IPv6 Autoconfiguration

Best Practice Document

Produced by the CESNET-led working group on IPv6
(CBPD117)

Authors:
Tomáš Podermański,
Matěj Grégr
August 2011

Document No:	GN3-NA3-T4-CBPD117
Version / date:	1 August 2011
Original language :	Czech
Original title:	“IPv6 Mýty a skutečnost, díl IV. - Podpora autokonfigurace”
Original version / date:	1 of 1 August 2011
Contact:	tpoder@cis.vutbr.cz

This translated version is based on the Czech version published in the electronic journal, Lupa.cz, on 4 March 2011.

The translation of this report has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 238875, relating to the project 'Multi-Gigabit European Research and Education Network and Associated Services (GN3)'.

Table of Contents

Executive Summary

One of the basic requirements of the IPv6 protocol is support for the autoconfiguration of network nodes. This document details the options for autoconfiguration in current operating systems and gives an overview of these options in IPv6 networks.

1 Stateless client configuration

The original idea of autoconfiguration was based on the notion of an IPv6 device connecting to a network and autoconfiguring everything automatically, without requiring any interaction from the user. Even though DHCP was widely used at the time of the first IPv6 proposals, the creators of the IPv6 protocol decided to use a different mechanism.

The very large address space offered by IPv6 suggests an interesting solution. Why dictate IPv6 addresses to end-user stations manually, or through some central authority (a DHCP server, for instance) if each end-user station can set the address it wants to use to communicate by itself? This address can be derived from information that it already has, such as, a network-card *link-address*. This creates a mechanism to define the host part of the network address via a modified EUI 64 algorithm, or through Privacy Extensions. Thus, the host part of the address has been determined, and only the rest of the address needs to be established, i.e., the network part or global network prefix. The only device with information about the network prefix is the router to which the end-user station is connected. Passing this router is usually the only path that packets may take from (and to) the Internet. Hence, there is nothing simpler than to give the router a mechanism to announce to end-user devices what network they are on (network prefix) and what is the way out (default gateway). From these simple thoughts arose the idea of a stateless client configuration, which is called SLAAC (Stateless Address Autoconfiguration, RFC 2462[1]).

SLAAC can be described in simple terms. The network router tells all the connected nodes in a network segment what network they appear in, and what router they should use for packets travelling to the Internet (RA – *Router Advertisement*). Of course, announcing alone would not be flexible enough. Hence, a newly connected device may send a request to the network (RS – *Router Solicitation*) asking for information about what network it is in, and what is the way out. The whole autoconfiguration mechanism is a part of Neighbour Discovery for IPv6 (RFC 2461[2]), and all communication takes place using the ICMPv6 protocol. Therefore, it appears that the autoconfiguration issues have been solved. Instead of DHCP, which is familiar from the IPv4 world, SLAAC can be used for IPv6. It is not necessary to look after the DHCP server configuration, define DHCP pools and set DHCP relay; only a global prefix needs to be set on the router interface and everything else will happen almost all by itself. That sounds too good to be true, and indeed there is a catch. Actually, there are several catches.

1.1 DNS servers

A study of the specifications of the information transferred in a Router Advertisement reveals that apart from definitions of types, options and validity times, there is only a definition of the prefixes for the network that the user is currently in. In order to have complete communication in the network, other details are required, such as, the IPv6 addresses of recursive DNS servers. However, this information is not in the Router Advertisement. The problem of the absence information about recursive DNS servers in SLAAC has been known for quite some time. In practice, the efforts to resolve this problem have taken three different routes.

[1] http://tools.ietf.org/html/rfc2462
2 http://tools.ietf.org/html/rfc2461

Adding recursive DNS server information to the information transmitted within SLAAC. The standard suggests the addition of two items to Router Advertisement messages, namely recursive DNS server addresses and a list of the domains that should be searched during address translation (*Domain Search List*). This effort was made a standard only in 2010 as RFC 6106[3]. It now depends on whether the creators of operating systems decide to implement this standard. So far, this support has been implemented in the RA daemon tool for Linux/UNIX (radvd[4]), but it is not supported in other current systems - both routers and client systems. At the moment, it is very difficult to estimate how willing manufacturers would be to implement this extension to their systems. SLAAC is usually processed at the OS core level, and expansion to other items would necessarily mean changing it directly. We probably cannot expect support for this addition during a normal update.

