Networking problems and Internet protocol version 6

Pavel Bezpalec, Lukáš Čepa

Authors:	Pavel Bezpalec, Lukáš Čepa
Title:	Networking problems and Internet protocol version 6
Published by:	Czech Technical University in Prague
Compiled by:	Faculty of Electrical Engineering
Contact address:	Technicka 2, Prague 6, Czech Republic
Phone Number:	+420 2 2435 2084
Print:	(only electronic form)
Number of pages:	73
Edition:	1st Edition
ISBN	978-80-01-05303-4
Reviewed by:	Radim Berger

Innovative Methodology for Promising VET Areas
http://improvet.cvut.cz

Lifelong Learning Programme

This project has been funded with support from the European Commission.
This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Explanatory notes

📚	Definition
💡	Interesting
✎	Note
⚙	Example
🧩	Summary
+	Advantage
–	Disadvantage

ANNOTATION

Diminishing the address space of Internet Protocol version 4 (IPv4) was the reason for the establishment Internet Protocol version 6 (IPv6). Several RFC documents that define IPv6 have been released in 1996. Despite the diminishing IPv4 address space, IPv4 is still the main Internet Protocol in 2012 due to its progressive modifications. But with the growing popularity of services that require direct communications (IP telephony, video conferencing, etc.) and the intended development of WiMAX (Worldwide Interoperability for Microwave Access) such as 4G mobile network, which provides IPv6 mobility, IPv6 could be more extended. Currently, IPv6 is implemented primarily at the core network of most Internet Service Providers.

OBJECTIVES

The main goal of the module is to introduce a student with the fundamental of IPv6. The first part of the module describes the datagram, notation and structure of IPv6 addresses. The second part deals with ICMPv6 protocol that is used to send error messages and informations, without which IPv6 cannot function properly. It also deals with the protocols that IPv6 uses for its activities. They are Neighbors Discovery protocol, IPv6 Address Auto-configuration, Domain Name System and Mobility.

LITERATURE

[1] SATRAPA, Pavel. *IPv6*: *internetový protokol verze 6* [online]. 3., aktualiz. a dopl. vyd. Praha: CZ.NIC, c2011, 407 s. [cit. 2012-03-01]. CZ.NIC. ISBN 978-80-904248-4-5 (BROž.). Dostupné z: http://ii.iinfo.cz/r/kd/internetovy-protokol-ipv6-treti-vydani.pdf.

[2] Deering, S. – Hinden, R. *Internet Protocol, Version 6 (IPv6) Specification*, RFC 2460, December 1998.

[3] Conta, A. – Deering, S. – Gupta, M. *Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification*, RFC 4443, March 2006.

[4] Johnson, D. – Perkins, C. – Arkko, J. *Mobility Support in IPv6*, RFC 6275, July 2011.

Index

1 IPv6 Features

1.1 IPv6 Features

Among main drivers for establishment of **IPv6** (*Internet Protocol version 6*) are the following features:

- Large address space, which was established to 128 bits (four times the IPv4). There are 3,4x10^{38} (2^{128}) addresses.
- Multiple levels of addressing hierarchy that enables efficient aggregation and summarization of routes.
- Single address scheme for the Internet and internal networks.
- Three type of addresses:
  - unicast
  - multicast
  - anycast
- Multiple addresses to a network interface.
- Auto-configuration of nodes.
- Forwarding optimization.
- Support **QoS** (*Quality of Service*).
- Improving security (includes encryption, authentication and tracking of the sender).
- Improving support for multicast.
- Support for mobility (notebooks, etc.).

IPv6 is essentially an extension of **IPv4** (*Internet Protocol version 4*). Most of the transmission and application layer protocols require little or no change in functionality for IPv6. Parts of the applications that work with IPv6 addresses are exception. The format of IPv6 datagram has undergone the greatest change.

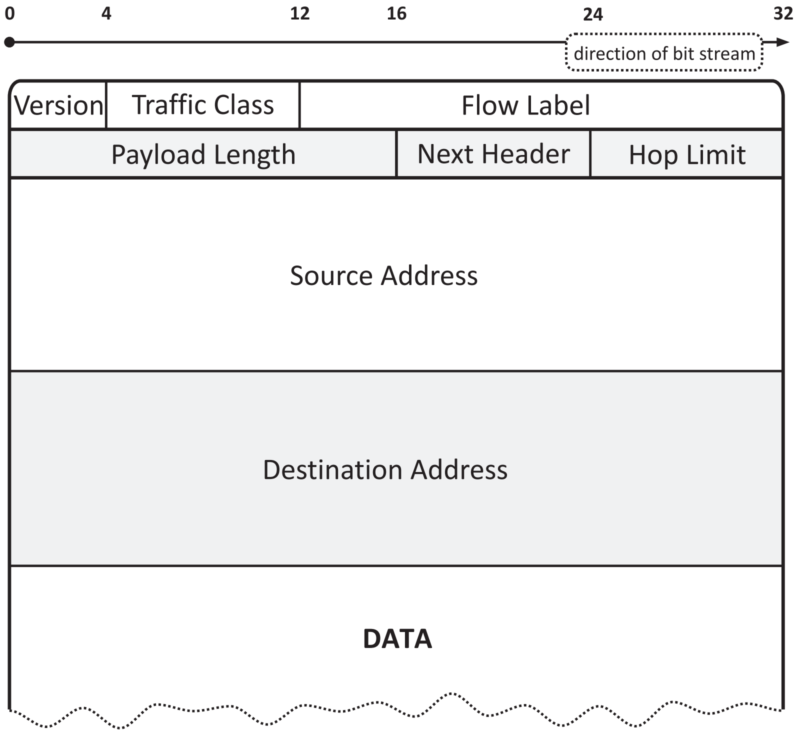
2 Format of IPv6 Datagram

2.1 Datagram

RFC 2460 is the basic building block of IPv6. It mainly defines the format of datagram. IPv6 datagram is composed of header, extension headers and data. It has a constant length and optional information is moved into separate extension headers as opposed to IPv4 datagram. Extension headers may or may not place after header. The total length of the IPv6 header is twice large than IPv4 header. It increased from 20 octets to 40 octets.

Checksum field was removed from the header because the lower layer of network architecture performs it.

IPv6 Header has the following format:



IPv6 Header Format

Description of individual fields of IPv6 header:

- Version – it identifies the protocol version and it contains a value of 6. The field has a size of 4 bits.

- Traffic Class – it expresses the priority of IPv6 datagram. The purpose of the field is to provide services with guaranteed quality (QoS). The field has a size of 8 bits.

- Flow Label – identifies the stream of datagrams with common parameters. Consequently, the router can route datagrams based on labels allowing faster forwarding. The field has a size of 20 bits.

- Payload Length – it expresses the length of the datagram in octets. The total length does not include the length of the header itself. The field has a size of 16 bits, which allows a maximum length of the IPv6 datagram up to 64 KB. To create a longer datagram, the Hop-by-Hop Options header with Jumbo Payload option has to be used.

- Next Header – it identifies the extension header or type of data that follows after the header. The field has a size of 8 bits.

- Hop Limit – it replaces the **TTL** (*Time to Live*) field, which is used in IPv4 datagram. The value of Hop Limit is decremented by 1 by each node that forwards the datagram. The datagram is discarded if Hop Limit is decremented to zero. This is reported to the sender by the ICMPv6 Time Exceeded message. The purpose of Hop Limit is to avoid cyclic forwarding. The field has a size of 8 bits.

- Addresses – they are the last two fields. These are the Source Address and Destination Address. Each address has a size of 128 bits and both addresses occupy 32 octets of the total length of the IPv6 header.

2.2 Concatenation of Headers

Optional and supplementary information are moved to a separate extension headers. These headers may or may not place after header. Each extension header is a separate block. To concatenate the extension headers, the Next Header Value field is used. This field contains a code that represents the type of extension or data. This way you can concatenate any number of headers.

IPv6 header Next Header Value = 43 (Routing header)	**Routing header** Next Header Value = 44 (Fragment header)	**Fragment header** Next Header Value = 6 (TCP segment)	**TCP SEGMENT**

Concatenation of Headers

The goal is that interesting information for the nodes are located immediately behind the header and other information that are interesting only for end nodes are located after them. When the concatenation of headers is used, the headers are concatenated in the following order:

1. IPv6 header
2. Hop-by-Hop Options header
3. Destination Options header – for the first destination address
4. Routing header
5. Fragment header
6. Authentication header
7. Encapsulating Security Payload header
8. Destination Options header – for the end node
9. Mobility header

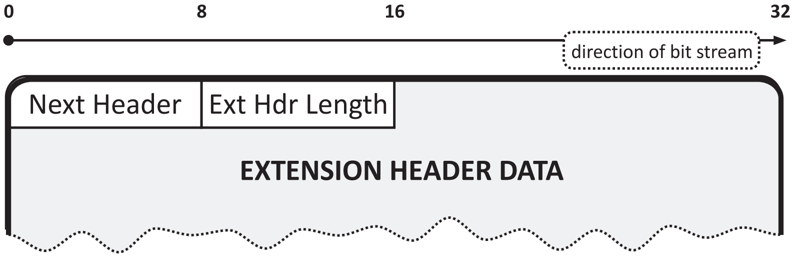
Each extension header can appear only once in the IPv6 datagram, except the Destination Options header.

2.3 Options

There are two extension headers of this type:

- Hop-by-Hop Options header – it is for all nodes in the path
- Destination Options header – this extension header is processed only by end node, others ignore it

Hop-by-Hop Options and Destination Options headers have the following format:



Hop-by-Hop Options Header and Destination Options Header

Hdr Ext Len field indicates the length of the option in 8-octet units, not including the first 8 octets. If the Hdr Ext Len field contains a value of 1 so the length of the option is 16 octets.

There are common options for both these extension headers:

- Pad1 – it is used to insert one octet of padding into the Options area of a header, to ensure that multi-octet values within Option Data fields fall on natural boundaries.
- PadN – it performs a similar function as the previous option, but it is used to insert two or more octets of padding into the Options area of a header.

These options are included in the Hop-by-Hop Options:

- Router Alert – this option alerts all routers in the path that the packet carries interesting data. It is defined in RFC 2711 and is mainly used in the reservation protocol **RSVP** (*Resource Reservation Protocol*), which sends control packets to allocate capacity in the path. Just these packets are addressed to all routers.
- Quick Start – it aims to increase the throughput of transport protocols, especially TCP. The node that starts communication adds this option into the request to establish a TCP connection. This option indicates the desired bit rate. If the bit rate is unsatisfactory for any routers, so this router will reduce its value to an acceptable level. Once the datagram reaches the target node, the Quick Start option includes an acceptable bit rate for all routers in the path. This bit rate may be updated during the communication.

- Jumbo Payload – it allows to create datagrams with length of 65 535 up to 4 294 967 295 octets. These datagrams are called jumbograms. Jumbograms are used only in the case when the transmission technology allows it.

These options are included in the Destination Options:

- Home Address – this option was introduced in the context of supporting mobility and it is used when the mobile node is outside the home network. It informs the target node about the home address of sender.

Router Alert, Jumbo Payload and Home Address options are the only options which have practical use currently.

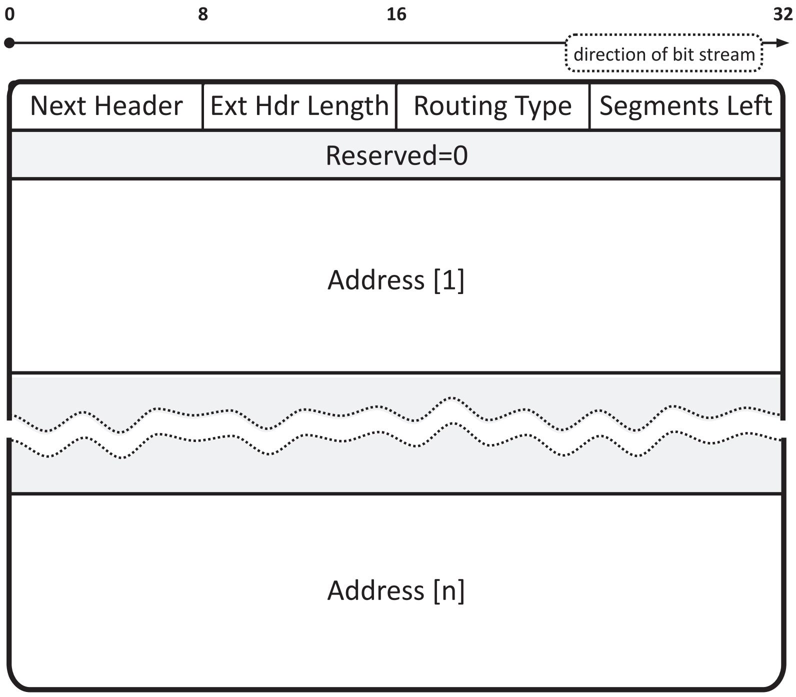
2.4 Routing Header

IPv6 datagram is still forwarded according to the longest compliance of destination address with an entry in the routing table. Routing header allows entering into this process and determines nodes through which the IPv6 datagram should be forwarded.

Currently, IPv6 provides two types of Routing header. Routing Type field is used to distinguish the types of Routing header.

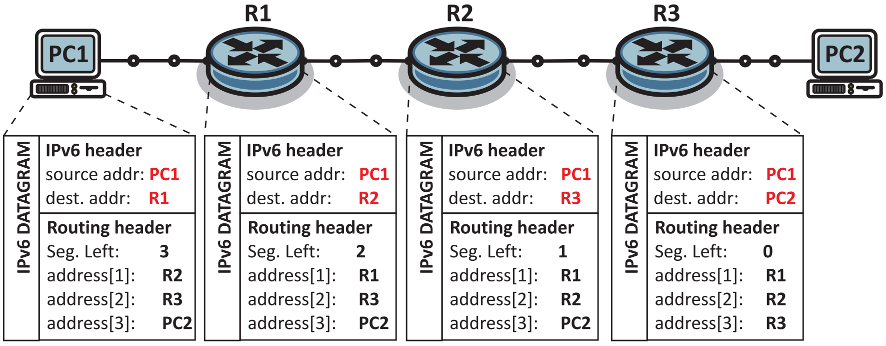
Type 0 Routing Header

This type of Routing header allows determining the nodes which IPv6 datagram has to pass in the given order. It also serves as a record of which nodes the IPv6 datagram has already passed. Type 0 Routing header has the following format:



The Type 0 Routing Header Format

The Type 0 Routing header contains a list of all nodes, which datagram has to pass. The sender takes the first address from the list and places it in the Destination Address field of IPv6 header and destination address of the target node is placed at the end of the list before the datagram is sent. When the datagram arrives at the destination address (a router in the path), the Segments Left field of the Routing header is decremented by 1. This field indicates the next node address in order from the end of the list. The router places this address in the Destination Address field of IPv6 header before the datagram is sent. This process is repeated at each router that is listed in the list until the datagram reaches its target node. This fact indicates zero value in the Segments Left field of the Routing header.



