# Preparing an IPv6 Addressing Plan

## Manual

**December 2010: Original text**
**March 2011: Translation provided by RIPE NCC**

SURFnet bv, Radboudkwartier 273, Postbus 19035, 3501 DA Utrecht
T +31 302 305 305, F +31 302 305 329, www.surfnet.nl

# TABLE OF CONTENTS

# 1 INTRODUCTION

## 1.1 What is the purpose of this document?

IPv4 addresses have almost run out, and more and more businesses and institutions see the necessity of migrating to an IPv6 addressing system. An IPv6 address is 128 bits, which means that, in theory, there are $2^{128}$ addresses available, a great deal more than the $2^{32}$ (=4.3 billion) addresses available with IPv4. To give you an idea of the volume: $2^{128}$ represents approximately the number of grains of sand on our planet.

An addressing plan using the IPv4 system limits the options available to an organisation because there are relatively few IPv4 addresses still available. This is why the IPv4 addressing system is based on efficient address assignment. If you apply for an IPv6 address range at many LIRs, you will be allocated $2^{80}$ addresses. This is such a huge amount that efficiency virtually ceases to be an issue. This is why it is worthwhile adopting an IPv6 addressing plan: a system in which you assign the IPv6 addresses to locations and/or use types.

In an efficient IPv6 addressing plan, the IPv6 addressing ranges are grouped effectively and logically. This has several advantages, including:

- Security policies are easier to implement, such as the configuration of access lists and firewalls
- Addresses are easier to trace: the address contains information about the use type or location where the address is in use
- An efficient addressing plan is scalable: it can be expanded, for example, to include new locations or use types
- An efficient IPv6 addressing plan also enables more efficient network management

This manual will show you how to prepare an effective IPv6 addressing plan. In making that plan, you will need to make a number of important choices. Please think carefully about these choices to ensure that the addressing plan will meet the requirements of your organisation. The manual will provide suggestions to help you to make the right choices.

## 1.2 For whom is this document intended?

This manual is intended for network architects and network managers implementing IPv6 in their organisations. We assume you have experience in setting up IPv4 networks.

# 2 STRUCTURE OF IPV6 ADDRESSES

## 2.1 Address Notation

An IPv6 address consists of 128 bits that can each have a value of 0 or 1. Because an address made up of 128 ones and zeroes is illegible, a more convenient format has been devised. This format is based on the hexadecimal system, which is much easier to decipher while being closely related to the binary format.

Each digit in the hexadecimal system is equivalent to 4 bits; an IPv6 address of 128 bits therefore consists of 128/4 = 32 hexadecimal digits. The notation is as follows:

```
2001:0db8:0000:0000:0000:0000:0000:0000
```

Since it is impractical to record all these zeroes, they may be skipped in accordance with certain conventions; leading zeroes can be dropped for any group of digits between two colons. The result would then be:

```
2001:db8:0:0:0:0:0:0
```

A series of zeroes and colons may also be abbreviated as two colons. The result is now:

```
2001:db8::
```

The precise rules for writing IPv6 addresses are recorded in RFC 5952.[1]

## 2.2 Grouping Addresses (prefixes)

IPv6 addresses are grouped using the binary value of the address. This grouping is carried out using a “prefix”. Prefixes are all addresses that start with the same series of bits. The length of the identical series is noted after the address, separated by a forward slash. The prefix

```
2001:db8::/32
```

thus contains all the addresses from

```
2001:0db8:0000:0000:0000:0000:0000:0000
```

through

```
2001:0db8:ffff:ffff:ffff:ffff:ffff:ffff
```

[1] http://tools.ietf.org/html/rfc5952#page-10

As can be seen above, the first 32 bits, i.e. the first eight hexadecimal digits, are identical. The prefix

**2001:db8:1234::/64**

contains all the addresses from

**2001:0db8:1234:0000:0000:0000:0000:0000**

through

**2001:0db8:1234:0000:ffff:ffff:ffff:ffff**

It is common practice to separate the different groups in multiples of four bits. This makes it much easier to decipher the addresses. This is why the prefixes /48, /52, /56, /60 and /64 are commonly used. If a group is not separated at a multiple of four, the addresses become much more difficult to decipher (see box).

### Prefix not a multiple of four

If the prefix length is not a round multiple of four, the binary separation will take place in the  middle of a hexadecimal number. This means that all hexadecimal numbers that start with the same series of bits will belong to this prefix. The prefix

**2001:db8::/61**

thus contains all the addresses from

**2001:0db8:0000:0000:0000:0000:0000:0000**

through

**2001:0db8:0000:0007:ffff:ffff:ffff:ffff**

because the hexadecimal numbers 0 through 7 all start with the binary value 0. For example, the prefix

**2001:db8:0:8::/61**

contains all the addresses from

**2001:0db8:0000:0008:0000:0000:0000:0000**

through

**2001:0db8:0000:000f:ffff:ffff:ffff:ffff**

because the hexadecimal numbers 8 to f all start with the binary value 1.

