Server)

1. Link-layer addresses (i.e., MAC addresses) are assigned in blocks. The smallest block is a single address. To request an assignment, the client sends a Solicit message with an IA_LL option in the message. The IA_LL option **MUST** contain an LLADDR option. In order to indicate the preferred SLAP quadrant(s), the IA_LL option includes the new OPTION_SLAP_QUAD option in the IA_LL-option field (alongside the LLADDR option).
2. The server, upon receiving an IA_LL option in a Solicit message, inspects its contents. For each of the entries in the OPTION_SLAP_QUAD, in order of the preference field (highest to lowest), the server checks if it has a configured MAC address pool matching the requested quadrant identifier and an available range of addresses of the requested size. If suitable addresses are found, the server sends back an Advertise message with an IA_LL option containing an LLADDR option that specifies the addresses being offered. If the server manages a block of addresses belonging to a requested quadrant, the addresses being offered **MUST** belong to a requested quadrant. If the server does not have a configured address pool matching the client's request, it **SHOULD** return the IA_LL option with the addresses being offered belonging to a quadrant managed by the server (following a local policy to select from which of the available quadrants). If the server has a configured address pool of the correct quadrant but no available addresses, it **MUST** return the IA_LL option containing a Status Code option with status set to NoAddrsAvail.
3. The client waits for available servers to send Advertise responses and picks one server, as defined in Section 18.2.9 of [RFC8415]. The client **SHOULD NOT** pick a server that does not advertise an address in the requested quadrant(s). The client then sends a Request message that includes the IA_LL container option with the LLADDR option copied from the Advertise message sent by the chosen server. It includes the preferred SLAP quadrant(s) in a new QUAD IA_LL option.

4. Upon reception of a Request message with an IA_LL container option, the server assigns requested addresses. The server **MAY** alter the allocation at this time (e.g., by reducing the address block). It then generates and sends a Reply message back to the client. Upon receiving a Reply message, the client parses the IA_LL container option and may start using all provided addresses. Note that a client that has included a Rapid Commit option in the Solicit message may receive a Reply message in response to the Solicit message and skip the Advertise and Request message steps above (following standard DHCPv6 procedures).
5. The client is expected to periodically renew the link-layer addresses, as governed by T1 and T2 timers. This mechanism can be administratively disabled by the server sending "infinity" as the T1 and T2 values (see Section 7.7 of [RFC8415]). The client sends a Renew option after T1. It includes the preferred SLAP quadrant(s) in the new QUAD IA_LL option, so in case the server is unable to extend the lifetime on the existing address(es), the preferred quadrants are known for the allocation of any "new" (i.e., different) addresses.
6. The server responds with a Reply message with an IA_LL option that includes an LLADDR option with extended lifetime.

The client **SHOULD** check if the received MAC address comes from one of the requested quadrants. It **MAY** repeat the process selecting a different DHCP server.

3.2. Address Assignment from the Preferred SLAP Quadrant Indicated by the Relay

Next, we describe the protocol operations for a relay to select a preferred SLAP quadrant using the DHCPv6 signaling procedures described in [RFC8947]. This is useful when a DHCPv6 server is operating over a large infrastructure split in different network regions, where each region might have different requirements. The signaling flow is shown in Figure 3.

```
+--------+                  +--------+                      +--------+
| DHCPv6 |                  | DHCPv6 |                      | DHCPv6 |
| client |                  | relay  |                      | server |
+--------+                  +--------+                      +--------+
   |                            |                                 |
   +-----1. Solicit(IA_LL)----->|                                 |
   |                            +----2. Relay-forward             |
   |                            |     (Solicit(IA_LL),QUAD)------>|
   |                            |                                 |
   |                            |<---3. Relay-reply               |
   |                            |     (Advertise(IA_LL(LLADDR)))--+
   |<4. Advertise(IA_LL(LLADDR))+                                 |
   |-5. Request(IA_LL(LLADDR))->|                                 |
   |                            +-6. Relay-forward                |
   |                            |  (Request(IA_LL(LLADDR)),QUAD)->|
   |                            |                                 |
   |                            |<--7. Relay-reply                |
   |                            |    (Reply(IA_LL(LLADDR)))-------+
   |<--8. Reply(IA_LL(LLADDR))--+                                 |
   |                            |                                 |
   .                            .                                 .
   .                    (timer expiring)                          .
   .                            .                                 .
   |                            |                                 |
   +--9. Renew(IA_LL(LLADDR))-->|                                 |
   |                            |--10. Relay-forward              |
   |                            |   (Renew(IA_LL(LLADDR)),QUAD)-->|
   |                            |                                 |
   |                            |<---11. Relay-reply              |
   |                            |      (Reply(IA_LL(LLADDR)))-----+
   |<--12. Reply(IA_LL(LLADDR))-+                                 |
   |                            |                                 |
```