IPv6 header	**IPv6 header**	**IPv6 header**	**IPv6 header**
source addr: PC1	source addr: PC1	source addr: PC1	source addr: PC1
dest. addr: R1	dest. addr: R2	dest. addr: R3	dest. addr: PC2
Routing header	**Routing header**	**Routing header**	**Routing header**
Seg. Left: 3	Seg. Left: 2	Seg. Left: 1	Seg. Left: 0
address[1]: R2	address[1]: R1	address[1]: R1	address[1]: R1
address[2]: R3	address[2]: R3	address[2]: R2	address[2]: R2
address[3]: PC2	address[3]: PC2	address[3]: PC2	address[3]: R3

The Principle of Forwarding and Change the Addresses in the IPv6 Header

The Type 0 Routing header allows determining the exact path of the datagram. It also allows verifying the functionality of this connection. This is an advantage. So this type of Routing header was introduced to test the reachability between any two nodes.

The main disadvantage of this type of Routing header is the possibility of overloading the transmission path. It allows concatenating any number of Routing headers in the IPv6 datagram, which causes that the datagram will be in the network for very long time. This fact can lead to creation of data flows with a huge volume. Another disadvantage is that a datagram can pass through the NAT (Network Address Translation) and firewall. In this case, the private address is placed at the end of the list of Routing header and public address of the router that performs the translation addresses is placed at the list as intermediate.

The above problems have led to prohibition of the use the Type 0 Routing header and definition RFC 5095 that describes the ways to deal with such forwarded datagrams.

The Type 2 Routing Header

This type of Routing header was specially defined for mobility and uses the same mechanisms as the type 0. However, the type 2 reduces the number of concatenated Routing headers only to one header with a single address and thereby reduces its abuse.

2.5 Fragment Header

Fragmentation is a process in which too large datagram is broken into several smaller ones so that the length of new datagrams matches the size of the **MTU** (*Maximum Transmission Unit*). MTU is the maximum datagram length in octets, which the link is able to convey. In the case of IPv6, the minimum value of MTU is 1280 octets. This MTU size is chosen in order to minimize the fragmentation.

Link is a communication facility or medium over which nodes can communicate at the link layer, i.e., the layer immediately below IPv6. Examples are Ethernets, PPP links etc.

Each data path consists of transit nodes and links where each link can have a different size of MTU. The total MTU of all nodes and links is called **PMTU** (*Path MTU*). So PMTU is given by the link with the lowest MTU in the path.

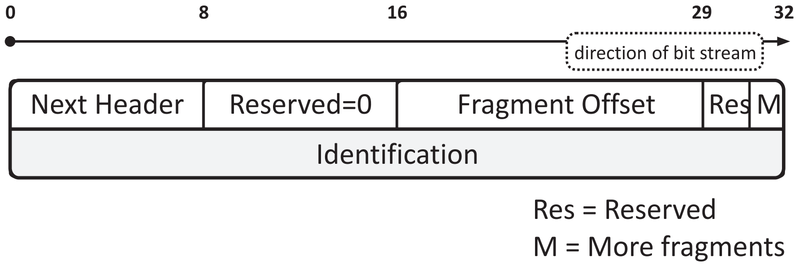
Fragmentation is a complex process that unnecessarily loads the transit nodes. This fact was taken into consideration during the development of IPv6. So, IPv6 allows fragmentation only and exclusively on the side of sender. It means that if a link with a smaller MTU than outgoing interface MTU is anywhere in the path, the datagram is discarded and the node that discards the datagram informs the sender about the fact using ICMPv6 Packet Too Big message.

The principle of fragmentation

IPv6 datagram is divided into two parts due to the fragmentation:

- Unfragmentable Part – it consists of the IPv6 header plus any extension headers that must be processed by nodes in the path to the target node, that is, all headers up to and including the Routing header if present, else the Hop-by-Hop Options header if present, else no extension headers.
- Fragmentable Part – it consists of the rest of the datagram, that is, any extension headers that need be processed only by the target node(s), plus the upper-layer header and data.

Fragment header has the following format:



Fragment Header Format

Fragmentation applies only to Fragmentable Part. The Fragmentable Part is broken into smaller parts whose lengths are a multiple of 8-octet units so that these parts are also smaller than the required MTU. These parts are called fragments. Each fragment is composed of:

- Unfragmentable Part is taken and the length of the datagram in Payload Length field of IPv6 header is adjusted in order to match the length of the fragment. The value 44 that corresponds to Fragment header is written to Next Header field in the last concatenated header.

- Fragment header is added. It contains:

  - The Identification value generated for the original datagram.

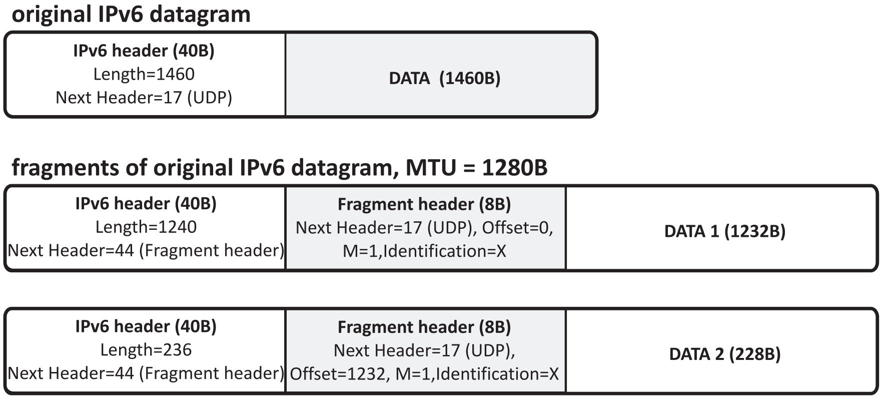
  - The Next Header value that identifies the first header of the Fragmentable Part of the original datagram.

  - A Fragment Offset field contains the offset of the fragment, in 8-octet units, relative to the start of the Fragmentable Part of the original datagram. The Fragment Offset of the first fragment is 0. The Fragment Offset of other fragments is equal to multiple of the fragment length. This offset is the same for all fragments, except the first and last one.

  - An M flag value of 0 if the fragment is the last one, else an M flag value of 1. This field represents the flag of the last fragment.

- The fragment itself.

Fragments can be considered as separate datagrams that are sent to the target node. At the target node, the original datagram is reassembled from the information in Fragment header.



Principle of Fragmentation

2.6 Datagram Length

IPv6 is designed in order to minimize the fragmentation. It is closely related to the length of outgoing datagrams. Each datagram should be as large as possible to avoid the network overload due to sending a large number of smaller datagrams. But the datagram length also may not exceed path MTU. To find the ideal datagram length, the Path MTU Discovery algorithm is used. The RFC 1981 defines this algorithm.

Path MTU Discovery Algorithm

To find the ideal datagram length, the Path MTU Discovery algorithm proceeds as follows:

- Sender sends a datagram whose length is equal to outgoing link MTU (total path MTU cannot be larger than outgoing link MTU).
- If the datagram arrives at the target node, the path MTU is found.
- If the datagram encounters a link with smaller MTU than the outgoing link MTU, so the node that detects this fact discards the datagram and informs the sender using the ICMPv6 Packet Too Big message.
- Sender reduces the datagram length according ICMPv6 message and sends the datagram again.
- These steps are repeated until the datagram arrives at target node.

This algorithm should be repeated at certain intervals, because the path MTU can be increased during communication. The recommended interval is 10 minutes.

2.7 Other Extension Headers

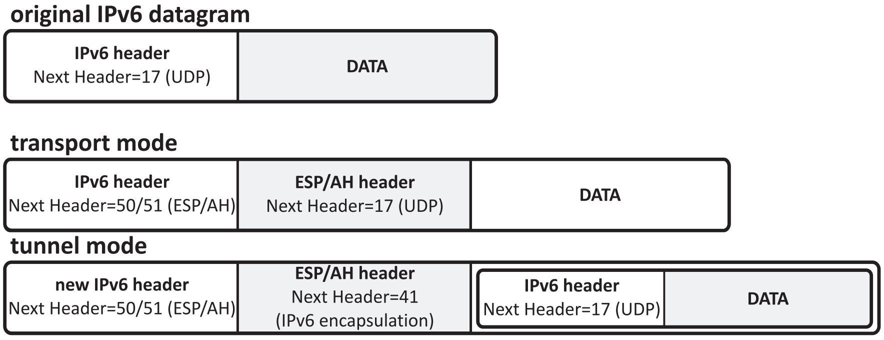
Authentication and Encapsulating Security Payload Extension Headers

Authentication and data encryption are known as **IPsec** (*IP security*). IPsec is a very broad topic and there is not enough space to describe it. Therefore, we mention only the extension headers that are used for IPsec in the context of IPv6. We also mention the modes of encapsulating these headers into IPv6 datagram.

AH (*Authentication Header*) and **ESP** (*Encapsulating Security Payload*) headers are used for IPsec. AH header provides only authentication and ESP header provides authentication and data encryption. Datagram can be provided with one or both headers at the same time. In the case of IPv6, the IPsec implementation is mandatory, unlike in IPv4.

Both extension headers can be encapsulated in two modes:

- Transport Mode – headers are encapsulated directly as part of the IPv6 datagram.
- Tunnel Mode – the original datagram is encapsulated into a new datagram as data.



IPsec Modes

No Next Header

No Next Header indicates the fact that all relevant information is only in the IPv6 header and extension headers and the datagram does not convey any data. For example, ICMPv6 protocol uses this extension header.

3 IPv6 Addresses

3.1 IPv6 Addresses

The main reason for establishment IPv6 is its enormously large address space. During the development of IPv6, the authors followed the motto "IPv6 addresses must not be ever exhausted" and therefore they chose the length of IPv6 addresses of 128 bits. The fundamental document that defines the IPv6 addresses is RFC 4291. This document defines lengths, structures and types of addresses. It also defines other conceptual elements. There are several RFC documents that describe individual types of IPv6 addresses in more detail. We mention only the fundamental of IPv6 addresses in this module. RFC 4291 defines the following types of addresses:

- Unicast – this address identifies a single network interface.
- Multicast – this address is used for addressing a group of interfaces. If the data is sent to this address, it has to be delivered to all interfaces in the multicast group (for example IPTV).
- Anycast – it is a new thing in IPv6. These addresses identify a group of devices. However, if the data is sent to anycast address, it is delivered to only device that is closest to the sender.

In IPv6, the broadcast addresses are not used. Multicast addresses take over the function of broadcast addresses. For this purpose, the special groups were defined. For example, the group for all IPv6 nodes within link-local scope that replace original broadcast addresses.

IPv6 addresses are assigned to network interfaces as well as in IPv4. In IPv6, interfaces can have more addresses of different types. Even it is ordered that the interface has to have several mandatory IPv6 addresses for each node.

Structure and Notation of IPv6 Addresses

IPv6 address has a length of 128 bits and is composed of 8 groups of 16 bits. Each group is expressed in four digits of hexadecimal system and the individual groups are separated by a colon. Example of IPv6 address is

2001:0718:0000:0000:28F6:19FF:FE00:1984

The possibility to compressing the addresses is an important feature of IPv6. There are the rules:

- It is possible to write a single zero instead of 0000.
- First zero may be omitted in each group.

- Several groups of zeros in a row can be replaced with "::" (double colon).
- Zero at the end of groups cannot be omitted.
- Notation "::" can only be used once in each address, otherwise the original address could not be unambiguously determined.

After applying these rules, the mentioned IPv6 address can be written as

2001:718::28F6:19FF:FE00:1984

Prefixes

Prefixes express the membership of a particular network or subnet and use the fact that all interfaces have the same begin of address within a single network. This approach has been used in IPv4 and it is known as **CIDR** (*Classless Inter-Domain Routing*). Prefix notation is taken from CIDR:

IPv6_address/Prefix_length

Prefix_length determines how many bits from the beginning of the address are regarded as a prefix. Example of 64-bit prefix is

2001:718::/64

3.2 Address Space

The huge IPv6 address space was divided into several groups (types of addresses) and each group associates addresses with a common characteristic. Addresses can be assigned to individual groups based on the prefix. Basic classification of addresses is in the following table.

The Basic Classification of Addresses

IPv6 notation	Address type
::/128	Unspecified Address
::1/128	Loopback
FC00::/7	Unique-Local Unicast Addresses
FE80::/10	Link-Local Unicast Addresses
FF00::/8	Multicast Addresses
Everything else	Global Unicast Addresses

Global Unicast addresses occupy the vast of majority of address space. These addresses are taken only from the prefix 2000::/3 today. Other prefixes are reserved for future use.

Anycast addresses are not taken from its own address space. They are taken from the same address space as well as Global Unicast addresses and are not syntactically distinguishable from unicast addresses.

Several small areas of address space were assigned a special meaning. The whole prefix ::/8 that is reserved for those areas is declared as unassigned. However, some addresses were taken from this prefix. They are ::0 and ::1. ::0 address is used when no IPv6 address is assigned to the interface. This address is called Unspecified. ::1 address is used as Loopback. This address is equivalent of 127.0.0.1 for IPv4.

Another group identifies the addresses with a limited range:

- Link-Local Unicast addresses – they were established for purpose of communication within a single link. These addresses start with the prefix FE80::/10 and they are mandatory for each interface.

- Site-Local Unicast addresses – they were originally designed to be used for addressing inside of a site (LAN) without the need for a global prefix. Site-local addresses are now deprecated.

- Unique-Local Unicast addresses –they have replaced the deprecated Site-Local Unicast addresses. These addresses start with the prefix FC00:/7 and they have the same meaning as the private IPv4 addresses.

3.3 Scope of IPv6 Addresses

Another new concept of IPv6 is the scope of addresses. This concept defines the network topology in which the address is unique. It essentially replaces TTL. RFC 4007 defines the scope of addresses.

