

Network Working Group  
Request for Comments: 2462  
Obsoletes: 1971  
Category: Standards Track

S. Thomson  
Bellcore  
T. Narten  
IBM  
December 1998

## IPv6 Stateless Address Autoconfiguration

### Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

### Copyright Notice

Copyright (C) The Internet Society (1998). All Rights Reserved.

### Abstract

This document specifies the steps a host takes in deciding how to autoconfigure its interfaces in IP version 6. The autoconfiguration process includes creating a link-local address and verifying its uniqueness on a link, determining what information should be autoconfigured (addresses, other information, or both), and in the case of addresses, whether they should be obtained through the stateless mechanism, the stateful mechanism, or both. This document defines the process for generating a link-local address, the process for generating site-local and global addresses via stateless address autoconfiguration, and the Duplicate Address Detection procedure. The details of autoconfiguration using the stateful protocol are specified elsewhere.

### Table of Contents

|   |    |
|---|----|
| 1. INTRODUCTION.....                          | 2  |
| 2. TERMINOLOGY.....                           | 4  |
| 2.1. Requirements.....                        | 6  |
| 3. DESIGN GOALS.....                          | 7  |
| 4. PROTOCOL OVERVIEW.....                     | 8  |
| 4.1. Site Renumbering.....                    | 10 |
| 5. PROTOCOL SPECIFICATION.....                | 10 |
| 5.1. Node Configuration Variables.....        | 11 |
| 5.2. Autoconfiguration-Related Variables..... | 11 |
| 5.3. Creation of Link-Local Addresses.....    | 12 |

|        |  |    |
|--------|--|----|
| 5.4.   | Duplicate Address Detection.....                                       | 13 |
| 5.4.1. | Message Validation.....  | 14 |
| 5.4.2. | Sending Neighbor Solicitation Messages.....                            | 14 |
| 5.4.3. | Receiving Neighbor Solicitation Messages.....                          | 15 |
| 5.4.4. | Receiving Neighbor Advertisement Messages.....                         | 16 |
| 5.4.5. | When Duplicate Address Detection Fails.....                            | 16 |
| 5.5.   | Creation of Global and Site-Local Addresses.....                       | 16 |
| 5.5.1. | Soliciting Router Advertisements.....                                  | 16 |
| 5.5.2. | Absence of Router Advertisements.....                                  | 17 |
| 5.5.3. | Router Advertisement Processing.....                                   | 17 |
| 5.5.4. | Address Lifetime Expiry.....   | 19 |
| 5.6.   | Configuration Consistency.....   | 19 |
| 6.     | SECURITY CONSIDERATIONS.....   | 20 |
| 7.     | References.....  | 20 |
| 8.     | Acknowledgements and Authors' Addresses.....                           | 21 |
| 9.     | APPENDIX A: LOOPBACK SUPPRESSION & DUPLICATE ADDRESS<br>DETECTION..... | 22 |
| 10.    | APPENDIX B: CHANGES SINCE RFC 1971.....                                | 24 |
| 11.    | Full Copyright Statement.....  | 25 |

## 1. INTRODUCTION

This document specifies the steps a host takes in deciding how to autoconfigure its interfaces in IP version 6. The autoconfiguration process includes creating a link-local address and verifying its uniqueness on a link, determining what information should be autoconfigured (addresses, other information, or both), and in the case of addresses, whether they should be obtained through the stateless mechanism, the stateful mechanism, or both. This document defines the process for generating a link-local address, the process for generating site-local and global addresses via stateless address autoconfiguration, and the Duplicate Address Detection procedure. The details of autoconfiguration using the stateful protocol are specified elsewhere.

IPv6 defines both a stateful and stateless address autoconfiguration mechanism. Stateless autoconfiguration requires no manual configuration of hosts, minimal (if any) configuration of routers, and no additional servers. The stateless mechanism allows a host to generate its own addresses using a combination of locally available information and information advertised by routers. Routers advertise prefixes that identify the subnet(s) associated with a link, while hosts generate an "interface identifier" that uniquely identifies an interface on a subnet. An address is formed by combining the two. In the absence of routers, a host can only generate link-local addresses. However, link-local addresses are sufficient for allowing communication among nodes attached to the same link.

In the stateful autoconfiguration model, hosts obtain interface addresses and/or configuration information and parameters from a server. Servers maintain a database that keeps track of which addresses have been assigned to which hosts. The stateful autoconfiguration protocol allows hosts to obtain addresses, other configuration information or both from a server. Stateless and stateful autoconfiguration complement each other. For example, a host can use stateless autoconfiguration to configure its own addresses, but use stateful autoconfiguration to obtain other information. Stateful autoconfiguration for IPv6 is the subject of future work [DHCPv6].