Another proposed route was to define specific **anycast addresses for recursive DNS servers**. The client would send the translation request to this unicast address and the nearest Recursive DNS Server would provide an answer. A list of domains to search (*Doman Search List*) definitely did not solve this problem. The specification never went further than draft-ietf-ipv6-dns-discovery-07[5]. Because it used *Site-Local* addresses, which were abolished by RFC 3879[6] in 2004, this proposal was abandoned. An implementation can be found on Microsoft systems.

A third proposed solution was the transmission of Recursive DNS Server addresses, and perhaps other parameters, with a very different protocol, independent of the SLAAC mechanism. Quite logically, there is an opportunity to use something already known, and that is **DHCP**.

[3] http://tools.ietf.org/html/rfc6106
[4] http://www.litech.org/radvd/
[5] http://tools.ietf.org/html/draft-ietf-ipv6-dns-discovery-07
[6] http://tools.ietf.org/html/rfc3879

2 DHCPv6

Assigning addresses with a DHCP server became the de-facto standard for IPv4. Reacting to this positive experience, there has been an effort to re-use this mechanism in the IPv6 world. However, contrary to what one might expect, DHCPv6 is not merely DHCP that has been adapted to IPv6, with mostly the same functions.

DHCPv6 features two basic modes. In practice, the first mode, Stateless DHCPv6, is only a layer on top of the autoconfiguration mechanism described above (SLAAC). Two special flags are used for this purpose in the router advertisement: *M* – managed, *O* – other. These tell the client that it should ask in the relevant network for more information related to the connection parameters, through DHCPv6. If the *M* flag is set, stateful DHCPv6 is used. If the *O* flag is set, SLAAC will most likely be combined with stateless DHCPv6. If both flags are reset, the end-user stations know that there is no DHCPv6 server available in the network.

In practice, communication after connecting the device to the network takes place in the following way:

- the client sends an RS (*Router Solicitation*, i.e., a request to send a *Router Advertisement*) to the network;
- it receives an answer from a router (a *Router Advertisement*);
- the client configures the interface parameters based on this answer, i.e., on the IPv6 address according to EUI 64, or Privacy Extensions;
- if the *M* or *O* flags were set in the RA, it continues by sending a DHCPv6 request, setting the parameters according to answers from the DHCPv6 server (such as, recursive DNS server addresses, search domains, or NTP servers).

The behaviour of stateful DHCPv6 is more like the behaviour of DHCP that is known from IPv4. The client learns from the Router Advertisement that it should get the network address with DHCPv6 (flag *M* set). The client asks the DHCPv6 server by multicast to have an address assigned. The server assigns an address to the client for a definite period, and the assignment is confirmed. However, this does not stop the client from also configuring addresses that it obtained based on the Router Advertisement. As a result, the client has an IPv6 address configured through SLAAC, and also an address obtained from the DHCPv6 server.

It would seem that the SLAAC mechanism could be completely de-activated, and everything would depend on DHCPv6 alone. This idea is certainly right - except for one detail. All of the required parameters can be transferred through DHCPv6 except the most important one, which is the default gateway. The client expects to receive this information only via the Router Advertisement. This means that the client can create "uncontrolled" addresses, either with EUI 64 or Privacy Extensions. This behaviour could be suppressed by setting (or resetting) the *Autonomous* option in the Router Advertisements. One practical problem remains: not on all devices, this option cannot be configured or changed.