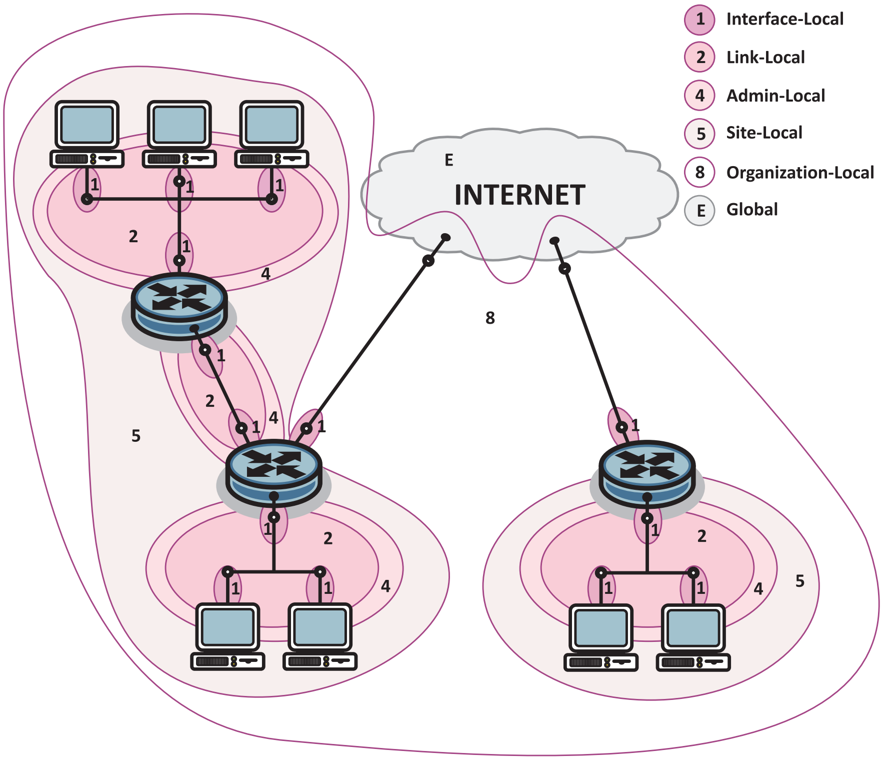
Available scopes depend on the type of addresses. Multicast addresses have the finest classification of scope. There are the following levels:

- Interface-Local scope (1) – this scope spans only a single interface on a node and is useful only for loopback transmission of multicast.
- Link-Local scope (2) – this scope spans the same topological region as the corresponding unicast scope.
- Admin-Local scope (4) – it is the smallest scope that must be administratively configured, i.e., not automatically derived from physical connectivity or other, non-multicast-related configuration.
- Site-Local scope (5) – it is intended to span a single site.
- Organization-Local scope (8) – it is intended to span multiple sites belonging to a single organization.
- Global scope (E)

There are two levels for Global Unicast and Anycast addresses:

- Local scope – it is intended to span a single link.
- Global scope

The zone concept is closely related to the scope of addresses. The Zone defines the part of network that corresponds to the scope of the address in which the address is unique. The boundaries of zones pass through devices, not link, and the whole zone is always included in the superior zone of greater scope. Zones of the same scope cannot overlap each other and are either identical or mutually separated. The zone must be continuous with regard to the forwarding; otherwise the datagram would be able to leave the zone during its transfer.



Example of IPv6 Addresses Scope

The individual zones are necessary to distinguish on the device. For this purpose, the Zone Index was introduced. It consists of Scope Zone that is derived from its own address whose notation is *address%zone*, and Sequence Number. The individual Zone Indices are assigned to each device internally and are not mutually synchronized with its neighbors within the same zone. The Zone Indices are typically used to identify zones in the routing tables within a single device.

The entry of IPv6 address with Zone Index FF02::1%1 can be an example. It is the multicast address for all nodes on the link.

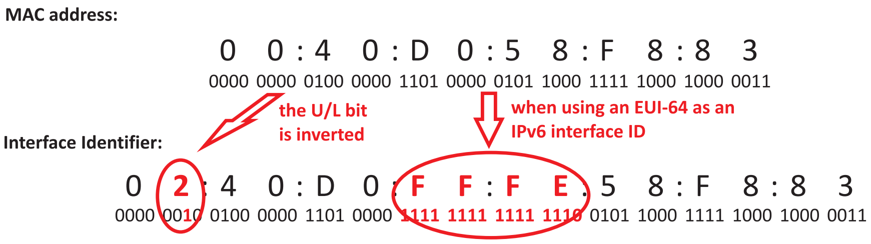
RFC 4007 also provides the implicitly zone (value 0) that is inserted if the address does not contain Zone Index. The Global Unicast addresses for which there is only one zone can be an example. However, this zone is not explicitly noted.

3.4 Interface Identifier

Before we discuss the individual addresses, it is necessary to explain how the Interface Identifier (Interface ID) is generated.

Each network interface generates its own Interface ID from IEEE EUI-64 standard. The Interface ID is subsequently taken to IPv6 address with a little modification. The penultimate bit (U/L) in the highest octet of identifier is inverted. This bit servers as a flag of globally unique address. The change is in order to facilitate the generation of Interface ID. We can consider a serial link whose Interface ID is 200:0:0:1 in the case of using the non-modified EUI-64 standard as an example. After modification, the Interface ID of serial link is 1.

In the case of Ethernet or wireless networks, the Interface ID is generated from globally unique **MAC** (*Media Address Control*) address. The procedure of generation is simple. 16 bits with the value FFFE (in hexadecimal system) are inserted between the third and fourth octet of MAC address and the flag of globally unique address is inverted according to the modified EUI-64 standard. Interface ID 0240:d0FF:FE58:F883 that is generated from the MAC address 00:40:d0:58:F8:83 is an example.



Generation of Interface ID from MAC Address

There is the unique interface identification and thus identification of the user's computer, due to generating the Interface ID according to the modified EUI-64 standard. The unambiguous identification of the interface can be undesirable because of safety communication. Therefore, the new mechanisms were defined that are based on a random generation of Interface ID. RFC 4941 defines these mechanisms.

The interface should have a fixed Interface ID that is used for establishing of the incoming connection, and random Interface ID that is used for establishing of the soutgoing connection to other computers. The lifetime of random Interface ID can be several hours or days. It depends on operating system. The incoming connection is always established using **DNS** (*Domain Name Server*), in which the entry with fixed Interface ID is stored. The random Interface ID must not be stored in the DNS.

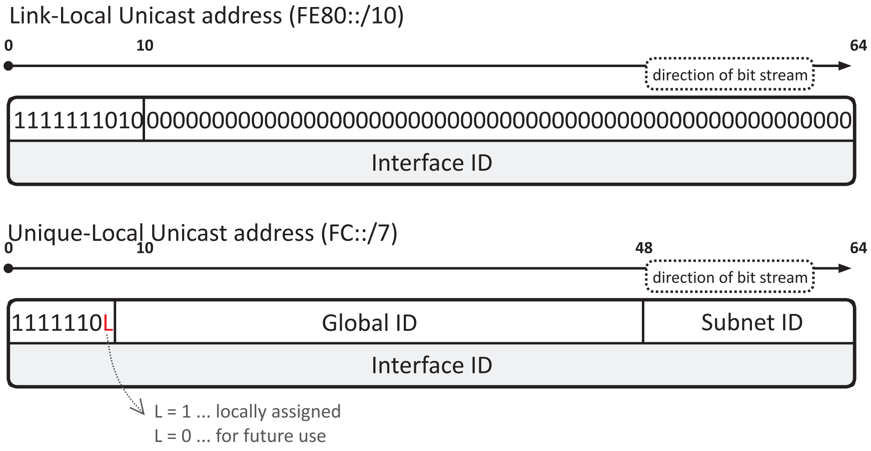
Operating systems based on Linux kernel primarily use the modified EUI-64 standard to generate Interface ID. However, generating a random Interface ID can be enabled using the kernel parameter. For Microsoft operating systems, it is different. In the case of Microsoft Windows XP, Interface ID is generated according to the modified EUI-64 standard. In the case of Microsoft Windows 7, Interface ID is primarily generated according to RFC 4941. The random generation of Interface ID can be disabled using the command line.

3.5 Local Addresses

Local addresses are similar to private IPv4 addresses and are used on a single link. These addresses are not forwarded in the global Internet and registered or centrally coordinated. They are used in combination with NAT today. The local addresses include:

- Link-Local Unicast address (FE80::/10)
- Unique-Local Unicast address (FC::/7)

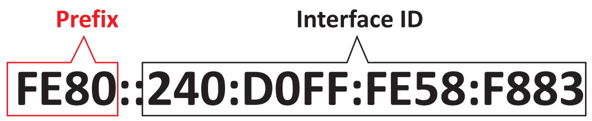
Local addresses have the following format:



Local Addresses

Link-Local Unicast Addresses

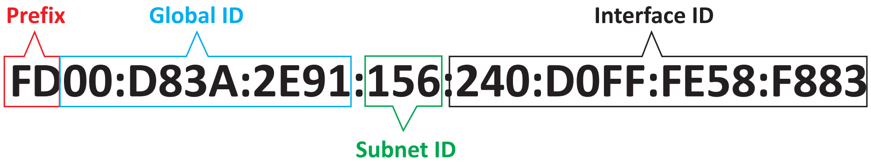
Link-Local addresses are designed to be used for addressing on a single link for purposes such as automatic address configuration, neighbor discovery, or when no routers are present. Routers must not forward any datagrams with Link-Local source or destination addresses to other links. RFC 1918 defines the Link-Local Unicast addresses.



Example of Link-Local Address

Unique-Local Unicast Addresses

Unique-Local addresses are designed to be used for the case when there are more networks (subnets) that the administrator considers as a single network and where he also wants to use both Link-Local addresses and these addresses. These subnets are usually connected using the core networks and the use of Unique-Local addresses does not cause the problems when the data are transferred over these networks. It is because each subnet has its own prefix and thus different local addresses. The probability that the two subnets choose the same Unique-Local addresses is approximately 10^{-12}.



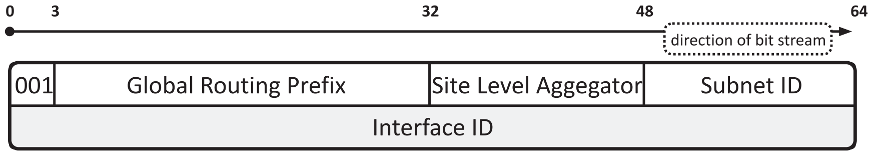
Example of Unique-Local Address

RFC 4193 defines the Unique-Local Unicast addresses. These addresses start with the prefix FC::/7. The L flag that is followed the prefix identifies whether the address is generated locally or otherwise. Currently, the Unique-Local addresses are generated only locally. It means that the L flag is set to 1 and therefore the all Unique-Local addresses start with the prefix FD::/8 today.

The next 40 bits contain a global identifier that is generated randomly. RFC 4193 recommends generating this identifier from the current time, node address and SHA-1 algorithm. The prefix FC::/7 together with a global identifier create a prefix with a length of 48 bits. 16-bit Subnet ID and 64-bit Interface ID that is generated according to the modified EUI-64 standard follow the 48-bit prefix.

3.6 Global Unicast Addresses

Global Unicast addresses identify their users on the Internet. They are globally unique addresses that can be recognized by the first three bits in the prefix. RFC 3487 defines the format of Global Unicast addresses. Global Unicast addresses have the following format:



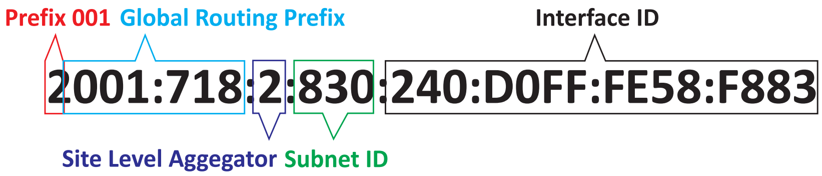
Global Unicast IPv6 Address Format

Description of individual fields of Global Unicast address:

- Global Routing Prefix – it is assigned to a site. Tier 1 **ISP** (*Internet Service Provider*) usually provides this prefix. This prefix together with the prefix 001 is known as Private Topology.

- Subnet ID – it is an identifier of a link within the site. It is known as Local Topology. 16 bits of the Subnet ID allow addressing up to 65 535 subnets.

- Interface ID – it is used to identify the interface within the subnet. 64 bits of Interface ID allow addressing up to 18x10^{18} interfaces.



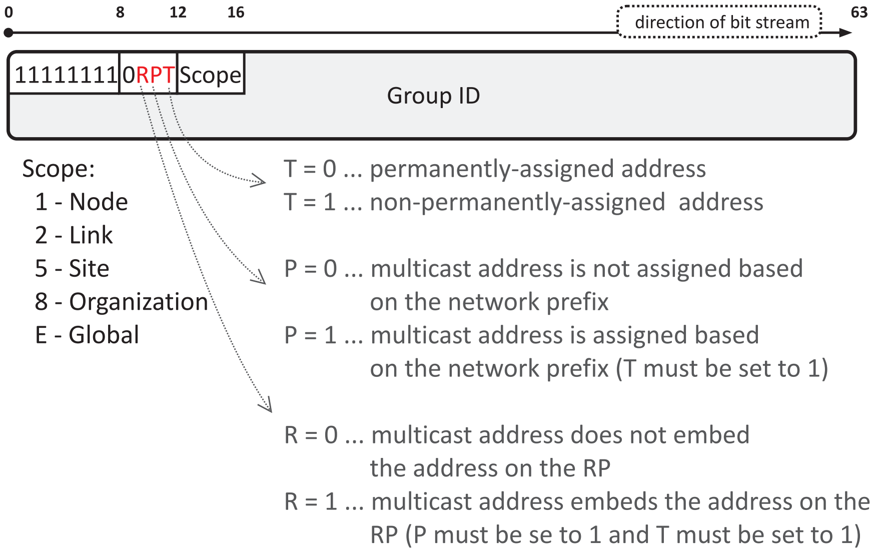
Example of Global Unicast IPv6 Address

Global Unicast addresses are aggregable. It means that the addresses are clustered into groups according to distance from which we look at them. The aggregation of addresses significantly reduces the number of entries in routing tables, which affects the speed of forwarding on the core routers. The fineness classification of routing information decreases with increasing distance. It means that the datagram is forwarded according to the initial bits of destination address (prefix) in the first stage and the examination of destination address becomes more accurate closer to the target node. It is called Hierarchical routing.

3.7 Multicast Addresses

An IPv6 multicast address is an identifier for a group of interfaces (typically on different nodes). An interface may belong to any number of multicast groups. If the data is sent to this address, it has to be delivered to all interfaces in the multicast group. The distribution of video and audio signals in real time (videoconferences, radio and television broadcasting, etc.) is a typical example of the use of multicast addresses.

The prefix of each multicast address starts with the value FF in hexadecimal system (binary 11111111). Therefore, these addresses are easily recognizable. Multicast addresses have the following format:


0
8
12
16
direction of bit stream
63
11111111
0
RPT
Scope
Group ID
Scope:
1 - Node
2 - Link
5 - Site
8 - Organization
E - Global
T = 0 ... permanently-assigned address
T = 1 ... non-permanently-assigned address
P = 0 ... multicast address is not assigned based on the network prefix
P = 1 ... multicast address is assigned based on the network prefix (T must be set to 1)
R = 0 ... multicast address does not embed the address on the RP
R = 1 ... multicast address embeds the address on the RP (P must be se to 1 and T must be set to 1)

Multicast Address Format

Description of individual fields of multicast address:

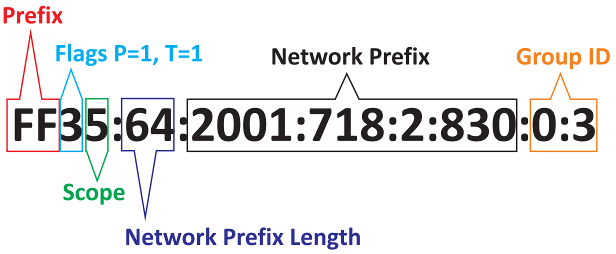
- Option – it consist of four flags. The first flag is reserved for future use. Other flags are described below.

- R flag – it uses for multicast groups that are related to **PIM-SM** (*Protocol Independent Multicast – Sparse Mode*) routing protocol. This flag identifies multicast addresses that include **RP** (*Rendezvous Point*).