The stateless approach is used when a site is not particularly concerned with the exact addresses hosts use, so long as they are unique and properly routable. The stateful approach is used when a site requires tighter control over exact address assignments. Both stateful and stateless address autoconfiguration may be used simultaneously. The site administrator specifies which type of autoconfiguration to use through the setting of appropriate fields in Router Advertisement messages [DISCOVERY].

IPv6 addresses are leased to an interface for a fixed (possibly infinite) length of time. Each address has an associated lifetime that indicates how long the address is bound to an interface. When a lifetime expires, the binding (and address) become invalid and the address may be reassigned to another interface elsewhere in the Internet. To handle the expiration of address bindings gracefully, an address goes through two distinct phases while assigned to an interface. Initially, an address is "preferred", meaning that its use in arbitrary communication is unrestricted. Later, an address becomes "deprecated" in anticipation that its current interface binding will become invalid. While in a deprecated state, the use of an address is discouraged, but not strictly forbidden. New communication (e.g., the opening of a new TCP connection) should use a preferred address when possible. A deprecated address should be used only by applications that have been using it and would have difficulty switching to another address without a service disruption.

To insure that all configured addresses are likely to be unique on a given link, nodes run a "duplicate address detection" algorithm on addresses before assigning them to an interface. The Duplicate Address Detection algorithm is performed on all addresses, independent of whether they are obtained via stateless or stateful autoconfiguration. This document defines the Duplicate Address Detection algorithm.

The autoconfiguration process specified in this document applies only to hosts and not routers. Since host autoconfiguration uses information advertised by routers, routers will need to be configured by some other means. However, it is expected that routers will generate link-local addresses using the mechanism described in this document. In addition, routers are expected to successfully pass the Duplicate Address Detection procedure described in this document on all addresses prior to assigning them to an interface.

Section 2 provides definitions for terminology used throughout this document. Section 3 describes the design goals that lead to the current autoconfiguration procedure. Section 4 provides an overview of the protocol, while Section 5 describes the protocol in detail.

## 2. TERMINOLOGY

IP - Internet Protocol Version 6. The terms IPv4 and are used only in contexts where necessary to avoid ambiguity.

node - a device that implements IP.

router - a node that forwards IP packets not explicitly addressed to itself.

host - any node that is not a router.

upper layer - a protocol layer immediately above IP. Examples are transport protocols such as TCP and UDP, control protocols such as ICMP, routing protocols such as OSPF, and internet or lower-layer protocols being "tunneled" over (i.e., encapsulated in) IP such as IPX, AppleTalk, or IP itself.

link - a communication facility or medium over which nodes can communicate at the link layer, i.e., the layer immediately below IP. Examples are Ethernets (simple or bridged); PPP links; X.25, Frame Relay, or ATM networks; and internet (or higher) layer "tunnels", such as tunnels over IPv4 or IPv6 itself.

interface - a node's attachment to a link.

packet - an IP header plus payload.

address - an IP-layer identifier for an interface or a set of interfaces.

unicast address - an identifier for a single interface. A packet sent to a unicast address is delivered to the interface identified by that address.

multicast address - an identifier for a set of interfaces (typically belonging to different nodes). A packet sent to a multicast address is delivered to all interfaces identified by that address.

anycast address - an identifier for a set of interfaces (typically belonging to different nodes). A packet sent to an anycast address is delivered to one of the interfaces identified by that address (the "nearest" one, according to the routing protocol's measure of distance). See [ADDR-ARCH].

solicited-node multicast address - a multicast address to which Neighbor Solicitation messages are sent. The algorithm for computing the address is given in [DISCOVERY].

link-layer address - a link-layer identifier for an interface. Examples include IEEE 802 addresses for Ethernet links and E.164 addresses for ISDN links.

link-local address - an address having link-only scope that can be used to reach neighboring nodes attached to the same link. All interfaces have a link-local unicast address.

site-local address - an address having scope that is limited to the local site.

global address - an address with unlimited scope.

communication - any packet exchange among nodes that requires that the address of each node used in the exchange remain the same for the duration of the packet exchange. Examples are a TCP connection or a UDP request- response.

tentative address - an address whose uniqueness on a link is being verified, prior to its assignment to an interface. A tentative address is not considered assigned to an interface in the usual sense. An interface discards received packets addressed to a tentative address, but accepts Neighbor Discovery packets related to Duplicate Address Detection for the tentative address.

preferred address - an address assigned to an interface whose use by upper layer protocols is unrestricted. Preferred addresses may be used as the source (or destination) address of packets sent from (or to) the interface.

deprecated address - An address assigned to an interface whose use is discouraged, but not forbidden. A deprecated address should no longer be used as a source address in new communications, but packets sent from or to deprecated addresses are delivered as expected. A deprecated address may continue to be used as a source address in communications where switching to a preferred address causes hardship to a specific upper-layer activity (e.g., an existing TCP connection).