## 2.3 Assigning Address Blocks

IPv6 address ranges are assigned as follows:

| Prefix | Assigned to | Number of addresses |
|---|---|---|
| /32 | LIR (Local Internet Registry, usually an Internet Service Provider) | $2^{96}$ |
| /48 | Organisation | $2^{80}$ |
| /64 | Organisation network (subnet) | $2^{64}$ |
| /128 | Host (PC, server, printer, router) | 1 |

In this manual, it is assumed that all networks use a /64 address block. Other address blocks may be used, but we do not recommend this because some equipment may work differently with other formats.

Also, it is assumed that your organisation has been allocated a /48 address block, and that 16 bits (64-48) are therefore available for assigning the addresses to networks. If your situation is different, you will need to adapt the calculations in this manual accordingly.

Based on the above information, the first 48 bits of your IPv6 plan are fixed. In this document, we use 2001:db8:1234::/48 as an example. This means you can use the /64 prefixes

**2001:db8:1234:0000::/64**

through

**2001:db8:1234:ffff::/64**

for your network – 16 bits in total.

For your own addressing plan, you will need to replace the numbers in the examples with the prefix allocated to you.

### 2.4 Notation of the Assigned Addresses

In this manual, we will subdivide the 16 available bits into groups. We distinguish the following types of groups:

**B:** bit is assignable
**L:** bit is assigned to a location
**G:** bit is assigned to a use type

The following notation is used for the assigned bits. The order of the letters here are random and are only used as an example:

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2001:db8:1234: | L | L | L | L | G | G | G | G | B | B | B | B | B | B | B | B | ::/64 |

Each box represents 1 bit. Four boxes together therefore represent one hexadecimal digit in the IPv6 address. For the above example, this produces the following address structure:

**2001:db8:1234:LGBB::/64**

Bits 1-4 are in this example assigned to a location, bits 5-8 are assigned to a use type and bits 9-16 remain available to be assigned to another purpose.

## 3 OPTIONS FOR WORKING WITHOUT AN ADDRESSING PLAN

### 3.1 No Addressing Plan

Small, flat organisations that do not have an internal organisational structure (with several departments within the organisation being authorised to assign IP addresses) or technical structure (distinguishing between various categories of use types and networks) can work without an addressing plan, instead assigning a random free IPv6 address as network host.

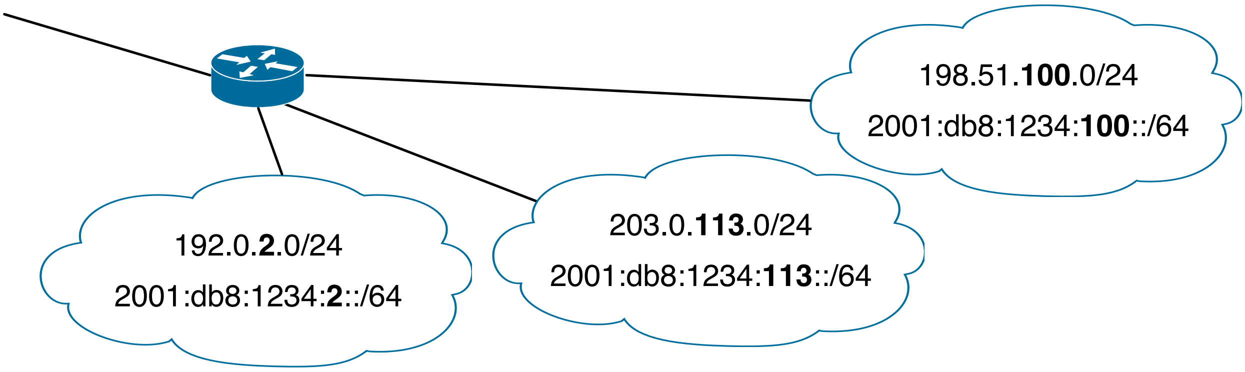
A disadvantage is that it can be difficult to recognise networks based on their address because this method lacks structure. For this reason, we recommend preparing an addressing plan.

Should you opt to work without an addressing plan, we recommend keeping a list of the assigned networks in a central location, such as an Excel spreadsheet, an internal Wiki page or in the reverse DNS configuration.

### 3.2 Direct Link Between IPv4 and IPv6 Addresses

If the existing IPv4 networks use only /24 subnets (for example, from 203.0.113.0 to 203.0.113.255), a direct link can be established between IPv4 addresses and the new IPv6 addresses. In this case, you can include the penultimate number of the IPv4 address (113 in 203.0.113.0/24, for example) in the IPv6 subnet. The IPv6 address will then be 2001:db8:1234:113::/64.

Such an IPv4-to-IPv6 transition could appear as follows:



In this addressing plan, the link between the existing IPv4 networks and the IPv6 networks is immediately visible.

For vital equipment, such as servers and routers, it may be practical to use the last number of the IPv4 address in the IPv6 Address too. The IPv4 address 192.0.2.123, for example, would become the IPv6 address 2001:db8:1234:2::123. It is also possible to incorporate the entire IPv4 address into the IPv6 address. In this case, the IPv6 address would be 2001:db8:1234:2:192:0:2:123.