Figure 3: DHCPv6 Signaling Flow (Client-Relay-Server)

1. Link-layer addresses (i.e., MAC addresses) are assigned in blocks. The smallest block is a single address. To request an assignment, the client sends a Solicit message with an IA_LL option in the message. The IA_LL option **MUST** contain an LLADDR option.
2. The DHCP relay receives the Solicit message and encapsulates it in a Relay-forward message. The relay, based on local knowledge and policies, includes in the Relay-forward message the QUAD option with the preferred quadrant. The relay might know which quadrant to request based on local configuration (e.g., the served network contains IoT devices only, thus requiring ELI/SAI) or other means. Note that if a client sends multiple instances of the IA_LL option in the same message, the DHCP relay **MAY** only add a single instance of the QUAD option.
3. Upon receiving a relayed message containing an IA_LL option, the server inspects the contents for instances of OPTION_SLAP_QUAD in both the Relay-forward message and the client's message payload. Depending on the server's policy, the instance of the option to process is selected (see the end of this section). For each of the entries in OPTION_SLAP_QUAD, in order of the preference field (highest to lowest), the server checks if it has a configured MAC address pool matching the requested quadrant identifier and an

available range of addresses of the requested size. If suitable addresses are found, the server sends back an Advertise message with an IA_LL option containing an LLADDR option that specifies the addresses being offered. This message is sent to the Relay in a Relay-reply message. If the server supports the semantics of the preferred quadrant included in the QUAD option and manages a block of addresses belonging to a requested quadrant, then the addresses being offered **MUST** belong to a requested quadrant. The server **SHOULD** apply the contents of the relay's supplied QUAD option for all of the client's IA_LLs, unless configured to do otherwise.

4. The relay sends the received Advertise message to the client.
5. The client waits for available servers to send Advertise responses and picks one server, as defined in Section 18.2.9 of [RFC8415]. The client then sends a Request message that includes the IA_LL container option with the LLADDR option copied from the Advertise message sent by the chosen server.
6. The relay forwards the received Request in a Relay-forward message. It adds, in the Relay-forward, a QUAD IA_LL option with the preferred quadrant.
7. Upon reception of the forwarded Request message with IA_LL container option, the server assigns requested addresses. The server **MAY** alter the allocation at this time. It then generates and sends a Reply message in a Relay-reply message back to the relay.
8. Upon receiving a Reply message, the client parses the IA_LL container option and may start using all provided addresses.
9. The client is expected to periodically renew the link-layer addresses, as governed by T1 and T2 timers. This mechanism can be administratively disabled by the server sending "infinity" as the T1 and T2 values (see Section 7.7 of [RFC8415]). The client sends a Renew option after T1.
10. This message is forwarded by the relay in a Relay-forward message, including a QUAD IA_LL option with the preferred quadrant.
11. The server responds with a Reply message, including an LLADDR option with extended lifetime. This message is sent in a Relay-reply message.
12. The relay sends the Reply message back to the client.

The server **SHOULD** implement a configuration parameter to deal with the case where the client's DHCP message contains an instance of OPTION_SLAP_QUAD and the relay adds a second instance in its Relay-forward message. This parameter configures the server to process either the client's or the relay's instance of option QUAD. It is **RECOMMENDED** that the default for such a parameter is to process the client's instance of the option.

The client **MAY** check if the received MAC address belongs to a quadrant it is willing to use/configure and **MAY** decide based on that whether to use/configure the received address.

4. DHCPv6 Option Definition

4.1. QUAD Option

The QUAD option is used to specify the preferences for the selected quadrants within an IA_LL. The option **MUST** be encapsulated either in the IA_LL-options field of an IA_LL option or in a Relay-forward message.

The format of the QUAD option is:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       OPTION_SLAP_QUAD        |           option-len          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   quadrant-1  |    pref-1     |   quadrant-2  |    pref-2     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  quadrant-n-1 |   pref-n-1    |   quadrant-n  |    pref-n     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: QUAD Option Format

option-code
: OPTION_SLAP_QUAD (140).

option-len
: 2 * number of included (quadrant, preference). This is a 2-byte field containing the total length of all (quadrant, preference) pairs included in the option.

quadrant-n
: Identifier of the quadrant (0: AAI, 1: ELI, 2: Reserved by IEEE [IEEEStd802c], and 3: SAI). Each quadrant **MUST** only appear once at most in the option. This is a 1-byte field.

pref-n
: Preference associated to quadrant-n. A higher value means a more preferred quadrant. This is a 1-byte field.