valid address - a preferred or deprecated address. A valid address may appear as the source or destination address of a packet, and the internet routing system is expected to deliver packets sent to a valid address to their intended recipients.

invalid address - an address that is not assigned to any interface. A valid address becomes invalid when its valid lifetime expires. Invalid addresses should not appear as the destination or source address of a packet. In the former case, the internet routing system will be unable to deliver the packet, in the later case the recipient of the packet will be unable to respond to it.

preferred lifetime - the length of time that a valid address is preferred (i.e., the time until deprecation). When the preferred lifetime expires, the address becomes deprecated.

valid lifetime - the length of time an address remains in the valid state (i.e., the time until invalidation). The valid lifetime must be greater than or equal to the preferred lifetime. When the valid lifetime expires, the address becomes invalid.

interface identifier - a link-dependent identifier for an interface that is (at least) unique per link [ADDR-ARCH]. Stateless address autoconfiguration combines an interface identifier with a prefix to form an address. From address autoconfiguration's perspective, an interface identifier is a bit string of known length. The exact length of an interface identifier and the way it is created is defined in a separate link-type specific document that covers issues related to the transmission of IP over a particular link type (e.g., [IPv6-ETHER]). In many cases, the identifier will be the same as the interface's link-layer address.

## 2.1. Requirements

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when they appear in this document, are to be interpreted as described in [KEYWORDS].

### 3. DESIGN GOALS

Stateless autoconfiguration is designed with the following goals in mind:

- o Manual configuration of individual machines before connecting them to the network should not be required. Consequently, a mechanism is needed that allows a host to obtain or create unique addresses for each of its interfaces. Address autoconfiguration assumes that each interface can provide a unique identifier for that interface (i.e., an "interface identifier"). In the simplest case, an interface identifier consists of the interface's link-layer address. An interface identifier can be combined with a prefix to form an address.
- o Small sites consisting of a set of machines attached to a single link should not require the presence of a stateful server or router as a prerequisite for communicating. Plug-and-play communication is achieved through the use of link-local addresses. Link-local addresses have a well-known prefix that identifies the (single) shared link to which a set of nodes attach. A host forms a link-local address by appending its interface identifier to the link-local prefix.
- o A large site with multiple networks and routers should not require the presence of a stateful address configuration server. In order to generate site-local or global addresses, hosts must determine the prefixes that identify the subnets to which they attach. Routers generate periodic Router Advertisements that include options listing the set of active prefixes on a link.
- o Address configuration should facilitate the graceful renumbering of a site's machines. For example, a site may wish to renumber all of its nodes when it switches to a new network service provider. Renumbering is achieved through the leasing of addresses to interfaces and the assignment of multiple addresses to the same interface. Lease lifetimes provide the mechanism through which a site phases out old prefixes. The assignment of multiple addresses to an interface provides for a transition period during which both a new address and the one being phased out work simultaneously.
- o System administrators need the ability to specify whether stateless autoconfiguration, stateful autoconfiguration, or both should be used. Router Advertisements include flags specifying which mechanisms a host should use.

#### 4. PROTOCOL OVERVIEW

This section provides an overview of the typical steps that take place when an interface autoconfigures itself. Autoconfiguration is performed only on multicast-capable links and begins when a multicast-capable interface is enabled, e.g., during system startup. Nodes (both hosts and routers) begin the autoconfiguration process by generating a link-local address for the interface. A link-local address is formed by appending the interface's identifier to the well-known link-local prefix.

Before the link-local address can be assigned to an interface and used, however, a node must attempt to verify that this "tentative" address is not already in use by another node on the link. Specifically, it sends a Neighbor Solicitation message containing the tentative address as the target. If another node is already using that address, it will return a Neighbor Advertisement saying so. If another node is also attempting to use the same address, it will send a Neighbor Solicitation for the target as well. The exact number of times the Neighbor Solicitation is (re)transmitted and the delay time between consecutive solicitations is link-specific and may be set by system management.

If a node determines that its tentative link-local address is not unique, autoconfiguration stops and manual configuration of the interface is required. To simplify recovery in this case, it should be possible for an administrator to supply an alternate interface identifier that overrides the default identifier in such a way that the autoconfiguration mechanism can then be applied using the new (presumably unique) interface identifier. Alternatively, link-local and other addresses will need to be configured manually.

Once a node ascertains that its tentative link-local address is unique, it assigns it to the interface. At this point, the node has IP-level connectivity with neighboring nodes. The remaining autoconfiguration steps are performed only by hosts; the (auto)configuration of routers is beyond the scope of this document.