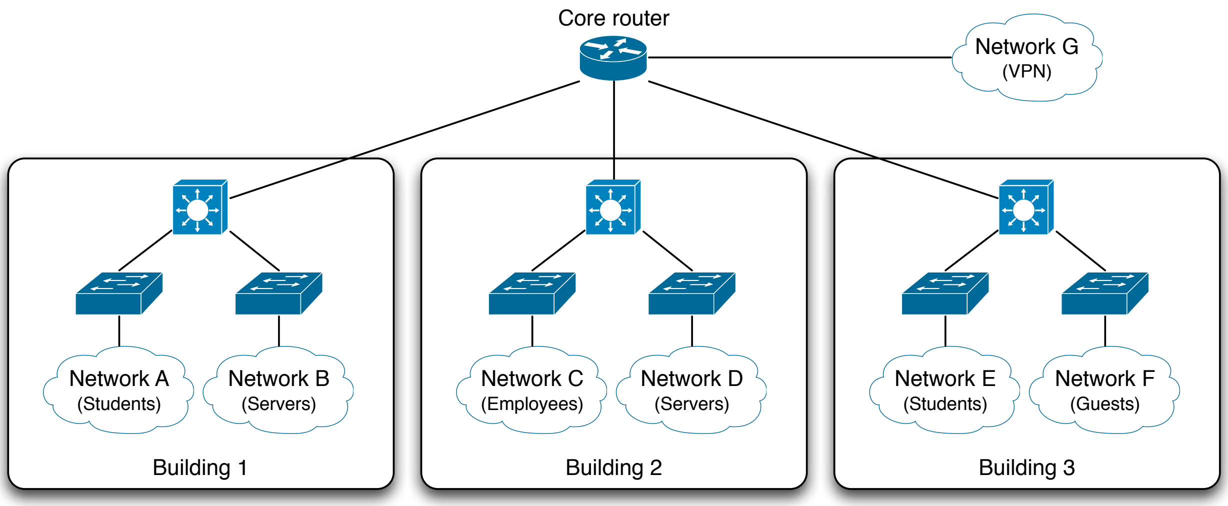
Should you choose this option, we recommend keeping a list of the assigned networks in a central location, such as an Excel spreadsheet, an internal Wiki page or in the reverse DNS configuration.

## 4 PREPARING AN ADDRESSING PLAN FOR A NETWORK

### 4.1 Introduction

When preparing your addressing plan you have to decide which system to use to assign the available addresses to the networks in the organisation. There are a number of convenient methods for assigning addresses.

The present chapter will describe the possible addressing plans, using the following example network:



## 4.2 Basic Structure of the Addressing Plan

With the IPv6 protocol, there are so many available addresses that we can create one or more primary subnets. We can, for example, assign the addresses per use type or per location, or use combinations. For example, we may assign the addresses first by use type and then by location. Once these subnets have been defined, there will still be bits remaining that can be assigned to another purpose.

Take the example in section 2.4:

| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2001:db8:1234: | L | L | L | L | G | G | G | G | B | B | B | B | B | B | B | B | ::/64 |

In this example, 4 bits are assigned to a location (L) and 4 bits are assigned to a use type (G). As a result, there are 8 bits remaining (B). Following this addressing plan, 16[1] locations can be addressed, with each location having 16[2] allocatable use types. Each of these locations can create 256[3] networks per use type.

## 4.3 Defining a Primary Subnet

We first need to decide on the addressing of the primary subnets. We recommend choosing the location or the use type (such as students, staff, servers, switches, routers, public, etc.) as the primary subnet. These options are discussed below.

### 4.3.1 Location

When the location is the primary subnet, each building, department etc. is assigned a number of dedicated addresses. The emphasis in this case lies on optimisation of the routing tables. All the networks within a single location will be aggregated to a single route in the routing table, so that the routing table will remain compact.

[1] Number of possible combinations of the 4 location bits
[2] Number of possible combinations of the 4 use type bits
[3] Number of possible combinations of the 8 assignable bits

### 4.3.2 Use type

When the use type is the primary subnet, the routing optimisation described above is not feasible, because the use types are divided across a number of locations. However, in practice this will not be a problem with most routers.

This option does make it much easier to implement a security policy. Most firewall policies are based on the type of use and not on the location of the network. This is why the firewalls often require only one policy per use type.

### 4.3.3 Recommendation

Based on the above information, we recommend use type-based primary subnets, because this is the easiest way to integrate with existing policies and procedures.

Possible reasons for location-based primary subnets are:

- Some locations will prepare their own addressing plan
- The routers cannot process such a large number of routes without aggregation

## 4.4 Determining the Address Space Required for the Addressing Plan of Choice

Now we need to determine which portion of the 16 bits of available address space (see section 2.3) is required for the addressing plan selected. The number of groups in the primary subnet determines the number of bits required. One bit can contain two groups ($2^1$), 2 bits can contain 4 groups ($2^2$), etc. (see also the appendix).

We can determine the number of groups as follows:

1. First determine the number of locations or use types within your organisation. Count each location or use type as one group.
2. Increase this number by one group (required for the backbone and other infrastructure).
3. If you chose to work with location-based primary subnets, add one extra group for all networks that do not have a fixed location. These are networks for VPNs and tunnels, for example.
4. Add one or two groups to allow for future expansion.
5. To create a practical addressing plan, the number of blocks into which we divide the address space should be to a power of 2. So we'll round up the number of bits counted in steps 1 to 4 to the nearest power of 2.

The result is the number of groups in the primary subnet, either by location or by use type.

