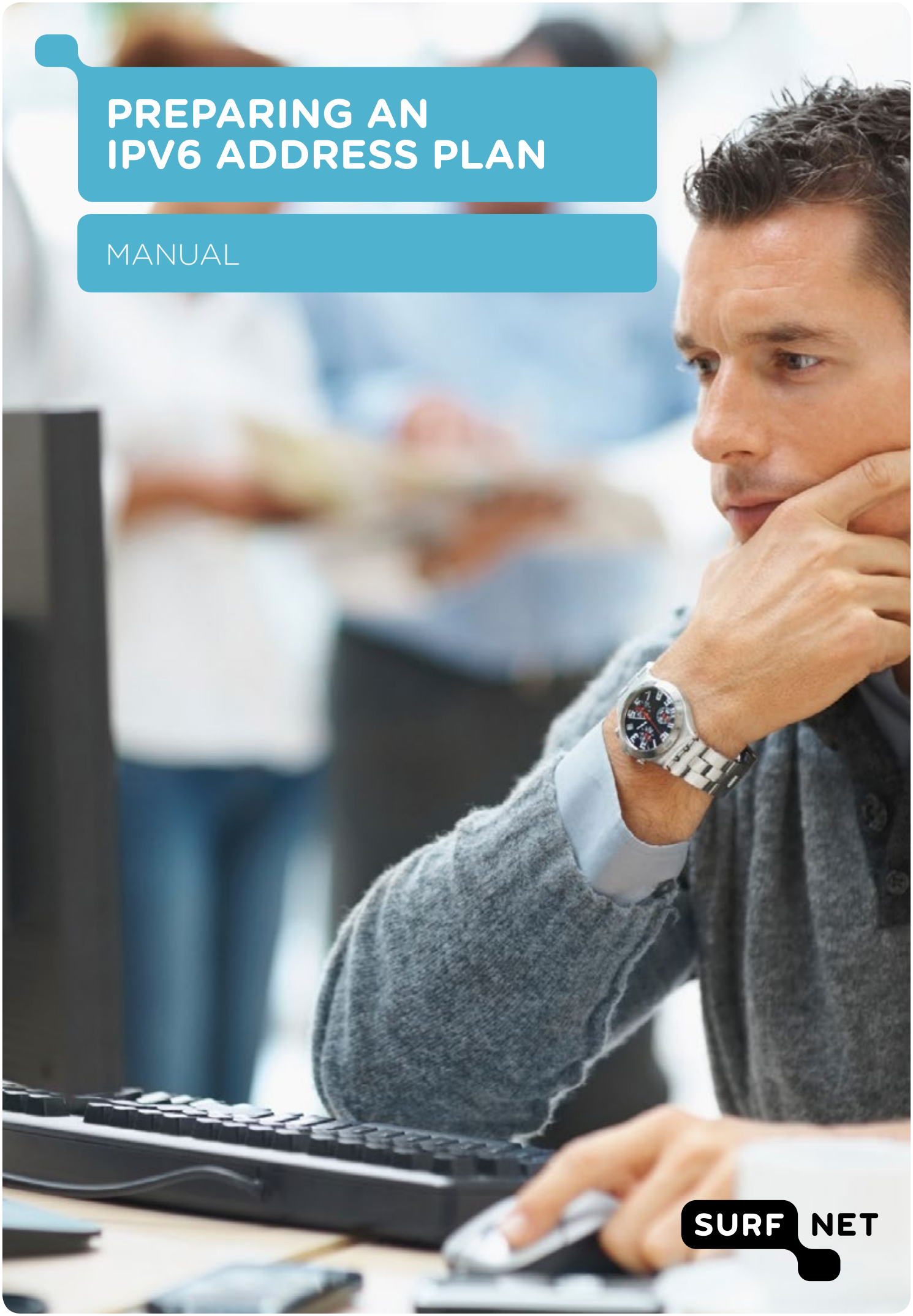
PREPARING AN IPV6 ADDRESS PLAN

MANUAL



SURF NET

PREPARING AN IPV6 ADDRESS PLAN

MANUAL

Version 2, 18 September 2013

CONTENTS

1. INTRODUCTION

IPv4 addresses have almost run out, and more and more businesses and institutions see the necessity to migrate to IPv6. As a result, they need an IPv6 address plan. An IPv6 address is 128 bits long, 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} or 340 282 366 920 938 463 463 374 607 431 768 211 456 or 340 billion billion billion billion represents approximately the number of grains of sand on our planet. This means that an IPv6 address plan will look very different from an IPv4 address plan.

An address 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 Internet Service Providers, you will be assigned 2^{80} addresses (a /48 prefix). This is such a huge amount that efficiency virtually ceases to be an issue. This is why it is worthwhile adopting an IPv6 address plan: a system in which you assign the IPv6 addresses to locations and/or use types.

In an efficient IPv6 address 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 address plan is scalable: it can be expanded, for example, to include new locations or use types
- An efficient IPv6 address plan also enables more efficient network management

However, an efficient IPv6 address plan may "waste" large numbers of IPv6 addresses. In almost all cases, this is a good trade-off: seemingly wasteful practices lead to more efficiency elsewhere, for instance, by avoiding unnecessary inflation of routing tables in routers. The addresses are there; you may as well use them.