The next phase of autoconfiguration involves obtaining a Router Advertisement or determining that no routers are present. If routers are present, they will send Router Advertisements that specify what sort of autoconfiguration a host should do. If no routers are present, stateful autoconfiguration should be invoked.

Routers send Router Advertisements periodically, but the delay between successive advertisements will generally be longer than a host performing autoconfiguration will want to wait [DISCOVERY]. To obtain an advertisement quickly, a host sends one or more Router



Solicitations to the all-routers multicast group. Router Advertisements contain two flags indicating what type of stateful autoconfiguration (if any) should be performed. A "managed address configuration" flag indicates whether hosts should use stateful autoconfiguration to obtain addresses. An "other stateful configuration" flag indicates whether hosts should use stateful autoconfiguration to obtain additional information (excluding addresses).

Router Advertisements also contain zero or more Prefix Information options that contain information used by stateless address autoconfiguration to generate site-local and global addresses. It should be noted that the stateless and stateful address autoconfiguration fields in Router Advertisements are processed independently of one another, and a host may use both stateful and stateless address autoconfiguration simultaneously. One Prefix Information option field, the "autonomous address-configuration flag", indicates whether or not the option even applies to stateless autoconfiguration. If it does, additional option fields contain a subnet prefix together with lifetime values indicating how long addresses created from the prefix remain preferred and valid.

Because routers generate Router Advertisements periodically, hosts will continually receive new advertisements. Hosts process the information contained in each advertisement as described above, adding to and refreshing information received in previous advertisements.

For safety, all addresses must be tested for uniqueness prior to their assignment to an interface. In the case of addresses created through stateless autoconfig, however, the uniqueness of an address is determined primarily by the portion of the address formed from an interface identifier. Thus, if a node has already verified the uniqueness of a link-local address, additional addresses created from the same interface identifier need not be tested individually. In contrast, all addresses obtained manually or via stateful address autoconfiguration should be tested for uniqueness individually. To accommodate sites that believe the overhead of performing Duplicate Address Detection outweighs its benefits, the use of Duplicate Address Detection can be disabled through the administrative setting of a per-interface configuration flag.

To speed the autoconfiguration process, a host may generate its link-local address (and verify its uniqueness) in parallel with waiting for a Router Advertisement. Because a router may delay responding to a Router Solicitation for a few seconds, the total time needed to complete autoconfiguration can be significantly longer if the two steps are done serially.

#### 4.1. Site Renumbering

Address leasing facilitates site renumbering by providing a mechanism to time-out addresses assigned to interfaces in hosts. At present, upper layer protocols such as TCP provide no support for changing end-point addresses while a connection is open. If an end-point address becomes invalid, existing connections break and all communication to the invalid address fails. Even when applications use UDP as a transport protocol, addresses must generally remain the same during a packet exchange.

Dividing valid addresses into preferred and deprecated categories provides a way of indicating to upper layers that a valid address may become invalid shortly and that future communication using the address will fail, should the address's valid lifetime expire before communication ends. To avoid this scenario, higher layers should use a preferred address (assuming one of sufficient scope exists) to increase the likelihood that an address will remain valid for the duration of the communication. It is up to system administrators to set appropriate prefix lifetimes in order to minimize the impact of failed communication when renumbering takes place. The deprecation period should be long enough that most, if not all, communications are using the new address at the time an address becomes invalid.

The IP layer is expected to provide a means for upper layers (including applications) to select the most appropriate source address given a particular destination and possibly other constraints. An application may choose to select the source address itself before starting a new communication or may leave the address unspecified, in which case the upper networking layers will use the mechanism provided by the IP layer to choose a suitable address on the application's behalf.

Detailed address selection rules are beyond the scope of this document.

#### 5. PROTOCOL SPECIFICATION

Autoconfiguration is performed on a per-interface basis on multicast-capable interfaces. For multihomed hosts, autoconfiguration is performed independently on each interface. Autoconfiguration applies primarily to hosts, with two exceptions. Routers are expected to generate a link-local address using the procedure outlined below. In addition, routers perform Duplicate Address Detection on all addresses prior to assigning them to an interface.

### 5.1. Node Configuration Variables

A node MUST allow the following autoconfiguration-related variable to be configured by system management for each multicast interface:

#### DupAddrDetectTransmits

The number of consecutive Neighbor Solicitation messages sent while performing Duplicate Address Detection on a tentative address. A value of zero indicates that Duplicate Address Detection is not performed on tentative addresses. A value of one indicates a single transmission with no follow up retransmissions.

Default: 1, but may be overridden by a link-type specific value in the document that covers issues related to the transmission of IP over a particular link type (e.g., [IPv6-ETHER]).