2.1 DHCPv6 and the default router

Historically, there were efforts to integrate the default gateway (default router) into DHCPv6 as the draft-droms-dhc-dhcpv6-default-router-00[7], but the proposal was evaluated as unsatisfactory at the very beginning, and was

[7] http://tools.ietf.org/html/draft-droms-dhc-dhcpv6-default-router-00

not processed further. Those interested may study the related correspondence of the IETF group dhcpwg[8], which took place in 2009. There was another attempt to solve that problem in a more complex way, as described in draft-dec-dhcpv6-route-option-05[9], but that expired in April 2011. Now, the IETF is working on the third draft, draft-ietf-mif-dhcpv6-route-option02[10], which may be accepted as a standard in the future.

If an administrator decides to assign addresses through DHCPv6 and makes an effort not to let the terminal station get another address assigned by the DHCPv6 server, he faces another problem. Quite naturally, there is an opportunity to derive IPv6 addresses from an existing database of MAC addresses, which is run to assign IPv4 addresses, and this would create a similar environment for both protocols. However, there is a trap.

2.2 DHCP Unique Identifier

DHCPv6 does not use a MAC address to identify the client; instead, it uses a specially created unique identifier called a DUID (DHCP Unique Identifier, RFC 3315[11]). The main idea behind creating such an identifier is to free the clients from dependence on hardware and on a specific network interface. The advantage is that a change of network adapter or a connection through another interface (such as WiFi instead of Ethernet) would not mean that the end-user station would start to behave as "someone else". The standard defines three ways to obtain a DUID. The creator of the DHCPv6 client decides which one he chooses to use. In practice, this means that each system creates a DUID in a different way. If a PC has *multiboot*, with more than one operating system, then each system will have a different DUID. Most likely, the DUID will also change after reinstallation of the operating system. To use DHCPv6 in a network, while retaining a sufficient overview of who has which address, there is no other solution than to create completely different mechanisms and methods to register clients and end-user stations.

Thus, the IPv6 autoconfiguration options are not straightforward. There are two different sets of mechanisms and protocols, and one cannot work effectively without the other. Currently, it is not possible to configure Recursive DNS Servers addresses with SLAAC, but with DHCPv6 it is not possible to configure the default gateway address (default router). As a result, the only working method is to use both protocols and these are processed in different parts of the operating system - SLAAC as part of the kernel and DHCPv6 as a user-space process. Failure of either mechanism, whether through faulty configuration, poorly debugged software or targeted attacks, leads to denial of the communication for the end-node, and thus, for the user. Moreover, diagnostics are fairly complicated in this situation and require a good understanding of the way both mechanisms work.

[8] http://www.ietf.org/mail-archive/web/dhcwg/current/msg09715.html
[9] http://tools.ietf.org/html/draft-dec-dhcpv6-route-option-05
[10] http://tools.ietf.org/search/draft-ietf-mif-dhcpv6-route-option-02
[11] http://tools.ietf.org/html/rfc3315

3 Support for autoconfiguration in specific systems

The autoconfiguration situation is further complicated by the different support options for autoconfiguration in specific systems.

3.1 Microsoft systems, i.e., Windows 7, Vista 2008

These systems support all of the above-mentioned autoconfiguration options. In their default setting, they first attempt configuration through SLAAC. If they manage to get at least one answer through a Router Advertisement during three Router Solicitations, they configure their addresses according to the Privacy Extensions (RFC 4941[12]). This behaviour can be suppressed by resetting the *Autonomous* flag in Router Advertisements for the relevant prefix. If the answer has the *M* (managed) or *O* (other) flag set, they will try to obtain more parameters through stateful or stateless DHCPv6.

If the system fails to obtain a Router Advertisement with three Router Solicitations, it will start to send requests to a stateful DHCPv6 server. If the network has a DHCPv6 server configured, it will give the client a network prefix, a global IPv6 address (normally just one, unless the configuration says otherwise) and other parameters managed by the stateful DHCPv6 server. As described above, this mode has no way of informing the client of the default gateway for outgoing packets.

3.2 Microsoft Windows XP

IPv6 is not active in the default settings. After installing the IPv6 support, it is only possible to configure the address through SLAAC. If there is a router present in the network that announces its existence through a Router Advertisement, the system will configure its IPv6 address using Privacy Extensions according to the older RFC 3041[13]. The obsolete Site-Local addresses or anycast addresses are used for Recursive DNS Server addresses.