This manual will show you how to prepare an effective IPv6 address 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 address plan will meet the requirements of your organisation. This manual will provide suggestions to help you to make the right choices.

1.1. 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

IPv6 addresses are 128 bits long, four times as long as IPv4 addresses. As a result, they're written down differently, and the large number of bits to work with allows for more internal structure within the IPv6 address.

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[1], which is much easier to understand for humans 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:

2001:0db8:0000:0000:0000:0000:0000:0001

Since it is impractical to record all these zeroes, some may be skipped in accordance with certain conventions. Leading zeroes can be dropped for each group of digits. The result would then be:

2001:db8:0:0:0:0:0:1

One (and only one) series of zeroes and colons may also be abbreviated as two colons. The result is now:

2001:db8::1

The precise rules for IPv6 address notation are specified in RFC 5952[2].

2.2. Prefixes: Grouping Addresses

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, similar to an area code for phone numbers (e.g., the prefix for Amsterdam is 020). 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

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

[1] http://betterexplained.com/articles/numbers-and-bases/

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

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

through

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

(Don't be fooled by the missing zeroes in 2001:db8:1234::/64 rather than 2001:db8:1234:0000::/64.)

Prefixes that are multiples of four bits are easiest to work with, so those are common. Examples are /32, /48, /52, /56, /60 and /64. If a prefix falls on a different boundary, this means it "slices" through a a hexadecimal digit, rendering the address range 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.

With IPv4, it's at least theoretically possible to have a non-contiguous subnet mask. For instance, with subnet mask 255.255.252.255 the addresses 192.0.2.3 and 192.0.3.3 are in

the same subnet, but 192.0.2.3 and 192.0.2.4 aren't. However, non-contiguous subnet masks have no corresponding prefix length. As IPv6 subnets are defined using prefix lengths, non-contiguous subnets are not possible with IPv6.

2.3. /64 Subnets

IPv6 addresses don't have a fixed structure, like the class A/B/C system originally used with IPv4. However, IPv6 subnets should be /64 prefixes. Other subnet sizes are possible, but may get in the way of mechanisms such as stateless address autoconfiguration (see section 5.1). So very small subnets, such as a point-to-point link, use the same size IPv6 address block as very large subnets, such as a large Ethernet containing a number of Ethernet switches.

2.4. Assigning Address Blocks

The original recommendation for assigning IPv6 address space to end users was as follows:

- **/48** (65 536 subnets) in the general case, except for very large subscribers
- **/64** (a single subnet) when it is known that one and only one subnet is needed by design
- **/128** (a single address) when it is absolutely known that one and only one device is connecting

However, RFC 6177[1] (also known as Best Current Practice 157) changes this, and recommends using an address block / prefix size tailored to the end user's needs. It also recommends against giving out single addresses. For instance, a /48 is much more than a home user needs, but a /64 only allows for a single subnet, which may be limiting, if not immediately, then in the future. So a /56 or /60 may be more appropriate for consumers.

That said, it is important to err on the side of assigning more rather than less, as adding a second address block or moving to a bigger one is costly. This is especially true when the original assignment to a medium-sized or larger organisation was a /56. With a /56 assignment, the network will already be fairly large before more address space is needed, so the impact of changes will be significant. If the original assignment was a /60, the network would have outgrown the assignment when it was much smaller and changes were still much easier to make.

So a /48 should be used when there is any doubt whether a /56 is sufficient in the long run. ISPs get much leeway in determining the prefix size they give to their customers up to /48–even in the case of home users.

ISPs which are LIRs (Local Internet Registries, sometimes called a "RIPE member" in Europe) get at least a /32, but large ISPs can get much larger address blocks / shorter prefixes, so they can use a dedicated sub-prefix per region/country where they're active.

In the rest of this document, it is assumed that your organisation has been assigned a /48 address block, and that 16 bits (64 - 48) are therefore available for assigning the addresses to subnets. 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

[1] RFCs are available from http://tools.ietf.org/html/

through

2001:db8:1234:ffff::/64

for your network–16 bits in total.

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

2.5. Representation of Subdivisions

As the first 48 bits are assigned by a service provider and the last 64 bits are used within each subnet, an IPv6 address plan is about bits 48 to 63[1], the 16 bits available to number subnets. 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
- T: bit is assigned to a use type

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

2001:db8:1234:	L	L	L	L	T	T	T	T	B	B	B	B	B	B	B	B	::/64

Each box represents 1 bit. Four boxes together represent a nibble (four bits) and thus one hexadecimal digit in the IPv6 address. For the above example, this produces the following address structure:

2001:db8:1234:LTBB::/64

2001:db8:1234:TLBB::/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 WITHOUT AN ADDRESS 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 address 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 address plan.