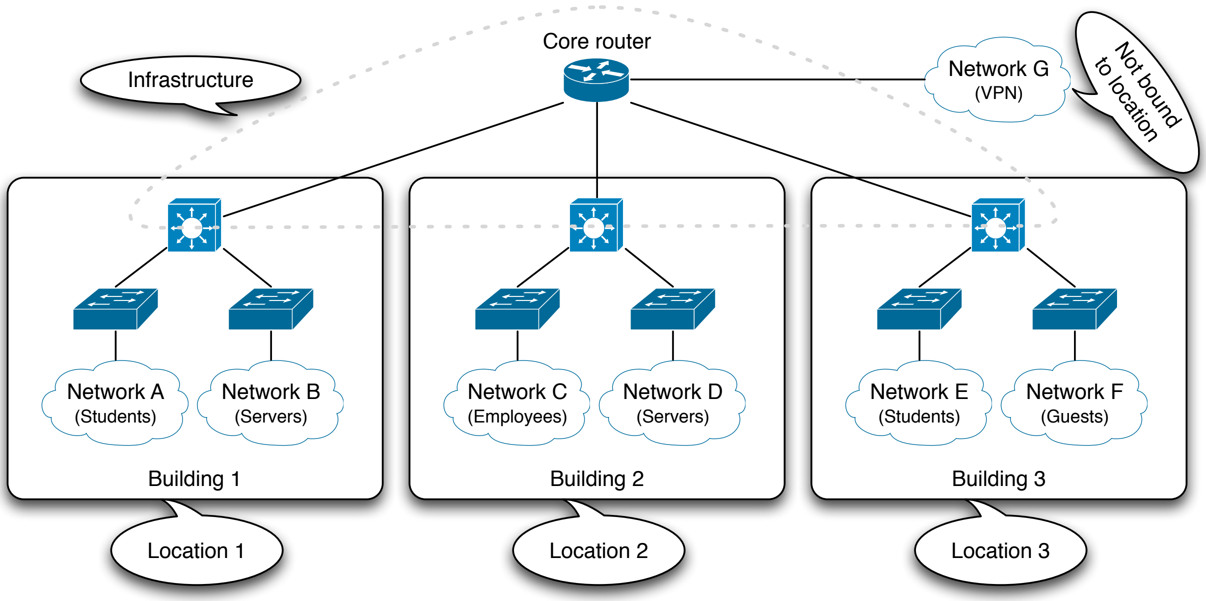
This method is explained using a number of examples. More detailed examples can be found in chapter 5.

### Example 1: Location-based subnet

In this example, we define the locations as the primary subnets. The number of groups required is then as follows:

| | |
|---|---|
| Number of locations: | Three groups |
| Backbone and other infrastructure: | One group |
| Non-location-based networks: | One group |
| Future locations: | Two groups |
| Total: | Seven groups |

This example network would then appear as follows:



If we round this up to the first power of 2, this results in eight subnets. Incorporating these primary subnets into the IPv6 address requires 3 bits (L) ($2^3$ = 8; see also the appendix) This results in the following bit distribution:

| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2001:db8:1234: | L | L | L | B | B | B | B | B | B | B | B | B | B | B | B | B | ::/64 |

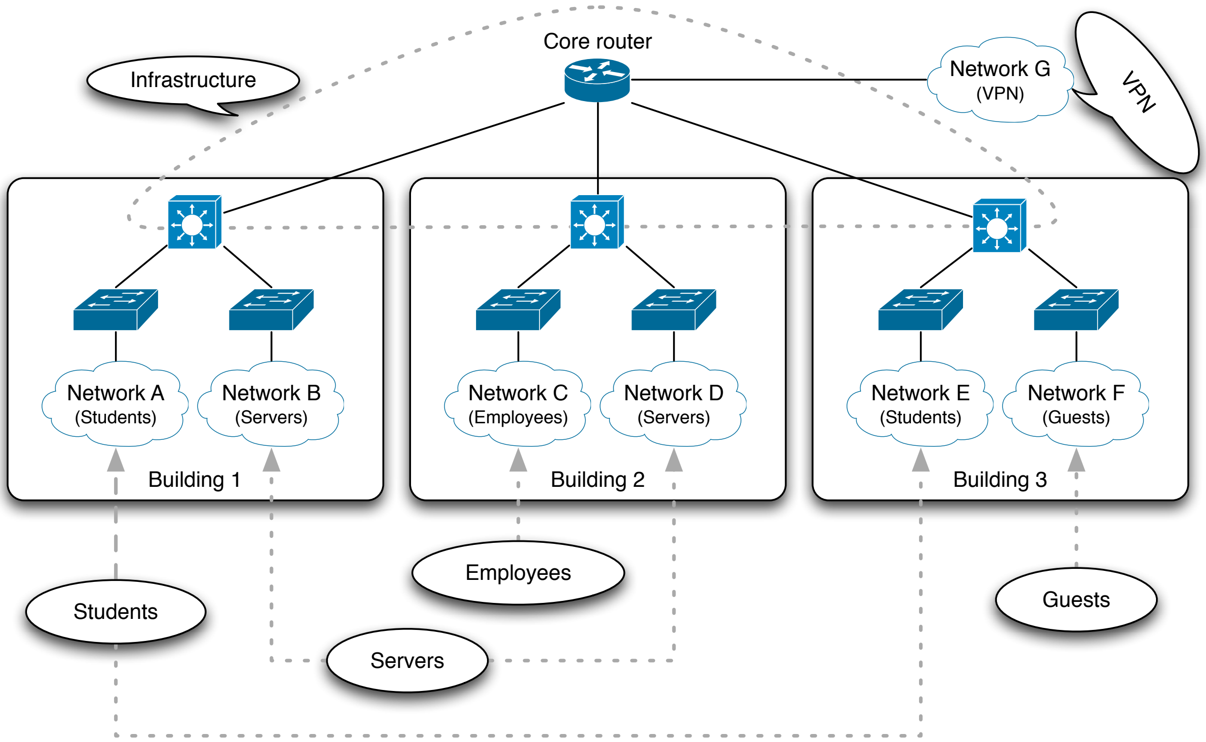
This leaves 13 available bits (B).