3.3 Apple systems, i.e., Mac OS X (PCs and laptops), iOS (iPhone, iPod, iPad)

These systems support autoconfiguration through SLAAC, and the latest software release, from July 2011 (MAC OS Lion, IOS 5), also supports DHCPv6. The system address is created by the EUI 64 algorithm, by

[12] http://tools.ietf.org/html/rfc4941
[13] http://tools.ietf.org/html/rfc3041

default. With Mac OS X[14], it is possible to activate Privacy Extensions according to RFC 4941[15]. The addresses of IPv6 Recursive DNS Servers can only be configured in the latest releases, which support DHCPv6.

3.4 Linux, FreeBSD and others

Generally, one can say that most UNIX-like systems support SLAAC, with addresses according to EUI 64. The DHCPv6 client is either part of the base installation, or it can usually be added from standard system packages. For instance, RedHat-based systems (Centos, Fedora) ask for the autoconfiguration method (SLAAC or DHCPv6) during installation. Privacy Extensions-based address support can usually be activated without difficulty (see Linux[16], FreeBSD[17]).

[14] http://ipv6int.net/systems/mac_os_x-ipv6.html
[15] http://tools.ietf.org/html/rfc4941
[16] http://ipv6int.net/systems/linux-ipv6.html
[17] http://ipv6int.net/systems/freebsd-ipv6.html

4 Which autoconfiguration method to choose

The choice of a suitable autoconfiguration method is presently quite problematic. From a management viewpoint, stateful DHCPv6 with de-activated support for autoconfiguration is the best choice. The clients on the network can be relatively well controlled. This method is not supported by all systems and that remains its biggest disadvantage.

From the client viewpoint, SLAAC is a good choice and is supported by almost all systems. Consideration should be given to the addition of stateless DHCPv6 configuration. The addresses of Recursive DNS Servers can best be left to DHCP(v4) until one type of autoconfiguration prevails. The type that will eventually be favoured depends mostly on whether Microsoft and Apple will accept the addition of Recursive DNS Servers to Router Advertisement (RFC 6106[18]), or support for DHCPv6 becomes widely implemented. Most likely, other manufacturers will then adapt to these key players. The IETF dhcpw working group might eventually succumb to the pressure of the manufacturers and include default route-support in DHCPv6. The results of that effort are currently being processed in draft-ietf-mif-dhcpv6-route-option-02[19] and await acceptance. The current situation, in which several protocols must be supported that are based on different principles, cannot be maintained for long and thwarts the whole idea of ‘simple’ autoconfiguration.

[18] http://tools.ietf.org/html/rfc6106
[19] http://tools.ietf.org/search/draft-ietf-mif-dhcpv6-route-option-02

5 Conclusion

IPv6 autoconfiguration represents perhaps the saddest and most painful story in the whole IPv6 standardisation process. Ambiguities and delays in the inclusion of Recursive DNS Servers into mechanisms of stateless configuration has led to a host of alternative solutions. These solutions complicate the securing of the end-user nodes significantly, while at the same time permit fairly simple attacks. So far, it is not clear what form of autoconfiguration will eventually dominate. This creates complications for hardware and software manufacturers, who do not know what they should implement in their devices and products and in what way. The autoconfiguration tools do not bring any practical improvement compared to IPv4; on the contrary, the whole process is quite complicated, poorly arranged and much more vulnerable.

Today, most end-user systems that declare IPv6 support (older MAC OS systems, Win XP, mobile telephones and other devices) cannot function autonomously, without support for the IPv4 protocol. This is not a serious issue for the time being, because pure IPv6 networks (IPv6-only) are rare. However, this situation prolongs the period during which both the IPv6 and the IPv4 infrastructure will need to be operated, even in situations where it is otherwise not necessary. This is a matter for the future; yet the process of replacing all end-user devices is usually a long one, and it is a great pity that the issue of autoconfiguring devices has not yet been resolved satisfactorily.