Autoconfiguration also assumes the presence of the variable RetransTimer as defined in [DISCOVERY]. For autoconfiguration purposes, RetransTimer specifies the delay between consecutive Neighbor Solicitation transmissions performed during Duplicate Address Detection (if DupAddrDetectTransmits is greater than 1), as well as the time a node waits after sending the last Neighbor Solicitation before ending the Duplicate Address Detection process.

### 5.2. Autoconfiguration-Related Variables

A host maintains a number of data structures and flags related to autoconfiguration. In the following, we present conceptual variables and show how they are used to perform autoconfiguration. The specific variables are used for demonstration purposes only, and an implementation is not required to have them, so long as its external behavior is consistent with that described in this document.

Beyond the formation of a link-local address and using Duplicate Address Detection, how routers (auto)configure their interfaces is beyond the scope of this document.

Hosts maintain the following variables on a per-interface basis:

**ManagedFlag** Copied from the M flag field (i.e., the "managed address configuration" flag) of the most recently received Router Advertisement message. The flag indicates whether or not addresses are to be configured using the stateful autoconfiguration mechanism. It starts out in a FALSE state.

**OtherConfigFlag** Copied from the O flag field (i.e., the "other stateful configuration" flag) of the most recently received Router Advertisement message. The flag indicates whether or not information other than addresses is to be obtained using the stateful autoconfiguration mechanism. It starts out in a FALSE state.

In addition, when the value of the ManagedFlag is TRUE, the value of OtherConfigFlag is implicitly TRUE as well. It is not a valid configuration for a host to use stateful address autoconfiguration to request addresses only, without also accepting other configuration information.

A host also maintains a list of addresses together with their corresponding lifetimes. The address list contains both autoconfigured addresses and those configured manually.

### 5.3. Creation of Link-Local Addresses

A node forms a link-local address whenever an interface becomes enabled. An interface may become enabled after any of the following events:

- The interface is initialized at system startup time.
- The interface is reinitialized after a temporary interface failure or after being temporarily disabled by system management.
- The interface attaches to a link for the first time.
- The interface becomes enabled by system management after having been administratively disabled.

A link-local address is formed by prepending the well-known link-local prefix FE80::0 [ADDR-ARCH] (of appropriate length) to the interface identifier. If the interface identifier has a length of N bits, the interface identifier replaces the right-most N zero bits of the link-local prefix. If the interface identifier is more than 118 bits in length, autoconfiguration fails and manual configuration is required. Note that interface identifiers will typically be 64-bits long and based on EUI-64 identifiers as described in [ADDR-ARCH].

A link-local address has an infinite preferred and valid lifetime; it is never timed out.

#### 5.4. Duplicate Address Detection

Duplicate Address Detection is performed on unicast addresses prior to assigning them to an interface whose DupAddrDetectTransmits variable is greater than zero. Duplicate Address Detection MUST take place on all unicast addresses, regardless of whether they are obtained through stateful, stateless or manual configuration, with the exception of the following cases:

- Duplicate Address Detection MUST NOT be performed on anycast addresses.
- Each individual unicast address SHOULD be tested for uniqueness. However, when stateless address autoconfiguration is used, address uniqueness is determined solely by the interface identifier, assuming that subnet prefixes are assigned correctly (i.e., if all of an interface's addresses are generated from the same identifier, either all addresses or none of them will be duplicates). Thus, for a set of addresses formed from the same interface identifier, it is sufficient to check that the link-local address generated from the identifier is unique on the link. In such cases, the link-local address MUST be tested for uniqueness, and if no duplicate address is detected, an implementation MAY choose to skip Duplicate Address Detection for additional addresses derived from the same interface identifier.

The procedure for detecting duplicate addresses uses Neighbor Solicitation and Advertisement messages as described below. If a duplicate address is discovered during the procedure, the address cannot be assigned to the interface. If the address is derived from an interface identifier, a new identifier will need to be assigned to the interface, or all IP addresses for the interface will need to be manually configured. Note that the method for detecting duplicates is not completely reliable, and it is possible that duplicate

addresses will still exist (e.g., if the link was partitioned while Duplicate Address Detection was performed).

An address on which the duplicate Address Detection Procedure is applied is said to be tentative until the procedure has completed successfully. A tentative address is not considered "assigned to an interface" in the traditional sense. That is, the interface must accept Neighbor Solicitation and Advertisement messages containing the tentative address in the Target Address field, but processes such packets differently from those whose Target Address matches an address assigned to the interface. Other packets addressed to the tentative address should be silently discarded.

It should also be noted that Duplicate Address Detection must be performed prior to assigning an address to an interface in order to prevent multiple nodes from using the same address simultaneously. If a node begins using an address in parallel with Duplicate Address Detection, and another node is already using the address, the node performing Duplicate Address Detection will erroneously process traffic intended for the other node, resulting in such possible negative consequences as the resetting of open TCP connections.