- P flag – it was defined in order to establish multicast addresses that are based on Global Unicast addresses. These addresses are called Unicast-Prefix-based IPv6 Multicast Addresses and they are used to generate unique multicast addresses, without determining whether the multicast address already exists on a site. It is achieved by the Global Unicast address prefix of a local site is included to multicast address.

- Flag T – it indicates whether the multicast address is assigned permanently (value 0) or temporarily (value 1). The permanently multicast addresses are assigned by IANA (Internet Assigned Numbers Authority) and the temporarily multicast addresses can be assigned by applications as needed.

- Scope – it indicates how the individual members of multicast group can be far from each other. It is a 4-bit field that allows defining up to 16 levels of address scope. The concept of address scope has already been discussed in section Scope of IPv6 Addresses.
- Group ID – this field identifies a multicast group that the data should be delivered.



Example of Unicast-Prefix-based IPv6 Multicast Address

Multicast address must never be used as a source address of the sender in an IPv6 datagram and it must not also be included in the Routing header.

The topic related to the multicast addresses is extensive. Therefore, we only discuss the individual types of multicast addresses. These are:

- Unicast-Prefix-based IPv6 Multicast addresses (RFC 3306)
- Link-Scoped IPv6 Multicast addresses (RFC 4489)
- Multicast addresses for **SSM** (*Source Specific Multicast*) (RFC 3569)
- Multicast addresses with Embedded-RP address (RFC 3956)

Pre-Defined Multicast Addresses

There are a number of multicast addresses with a special significance. These addresses are defined in RFC 4291. Pre-Defined Multicast addresses essentially replace the broadcast addresses. There are:

- All IPv6 nodes on the same interface (FF01::1)
- All IPv6 nodes on the same link (FF02::1)
- All IPv6 routers on the same interface (FF01::2)
- All IPv6 routers on the same link (FF02::2)
- All IPv6 routers on the same site (FF05::2)

- Solicited-Node multicast address. These addresses have the following notation FF02:0:0:0:0:1:FFxx:xxxx, where the last three octets are taken from a search address. These addresses are used for Neighbor Discovery which is equivalent of **ARP** (*Address Resolution Protocol*) protocol for IPv4.

Pre-Defined Multicast addresses are used for some internal IPv6 mechanisms.

RFC 2375 defines other multicast groups that are used for various network protocols and services. For example, the multicast addresses FF0x::101 are used for **NTP** (*Network Time Protocol*).

3.8 Anycast Addresses

An IPv6 anycast address is an address that is assigned to more than one interface (typically belonging to different nodes), with the property that a datagram sent to an anycast address is routed to the "nearest" interface having that address, according to the routing protocols' measure of distance.

+ All anycast addresses can be forwarded using standard methods. It is enough that the computer, which joins the anycast address, announces this fact to any router. This router already arranges the distribution of the information.

− Anycast addresses are allocated from the unicast address space, using any of the defined unicast address formats. Thus, anycast addresses are syntactically indistinguishable from unicast addresses. When a unicast address is assigned to more than one interface, thus turning it into an anycast address, the nodes to which the address is assigned must be explicitly configured to know that it is an anycast address.

For any assigned anycast address, there is a longest prefix P of that address that identifies the topological region in which all interfaces belonging to that anycast address reside. Within the region identified by P, the anycast address must be maintained as a separate entry in the routing system (commonly referred to as a "host route"). Outside the region identified by P, the anycast address may be aggregated into the routing entry for prefix P.

Note that in the worst case, the prefix P of an anycast set may be the null prefix, i.e., the members of the set may have no topological locality. In that case, the anycast address must be maintained as a separate routing entry throughout the entire Internet, which presents a severe scaling limit on how many such "global" anycast sets may be supported.

Another restriction of use anycast addresses is a dynamic routing and routing policy of tier 1 ISP. Dynamic routing causes problems for protocols, such as TCP protocol, and routing policy of tier 1 ISP refuse too long prefixes and thus the anycast addresses with too long prefix P. In addition, the different parts of the Internet core network are managed by different organizations whose routing policy we cannot control.

Therefore, it is expected that support for global anycast sets may be unavailable or very restricted.

One expected use of anycast addresses is to identify the set of routers belonging to an organization providing Internet service. Such addresses could be used as intermediate addresses in an IPv6 Routing header, to cause a packet to be delivered via a particular service provider or sequence of service providers.

Some other possible uses are to identify the set of routers attached to a particular subnet, or the set of routers providing entry into a particular routing domain.

Currently, it is expected that global anycast sets will mainly be used for DNS servers. On the other hand, anycast addresses can be a more effective way of addressing for the smaller sites that are managed by a single ISP.

3.9 A Node's Required Addresses

In the case of IPv4, the interface has almost one address. In IPv6, this fact is changed. Each interface has to have more addresses. Therefore, there are a set with a minimum number of addresses that have to be assigned to each interface.

In the case of a computer, an interface has to have the following addresses:

- Loopback address
- Link-Local address
- All Global Unicast and Anycast addresses that are assigned to the interface
- Multicast addresses for all IPv6 nodes
- Solicited-Node multicast address for each assigned Global Unicast and Anycast address
- All multicast addresses, which the interface is member

In the case of a router, an interface has to have the same addresses as a computer and the following:

- Anycast address for routers on the same subnet
- Multicast addresses for all routers

3.10 Default Address Selection

With several different addresses assigned to one interface, there is a problem what address does a host use for communication? For this purpose, the algorithm for selection of addresses was defined. RFC 3484 describes this algorithm.

Candidate Addresses

The candidate addresses are the fundamental block of Address Selection algorithm. The list of candidate addresses for destination addresses is usually obtained through a DNS query (translation of the domain name to IPv6 address). The list is sorted from the most suitable to the least suitable address from which the most suitable address is selected. In the case that the application has a destination IPv6 address, there is no choice of destination address.

It is necessary to select the most suitable source address to candidate destination address. The list of candidate source addresses contains all global unicast addresses that are assigned to interface through which data will be sent to a target node. It means that each destination address may have different lists of candidate source addresses. In the case of routers, the list of candidate source addresses may contain global unicast addresses of other interfaces on which data is forwarded.

The most suitable source/destination address is selected from the list of candidate source/destination addresses based on rules that are mentioned in the subsection Source Address Selection.

In the case that the communication fails, the address selection can be repeated.

Policy Table

The policy table is used to determine local precedence of address. Individual entries contain Prefix, Precedence and Label, where Precedence is used for sorting destination addresses and Label determines whether the selected pair of addresses fits together. In the case of identical labels both addresses, the pair of addresses is preferred. Precedence and Label are determined by the longest matching prefix.

If the administrator does not configure the policy table, the default policy table is used.

Default Policy Table

Prefix	Precedence	Label
::1/128	50	0
::/0	40	1
2002::/16	30	2
::/96	20	3
::FFFF:0:0/96	10	4

Source Address Selection

The Address Selection algorithm has two sets of rules. The first set describes source address selection. The Source Address Selection algorithm produces as output a single source address for use with a given destination address. The algorithm is demonstrated through an example, when there are two source addresses SA and SB for target node. The rules are applied gradually and when the decision is done, the others are ignored. If the decision is not done, implementations should provide a source address selection. The order of rules is:

1. Prefer same address.
2. Prefer appropriate scope – if *Scope(SA) < Scope(SB)*: If *Scope(SA) < Scope(destination address)*, then prefer SB and otherwise prefer SA.
3. Avoid deprecated addresses.
4. Prefer home addresses.
5. Prefer outgoing interface – if SA is assigned to the interface that will be used to send to destination address and SB is assigned to a different interface, then prefer SA.
6. Prefer matching label – if *Label(SA) = Label(destination address)* and *Label(SB) ≠ Label(destination address)*, then prefer SA.
7. Prefer public addresses.
8. Use longest matching prefix.

The algorithm is a symmetric, so each rule is performed once more with swapped roles of source addresses. The most suitable source address for target node is determined by applying the rules.

Destination Address Selection

The suitability of source and destination addresses, among other things, is considered when the destination address selection is done. Therefore, the most suitable source address is determined for each destination address. Then there is the arrangement of destination addresses in the candidate list according to the following rules:

1. Avoid unusable destinations – if destination address DB is known to be unreachable or if *Source(DB)* is undefined, then prefer destination address DA.
2. Prefer matching scope – if *Scope(DA) = Scope(Source(DA))* and *Scope(DB) ≠ Scope(Source(DB))*, then prefer DA.
3. Avoid deprecated addresses.
4. Prefer home addresses.

5. Prefer matching label – if *Label(Source(DA))* = *Label(DA)* and *Label(Source(DB))* ≠ *Label(DB)*, then prefer DA.

6. Prefer higher precedence.

7. Prefer native transport – if DA is reached via an encapsulating transition mechanism (e.g., IPv6 in IPv4) and DB is not, then prefer DB.

8. Prefer smaller scope – if *Scope(DA)* < *Scope(DB)*, then prefer DA.

9. Use longest matching prefix.

10. Otherwise, leave the order unchanged – if DA preceded DB in the original list, prefer DA. Otherwise prefer DB.

The computer has one network interface to which the following addresses were assigned:

1. FE80::02B5:32FF:FE01:1984 (Link-Local address)

2. 2001:718:10:1:02B5:32FF:FE01:1984 (Global Unicast address)

3. 2001:: 02B5:32FF:FE01:1984 (Global Unicast address)

The computer has to send data to the address 2001:718:20:56:E859:A1FF:FE82:AA01 and needs to select the most suitable source address. The default policy table is used during the address selection. The precedence value of 40 and label value of 1 are assigned to all addresses.

Link-Local address is not selected because of rule 2. Its scope is smaller than the global scope of the target node. The most suitable source address is selected using rule 8. The second address is selected as source address because it has the same longest matching prefix 2001:718::/32 as the target node.

Destination address selection is based on the candidate list which the DNS server sent in response to a request for translating a domain name. Candidate list contains the following addresses:

1. 2001:DB8:A29C:5::02 (source address 2001:: 02B5:32FF:FE01:1984)

2. 2001:718::B5:18FF:FE09:1 (source address 2001:718:10:1:02B5:32FF:FE01:1984)

The list also contains a suitable source addresses that are selected based on the source address selection algorithm. The precedence value of 40 and label value of 1 are assigned to both addresses. The algorithm decides on the basis of rule 8 when the second address has the same longest matching prefix 2001:718::/32 as the target node. The candidate list for the destination address is sorted as follows:

1. 2001:718::B5:18FF:FE09:1

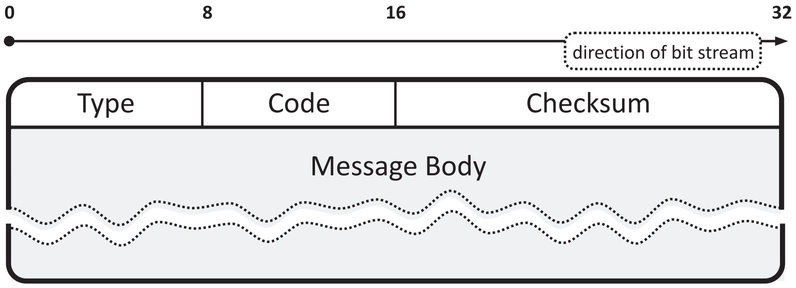
2. 2001:DB8:A29C:5::02

Then address 2001:718:10:1:02B5:32FF:FE01:1984 is selected as the source address and address 2001:718::B5:18FF:FE09:1 is selected as the destination address for the connection.

4 ICMPv6

4.1 ICMPv6 Protocol

ICMPv6 (*Internet Control Message Protocol for the Internet Protocol Version 6*) protocol is part of the Internet protocol family and is defined in RFC 4443. It is used primarily for reporting error states, testing for reachability and exchange of certain operating information. RFC 4443 defines only the fundamental elements such as the message format and individual types of messages that are divided into ICMPv6 error message and ICMPv6 informational message. Individual messages are conveyed using an IPv6 datagram.



Message General Format

There are four ICMPv6 error messages:

- Destination Unreachable Message – it should be generated by a router, or by the IPv6 layer in the originating node, in response to a packet that cannot be delivered to its destination address for reasons other than congestion. (An ICMPv6 message must not be generated if a packet is dropped due to congestion.) Code field specifies the reason in detail why the packet cannot by delivered.

- Packet Too Big Message – it must be sent by a router in response to a packet that it cannot forward because the packet is larger than the MTU of the outgoing link. The information in this message is used as part of PMTU process. Originating a Packet Too Big Message makes an exception to one of the rules as to when to originate an ICMPv6 error message. Unlike other messages, it is sent in response to a packet received with an IPv6 multicast destination address, or with a link-layer multicast or link-layer broadcast address. Actual MTU size is conveyed in a 4-octet field for the checksum.

- Time Exceed Message – if a router receives a packet with a Hop Limit of zero, or if a router decrements a packet's Hop Limit to zero, it must discard the packet and originate an ICMPv6 Time Exceeded message with Code 0 to the source of the packet. This indicates either a routing loop or too small an initial Hop Limit value. An ICMPv6 Time Exceeded message with Code 1 is used to report fragment reassembly timeout.

- Parameter Problem Message – if an IPv6 node processing a packet finds a problem with a field in the IPv6 header or extension headers such that it cannot complete processing the packet, it must discard the packet and should originate an ICMPv6 Parameter Problem message to the packet's source, indicating the type and location of the problem. The pointer identifies the octet of the original packet's header where the error was detected.

There are four ICMPv6 informational messages:

- Echo Request Message and Echo Reply Message – these messages are used by PING program.

- ICMP Node Information Query Message and ICMP Node Information Response – these messages allow getting fundamental information about nodes (node name, IPv4 or IPv6 address, etc.). They are especially used for a network management.

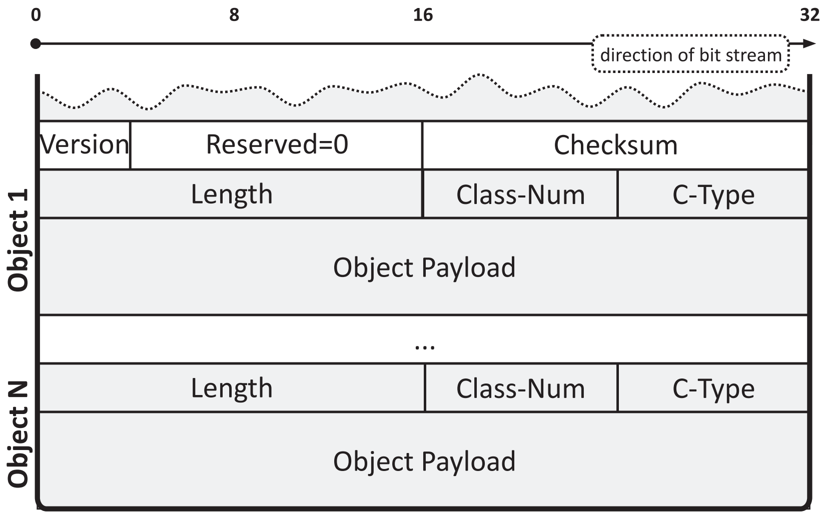
Internal IPv6 mechanisms are complement other types of messages and rules for their generation. For example, they are Router Solicitation, Router Advertisement, Neighbor Solicitation, Neighbor Advertisement, Redirect and messages that are relate to mobility.