### Example 2: Use type-based subnet

In this example we define the use types as the primary subnets. The number of groups required is then as follows:

| | |
|---|---|
| Number of use types (staff, students, guests, servers and VPNs): | Five groups |
| Backbone and other infrastructure: | One group |
| Future use types: | Four groups |
| Total: | Ten groups |

This example network would then appear as follows:



If we round this up to the first power of 2, this results in 16 subnets. Incorporating these subnets into the IPv6 address requires 4 bits (G) ($2^4 = 16$). This leaves 12 available bits (B).

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2001:db8:1234: | G | G | G | G | B | B | B | B | B | B | B | B | B | B | B | B | ::/64 |

## 4.5 Optional Secondary Subnets

The remaining bits can be used for numbering secondary subnetworks within the selected addressing plan. If the primary subnets are location-based, multiple networks can be addressed by location, whereas if the primary subnets are use type-based, multiple student networks or server networks can be addressed, for example.

The remaining bits can also be used to combine subnets by location and use type. If the subnet is location-based, as in example 1, and we create a use type-based secondary subnet, as in example 2, the result is as follows:

| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2001:db8:1234: | L | L | L | G | G | G | G | B | B | B | B | B | B | B | B | B | ::/64 |

In this addressing plan, there is space for eight locations, each with 16 use types. Within each use type, there are yet another 512 ($2^9$ due to 9 available bits) secondary subnetworks available.

This combination of primary and secondary subnets uses location-based primary subnets. This will make it easier to optimise routing tables, but it will complicate the design of the security policy. This is because firewall policies can only be applied on the basis of the first numbers of an address, while in this example the location is at the start of the address, not the use type.

To facilitate the design process for the security policy, the combination can be reversed by making the primary subnet use type-based, as in example 2, and the secondary subnets location-based, as in example 1. The result is then as follows:

| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2001:db8:1234: | G | G | G | G | L | L | L | B | B | B | B | B | B | B | B | B | ::/64 |

In this example, the use type is at the start of the address, making it easier to apply firewall policies per use type. Since the use type is typically more relevant for security policies than the location, we recommend using this system.

## 4.6 Control

We can check whether the addressing plan we have created meets our requirements by counting the number of bits remaining after creation of the primary and secondary subnets. If, for example, after the creation of use type-based primary subnets containing location-based secondary subnets we also require multiple student networks at each location, there will have to be enough bits left to create these.

In the example in section 4.5 there are 9 bits remaining, which results in 512 ($2^9$) possible values per use type per location. This will usually be more than enough.

## 4.7 Leeway

If the number of remaining bits is not quite sufficient, this can be compensated for in the assignment of the primary and secondary subnets.

In the above example we used 4 bits for the use types and 3 bits for the locations. That leaves 9 bits, so we can create 512 ($2^9$) networks per use type per location.

| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2001:db8:1234: | G | G | G | G | L | L | L | B | B | B | B | B | B | B | B | B | ::/64 |

This will be sufficient in most cases. However, it may be that we require 2,048 VPN networks per location. This can be done by changing the assignment of primary and secondary subnets. However, the IPv6 address will become more difficult to decipher in hexadecimal format. Another option is to reserve four groups within the use type subnets for VPN networks. If we sort the numbers of this group of four in binary number groupings (decimal: 0-3, 4-7, 8-11 of 12-15; hexadecimal: 0-3, 4-7, 8-B, C-F) then these can still be covered by a single firewall policy.

The following groups might result:

| | |
|---|---|
| 0 | **Backbone and other infrastructure** |
| 1 | **Servers** |
| 2 | Future expansion |
| 3 | Future expansion |
| 4 | **Staff** |
| 5 | **Students** |
| 6 | **Guests** |
| 7 | Future expansion |
| 8 | **VPNs** |
| 9 | **VPNs** |
| A | **VPNs** |
| B | **VPNs** |
| C | Future expansion |
| D | Future expansion |
| E | Future expansion |
| F | Future expansion |

## 4.8 Legibility

If there are enough bits remaining, we can use them to make IPv6 addresses more legible. Each hexadecimal digit in a IPv6 address is equivalent to 4 bits. By choosing an assignment system based on multiples of 4 bits, each group will correspond with a digit in the IPv6 address.