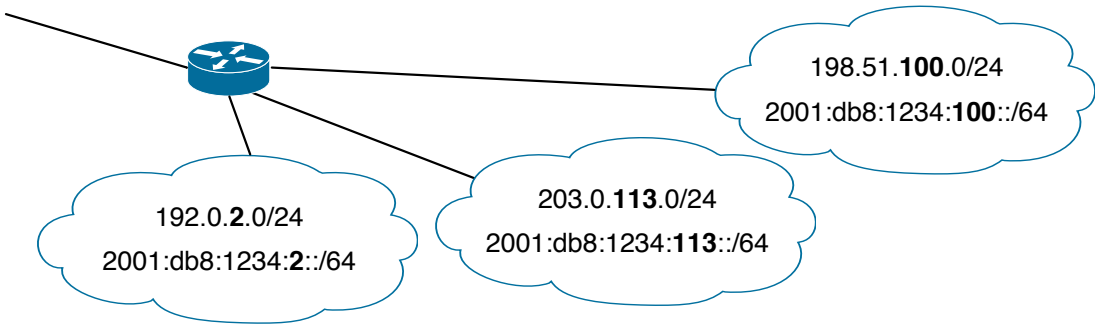
Should you opt to work without an address plan, we recommend keeping a list of the assigned networks in a central location, such as an Excel spreadsheet, an internal Wiki page, a configuration management database (CMDB) or in the reverse DNS configuration.

[1] The IETF numbers bits starting at zero from left (most significant) to right (least significant).

3.1. Direct Link Between IPv4 and IPv6 Subnets

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 address plan, the link between the existing IPv4 networks and the IPv6 networks is immediately visible. However, this scheme only works well when IPv4 subnets are /24s.

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.

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.

The reason this approach only works with /24 IPv4 (sub-)networks is that with a different (sub)net size, the IPv6 subnets can't be /64s while making each IPv4 subnet an IPv6 subnet and vice versa. For instance, with a /28 subnet size in effect, the two addresses 172.31.5.14 and 172.31.5.18 are in different IPv4 subnets (172.31.5.0/28 and 172.31.5.16/28), but when mapped to 2001:db8:1234:5::14 and 2001:db8:1234:5::18, they would be in the same IPv6 /64. If the subnet size is larger than /24, like with 10.0.8.250 and 10.0.9.5, which are both in 10.0.8.0/23, the addresses would map to 2001:db8:1234:8::250 and 2001:db8:1234:9::5, which are in different /64s.

Note that this is *not* an example of efficient addressing plan as discussed in chapter 1.

3.2. Direct Link Between IPv4 and IPv6 Addresses

If the IPv4 subnet size is not /24, it will be impossible to maintain a direct relationship between IPv4 and IPv6 subnets. In that case, it can still be useful to include the (whole) IPv4 address in the IPv6 address. For instance, by giving the system with IPv4 address 192.0.2.123 the IPv6 address 2001:db8:1234:2:192:0:2:123.

Note that there is a variation in IPv6 notation that allows the inclusion of an IPv4 address, but in this case the IPv4 address must occupy the lowest 32 bits of the IPv6 address, for example:

2001:db8:1234:c0:ff:ee:192.0.2.123

However, the fact that an address was entered in this form is generally not retained, so when displaying network settings, it will generally appear in standard IPv6 notation:

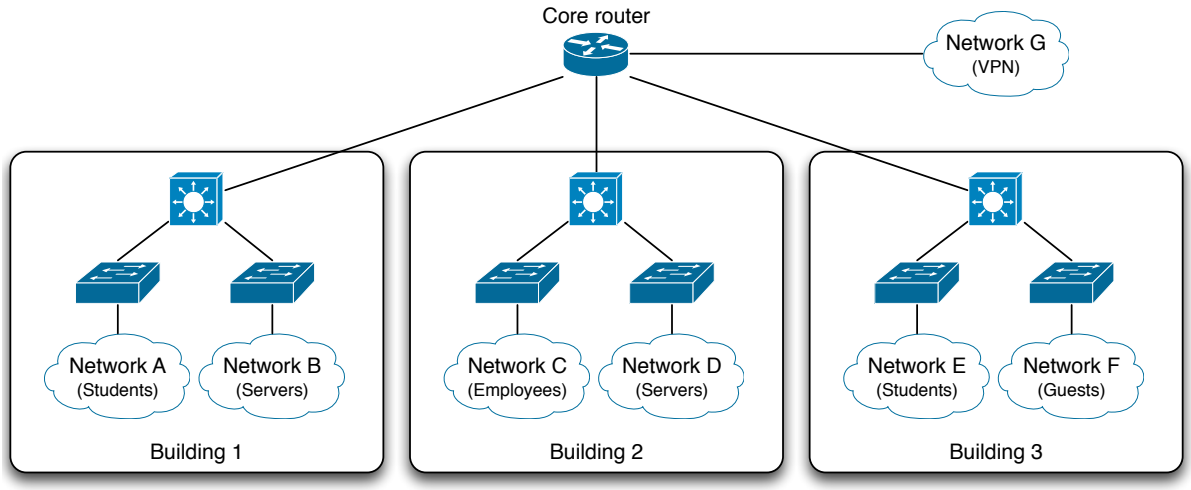
2001:db8:1234:c0:ff:ee:c000:27b

At this point, the relationship with IPv4 address 192.0.2.123 is no longer obvious.