ICMPv6 protocol has implemented security mechanisms unlike ICMPv4. In the case of IPv4, the ICMPv4 messages can be misused and thus it can result to limit functionality of the network. In short, the target node could be overload by many ICMP messages and other traffic could not pass through the node. Security mechanisms use following arrangements:

- The minimum interval between messages and the maximum proportion of the total bandwidth that the node generates. It results to guarantee the sufficient bandwidth for the transmission of real data.

- Individual messages can be provided with encryption header. These headers must be checked; otherwise the ICMPv6 messages are dropped. The administrator should be able to set the node to accept only secure messages and others ignore.

RFC 4884 defines the extended version of ICMPv6 protocol that allows inserting additional information in the ICMPv6 message and slightly modifies some existing messages such as Destination Unreachable message and Time Exceed message. ICMPv6 extension header is placed at the end of ICMPv6 message and the ICMPv6 extension data follows this header.



ICMPv6 Extension Header Format

The implementation of ICMPv6 is mandatory in any node that supports IPv6.

5 Neighbor Discovery

5.1 Neighbor Discovery

ND (*Neighbor Discovery*) is similar to the **ARP** (*Address Resolution Protocol*) that uses for determining the link-layer address. ND is more complex mechanism that is defined in RFC 4861 as a fundamental part of IPv6. It primarily serves the following purposes:

- Determining and updating link-layer addresses of nodes on the same link
- Discovering routers
- Redirect
- Discovering prefixes, network parameters and other information for address auto-configuration of nodes
- Maintaining reachability information about the paths to other active neighbor nodes
- Duplicate address detection

ND uses the following types of ICMPv6 messages:

- **NS** (*Neighbor Solicitation*)
- **NA** (*Neighbor Advertisement*)

The **SEND** (*Secure Neighbor Discovery*) was defined to ensure the security of computer systems. RFC 3971 describes the SEND fundamentals. The goal of SEND is to provide a sufficient secure level of exchange of ICMPv6 messages.

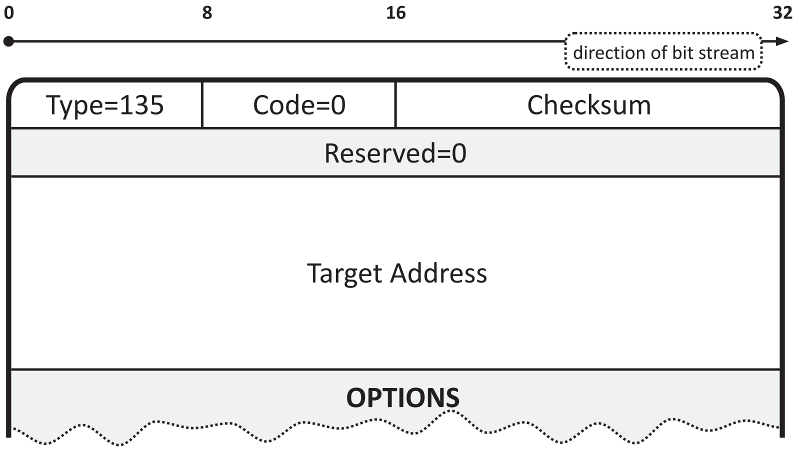
5.2 Link-Layer Address Determining

ND uses the Solicited-Node Multicast address for its activities. This address is based on the prefix FF02:0:0:0:0:1:FF00::/104 and the IPv6 address of a neighbor whose link-layer address is looking for.

Each network interface can be a member of several networks with different prefixes. For this reason, the Solicited-Node Multicast address is generated by takeover the last 24 bits of Interface ID. By this, all IPv6 addresses of the node have the same Solicited-Node Multicast address. To determining the link-layer address of neighbor, a node must send requests to all networks, which is a member. Each node uses the Neighbor Cache internal data structure for preserve the IPv6 addresses with their corresponding link-layer addresses.

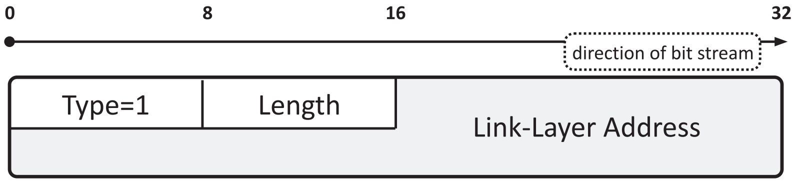
If the node knows the IPv6 address of a neighbor and wants to determine its link-layer address, the determining procedure is as follows:

- The node generates Solicited-Node Multicast address from destination IPv6 address and sends a Neighbor Solicitation message to this address. Optionally, a node provides its link-layer address and link MTU using the Source Link-Layer Address option and MTU option, respectively.

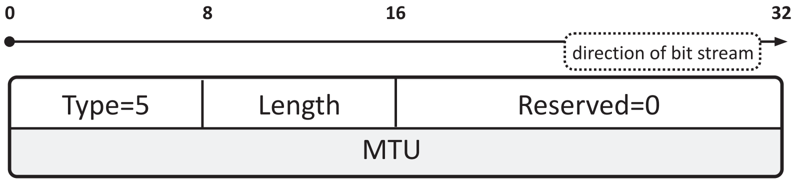


Neighbor Solicitation Message Format

- If a searched neighbor is active, it replies using the Neighbor Advertisement message that contains the Source Link-Layer Address option. This option contains link-layer address of a neighbor. Neighbor can also provide the link MTU.

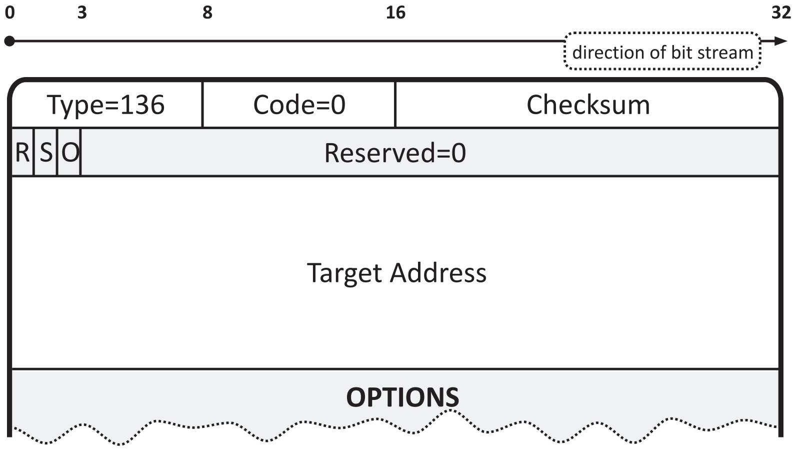


Source/Target Link-Layer Address Option Format



MTU Option Format

A node can send the Neighbor Advertisement message even if its link-layer address is changed. The node announces the fact by sending several Neighbor Advertisement messages to multicast address for all IPv6 nodes on the same link (FF02::1). The nodes that have stored the entry with the IPv6 address in its Neighbor Cache update this entry, others ignore it.



Neighbor Advertisement Message Format

Neighbor Advertisement message contains additional information that are conveyed using the following flags:

- R flag (*Router*) – it indicates that the sender is a router.
- S flag (*Solicited*) – it indicates whether the Neighbor Solicitation message is requested or not.
- O flag (*Override*) – it indicates whether information about link-layer address has to override the existing entry in Neighbor Cache.

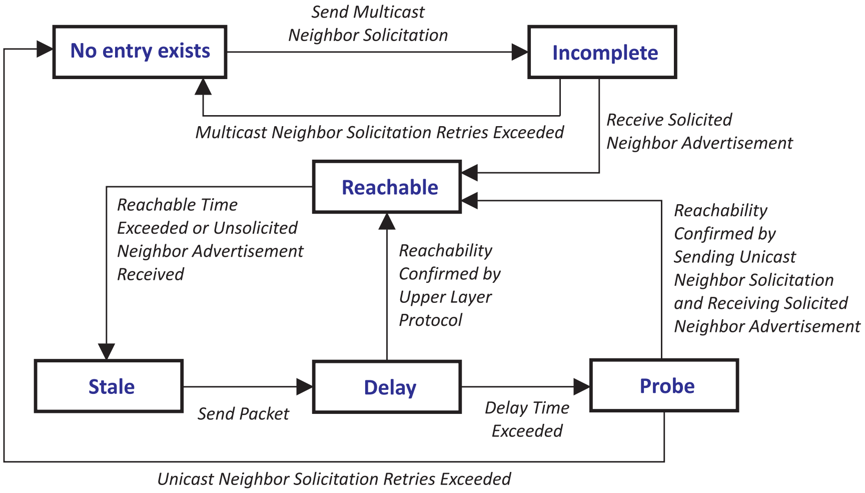
5.3 Neighbor Unreachability Detection

Neighbor Cache update algorithm is not absolutely reliable, so each node actively monitors the states of reachable neighbors with which the node communicates. There are two algorithms for verifying the neighbor reachability. The first algorithm uses the messages of upper-layers, i.e. TCP, and the second algorithm is based on sending Neighbor Solicitation messages and receiving Neighbor Advertisement messages as a response.

The following states that are assigned to individual entries in Neighbor Cache are the fundamental for the second mechanism:

- Incomplete – Address resolution is in progress and the link-layer address of the neighbor has not yet been determined. If the response does not arrive, the entry will be removed from Neighbor Cache.
- Reachable – Roughly speaking, the neighbor is known to have been reachable recently (within tens of seconds ago). This state takes a certain time interval that a default router announces to nodes.
- Stale – The neighbor is no longer known to be reachable but until traffic is sent to the neighbor, no attempt should be made to verify its reachability.
- Delay – The neighbor is no longer known to be reachable, and traffic has recently been sent to the neighbor. Rather than probe the neighbor immediately, however, delay sending probes for a short while in order to give upper-layer protocols a chance to provide reachability confirmation.
- Probe – The neighbor is no longer known to be reachable, and unicast Neighbor Solicitation probes are being sent to verify reachability.

The course of changes in individual states during the Neighbor Unreachability Detection algorithm is depicted in the following diagram.



Neighbor Cache Entry States

5.4 Inverse Neighbor Discovery

IND (*Inverse Neighbor Discovery*) is defined in RFC 3122. IND allows a node that knows the link-layer address of a directly connected remote node to learn the IPv6 addresses of that node.

The principle of IND is a very simple. A node sends an Inverse Neighbor Discovery Solicitation message to request an IPv6 address corresponding to a link-layer address of the target node while also providing its own link-layer address to the target. The sender must send the following options in the Solicitation message:

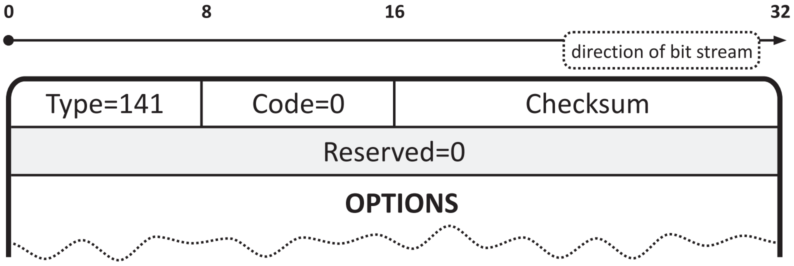
- Source Link-Layer Address option – it contains the link-layer address of the sender.

- Target Link-Layer Address option – it contains the link-layer address of the target node.

These options are optional:

- Source Address List – the list of one or more IPv6 addresses of the interface identified by the Source Link-Layer Address. Code 9 identifies this option.

- MTU – the MTU configured for this link.

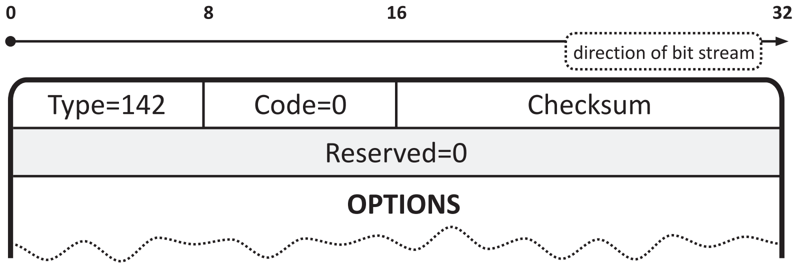


Inverse Neighbor Discovery Solicitation Message Format

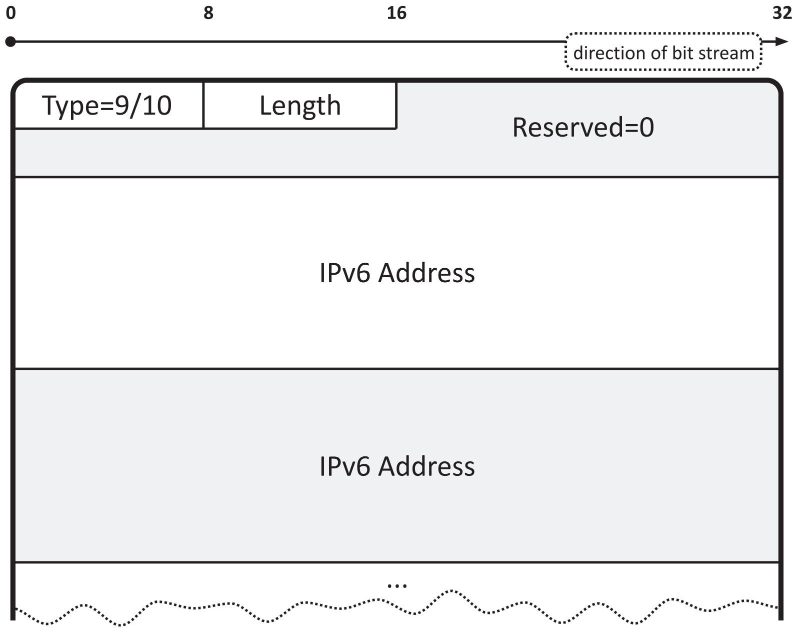
A solicited node sends Inverse Neighbor Discovery Advertisements message in response to IND Solicitation messages to a link-layer address of the target node. The sender node must send the following options in Advertisement message:

- Target Link-Layer Address option – the link-layer address of the target, that is, the sender of the advertisement.

- Target Address List – the list of one or more IPv6 addresses of the interface identified by the Target Link-Layer Address in the Inverse Neighbor Discovery Solicitation message that prompted this advertisement. Code 10 identifies this option.



Inverse Neighbor Discovery Advertisement Message Format



Source/Target Address List Format

The sender of the Solicitation message stores the received information in its Neighbor Cache.

The sender node may send the MTU option in the Advertisement message. The Source/Target Link-Layer Address and MTU options are depicted in Link-Layer Address Determining section.

6 IPv6 Address Auto-configuration

6.1 IPv6 Address Auto-configuration

IPv6 Address Auto-configuration operates on the principle of plug and play. It means that a node generates an IPv6 address itself and finds out necessary parameters from routers for proper communication with the outside IPv6 world after connection to the IPv6 network.