The following subsections describe specific tests a node performs to verify an address's uniqueness. An address is considered unique if none of the tests indicate the presence of a duplicate address within RetransTimer milliseconds after having sent DupAddrDetectTransmits Neighbor Solicitations. Once an address is determined to be unique, it may be assigned to an interface.

#### 5.4.1. Message Validation

A node MUST silently discard any Neighbor Solicitation or Advertisement message that does not pass the validity checks specified in [DISCOVERY]. A solicitation that passes these validity checks is called a valid solicitation or valid advertisement.

#### 5.4.2. Sending Neighbor Solicitation Messages

Before sending a Neighbor Solicitation, an interface MUST join the all-nodes multicast address and the solicited-node multicast address of the tentative address. The former insures that the node receives Neighbor Advertisements from other nodes already using the address; the latter insures that two nodes attempting to use the same address simultaneously detect each other's presence.

To check an address, a node sends DupAddrDetectTransmits Neighbor Solicitations, each separated by RetransTimer milliseconds. The solicitation's Target Address is set to the address being checked,

the IP source is set to the unspecified address and the IP destination is set to the solicited-node multicast address of the target address.

If the Neighbor Solicitation is the first message to be sent from an interface after interface (re)initialization, the node should delay sending the message by a random delay between 0 and MAX\_RTR\_SOLICITATION\_DELAY as specified in [DISCOVERY]. This serves to alleviate congestion when many nodes start up on the link at the same time, such as after a power failure, and may help to avoid race conditions when more than one node is trying to solicit for the same address at the same time. In order to improve the robustness of the Duplicate Address Detection algorithm, an interface **MUST** receive and process datagrams sent to the all-nodes multicast address or solicited-node multicast address of the tentative address while delaying transmission of the initial Neighbor Solicitation.

#### 5.4.3. Receiving Neighbor Solicitation Messages

On receipt of a valid Neighbor Solicitation message on an interface, node behavior depends on whether the target address is tentative or not. If the target address is not tentative (i.e., it is assigned to the receiving interface), the solicitation is processed as described in [DISCOVERY]. If the target address is tentative, and the source address is a unicast address, the solicitation's sender is performing address resolution on the target; the solicitation should be silently ignored. Otherwise, processing takes place as described below. In all cases, a node **MUST NOT** respond to a Neighbor Solicitation for a tentative address.

If the source address of the Neighbor Solicitation is the unspecified address, the solicitation is from a node performing Duplicate Address Detection. If the solicitation is from another node, the tentative address is a duplicate and should not be used (by either node). If the solicitation is from the node itself (because the node loops back multicast packets), the solicitation does not indicate the presence of a duplicate address.

Implementor's Note: many interfaces provide a way for upper layers to selectively enable and disable the looping back of multicast packets. The details of how such a facility is implemented may prevent Duplicate Address Detection from working correctly. See the Appendix for further discussion.

The following tests identify conditions under which a tentative address is not unique:

- If a Neighbor Solicitation for a tentative address is received prior to having sent one, the tentative address is a duplicate. This condition occurs when two nodes run Duplicate Address Detection simultaneously, but transmit initial solicitations at different times (e.g., by selecting different random delay values before transmitting an initial solicitation).
- If the actual number of Neighbor Solicitations received exceeds the number expected based on the loopback semantics (e.g., the interface does not loopback packet, yet one or more solicitations was received), the tentative address is a duplicate. This condition occurs when two nodes run Duplicate Address Detection simultaneously and transmit solicitations at roughly the same time.

#### 5.4.4. Receiving Neighbor Advertisement Messages

On receipt of a valid Neighbor Advertisement message on an interface, node behavior depends on whether the target address is tentative or matches a unicast or anycast address assigned to the interface. If the target address is assigned to the receiving interface, the solicitation is processed as described in [DISCOVERY]. If the target address is tentative, the tentative address is not unique.

#### 5.4.5. When Duplicate Address Detection Fails

A tentative address that is determined to be a duplicate as described above, MUST NOT be assigned to an interface and the node SHOULD log a system management error. If the address is a link-local address formed from an interface identifier, the interface SHOULD be disabled.

### 5.5. Creation of Global and Site-Local Addresses

Global and site-local addresses are formed by appending an interface identifier to a prefix of appropriate length. Prefixes are obtained from Prefix Information options contained in Router Advertisements. Creation of global and site-local addresses and configuration of other parameters as described in this section SHOULD be locally configurable. However, the processing described below MUST be enabled by default.

#### 5.5.1. Soliciting Router Advertisements

Router Advertisements are sent periodically to the all-nodes multicast address. To obtain an advertisement quickly, a host sends out Router Solicitations as described in [DISCOVERY].