4. Preparing an Address Plan

When preparing your address 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 address plans, using the following example network:



4.1. Basic Structure of the Address 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 2.5:

2001:db8:1234:	L	L	L	L	T	T	T	T	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 (T). As a result, there are 8 bits remaining (B). Following this address plan, with 4 location bits, a maximum of 16 locations can be addressed, with each location having 16 (also 4 bits) allocatable use types. 8 bits are left to number a maximum of 256 different subnets for each location and use type combination.

4.2. IPv6, NAT and Firewalls

With IPv4, Network Address Translation (NAT) is widely used to allow multiple hosts to gain access to the public internet using a single IPv4 address. With IPv6, there is no longer a lack of address space, so using NAT for this purpose is no longer necessary.

A side effect of NAT is that it (mostly) prevents unsolicited packets from the public internet from reaching hosts residing behind the NAT device. As such, protocols that require two hosts that both reside behind a NAT to communicate directly (peer-to-peer protocols) need to employ workarounds to set up communication through NAT. However, IPv6 peer-to-peer applications, and some applications that aren't obviously peer-to-peer, such as FTP, may not have these workarounds built in, or they may not be compatible with IPv6. So networks deploying NAT with IPv6 are likely to see more NAT-related problems than in a similar IPv4 deployment. Also, there is no RFC describing IPv6 NAT, only IPv6 IPv6-to-IPv6 Network Prefix Translation (RFC 6296, with status "experimental").

To prevent unwanted packets coming in from the public internet from reaching hosts, a stateful firewall may be employed, in order to make sure that incoming packets match communication sessions set up by internal hosts. See RFC 4864 for more information.

4.3. Router vs Firewall: Location or Use Type First

We first need to decide on whether to use location first, and then use type (such as students, staff, servers, switches, routers, public, etc.), or the other way around. These options are discussed below.

4.3.1. Location First

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.

4.3.2. Use Type First

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.

The advantage of first grouping all subnets by use type is that it makes 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 address plan
- The routers cannot process such a large number of routes without aggregation

4.4. Determining the Address Space Required for the Address Plan

Now we need to determine which portion of the 16 bits of available address space (see section 2.4) is required for the address plan selected, depending on how many groups of use types and locations there are. For both, the number of groups determines how many of the 16 bits

available must be used. One bit can contain two groups (2^1), 2 bits can contain 4 groups (2^2), etc. (see the table).

We can calculate 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 choose 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.

To create a practical address 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 groups counted in steps 1 to 4 to the nearest power of 2. See the table for the number of bits required for each possible number of groups.

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.

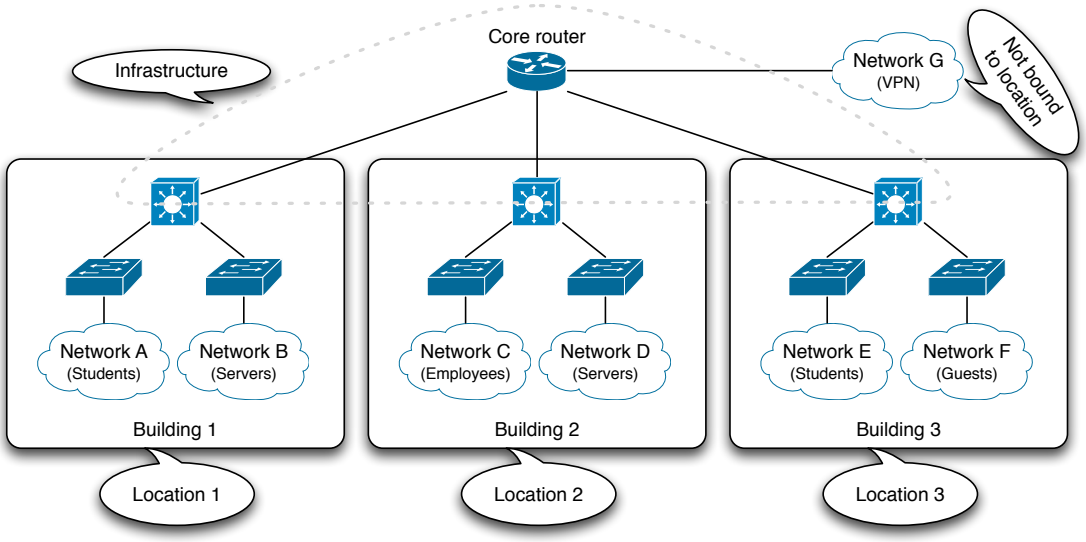
Bits	Locations or use types
1	2
2	3 or 4
3	5 - 8
4	9 - 16
5	17 - 32
6	33 - 64
7	65 - 128
8	129 - 256
9	257 - 512
10	513 - 1024
11	1025 - 2048
12	2049 - 4096

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). 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	::/64