There are two types of IPv6 Address Auto-configuration:

- Stateless auto-configuration (**SLAAC**, *Stateless Address Auto-configuration*) – it represents a completely new way of address auto-configuration. This reflects the fact that each local site has a default router, which knows all the necessary configuration parameters (prefixes, etc.) for communication with the outside IPv6 world. This "smart" router advertises the parameters at certain time intervals to all sites to which it is connected. So the node just listens on an interface connected to the site or requests a "smart" router to send the parameters. SLAAC is defined in RFC 4862.

- Stateful auto-configuration (**DHCPv6**, *Dynamic Host Configuration Protocol for IPv6*) – DHCPv6 works in a client-server model. The fundamental of DHCPv6 is a server that maintains the configuration parameters. DHCPv6 server tells the configuration parameters to clients on a request. DHCPv6 is defined in RFC 3315.

Most of the current IPv6 implementation allows a combination of both types of Address Auto-configuration.

7 Stateless Auto-configuration

7.1 ICMPv6 Messages

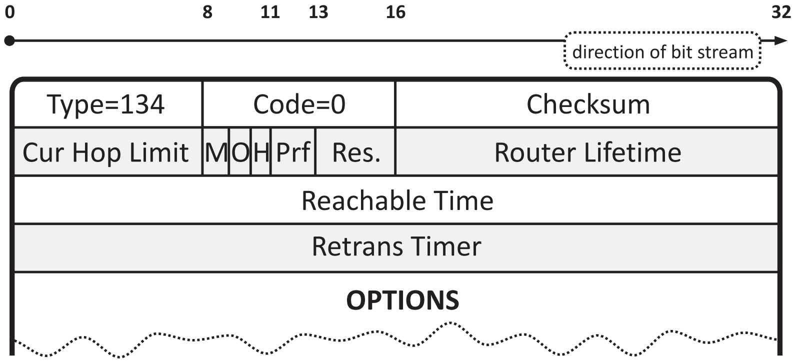
Stateless auto-configuration uses the following ICMPv6 messages for its activities:

- **RA** (*Router Advertisement*)
- **RS** (*Router Solicitation*)
- Redirect

Before we discuss the principle of SLAAC, it is necessary to explain a Router Advertisement message.

Router Advertisement

The fundamental block of SLAAC is a Router Advertisement message. RA message is used for sending all necessary configuration parameters to nodes. Routers send out RA messages periodically, or in response to RS message.



Router Advertisement Message Format

Description of individual fields of RA message:

- Cur Hop Limit – this field indicates the default value that should be placed in the Hop Limit field of IPv6 header for outgoing datagram. A value of zero means unspecified (by this router).
- **M** flag (*Managed address configuration*) – when set, it indicates that addresses are available via DHCPv6.
- **O** flag (*Other configuration*) – when set, it indicates that other configuration information is available via DHCPv6. Examples of such information are DNS-related information or information on other servers within the network.

The significance of possible combination of M and O flags is shown in the following table.

The Combination of M and O Flags

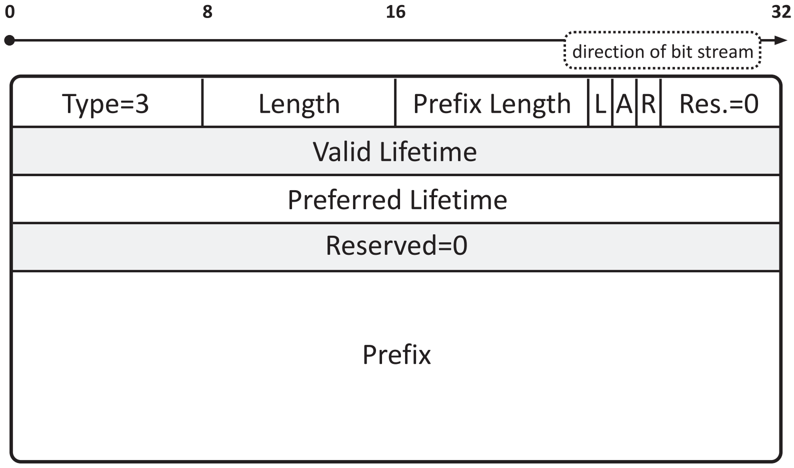
M flag	O flag	Significance
1	-	DHCPv6 assigns all parameters
0	1	Combination of SLAAC (IPv6 address, prefix, routing) and DHCPv6 (other configuration parameters)
0	0	DHCPv6 is not available

- **H** flag (*Home agent*) – it indicates that the default router works also as home agent.
- Prf *(Default Router Preference*) – it allows to distinguish the default router preferences. If the Router Lifetime is zero, the preference value must be set to 00 by the sender and must be ignored by the receiver. There are four levels:

Default Router Preference

Prf	Significance
01	High
00	Medium (default)
11	Low
10	Reserved (must not be sent)

- Router Lifetime – The lifetime associated with the default router in units of seconds. A Lifetime of 0 indicates that the router is not a default router and should not appear on the default router list.
- Reachable Time – The time, in milliseconds, that a node assumes a neighbor is reachable after having received a reachability confirmation. A value of zero means unspecified (by this router).
- Retrans Timer – the time, in milliseconds, between retransmitted Neighbor Solicitation messages. A value of zero means unspecified (by this router).
- Options – it allows adding the information on a link-layer address, link MTU and it is especially used for inserting information on prefixes. Prefix Information option is added for each prefix.



Prefix Information Option Format

Description of individual fields of Prefix Information option:

- Prefix Length – it expresses the size of prefix in bits.
- **L** flag (*on-Link*) – when set, indicates that this prefix can be used for on-link determination. When not set the advertisement makes no statement about on-link or off-link properties of the prefix.
- **A** flag (*Autonomous address-configuration*) – when set indicates that this prefix can be used for stateless address configuration. When A flag is set to 0, the SLAAC is disabled. When A and L flag are set, the node can use both of address auto-configuration.
- **R** flag (*Router address*) – it indicates that the Prefix field contains a Global Unicast address of router. This flag was defined for the mobility. When R flag is set, the prefix is only used for address auto-configuration. Interface ID is ignored.
- Valid Lifetime – The length of time in seconds (relative to the time the packet is sent) that the prefix is valid for the purpose of on-link determination. A value of all one bits (0xffffffff) represents infinity.
- Preferred Lifetime – The length of time in seconds (relative to the time the packet is sent) that addresses generated from the prefix via stateless address auto-configuration remain preferred. A node can use preferred address without any restrictions. The address is marked as deprecated after the expiration of the Preferred Lifetime. It means that the address is valid, but should not already be used for establishing a new connection. The address is not be used after the expiration of the Valid Lifetime. It becomes invalid. A value of all one bits (0xffffffff) represents infinity.

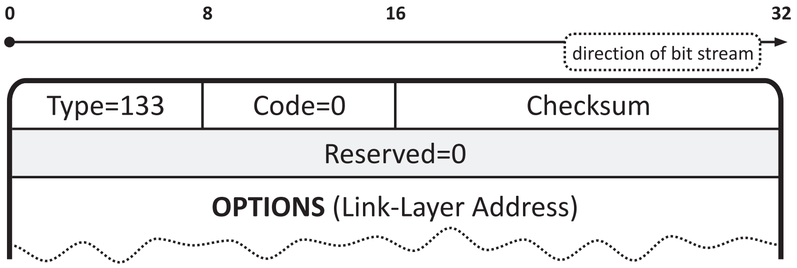
7.2 Principle of Stateless Address Auto-configuration

The stateless mechanism allows a node to generate its own addresses using a combination of locally available information and information advertised by routers. Routers advertise prefixes that identify the subnet(s) associated with a link, while nodes generate an Interface ID that uniquely identifies an interface on a subnet. An address is formed by combining the two. In the absence of routers, a node can only generate link-local addresses. However, link-local addresses are sufficient for allowing communication among nodes attached to the same link.

So the first node generates its own link-local address. Before the link-local address can be assigned to an interface and used, however, a node must attempt to verify that this "tentative" address is not already in use by another node on the link. Specifically, it sends a NS message containing the tentative address as the target. If another node is already using that address, it will return a NA message saying so. If a node determines that its tentative link-local address is not unique, auto-configuration stops and manual configuration of the interface is required. This mechanism is called Duplicate Address Detection. Duplication of addresses is also tested in the cases of manual and stateful address auto-configuration.

Once a node ascertains that its tentative link-local address is unique, it assigns the address to the interface. At this point, the node has IP-level connectivity with neighboring nodes. The next phase of auto-configuration involves obtaining a RA messages or determining that no routers are present. If routers are present, they will send RA messages that specify what sort of auto-configuration a node can do. Note that the DHCPv6 service for address auto-configuration may still be available even if no routers are present. It should be noted that a node may use both stateless address auto-configuration and DHCPv6 simultaneously. The A flag indicates whether or not the option even applies to stateless auto-configuration. If it does, additional option fields contain a subnet prefix, together with lifetime values, indicating how long addresses created from the prefix remain preferred and valid.

Routers send RA messages periodically, but the delay between successive advertisements will generally be longer than a node performing auto-configuration will want to wait. To obtain an advertisement quickly, a node sends one or more RS messages to the all-routers multicast group.



Router Solicitation Message Format

Because routers generate RA messages periodically, nodes will continually receive new advertisements. Nodes process the information contained in each advertisement as described above, adding to and refreshing information received in previous advertisements.

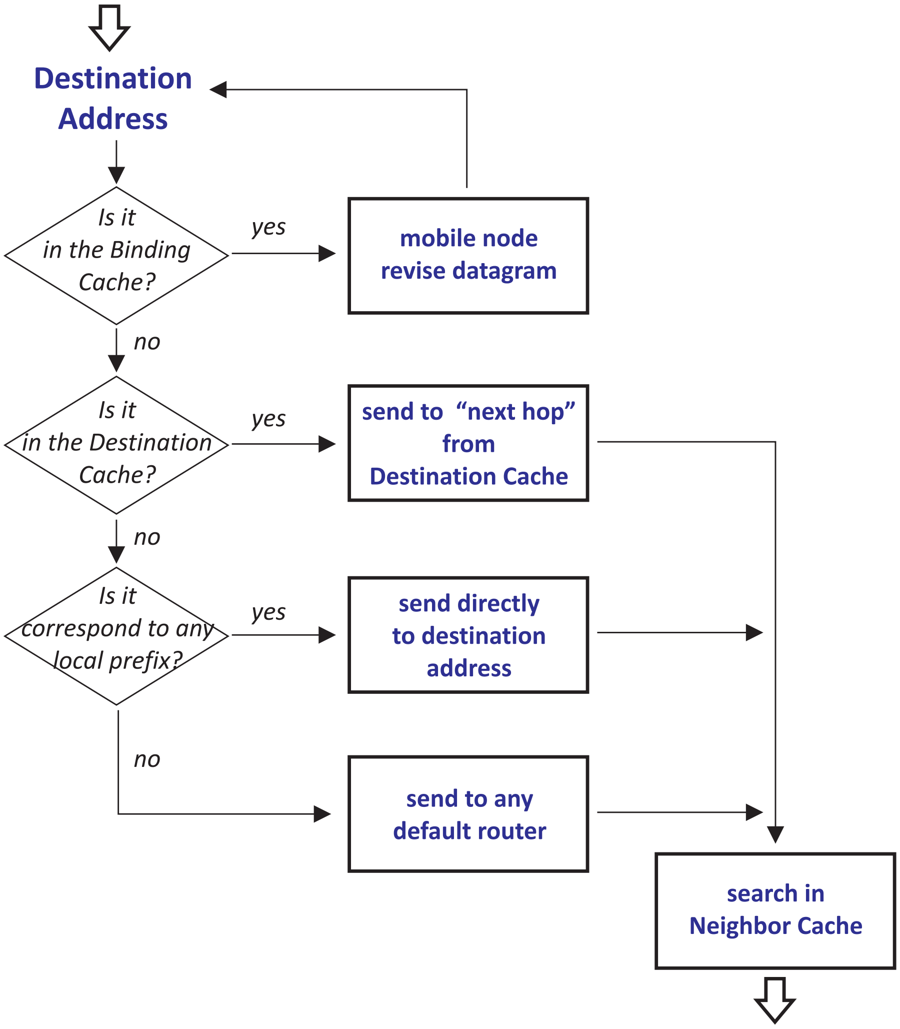
7.3 Routing configuration and DNS

Address auto-configuration allows also determining the routing information. It uses the following fundamental data structures that can be implemented as a single entity using the routing table:

- Destination Cache – it contains the routing information for the concrete addresses. Each entry contains the address of a next router (next hop address).
- Prefix List – it is used to determine whether target node is on the link or not.
- Default Router List – it contains a list of all available default routers.

In the case of support for mobility, the Binding Cache is added. It indicates that the target node communicates at temporarily assigned address. It means that the Routing header will be added to the outgoing datagram and its destination address will be changed.

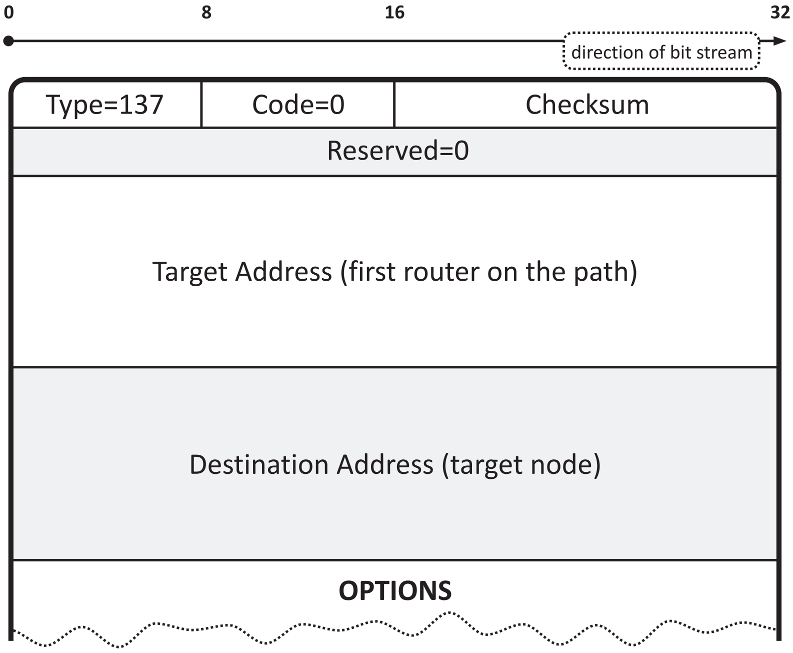
The procedure for sending a datagram is depicted in the following diagram.


Destination Address
Is it in the Binding Cache?
yes
mobile node revise datagram
no
Is it in the Destination Cache?
yes
send to "next hop" from Destination Cache
no
Is it correspond to any local prefix?
yes
send directly to destination address
no
send to any default router
search in Neighbor Cache

The Procedure for Sending a Datagram

Redirect

Routers send Redirect message to inform a host of a better first-hop node on the path to a target node. Hosts can be redirected to a better first-hop router but can also be informed by a redirect message that the target node is in fact a neighbor. Redirect message contains the destination address of datagram and the address of router or target node itself, to which the datagrams are sent to the target node. This address is written in the Target Address field. It is also possible to include the link-layer address of the router and the datagram header that caused a redirection. This header must be limited in size not to exceed a total length of the message that must be smaller than 1280 octets.