A quadrant identifier value **MUST** only appear, at most, once in the option. If an option includes more than one occurrence of the same quadrant identifier, only the first occurrence is processed, and the rest **MUST** be ignored by the server.

If the same preference value is used for more than one quadrant, the server **MAY** select which quadrant should be preferred (if the server can assign addresses from all or some of the quadrants with the same assigned preference). Note that this is not a simple list of quadrants ordered by preference with no preference value, but a list of quadrants with explicit preference values. This way it can support the case whereby a client really has no preference between two or three quadrants, leaving the decision to the server.

If the client or relay agent provides the OPTION_SLAP_QUAD, the server **MUST** use the quadrant-n/pref-n values to order the selection of the quadrants. If the server can provide an assignment from one of the specified quadrants, it **SHOULD** proceed with the assignment. If the server does not have a configured address pool matching any of the specified quadrant-n fields or if the server has a configured address pool of the correct quadrant but no available addresses, it **MUST** return the IA_LL option containing a status of NoAddrsAvail.

There is no requirement that the client or relay agent order the quadrant/pref values in any specific order; hence, servers **MUST NOT** assume that quadrant-1/pref-1 have the highest preference (except if there is only one set of values).

For cases where a server may not be configured to have pools for the client or relay quadrant preferences, clients and relays **SHOULD** specify all quadrants in the QUAD option to assure the client gets an address (or addresses) -- if any are available. Specifying all quadrants also results in a QUAD option supporting server responding like a non-QUAD option supporting server, i.e., an address (or addresses) from any available quadrants can be returned.

5. IANA Considerations

IANA has assigned the QUAD (140) option code from the "Option Codes" subregistry of the "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)" registry maintained at <http://www.iana.org/assignments/dhcpv6-parameters>:

Value: 140
Description: OPTION_SLAP_QUAD
Client ORO: No
Singleton Option: Yes
Reference: RFC 8948

6. Security Considerations

See [RFC8415] and [RFC7227] for the DHCPv6 security and privacy considerations. See [RFC8200] for the IPv6 security considerations.

Also, see [RFC8947] for security considerations regarding link-layer address assignments using DHCP.

7. References

7.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8415] Mrugalski, T., Siodelski, M., Volz, B., Yourtchenko, A., Richardson, M., Jiang, S., Lemon, T., and T. Winters, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 8415, DOI 10.17487/RFC8415, November 2018, <https://www.rfc-editor.org/info/rfc8415>.

[RFC8947] Volz, B., Mrugalski, T., and CJ. Bernardos, "Link-Layer Address Assignment Mechanism for DHCPv6", RFC 8947, DOI 10.17487/RFC8947, December 2020, <https://www.rfc-editor.org/info/rfc8947>.

7.2. Informative References

[IEEE-P802.1CQ-Project] IEEE, "P802.1CQ - Standard for Local and Metropolitan Area Networks: Multicast and Local Address Assignment", <https://standards.ieee.org/project/802_1CQ.html>.

[IEEEStd802] IEEE, "IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture", IEEE Std 802-2014, DOI 10.1109/IEEESTD.2014.6847097, June 2014, <https://doi.org/10.1109/IEEESTD.2014.6847097>.

[IEEEStd802c] IEEE, "IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture -- Amendment 2: Local Medium Access Control (MAC) Address Usage", IEEE Std 802c-2017, DOI 10.1109/IEEESTD.2017.8016709, August 2017, <https://doi.org/10.1109/IEEESTD.2017.8016709>.

[RFC7227] Hankins, D., Mrugalski, T., Siodelski, M., Jiang, S., and S. Krishnan, "Guidelines for Creating New DHCPv6 Options", BCP 187, RFC 7227, DOI 10.17487/RFC7227, May 2014, <https://www.rfc-editor.org/info/rfc7227>.

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, DOI 10.17487/RFC7228, May 2014, <https://www.rfc-editor.org/info/rfc7228>.

[RFC7548] Ersue, M., Ed., Romascanu, D., Schoenwaelder, J., and A. Sehgal, "Management of Networks with Constrained Devices: Use Cases", RFC 7548, DOI 10.17487/RFC7548, May 2015, <https://www.rfc-editor.org/info/rfc7548>.

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

Appendix A. Example Uses of Quadrant Selection Mechanisms

This appendix describes some examples of how the quadrant preference mechanisms could be used.