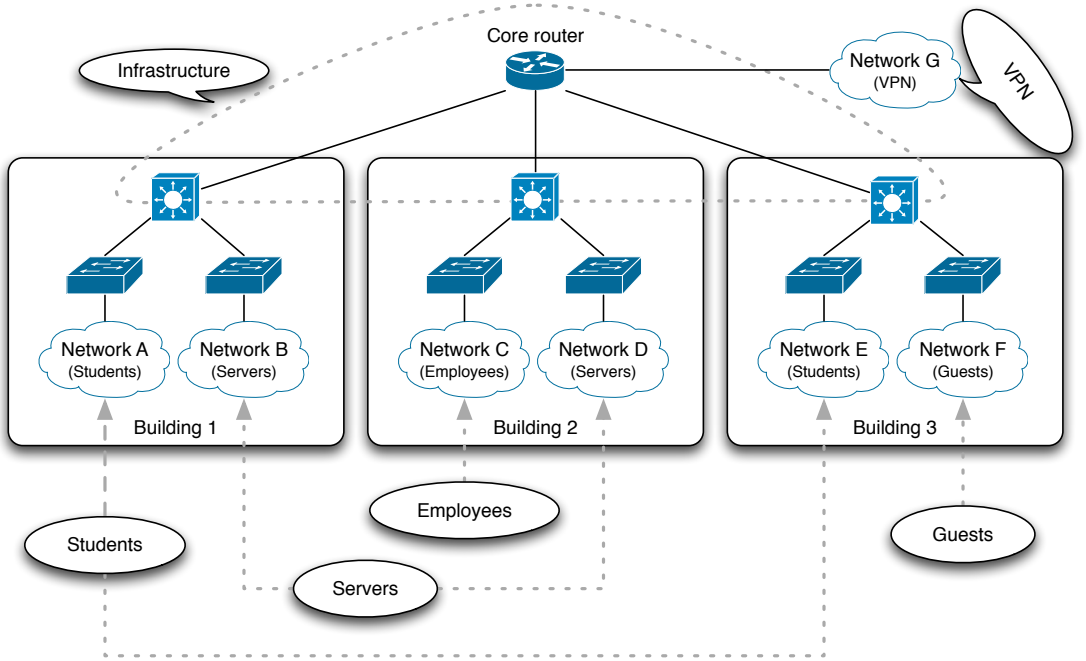
This leaves 13 available bits (B).

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 (T) ($2^4 = 16$). This leaves 12 available bits (B).

2001:db8:1234:	T	T	T	T	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 address 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	T	T	T	T	B	B	B	B	B	B	B	B	B	::/64

In this address 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:	T	T	T	T	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. Double Check

We can check whether the address 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:	T	T	T	T	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 an IPv6 address is equivalent to 4 bits. By choosing an assignment system based on multiples of 4 bits, each group will correspond to a digit in the IPv6 address.

In the example in this section, we used 3 bits per location and 4 bits per use type. However, the hexadecimal digits in the IPv6 address each represent 4 bits. So if the location or use type only needs two or three bits, it can be useful to make it 4 bits instead, and make IPv6 addresses easier to read. For example:

2001:db8:1234:	L	L	L	T	T	T	T	B	B	B	B	B	B	B	B	B	::/64

2001:db8:1234:	T	T	T	L	L	L	L	B	B	B	B	B	B	B	B	B	::/64

Respectively become:

2001:db8:1234:	L	L	L	L	T	T	T	T	B	B	B	B	B	B	B	B	::/64

2001:db8:1234:	T	T	T	T	L	L	L	L	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 T bits are also displayed as one hexadecimal digit. This results in the following IPv6 address structure:

2001:db8:1234:LTBB::/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 (T).

4.9. Flexibility for Future Growth

If the number of locations and/or use types may grow in ways that are hard to predict at the time the address plan is created, it may be beneficial to keep the boundaries between the different groups of bits as flexible as possible. This can be done using the strategy outlined in RFCs 1219 and 3531. The downsides of this approach is that a good understanding of bit manipulation is required, and from time to time, as the boundaries between the fields move, firewall rules and the like need to be updated. Assuming that the location occupies the top bits and the use type comes next, an address plan with flexible boundaries may start as follows, with five locations, three use types and two subnets per location/use type:

2001:db8:1234:	L	L	L						T	T						B	::/64

Then, perhaps the number of subnets per use type goes up from two to ten, requiring four bits:

2001:db8:1234:	L	L	L						T	T			B	B	B	B	::/64

After this, the number of use types increases to five, requiring a third bit. There are more free bits on the left side, so the use type field grows in that direction:

2001:db8:1234:	L	L	L					T	T	T			B	B	B	B	::/64

The number of locations grows to 50, requiring six bits:

2001:db8:1234:	L	L	L	L	L	L		T	T	T			B	B	B	B	::/64

The number of use types grows to 13, requiring a fourth bit, which is taken from the right where there are more remaining bits:

2001:db8:1234:	L	L	L	L	L	L		T	T	T	T		B	B	B	B	::/64

And so on. Note that the L and T fields have gained bits to the right, which means that in order to avoid renumbering, encoding a number in these bits must be done in a somewhat unusual way. Please refer to RFC 3531 for more information.

4.10. Using VLAN Numbers

Another approach is to use VLAN numbers as the subnet number. In networks where most subnets are VLANs and thus already have a VLAN number, this simplifies the administration of VLAN and subnet numbers, because now only one number has to be managed.

VLAN numbers are 12 bits in size, while subnet numbers are 16 bits (assuming a /48 prefix for the organisation). So it's possible to simply use either of the following approaches:

2001:db8:1234:	V	V	V	V	V	V	V	V	V	V	V	V	B	B	B	B	::/64
2001:db8:1234:	B	B	B	B	V	V	V	V	V	V	V	V	V	V	V	V	::/64

However, IPv6 addresses are written as hexadecimal, but VLAN numbers are written in decimal. So the above approaches require conversion between hex and decimal, obscuring the relationship between the VLAN and subnet numbers and thereby negating the advantage of simplicity.