0
8
16
32
direction of bit stream
Type=137
Code=0
Checksum
Reserved=0
Target Address (first router on the path)
Destination Address (target node)
OPTIONS

Redirect Message Format

DNS configuration

The fundamental mechanisms of stateless address auto-configuration do not support the configuration of DNS server addresses, which is a disadvantage because the form of IPv6 addresses. Therefore, it was defined the RFC 6106 that adds two new options to the RA message for DNS:

- **RDNSS** (*Recursive DNS Server*) – this option contains one or more IPv6 addresses of recursive DNS servers.
- **DNSSL** (*DNS Search List*) – this option contains one or more domain names of DNS suffixes. The principle is based on the progressive search and adding domain names from Search List if a domain name is not specified absolutely. Such a domain name ends with (.).

For example, if Search List contains an entry *cvut.cz*, so it is possible to access *www.cvut.cz* website just typing *www* in a host. The first a host will try to find the IPv6 address for *www* in DNS server and if it fails, it will repeat the procedure with prefix according the list. So, it will find the IPv6 address for *www.cvut.cz*.

All of the addresses share the same lifetime value. If it is desirable to have different Lifetime values, multiple DNS options can be used. The recommended lifetime ranges from the maximum interval between two RA messages to double value of this interval. The zero value of lifetime has a special meanings, the corresponding DNS address is deleted from the IPv6 host.

When an IPv6 host receives RA message with DNS options, the validity of DNS options is checked. If the DNS options are valid, the host should copy, update or delete the values of the options into the DNS Repository and the Resolver Repository based on RA message content. Otherwise, the host must discard the options. The recommended number of DNS addresses is three. When the number of DNS addresses is already the sufficient number, the new one replaces the old one that will expire first in terms of lifetime.

In the case where the DNS options of RDNSS and DNSSL can be obtained from multiple sources, such as RA and DHCPv6, the IPv6 host should keep some DNS options from all sources.

8 DHCPv6

8.1 Basics about DHCPv6

DHCP, *Domain Host Configuration Protocol,* is server-client service, which provides managed configuration of all necessary parameters for full computer network life (i.e. address, net mask, DNS server address, etc.).

- DHCPv6 process operates in both stateless or stateful mode. **In stateful mode**, address and other network parameters, such as NTP or DNS server address, are provided to clients. DHCPv6 node knows all assigned addresses and it is able to operate on per client (DUID) basis.

- **Stateless mode** is used in situation where addresses are provided by stateless service (SLAAC) and DHCPv6 service is used to provide additional parameters, which cannot be delivered by stateless service. There is a special flag in Router Advertisement (RA) messages called "O" - other stateful configuration. This flag instructs clients to query for DHCPv6. If no RDNSS option is provided in RA then client may check for DHCPv6 regardless of whether the O flag is set or not (see RFC4339).

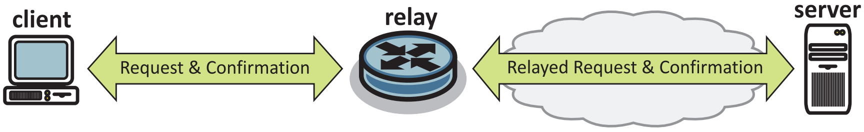
8.2 DHCPv6 Components

DHCPv6 procedure consists of three types of components

- Client – endpoint, which wants to gather information
- Server – provides information
- Relay – arrange connection between Client and Server, if the client and server are in different networks

Agent is common term for *Servers* or *Relays*.

The *Agent* is someone who provides DHCP response (either their own or mediated) and is located in the local network.



The Components of DHCPv6

8.3 DHCPv6 Identification

Identification of clients and servers plays very important role in process of DHCP.

Previous version (DHCP for IPv4) used MAC address of network interface as an unique client identificator. There are two new client identificators defined in DHCPv6

- **DUID** (*DHCP Unique IDentificator*)
- **IA** (*Identity Association*)

DUID is automatically generated on the first boot of operating system and it is generated according to link-layer or link-layer + boot time. The second option is usually used as default in most of the implementations. This cause problem, because network administrator is not able to determine DUID of newly added clients before the first connection and it changes with every reinstall of operating system. Each client or server has just one its own DUID.

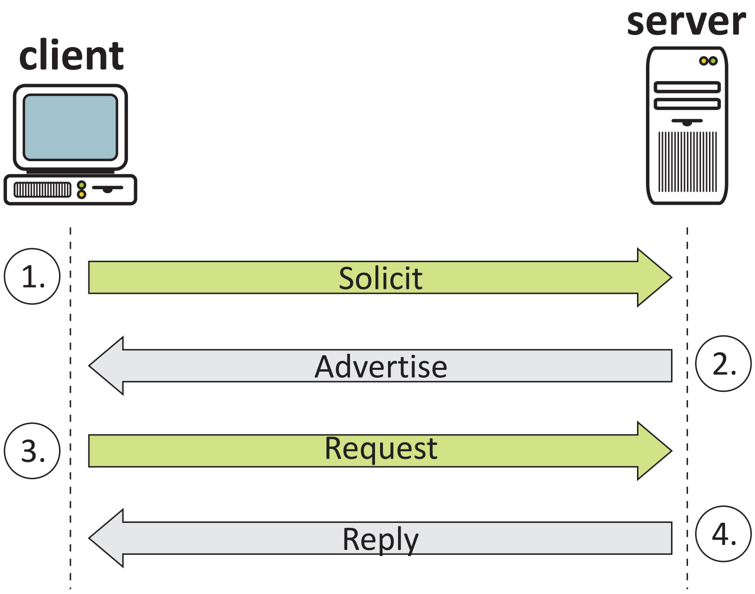
IA is typically assigned to a configuration information set of one interface, equipped with a unique identifier (IAID).

The client is uniquely identified by its DUID, the interface within the client are then differentiated by IA.

8.4 DHCPv6 Dialog Stage

Basic phase of DHCPv6 dialogue is not significantly changed comparing to its predecessors. DHCPv6 configuration is essential in four steps:

1. **Solicit** – the client looks for friendly server. Since the client does not know anything about the network sends its request to the multicast address. The request contains all its identification data (DUID and IA).
2. **Advertise** – a server sends this message to indicate that it is available for DHCP service, in response to a Solicit message received from a client. Message is sent by servers that are willing to give clients some parameters (usually only one, but there can be more).
3. **Request** – the client chooses bid, which he seems to be the best, and the appropriate server sends a request to award the offered parameters.
4. **Reply** – the whole process ends with sending the confirmation, which the server notifies the client that he actually assigned parameters.



The Procedure of DHCPv6

Multicast address allocated to the DHCPv6 procedure

FF02::1:2	all DHCPv6 agents and servers (it is link-local address !)
FF05::1:3	all DHCPv6 servers

8.5 DHCPv6 Life Cycle, Security

Address Life Cycle

Assigned addresses are allocated for a limited time period.

The client must request for a renewal before half of assigned period expires.

All the activity of DHCPv6 procedure typically comes from the client. However, sometimes server has to invoke DHCPv6 dialog. There could be situations where change of network parameters causes that server needs the client to adapt to the new situation.

Security

Some network administrators may wish to provide authentication of the source and contents of DHCPv6 messages. For example, clients may be subject to denial of service attacks through the use of bogus DHCPv6 servers, or may simply be misconfigured due to unintentionally instantiated DHCP servers.

The client that wants to authenticate DHCPv6 communication, will include the *Authentication option* into initial message (*Solicit*). Server must respond with *Authentication option* included into its *Advertise* message. The authentication information carried in the *Authentication option* can be used to reliably identify the source of a DHCPv6 message and to confirm that the contents of the DHCP message have not been tampered with.

9 Domain Name System

9.1 Domain Name System

DNS is a hierarchical distributed naming system for computers, services, or any resources connected to the Internet or a private network, which is implemented by DNS servers and DNS protocol. The rules for the creation of domain names and domain themselves as well as mechanisms for mutual translation of domain names and IP addresses of network nodes are part of the DNS system. The fundamentals of DNS protocol are defined in RFC 1035. Domain names are used for convenience of users and allow using symbolic names of network nodes instead of IP addresses that are arranged in a hierarchical structure.

Amendments and differences in DNS, which the IPv6 implementation has requested are listed below. Therefore, it is necessary that the student should have basic knowledge of the DNS protocol.

Standardization of the DNS for IPv6 was complex and walked a long way before it has stabilized. Current DNS implementation for IPv6 solves two problems:

- Store IPv6 addresses into DNS – currently, RFC 3596 solves this problem.
- Communication between DNS servers and hosts via IPv6 – a communication is a matter of DNS server implementation and therefore its administrators.

These problems are substantially independent. The host can communicate with DNS server via IPv4 and exchange information about IPv6 and vice versa.

There are several informational RFC documents dealing with the issues of IPv6 and DNS coexistence. RFC 3901 deals with recursive servers in local networks that solve DNS queries of local hosts. RFC 4074 describes the possible error states of DNS servers when they receive a DNS query for AAAA record type. RFC 4472 describes the issues of IPv6 and DNS coexistence complexly. It mainly deals with questions like: "What IPv6 addresses should or should not be written into DNS?" and "What communication protocol should be used if the computer uses both protocols?" The answers to both questions are briefly described below.

9.2 IPv6 Addresses in DNS

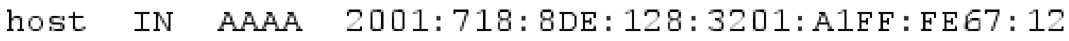
IPv6 addresses are stored into DNS in the same way as in IPv4. The client uses the following types of queries to communicate with DNS server:

- Standard DNS Queries
- Inverse DNS Queries

Standard DNS Queries

A new AAAA record type was introduced for DNS queries. The name of the new record type is derived from the length of an IPv6 address, which is four times larger in comparison with the length of an IPv4 address. In the case of IPv4, the DNS query uses A record type.

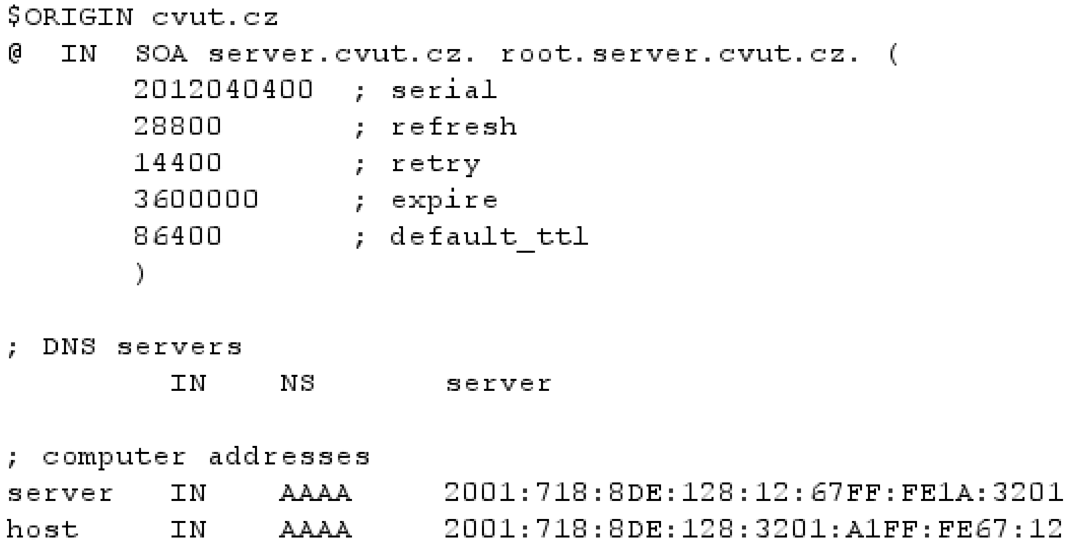
If *host.cvut.cz* computer has the address 2001:718:8DE:128:3201:A1FF:FE67:12, the following entry will be included in a zone file for *cvut.cz* domain:



```
host   IN   AAAA   2001:718:8DE:128:3201:A1FF:FE67:12
```

Entry

Then the zone file of *cvut.cz* domain in which there is an authoritative name server and a computer, it may look like this:



```
$ORIGIN cvut.cz
@   IN   SOA  server.cvut.cz. root.server.cvut.cz. (
       2012040400   ; serial
       28800        ; refresh
       14400        ; retry
       3600000      ; expire
       86400        ; default_ttl
       )

; DNS servers
         IN    NS       server

; computer addresses
server   IN    AAAA     2001:718:8DE:128:12:67FF:FE1A:3201
host     IN    AAAA     2001:718:8DE:128:3201:A1FF:FE67:12
```

Zone File

When a network uses multiple prefixes, computers typically have multiple IPv6 addresses. Then, each IPv6 address must have a corresponding AAAA record in a zone file in DNS server.

Inverse DNS Queries

Inverse DNS query is used to obtain the domain name to well-known IPv6 address. PTR records are used as in the case of IPv4. Inverse DNS query is formed by the inverse sequence of hexadecimal digits of IPv6 address to whose end the ip6.arpa domain is connected. IPv6 address must be complete. It means that it must contain all zeros. Inverse DNS query for above mentioned address 2001:718:8DE:128:3201:A1FF:FE67:12 has the form:

```
2.1.0.0.7.6.E.F.F.F.1.A.1.0.2.3.8.2.1.0.E.D.8.0.8.1.7.0.1.0.0
.2.ip6.arpa
```

Inverse DNS Query

The prefix will be at the end of inverse DNS query due to an inverse sequence of digits, which allows implementing distributed management of inverse domains.

If the CTU network has the prefix 2001:718:8DE::/48, the DNS server of CTU will manage E.D.8.0.8.1.7.0.1.0.0.2.ip6.arpa inverse domain. The following entry will be included in a zone file for this inverse domain:

```
2.1.0.0.7.6.E.F.F.F.1.A.1.0.2.3.8.2.1.0 PTR host.cvut.cz.
```

Entry for Inverse Domain

Then the zone file of inverse domain that corresponds to *cvut.cz* domain may look like this:

```
$ORIGIN E.D.8.0.8.1.7.0.1.0.0.2.ip6.arpa.
@       IN      SOA server.cvut.cz. root.server.cvut.cz. (
                2012040400  ; serial
                28800       ; refresh
                14400       ; retry
                3600000     ; expire
                86400       ; default_ttl
                )

; DNS servers
        IN      NS     server

; inverse records
1.0.2.3.A.1.E.F.F.F.7.6.2.1.0.0.8.2.1.0  PTR  server.cvut.cz.
2.1.0.0.7.6.E.F.F.F.1.A.1.0.2.3.8.2.1.0  PTR  host.cvut.cz.
```

Zone File of Inverse Domain

9.3 Zone File Contents

Each interface node has more IPv6 addresses with different scope and lifetime. For this reason, the following question arose: "What IPv6 addresses should be written into a zone file?" All IPv6 Global Unicast addresses and addresses used for transition mechanisms with long-term lifetime should certainly be written into a zone file. On the other hand, the Link-Local addresses and randomly generated addresses with short-term lifetime that are used for privacy should not be written into a zone file.