First, let's take an IoT scenario as an example. An IoT device might decide on its own the SLAP quadrant it wants to use to obtain a local MAC address, using the following information to make the decision:

- Type of IoT deployment: For example, industrial, domestic, rural, etc. For small deployments, such as domestic ones, the IoT device itself can decide to use the AAI quadrant (this might not even involve the use of DHCP, by the device just configuring a random address computed by the device itself). For large deployments, such as industrial or rural ones, where thousands of devices might coexist, the IoT can decide to use the ELI or SAI quadrants.
- Mobility: If the IoT device can move, then it might prefer to select the SAI or AAI quadrants to minimize address collisions when moving to another network. If the device is known to remain fixed, then the ELI is probably the most suitable one to use.
- Managed/Unmanaged: Depending on whether the IoT device is managed during its lifetime or cannot be reconfigured [RFC7548], the decision of what quadrant is more appropriate might be different. For example, if the IoT device can be managed (e.g., configured) and network topology changes might occur during its lifetime (e.g., due to changes on the deployment, such as extensions involving additional devices), this has an impact on the preferred quadrant (e.g., to avoid potential collisions in the future).
- Operation / Battery Lifetime: Depending on the expected lifetime of the device, a different quadrant might be preferred (as before, to minimize potential address collisions in the future).

The previous parameters are considerations that the device vendor/administrator may wish to use when defining the IoT device's MAC address request policy (i.e., how to select a given SLAP quadrant). IoT devices are typically very resource constrained, so there may only be a simple decision-making process based on preconfigured preferences.

We now take the Wi-Fi device scenario, considering, for example, that a laptop or smartphone connects to a network using its built-in MAC address. Due to privacy/security concerns, the device might want to configure a local MAC address. The device might use different parameters and context information to decide, not only which SLAP quadrant to use for the local MAC address configuration, but also when to perform a change of address (e.g., it might be needed to change address several times). This information includes, but it is not limited to:

- Type of network the device is connected: public, work, home.
- Trusted network: Yes/No.
- First time visited network: Yes/No.
- Network geographical location.
- Mobility: Yes (the device might change its network attachment point) / No (the device is not going to change its network attachment point).
- Operating System (OS) network profile, including security/trust-related parameters: Most modern OSs keep metadata associated with the networks they can attach to as, for example, the level of trust the user or administrator assigns to the network. This information is used to configure how the device behaves in terms of advertising itself on the network, firewall settings, etc. But this information can also be used to decide whether or not to configure a

local MAC address, from which SLAP quadrant it should be assigned, and how often it may be assigned.
- Triggers coming from applications regarding location privacy: An app might request that the OS maximize location privacy (due to the nature of the application), and this might require the OS to force the use of a local MAC address or the local MAC address to be changed.

This information can be used by the device to select the SLAP quadrant. For example, if the device is moving around (e.g., while connected to a public network in an airport), it is likely that it might change access points several times; therefore, it is best to minimize the chances of address collision, using the SAI or AAI quadrants. If the device is not expected to move and is attached to a trusted network (e.g., in some scenarios at work), then it is probably best to select the ELI quadrant. These are just some examples of how to use this information to select the quadrant.

Additionally, the information can also be used to trigger subsequent changes of MAC address to enhance location privacy. Besides, changing the SLAP quadrant might also be used as an additional enhancement to make it harder to track the user location.

Last, if we consider the data-center scenario, a hypervisor might request local MAC addresses be assigned to virtual machines. As in the previous scenarios, the hypervisor might select the preferred SLAP quadrant using information provided by the cloud management system or virtualization infrastructure manager running on top of the hypervisor. This information might include, but is not limited to:

- Migratable VM: If the function implemented by the VM is subject to be moved to another physical server or not, this has an impact on the preference for the SLAP quadrant, as the ELI and SAI quadrants are better suited for supporting migration in a large data center.
- VM connectivity characteristics: For example, standalone, part of a pool, and part of a service graph/chain. If the connectivity characteristics of the VM are known, this can be used by the hypervisor to select the best SLAP quadrant.

Acknowledgments

The authors would like to thank Bernie Volz for his very valuable comments on this document. We also want to thank Ian Farrer, Tomek Mrugalski, Éric Vyncke, Tatuya Jinmei, Carl Wallace, Ines Robles, Ted Lemon, Jaime Jimenez, Robert Wilton, Benjamin Kaduk, Barry Leiba, Alvaro Retana, and Murray Kucherawy for their very detailed and helpful reviews. And thanks to Roger Marks and Antonio de la Oliva for comments related to IEEE work and references.

The work in this document has been supported by the H2020 5GROWTH (Grant 856709) and 5G-DIVE projects (Grant 859881).

Authors' Addresses

Carlos J. Bernardos
Universidad Carlos III de Madrid
Av. Universidad, 30
28911 Leganes, Madrid
Spain
Phone: +34 91624 6236
Email: cjbc@it.uc3m.es
URI: http://www.it.uc3m.es/cjbc/

Alain Mourad
InterDigital Europe
Email: Alain.Mourad@InterDigital.com
URI: http://www.InterDigital.com/