A preferable approach is to take the decimal VLAN number, and use it in the place of the hexadecimal subnet number. So VLAN 2783 simply becomes subnet 2001:db8:1234:2783::/64. Note that this leaves the subnet numbers above 4095 as well as any subnet number with one or more letters in it available for additional subnets that aren't tied to a VLAN. The table shows both options and the two hexadecimal variations.

VLAN ID	IPv6 decimal	IPv6 hexadecimal (high)	IPv6 hexadecimal (low)
1	2001:db8:1234:000**1**::/64	2001:db8:1234:00**10**::/64	2001:db8:1234:000**1**::/64
12	2001:db8:1234:00**12**::/64	2001:db8:1234:00**c0**::/64	2001:db8:1234:000**c**::/64
2783	2001:db8:1234:**2783**::/64	2001:db8:1234:**adf0**::/64	2001:db8:1234:0**adf**::/64
4094	2001:db8:1234:**4094**::/64	2001:db8:1234:**ffe0**::/64	2001:db8:1234:0**ffe**::/64

In this approach, 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 unless this was planned for when assigning VLAN IDs. With hexadecimal encodings, four bits remain free for other purposes, such as encoding the location.

4.11. VLAN Numbers That Encode Location/Use Type

It is also possible to combine the encoding of location and use type with 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. In this case, it's easiest to stick to the 4-bit boundaries and not use numbers higher than 9 in one of the following schemes.

This is an example of encoding location first and then use type into the VLAN number and then the VLAN number into the subnet bits of the IPv6 address:

2001:db8:1234:	V	V	V	V	V↓L	V↓L	V↓L	V↓L	V↓T	V↓T	V↓T	V↓T	V↓B	V↓B	V↓B	V↓B	::/64

And this is an example of encoding the use type first and then the location into the VLAN number and then the VLAN number into the subnet bits of the IPv6 address:

2001:db8:1234:	V	V	V	V	V↓T	V↓T	V↓T	V↓T	V↓L	V↓L	V↓L	V↓L	V↓B	V↓B	V↓B	V↓B	::/64

Unfortunately, such strict subdivision of the bits and the imposed decimal/hex compatibility reduce the number of usable subnets significantly. So only small to medium sized organisations will be able to stick to such a numbering plan as the network grows.

4.11.1. Reversing VLAN Notation 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 parentheses:

VLAN number	Location	Use Type
0001 (**0**01)	0 (**0**)	1 (**0**1)
0529 (**2**11)	2 (**2**)	17 (**1**1)
4094 (**F**FE)	15 (**F**)	254 (**F**E)

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.

4.11.2. 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 parentheses.

VLAN number	Location	Use type	IPv6 hexadecimal
0001 (**0**01)	0 (**0**)	1 (**0**1)	2001:db8:1234:00**1**0::/64
0529 (**2**11)	2 (**2**)	17 (**1**1)	2001:db8:1234:01**1**2::/64
4094 (FFE)	15 (F)	254 (FE)	2001:db8:1234:0**fe**f::/64

4.11.3. 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:**01**00::/64
0529	**05**	29	2001:db8:1234:**29**05::/64
4094	**40**	94	2001:db8:1234:**94**40::/64

4.12. 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. In some cases, unused addresses in the /64 system may be bounced back by the routers on either side of the link. Data packets 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 prefix longer than /64 for these links.

Please note: using a subnet prefix size other than /64 goes against RFC 4291[1].

But if not /64, what is the appropriate subnet size for point-to-point links?

- **/127** is possible because IPv6 doesn't have any broadcast addresses. However, the all-zeroes address in each subnet is the "all routers anycast address", which means that all routers are supposed to receive packets at that address. Some router vendors do not implement this feature, so /127 subnets work on their routers. But problems may arise when this equipment is replaced with that from another vendor.
- **/126** allows the all-zeroes address to be skipped. However, the highest 128 addresses in any subnet are also reserved for various anycast addresses (RFC 2526). But in practice this is typically not problematic.
- **/120** makes it possible to avoid all reserved anycast addresses.
- **/112** makes it possible to avoid reserved anycast addresses and has the advantage that the entire four-digit hex value after the last colon in the IPv6 address identifies systems within the subnet.

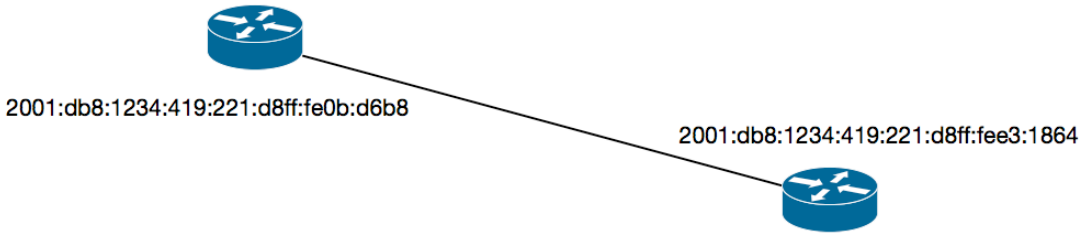
So /112 seems like the best non-/64 alternative. But it may also be useful to minimize the number of addresses in a subnet to avoid neighbor cache exhaustion attacks, where an attacker scans all the addresses in a subnet and routers must execute Neighbor Discovery for all those addresses, exhausting their resources. So an alternative may be a /126 or /120 but reserve a /112 for human readability or even an entire /64 so it's still possible to switch to /64 at a later date.