In the example in this section, we used 3 bits per location and 4 bits per use type. By using 4 bits for the location, we increase the legibility, because then both the location and the use type are conveniently arranged into a 4-bit group:

| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2001:db8:1234: | L | L | L | G | G | G | G | B | B | B | B | B | B | B | B | B | ::/64 |
| 2001:db8:1234:<br>G G<br>G G<br>L L<br>L B<br>B B<br>B B<br>B B<br>B B<br>::/64 | | | | | | | | | | | | | | | | | |

Becomes:

| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2001:db8:1234: | L | L | L | L | G | G | G | G | B | B | B | B | B | B | B | B | ::/64 |

The 4 L bits are displayed as one hexadecimal digit in the IPv6 address. The four G bits are also displayed as one hexadecimal digit. This results in the following IPv6 address structure:

**2001:db8:1234:LGBB::/64**

Using this system, the location and use type can easily be identified in the IPv6 address. The first symbol indicates the location (L) and the second indicates the use type (G).

## 4.9 Using VLAN Numbers

Some organisations have already included a location and/or use type description in their VLAN numbering plan. If this is the case, you might consider transferring these numbers directly to the IPv6 addressing system rather than using the method described above.

If the VLAN number system is not location-based or use type-based, it is also possible to transfer the VLAN numbers directly to the IPv6 address. This makes it easy for system managers to identify the relationship between the VLAN and the IPv6 address. However, as the use type and location do not appear in the IPv6 address, it will not be possible to optimise the security policies and routing tables. For this reason, we do not recommend this method of addressing.

A VLAN number is 12 bits long, while we have 16 bits available with an IPv6 address. If we only use the VLAN number in our addressing plan, this will lead to a waste of 15/16ths of the IPv6 addresses. This will probably not be a problem for smaller organisations, and it might even be practical.

There are two methods of dealing with this:

1. We might incorporate the VLAN number decimally into the IPv6 address. The hexadecimal digits A to F will remain unused. This format is easily legible, but it

is less suited to implementing firewall policies, which are based on the bits in the address and are thus compatible with the hexadecimal format.

2. The VLAN number can be incorporated into the IPv6 address hexadecimally, so that one hexadecimal digit in the IPv6 address will remain unused. If, for example, we choose to leave the left digit as 0, we can still use the digits 1 through F for the hexadecimal notation.

   For instance:

| VLAN number | IPv6 decimal | IPv6 hexadecimal |
|---|---|---|
| **1** | 2001:db8:1234:000**1**::/64 | 2001:db8:1234:000**1**::/64 |
| **12** | 2001:db8:1234:00**12**::/64 | 2001:db8:1234:000**c**::/64 |
| **4094** | 2001:db8:1234:**4094**::/64 | 2001:db8:1234:0**ffe**::/64 |

### 4.9.1 Reversing VLAN notations in an IPv6 structure

There are also options for reversing previously defined VLAN notations. If, for example, the VLAN numbers are assigned first by location and then by use type, it is still possible to assign the IPv6 addresses in the reverse order: first by use type and then by location.

In the following example of a VLAN structure, the hexadecimal notation of the VLAN number, location and use type is placed between brackets:

| VLAN number | Location | Use type |
|---|---|---|
| 0001 (**001**) | 0 (**0**) | 1 (**01**) |
| 0529 (**211**) | 2 (**2**) | 17 (**11**) |
| 4094 (**FFE**) | 15 (**F**) | 254 (**FE**) |

In this example, the first 4 bits of the VLAN number identify the location. The remaining 8 bits describe the use type. By copying this directly to the IPv6 address, we are able to optimise the routing table, but not the security policy. The reason for this is that the location is at the start of the address while the use type follows it. However, if we wish to use the IPv6 addresses to optimise the security policies, the use type has to be at the start of the address.

To arrange this, we can move the first 4 bits of the VLAN number (which describe the location) to the back to become the last 4 bits of the IPv6 subnet. The last 8 bits of the VLAN number (which describe the use type) can be placed in front of these.

### 4.9.2 Hexadecimal notation

Hexadecimal notation is further explained using the example below. The hexadecimal notation of the VLAN number, location and use type is placed between brackets.

| VLAN number | Location | Use type | IPv6 hexadecimal |
|---|---|---|---|
| 0001 (**001**) | 0 (**0**) | 1 (**01**) | 2001:db8:1234:0**010**::/64 |
| 0529 (**211**) | 2 (**2**) | 17 (**11**) | 2001:db8:1234:0**112**::/64 |
| 4094 (**FFE**) | 15 (**F**) | 254 (**FE**) | 2001:db8:1234:0**fef**::/64 |

### 4.9.3 Decimal notation

A similar notation system can be used if the VLAN number is divided decimally. If, for example, the first two digits indicate the location and the last two digits the use type, this can be reversed in the IPv6 address. For example:

| VLAN number | Location | Use type | IPv6 decimal |
|---|---|---|---|
| **0001** | **00** | **01** | 2001:db8:1234:**0100**::/64 |
| **0529** | **05** | **29** | 2001:db8:1234:**2905**::/64 |
| **4094** | **40** | **94** | 2001:db8:1234:**9440**::/64 |

## 4.10 Addressing Point-to-point Links

If you use point-to-point links, using a /64 address may present problems in combination with certain router configurations. Unused addresses in the /64 system are bounced back by the routers on either side of the link. Data packages sent to this address will thus be sent back and forth between the routers like ping pong balls. This places an unwanted burden on the network. It might therefore, be practical in some cases to configure a /127 prefix instead of a /64 for these links.