For other IPv6 addresses, especially those that are assigned by stateless or stateful address auto-configuration, there is not any universal recommendation. These addresses have mostly short-term lifetime, so their inclusion would require the deployment of dynamic DNS updates.

Another question arises when a computer communicates via two protocols: "What address type does a computer use?" There are two fundamental approaches:

- The same domain name for both types of addresses
- Different domain names for individual types of addresses

The Same Domain Name

If *host.cvut.cz* computer has the IPv4 address 192.168.121.57 and IPv6 address 2001:718:8DE:128:3201:A1FF:FE67:12, the following entries will be included in a zone file for *cvut.cz* domain:

```
host    IN    A         192.168.121.57
        IN    AAAA      2001:718:8DE:128:3201:A1FF:FE67:12
```

Example of Entries for the Same Domain Name

If the *pc* computer wants to communicate with *host.cvut.cz* computer, the DNS server sends both addresses to the *pc* computer that selects the appropriate address based on the communication protocol. When the *pc* computer communicates with both protocols, IPv6 will take precedence because the current operating systems prefer IPv6.

Currently, this approach is not entirely reliable due to lack of IPv6 implementation. If the establishing an IPv6 connection fails, an IPv4 connection will be established for TCP communication. The TCP protocol provides mechanisms for reliable delivery of data due to which the computer finds out that the IPv6 network is not functional. But the interval required for detecting malfunction of IPv6 network can be up to several minutes. In the case of UDP communication, the connection is not established at all.

Different Domain Names

In the case of different domain names, it is usually specify a subdomain (often called ip6 or ipv6) in which IPv6 addresses are included. From the domain name directly implies the use of the communication protocol.

Then the zone file for *host.cvut.cz* computer should contain the following entries:

```
host        IN     A       192.168.121.57
host.ip6    IN     AAAA    2001:718:8DE:128:3201:A1FF:FE67:12
```

Example of Entries for Different Domain Names

Most likely, this is a temporary option than the IPv6 implementation will be wholly accessible.

10 Mobility of Devices in IPv6

10.1 IPv6 Mobility Support

IPv6 allows devices remain reachable while moving around in the IPv6 Internet. Each mobile device is always identified by its home address, regardless of its current point of attachment to the Internet. While situated away from its home, a mobile device is also associated with a care-of address, which provides information about the mobile device’s current location. IPv6 packets addressed to a mobile device's home address are transparently routed to its care-of address. The protocol enables IPv6 devices to cache the binding of a mobile device's home address with its care-of address, and to then send any packets destined for the mobile node directly to it at this care-of address.

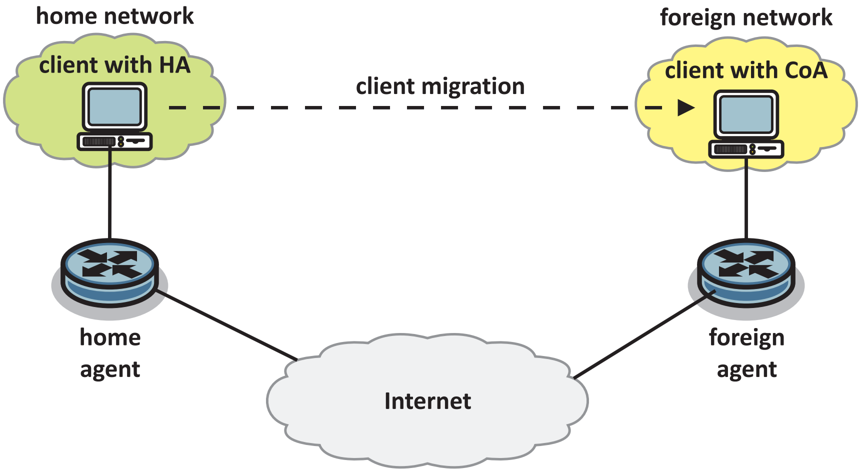
All IPv6 devices, whether mobile or stationary, can communicate with mobile devices.

10.2 Basic Principle of IPv6 Mobility

Even the mobile device is somewhere at home.

At home network mobile device has registered its **HA**, *home address,* which is fixed and linked to DNS record. This mobile device establishes also a Home Agent, which stores information about mobile devices whose permanent home address is in the home agent's network.

While away from its home network, a mobile device is associated with a temporary address, a **CoA**, *care-of address,* which is associated with the network the mobile node is visiting and establishes a Foreign Agent. If there is no foreign agent in the host network, the mobile device has to take care of getting an address and advertising that address by its own means.

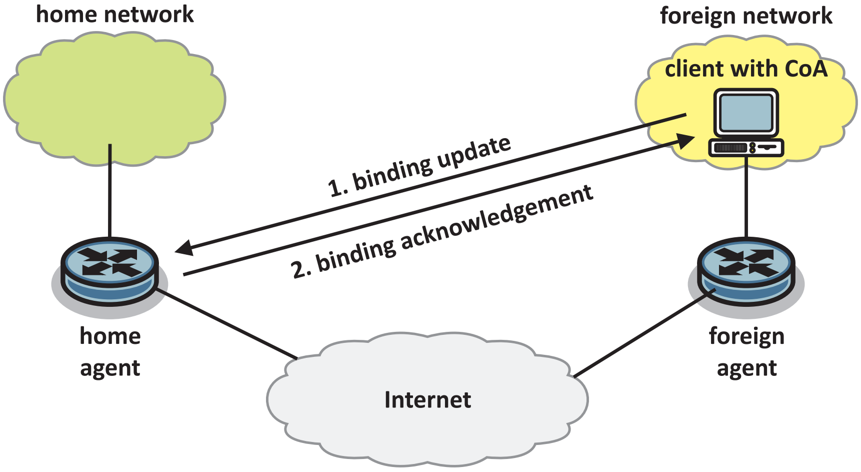


Basic IPv6 Mobility Scenario

Mobility in IPv6 specifies how a mobile device registers with its Home Agent and how the Home Agent routes datagrams to the mobile device through the *tunnel* between Home and Foreign Agents.

10.3 Bindings

The relationship between a mobile device's home address and care-of address is known as a binding for the mobile device. While away from home, a mobile device registers its primary care-of address with a router, requesting this router to function as the *Home Agent* for the mobile device. The mobile device performs this binding registration by sending a *Binding Update* message to the *Home Agent*. The *Home Agent* replies to the mobile device by returning a *Binding Acknowledgement* message.

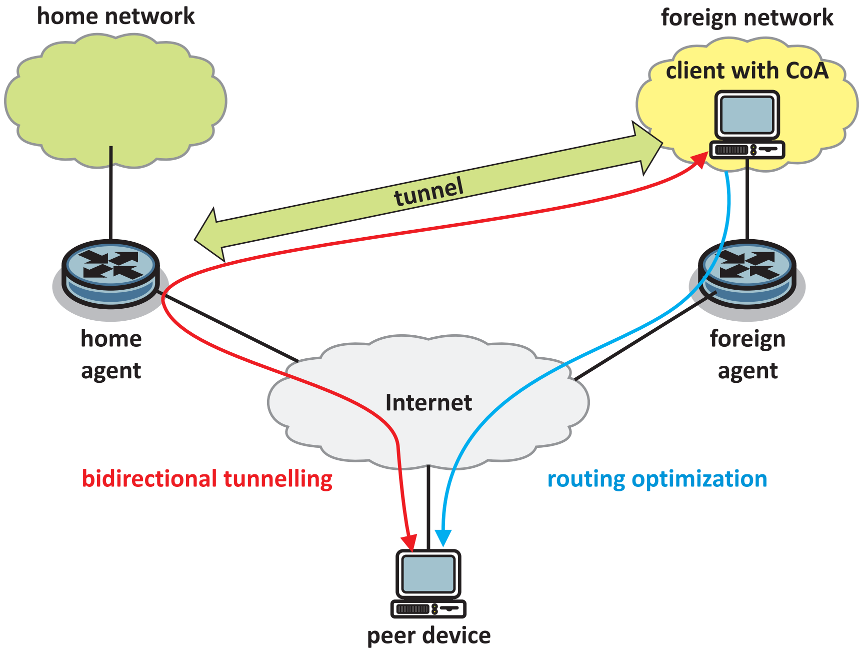


Mobility Bindings

10.4 Routing Scenarios

There are two possible modes for communications between the *mobile device* and a *peer device*.

- **Bidirectional tunneling** – is available even if the mobile device has not registered its current binding with the peer device.
    - Packets **from** the peer device are routed to the *Home Agent* and then tunneled to the mobile device.
    - Packets **to** the peer device are tunneled from the mobile device to the *Home Agent* (reverse tunneled) and then routed normally from the home network to the peer device.
- **Route optimization** – requires the mobile device to register its current binding at the peer device.
    - Packets **from** the peer device can be routed directly to the care-of address of the mobile node.
    - When sending a packet **to** any IPv6 destination, the peer device checks its cached bindings for an entry for the packet's destination address. If a cached binding for this destination address is found, the device uses a *IPv6 routing header* to route the packet to the mobile device by way of the care-of address indicated in this binding.



Routing Scenarios for IPv6 Mobility

11 Transition Mechanisms

11.1 Overview of Transition Mechanisms

IPv4 to IPv6 changeover is not a sudden process. Therefore, there must exist transition mechanisms that allow the simultaneous operation of both protocols. There are lots of proposals enabling mutual cooperation between these two protocols. These proposals can be divided into three groups:

- Dual Stack – the device includes two protocols enabling communication between IPv4 and IPv6 world. If it is needed, the cooperation between the two protocols takes place up to the application layer. The relevant application collects data that arrived through one protocol, and adjust them depending on their properties and potential configuration and sends them to target application by the second protocol. This principle is not ideal because it allows for the existence of IPv4 in the future. However, it is the basis for the remaining groups because all other methods require the support of both protocols in some devices.

- Tunneling – tunnels are used to connect two IPv6 networks through IPv4 network where an IPv6 datagram is transmitted as data encapsulated in an IPv4 datagram.

- Translators – they translate IPv6 datagrams into IPv4 datagrams and vice versa.

The following table is presenting short list of transition mechanisms. The table contains the RFC documents where the individual mechanisms are described in more detail.

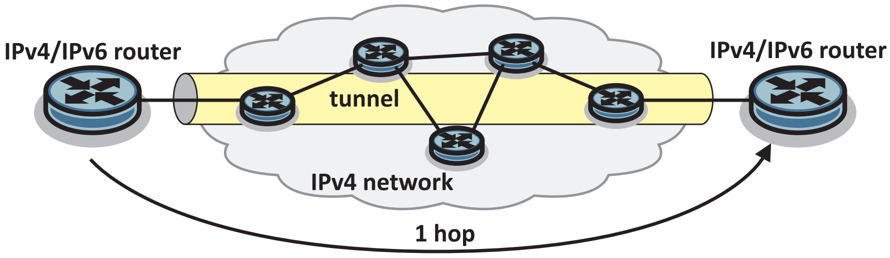
The List of Transition Mechanisms

Tunneling		**Translators**	
tunnel server/broker	RFC 3053	SIIT	RFC 6145
6to4	RFC 3056	NAT64	RFC 6146
6rd	RFC 5569	TRT	RFC 3142
6over4	RFC 2529	BIH	draft
ISATAP	RFC 5214	SOCKS64	RFC 3089
Teredo	RFC 4380		
Dual-Stack Lite	RFC 6333		

The general principles of tunneling and translating are only described in the following sections.

11.2 Tunneling

The general principle of tunneling is depicted in the figure below.



Tunneling

Each tunnel has generally the two ends and each end has its own IP address. If the device at one end of tunnel decides that the IPv6 datagram should be sent through the tunnel, the device takes the IPv6 datagram and inserts them as the data to newly created IPv4 datagram. The decision is done based on the routing table or special address of the target node in the incoming IPv6 datagram. Destination address of new IPv4 datagram will be an IPv4 address of the second end of the tunnel and source address will be an IPv4 address of the local end of the tunnel. The value 41 is inserted into the Protocol field in IPv4 datagram header. The value 41 represents the tunneled IPv6 datagram.

After that the datagram is sent in the normal way to the second end of the tunnel through IPv4 network. According the value 41, the devices at the second end of the tunnels recognizes that it is the tunneled IPv6 datagram and unpacks an IPv6 datagram and processes it based on its destination IPv6 address and routing table. Passage of the IPv6 datagram through the tunnel is counted as one hop in terms of IPv6. Therefore, the device at the second end of the tunnel will reduce the Hop Limit field in IPv6 datagram header by one.

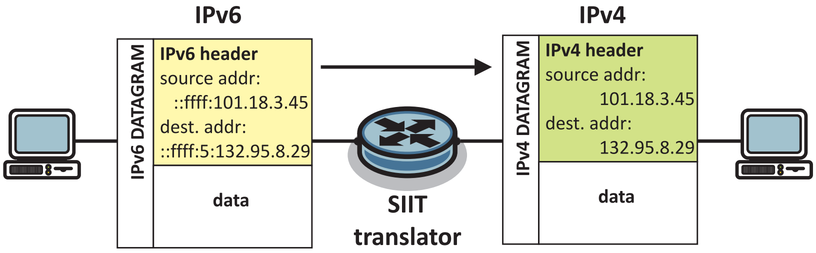
There are two tunneling modes in the case of IPv4 and IPv6 cooperation:

- Manual – the tunnels are manually configured by administrator. RFC 4213 describes the manual mode in more detail.
- Automatic – these tunnels are formed on the basis of information contained in IP addresses. RFC 2893 describes the automatic mode in more detail.

11.3 Translators

The general set of translating rules of datagram header is defined in RFC 2765: *Stateless IP/ICMP Translation Algorithm (SIIT)* and RFC 6145: *IP/ICMP Translation Algorithm*. Generally, these rules are used by all translators and they are very limited and do not support any extensions such as the IPv4 options or IPv6 extension headers. It means that translators drop these extensions.

Translators translate each datagram independently without any bindings to the previous datagrams and without preservation of data structures with information about the history or current status of ongoing communication. Passage of datagrams through **SIIT** (*Stateless IP/ICMP Translation*) translator is depicted in the figure below.



Passage of Datagrams Through SIIT Translator

Address mapping is part of the translating. That means the translation of IPv6 addresses to IPv4 addresses and vice versa. To do this the IPv4-embedded addresses are currently used. The 96-bit prefix is usually reserved for these addresses. It can be either an universal prefix 64:FF9B::/96 or prefix assigned by local ISP. IPv4 address follows the prefix. The IPv4-embedded address format is described in RFC 6052.

Translation of IPv4 address to IPv6 address can be made stateless. That means that IPv4 address is added for the IPv6 prefix. Otherwise, it is not that simple. There is most often used the dynamic mapping to translate IPv6 address to IPv4 address. This is similar to the **NAT** (*Network Address Translation*).