4.13. EUI-64 Addressing

On subnets with only routers present, it can be useful to have the routers generate the lower 64 bits of their IPv6 address in that subnet automatically. This way, there is no need to track which router has which address. It may be possible to configure the use of "EUI-64" (the 64-bit form of an Ethernet MAC address) addressing. Routers will then generate an IPv6-address from their MAC address similar to how hosts may generate the lower 64 bits of their IPv6 address from their MAC address when stateless address autoconfiguration is used. Like stateless address autoconfiguration, EUI-64 addressing is only possible on /64 subnets.

The figure below shows a subnet containing wo Cisco routers configured with "ipv6 address 2001:db8:1234:419::/64 eui-64" on the interface connecting to the subnet. Despite the identical configuration, each router configures a unique address.

[1] http://tools.ietf.org/html/rfc4291#section-2.5.4



5. MANAGING AND ADDRESSING HOSTS

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

- Stateless address autoconfiguration
- Dynamic Host Configuration Protocol for IPv6 (DHCPv6)
- Manual configuration

We recommend automatic configuration via stateless address autoconfiguration (sometimes unofficially abbreviated as SLAAC) or DHCPv6 for most clients because this makes management considerably easier. If properly implemented, it also increases the privacy of the users. Manual configuration is recommended only for equipment such as routers, switches, firewalls and servers.

5.1. Stateless Address Autoconfiguration

Stateless address autoconfiguration 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. Ethernet MAC addresses are 48 bits, but these are extended to 64 bits first by adding the bits FFFE in the middle, creating a 64-bit Extended Unique Identifier (EUI-64). Then, the unique/local bit in the MAC address is flipped to avoid manually configured addresses from looking like addresses derived from a globally unique MAC address. The lower 64 bits in an IPv6 address are known as the "interface identifier". For example, the MAC address 04:0c:ce:e9:38:60 results in the interface identifier 60c:ceff:fee9:3860.

An RA may contain multiple prefixes. If multiple routers on a single subnet send RAs, the hosts will listen to all RAs sent by all routers and configure addresses in all prefixes in all RAs. This feature can be used to incorporate a degree of redundancy.

5.2. Privacy Addresses

Current mainstream operating systems use privacy extensions in addition to EUI-64 based stateless address autoconfiguration. These privacy extensions avoid the situation where a host may connect to different IPv6 networks at different times and is then tracked by remote systems that observe the same MAC address in connections made from those different networks. The privacy extensions work by using a random number as the interface identifier rather than an EUI-64. A new random number is generated every 24 hours or whenever the system (re)connects to a network.

Typically, hosts generate a MAC based address as well as a privacy address, and then use the privacy address for outgoing connections. The MAC address based address may be used as a stable address for receiving incoming connections.

5.3. Dynamic Host Configuration Protocol for IPv6 (DHCPv6)

The use of privacy addresses can be problematic if network managers wish to trace who is using which IPv6 address and when. The solution is to use centrally coordinated address assignment via DHCPv6. Until a few years ago Mac OS X didn't support DHCPv6. However, in Mac OS X 10.7 DHCPv6 support was added. Windows has supported DHCPv6 as of Windows Vista. Today all mainstream operating systems support DHCPv6, but DHCPv6 software may not be installed by default on all Linux and BSD distributions.

DHCPv6 can provide IPv6 addresses as well as other information, such as nameserver addresses, similar to how IPv4 DHCP works. It can be deployed in two ways:

- DHCPv6 is used to hand out IPv6 addresses as well as other information.
- Stateless address autoconfiguration is used to configure IPv6 addresses, DHCPv6 is used to hand out other information.

Two flags in router advertisements determine which option is selected by hosts. For this reason, and also because DHCPv6 doesn't provide a default gateway address, IPv6 routers must always send router advertisements to hosts, even when stateless address autoconfiguration isn't used.

Note that depending on how RAs are configured, DHCPv6 and stateless address autoconfiguration may be active at the same time, resulting in hosts obtaining two addresses, three if privacy extensions are used.

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. So then the only remedy in case of problems is tracing the MAC address of the misbehaving system.

5.4. Manual Address Configuration

We recommend manually 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 stateless address autoconfiguration 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 3.1 and section 3.2.

5.5. Router Advertisement Guard

Sometimes incorrectly configured (home) routers and hosts to send out IPv6 router advertisements advertising non-working IPv6 connectivity. As many hosts prefer IPv6 connectivity (when available) over IPv4 connectivity, this means that destinations that have an IPv6 address in the DNS become unreachable. When this happens by accident, it's a nuisance. It's also possible that RAs are maliciously injected to reroute traffic so it can be observed or manipulated.