### 5.5.2. Absence of Router Advertisements

If a link has no routers, a host **MUST** attempt to use stateful autoconfiguration to obtain addresses and other configuration information. An implementation **MAY** provide a way to disable the invocation of stateful autoconfiguration in this case, but the default **SHOULD** be enabled. From the perspective of autoconfiguration, a link has no routers if no Router Advertisements are received after having sent a small number of Router Solicitations as described in [DISCOVERY].

### 5.5.3. Router Advertisement Processing

On receipt of a valid Router Advertisement (as defined in [DISCOVERY]), a host copies the value of the advertisement's M bit into ManagedFlag. If the value of ManagedFlag changes from FALSE to TRUE, and the host is not already running the stateful address autoconfiguration protocol, the host should invoke the stateful address autoconfiguration protocol, requesting both address information and other information. If the value of the ManagedFlag changes from TRUE to FALSE, the host should continue running the stateful address autoconfiguration, i.e., the change in the value of the ManagedFlag has no effect. If the value of the flag stays unchanged, no special action takes place. In particular, a host **MUST NOT** reinvoke stateful address configuration if it is already participating in the stateful protocol as a result of an earlier advertisement.

An advertisement's O flag field is processed in an analogous manner. A host copies the value of the O flag into OtherConfigFlag. If the value of OtherConfigFlag changes from FALSE to TRUE, the host should invoke the stateful autoconfiguration protocol, requesting information (excluding addresses if ManagedFlag is set to FALSE). If the value of the OtherConfigFlag changes from TRUE to FALSE, the host should continue running the stateful address autoconfiguration protocol, i.e., the change in the value of OtherConfigFlag has no effect. If the value of the flag stays unchanged, no special action takes place. In particular, a host **MUST NOT** reinvoke stateful configuration if it is already participating in the stateful protocol as a result of an earlier advertisement.

For each Prefix-Information option in the Router Advertisement:

- a) If the Autonomous flag is not set, silently ignore the Prefix Information option.

- b) If the prefix is the link-local prefix, silently ignore the Prefix Information option.
- c) If the preferred lifetime is greater than the valid lifetime, silently ignore the Prefix Information option. A node MAY wish to log a system management error in this case.
- d) If the prefix advertised does not match the prefix of an address already in the list, and the Valid Lifetime is not 0, form an address (and add it to the list) by combining the advertised prefix with the link's interface identifier as follows:

|              |                      |
|--------------|----------------------|
| 128 - N bits | N bits               |
| link prefix  | interface identifier |

If the sum of the prefix length and interface identifier length does not equal 128 bits, the Prefix Information option MUST be ignored. An implementation MAY wish to log a system management error in this case. It is the responsibility of the system administrator to insure that the lengths of prefixes contained in Router Advertisements are consistent with the length of interface identifiers for that link type. Note that interface identifiers will typically be 64-bits long and based on EUI-64 identifiers as described in [ADDR-ARCH].

If an address is formed successfully, the host adds it to the list of addresses assigned to the interface, initializing its preferred and valid lifetime values from the Prefix Information option.

- e) If the advertised prefix matches the prefix of an autoconfigured address (i.e., one obtained via stateless or stateful address autoconfiguration) in the list of addresses associated with the interface, the specific action to perform depends on the Valid Lifetime in the received advertisement and the Lifetime associated with the previously autoconfigured address (which we call `StoredLifetime` in the discussion that follows):
  - 1) If the received Lifetime is greater than 2 hours or greater than `StoredLifetime`, update the stored Lifetime of the corresponding address.
  - 2) If the `StoredLifetime` is less than or equal to 2 hours and the received Lifetime is less than or equal to `StoredLifetime`, ignore the prefix, unless the Router Advertisement from which

this Prefix Information option was obtained has been authenticated (e.g., via IPsec [RFC2402]). If the Router Advertisement was authenticated, the StoredLifetime should be set to the Lifetime in the received option.

- 3) Otherwise, reset the stored Lifetime in the corresponding address to two hours.

The above rules address a specific denial of service attack in which a bogus advertisement could contain prefixes with very small Valid Lifetimes. Without the above rules, a single unauthenticated advertisement containing bogus Prefix Information options with short Lifetimes could cause all of a node's addresses to expire prematurely. The above rules insure that legitimate advertisements (which are sent periodically) will "cancel" the short lifetimes before they actually take effect.

#### 5.5.4. Address Lifetime Expiry

A preferred address becomes deprecated when its preferred lifetime expires. A deprecated address SHOULD continue to be used as a source address in existing communications, but SHOULD NOT be used in new communications if an alternate (non-deprecated) address is available and has sufficient scope. IP and higher layers (e.g., TCP, UDP) MUST continue to accept datagrams destined to a deprecated address since a deprecated address is still a valid address for the interface. An implementation MAY prevent any new communication from using a deprecated address, but system management MUST have the ability to disable such a facility, and the facility MUST be disabled by default.