**Please note:** this configuration often works, but it is not in accordance with IPv6 standards.

We therefore recommend reserving a /64 prefix for such links in the addressing plan, even if you use only a /127. As soon as the router configuration has been corrected by the supplier, you may proceed to configure a /64 prefix without having to modify the addressing plan.

# 5 DETAILED EXAMPLES

## 5.1 Assigning by Use Type Only

A use type-based subnet has been adopted in which the following groups are distinguished:

| | |
|---|---|
| Number of use types (students, staff, guests, servers): | Four groups |
| Backbone and other infrastructure: | One group |
| Total: | Five groups |

If we round this up to the first power of 2, this results in eight primary subnets. Incorporating these groups into the IPv6 address requires 3 bits ($2^3 = 8$). With three unused groups, there is enough space for future expansion.

We have used 3 of the available 16 bits; as a result, 13 still remain. We decide not to divide these into secondary subnets. To increase legibility, we use 4 bits per use type so that 12 bits remain. This leaves address space for 4,096 ($2^{12}$) possible networks per use type. Now there are 12 bits still available:

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2001:db8:1234: | G | G | G | G | B | B | B | B | B | B | B | B | B | B | B | B | ::/64 |

This results in the following address structure:

**2001:db8:1234:GBBB::/64**

The table below illustrates this:

| Use type (G) | Assignable (B) | Network |
|---|---|---|
| Infrastructure (**0**) | **0** | 2001:db8:1234:**0000**::/64 |
| Infrastructure (**0**) | **1** | 2001:db8:1234:**0001**::/64 |
| Infrastructure (**0**) | **12** | 2001:db8:1234:**000c**::/64 |
| Infrastructure (**0**) | **100** | 2001:db8:1234:**0064**::/64 |
| Students (**1**) | **0** | 2001:db8:1234:**1000**::/64 |
| Students (**1**) | **12** | 2001:db8:1234:**100c**::/64 |
| Students (**1**) | **321** | 2001:db8:1234:**1141**::/64 |

Etc.

## 5.2 Assigning by Use Types and Locations

We want to set up a use type-based primary subnet and a location-based secondary subnet.

The following use types are distinguished:

| | |
|---|---|
| Number of use types (students, staff, guests, servers): | Four groups |
| Backbone and other infrastructure: | One group |
| Total: | Five groups |

If we round this up to the first power of 2, this results in eight groups. Incorporating these groups into the IPv6 address requires 3 bits ($2^3$ = 8). With three unused groups, there is enough space for future expansion.

This example is based on 35 locations. If we use 6 bits, we will have address space for 64 (26) locations, which is more than enough.

We have now assigned 9 bits to the primary and secondary subnets, and so 7 remain.  We can use these 7 bits to create 128 ($2^7$) networks per use type per location.

In this example, we decide not to make any modifications to improve legibility. We recommend that you do make such modifications in practice, but in this example we wish to demonstrate the impact of a sub-optimum addressing plan on legibility.

We now have the following IPv6 address structure:

| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2001:db8:1234: | G | G | G | L | L | L | L | L | L | B | B | B | B | B | B | B | ::/64 |

This results in the following example addresses. It is obvious that the groups cannot be traced in the address:

| Use type | Location | Assignable | Network |
|---|---|---|---|
| Infrastructure (0) | Non-location-based (0) | 0 | 2001:db8:1234:**0000**::/64 |
| Infrastructure (0) | Non-location-based (0) | 1 | 2001:db8:1234:**0001**::/64 |
| Infrastructure (0) | Non-location-based (0) | 2 | 2001:db8:1234:0**002**::/64 |
| Infrastructure (0) | Location 1 | 0 | 2001:db8:1234:**0080**::/64 |
| Infrastructure (0) | Location 35 | 0 | 2001:db8:1234:**1180**::/64 |
| Students (1) | Non-location-based (0) | 0 | 2001:db8:1234:**2000**::/64 |

| | | | |
|---|---|---|---|
| Students (1) | Location 1 | 12 | 2001:db8:1234:**208c**::/64 |
| Students (1) | Location 35 | 9 | 2001:db8:1234:**3189**::/64 |

Etc.

## 5.3 Improving Legibility

Although the assignment method used in the previous example may well function perfectly, it makes it very difficult to decipher addresses. To improve legibility, we will divide the addresses into groups of 4 bits.

We will use 4 bits for the use type and 8 bits for the locations. This leaves us 4 bits to create networks per use type per location. It is important to check whether these 4 bits are sufficient. An example of a situation where 4 bits would be insufficient is if there are to be more than 16 (24) student networks per location. We can then use the extra leeway created by using an extra bit for the use type as described in section 2.4.