RFC 6105 proposes Router Advertisement Guard, a system that prevents undesired RAs from reaching IPv6 hosts. RA Guard is a feature that is implemented in layer 2 devices (i.e., Ethernet switches). These filter out RAs that come from devices that have no business sending them, while allowing through the RAs from the "real" IPv6 router or routers.

When buying Ethernet switches and Wi-Fi base stations, investigate whether they support RA Guard.

5.6. DNS Considerations

DNS addresses are the addresses that are typed in most frequently. As such, it's useful for them to be short. They appear in many configuration files locally, and if a DNS server hosts one or more domains, its address will also be listed in the configurations of far away servers. This makes it important that DNS addresses are stable over time. With these two considerations in mind, it's useful to give each DNS server its own /64 subnet, with a short manually configured address. For instance:

DNS1: 2001:db8:1234:a::53

DNS2: 2001:db8:1234:b::53

Manual configuration of a static address removes dependencies from DHCPv6, stateless address autoconfiguration and the server's MAC address. Having a separate /64 allows the server to be moved physically without the need to renumber; the entire subnet simply moves with the server.

With IPv4, it's standard practice to generate a DNS zone file with a name for each possible address. So without further action, any host that is connected to the network has a name in the DNS–although it may be an ugly one.

With IPv6, this is no longer possible, as the number of addresses per address block is way too large. In the case of manual configuration of addresses, it is of course no problem to add the addresses in question to the DNS. With DHCPv6, either the DHCPv6 server can be configured to hand out the same address to a given system every time, so that address can be entered in the DNS. However, this is a lot of work. A better solution is to configure the DHCPv6 server to do a dynamic DNS update whenever it gives out an address lease. But it is straightforward to generate DNS names for the entire pool of addresses used by the DHCPv6 server.

With stateless address autoconfiguration, hosts create their own addresses. In the case of MAC address based addresses of fixed location machines, it's possible to add their addresses to the DNS manually. But for mobile hosts and privacy addresses, this is not workable. It is possible to run a dynamic DNS (RFC 2136) client on such hosts, but those clients need to have the right settings and may not update the reverse DNS.

As a result of the above, it is very common for IPv6 hosts to not have a valid reverse DNS name. As such, it is not recommended to check for reverse DNS name as part of access restrictions.

ACKNOWLEDGEMENTS

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

- 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)
- Thejo van Vlaanderen (Routz Group)
- Joost Tholhuijsen
- Peter Verhage (HZ University of Applied Sciences)
- Sander Steffann (text version 1)
- Miguel Bastos and Nathalie Trenaman (RIPE NCC) (design)
- Olly Pekelharing (English translation version 1)
- Iljitsch van Beijnum (text version 2)

APPENDIX: DETAILED EXAMPLES

This appendix shows several more detailed examples of the address plan options explained in section 4.1 and later.

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 4096 (2^{12}) possible networks per use type. Now there are 12 bits still available:

2001:db8:1234:	T	T	T	T	B	B	B	B	B	B	B	B	B	B	B	B	::/64

This results in the following address structure:

2001:db8:1234:TBBB::/64

The table below illustrates this:

Use type (T)	Assignable (B)	Network
Infrastructure (**0**)	**0**	2001:db8:1234:**0**000::/64
Infrastructure (**0**)	**1**	2001:db8:1234:**0**001::/64
Infrastructure (**0**)	**12**	2001:db8:1234:**0**00c::/64
Infrastructure (**0**)	**100**	2001:db8:1234:**0**064::/64
Students (**1**)	**0**	2001:db8:1234:**1**000::/64
Students (**1**)	**12**	2001:db8:1234:**1**00c::/64
Students (**1**)	**321**	2001:db8:1234:**1**141::/64
Etc.		

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 (2^6) 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 address plan on legibility.

We now have the following IPv6 address structure:

2001:db8:1234:	T	T	T	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	Assign-able	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:**0002**::/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.			

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, as explained in section 4.8.

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 (2^4) 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 2.5.

This results in the following situation:

2001:db8:1234:	T	T	T	T	L	L	L	L	L	L	L	L	B	B	B	B	::/64

The following address structure is created:

2001:db8:1234:TLLB::/64

The table below illustrates this:

Use type	Location	Assign-able	Network
Infrastructure (**0**)	Non-location-based (**0**)	0	2001:db8:1234:**000**0::/64
Infrastructure (**0**)		1	2001:db8:1234:**000**1::/64
Infrastructure (**0**)		2	2001:db8:1234:**000**2::/64
Infrastructure (**0**)	Location**1**	0	2001:db8:1234:**001**0::/64
Infrastructure (**0**)	Location**35**	0	2001:db8:1234:**023**0::/64
Students (**1**)	Non-location-based (**0**)	0	2001:db8:1234:**100**0::/64
Students (**1**)	Location **1**	12	2001:db8:1234:**101**c::/64
Students (**1**)	Location **35**	9	2001:db8:1234:**123**9::/64
Etc.			

SURFnet
Radboudkwartier 273

PO Box 19035
3501 DA Utrecht
The Netherlands

T +31 (0)30 2 305 305
F +31 (0)30 2 305 329

admin@surfnet.nl
www.surfnet.nl