This results in the following situation:

| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2001:db8:1234: | G | G | G | G | L | L | L | L | L | L | L | L | B | B | B | B | ::/64 |

The following address structure is created:

2001:db8:1234:**GLLB**::/64

The table below illustrates this:

| Use type | Location | Assignable | Network |
|---|---|---|---|
| Infrastructure (**0**) | Non-location-based (**0**) | **0** | 2001:db8:1234:**0000**::/64 |
| Infrastructure (**0**) | | **1** | 2001:db8:1234:**0001**::/64 |
| Infrastructure (**0**) | | **2** | 2001:db8:1234:**0002**::/64 |
| Infrastructure (**0**) | Location **1** | **0** | 2001:db8:1234:**0010**::/64 |
| Infrastructure (**0**) | Location **35** | **0** | 2001:db8:1234:**0230**::/64 |
| Students (**1**) | Non-location-based (**0**) | **0** | 2001:db8:1234:**1000**::/64 |
| Students (**1**) | Location **1** | **12** | 2001:db8:1234:**101c**::/64 |

|  |  |  |  |
|---|---|---|---|
| Students (**1**) | Location **35** | **9** | 2001:db8:1234:**1239**::/64 |

Etc.

# 6 MANAGING HOSTS

## 6.1 Addressing Hosts

### 6.1.1 Introduction

Once we have an addressing plan for the IPv6 networks, we can proceed to address the hosts in the network. There are three common methods for doing this:

- StateLess Address Auto Configuration (SLAAC)
- Dynamic Host Configuration Protocol for IPv6 (DHCPv6)
- Static IPv6 addresses

We recommend automatic configuration via SLAAC or DHCPv6 for most clients because this makes management considerably easier. If properly implemented, it also increases the privacy of the users. A static configuration is recommended only for equipment such as routers, switches, firewalls and servers.

### 6.1.2 StateLess Address Auto Configuration (SLAAC)

SLAAC is the simplest way to link hosts to an IPv6 network. The router sends “Router Advertisements” (RAs) and the hosts use the information in the RA in combination with their MAC address to assign an IPv6 address.

If multiple routers in a single network send RAs, the hosts will listen to both RAs and configure addresses on both networks. This feature can be used to incorporate a degree of redundancy.

### 6.1.3 Privacy extensions

Some operating systems have privacy extensions in addition to SLAAC. Depending on the operating system, these will be either active or inactive by default. Privacy extensions ensure that third parties cannot see what kind of network card a host has. In standard SLAAC this is visible because the MAC address is used for logging on to a network. Because the same MAC address is used for logging on to various networks (for example, with a smartphone), third parties can see that one and the same host is being used. Privacy extensions use multiple random addresses per host to prevent the host from being traced.

However, the use of privacy extensions in combination with SLAAC is problematic if network managers wish to trace who is using which IPv6 address and when. One solution is to use centrally coordinated address assignment via DHCPv6 (see section 6.1.4). However, this is not supported by all operating systems at present. One common operating system that does not support DHCPv6 is Mac OS X. Also, DHCPv6 software is not installed as standard by all Linux and BSD distributors.

### 6.1.4 Dynamic Host Configuration Protocol for IPv6 (DHCPv6)

DHCPv6 can be implemented in two ways:

- DHCPv6 can collaborate with SLAAC to provide hosts with all the information that SLAAC does not provide, such as DNS domains and servers and NTP servers.
- DHCPv6 can be used as a substitute for SLAAC. In this case the IPv6 addresses are explicitly distributed by the DHCPv6 server.

DHCPv6 can provide more control, but it is not supported by all hosts. If addresses are assigned with DHCPv6, it is recommended to program the switches so that hosts can only use the addresses assigned to them via DHCPv6.

**Please note:** this is not standard for either IPv4 or IPv6. Many suppliers have implemented their own security policies for IPv4. There are only a few switches available that offer this security with IPv6.Therefore, traceability based on the MAC address would still be the best option in this situation.

### 6.1.5 Static Addresses

We recommend assigning static IPv6 addresses only for equipment such as routers, switches, firewalls and servers. Automatic configuration of such equipment will lead to problems in the long run. For example, if the network adaptor on a server is replaced, the SLAAC address of that server will also change, and there is a major risk that the person replacing the network adaptor will forget to modify the DNS entries for the server.

To increase the traceability of vital equipment, we recommend incorporating all or part of the IPv4 address into the IPv6 address. This is explained in more detail in section 3.2.

## APPENDIX: RATIO BETWEEN GROUPS AND BITS

| Number of groups | Number of bits |
|---|---|
| 2 | 1 |
| 4 | 2 |
| 8 | 3 |
| 16 | 4 |
| 32 | 5 |
| 64 | 6 |
| 128 | 7 |
| 256 | 8 |
| 512 | 9 |
| 1024 | 10 |

| | |
|---|---|
| 2048 | 11 |
| 4096 | 12 |
| 8192 | 13 |
| 16384 | 14 |
| 32768 | 15 |
| 65536 | 16 |

## Acknowledgements

This manual was produced with the assistance of the following people, whom we would like to thank:

Sander Steffann, author

Roel Hoek (University of Twente)

Jeroen van Ingen Schenau (University of Twente)

Magda Pattiapon (RUG)

Mente Heemstra (RUG)

Rogier Spoor (SURFnet)

Wim Biemolt (SURFnet)

Niels den Otter (SURFnet)

Maurice van den Akker (SURFnet)

Jan Michielsen (SURFnet)

Translation provided by RIPE NCC