An address (and its association with an interface) becomes invalid when its valid lifetime expires. An invalid address MUST NOT be used as a source address in outgoing communications and MUST NOT be recognized as a destination on a receiving interface.

#### 5.6. Configuration Consistency

It is possible for hosts to obtain address information using both stateless and stateful protocols since both may be enabled at the same time. It is also possible that the values of other configuration parameters such as MTU size and hop limit will be learned from both Router Advertisements and the stateful autoconfiguration protocol. If the same configuration information is provided by multiple sources, the value of this information should be consistent. However, it is not considered a fatal error if information received from multiple sources is inconsistent. Hosts accept the union of all information received via the stateless and

stateful protocols. If inconsistent information is learned from different sources, the most recently obtained values always have precedence over information learned earlier.

## 6. SECURITY CONSIDERATIONS

Stateless address autoconfiguration allows a host to connect to a network, configure an address and start communicating with other nodes without ever registering or authenticating itself with the local site. Although this allows unauthorized users to connect to and use a network, the threat is inherently present in the Internet architecture. Any node with a physical attachment to a network can generate an address (using a variety of ad hoc techniques) that provides connectivity.

The use of Duplicate Address Detection opens up the possibility of denial of service attacks. Any node can respond to Neighbor Solicitations for a tentative address, causing the other node to reject the address as a duplicate. This attack is similar to other attacks involving the spoofing of Neighbor Discovery messages and can be addressed by requiring that Neighbor Discovery packets be authenticated [RFC2402].

## 7. References

- [RFC2402] Kent, S. and R. Atkinson, "IP Authentication Header", RFC 2402, November 1998.
- [IPv6-ETHER] Crawford, M., "A Method for the Transmission of IPv6 Packets over Ethernet Networks", RFC 2464, December 1998.
- [KEYWORDS] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC1112] Deering, S., "Host Extensions for IP Multicasting", STD 5, RFC 1112, August 1989.
- [ADDR-ARCH] Hinden, R. and S. Deering, "Internet Protocol Version (IPv6) Addressing Architecture", RFC 2373, July 1998
- [DHCPv6] Bound, J. and C. Perkins, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", Work in Progress.

[DISCOVERY] Narten, T., Nordmark, E. and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", RFC 2461, December 1998.

## 8. Acknowledgements

The authors would like to thank the members of both the IPNG and ADDRCONF working groups for their input. In particular, thanks to Jim Bound, Steve Deering, Richard Draves, and Erik Nordmark. Thanks also goes to John Gilmore for alerting the WG of the "0 Lifetime Prefix Advertisement" denial of service attack vulnerability; this document incorporates changes that address this vulnerability.

## AUTHORS' ADDRESSES

Susan Thomson  
Bellcore  
445 South Street  
Morristown, NJ 07960  
USA

Phone: +1 201-829-4514  
EMail: set@thumper.bellcore.com

Thomas Narten  
IBM Corporation  
P.O. Box 12195  
Research Triangle Park, NC 27709-2195  
USA

Phone: +1 919 254 7798  
EMail: narten@raleigh.ibm.com

## 9. APPENDIX A: LOOPBACK SUPPRESSION &amp; DUPLICATE ADDRESS DETECTION

Determining whether a received multicast solicitation was looped back to the sender or actually came from another node is implementation-dependent. A problematic case occurs when two interfaces attached to the same link happen to have the same identifier and link-layer address, and they both send out packets with identical contents at roughly the same time (e.g., Neighbor Solicitations for a tentative address as part of Duplicate Address Detection messages). Although a receiver will receive both packets, it cannot determine which packet was looped back and which packet came from the other node by simply comparing packet contents (i.e., the contents are identical). In this particular case, it is not necessary to know precisely which packet was looped back and which was sent by another node; if one receives more solicitations than were sent, the tentative address is a duplicate. However, the situation may not always be this straightforward.

The IPv4 multicast specification [RFC1112] recommends that the service interface provide a way for an upper-layer protocol to inhibit local delivery of packets sent to a multicast group that the sending host is a member of. Some applications know that there will be no other group members on the same host, and suppressing loopback prevents them from having to receive (and discard) the packets they themselves send out. A straightforward way to implement this facility is to disable loopback at the hardware level (if supported by the hardware), with packets looped back (if requested) by software. On interfaces in which the hardware itself suppresses loopbacks, a node running Duplicate Address Detection simply counts the number of Neighbor Solicitations received for a tentative address and compares them with the number expected. If there is a mismatch, the tentative address is a duplicate.

In those cases where the hardware cannot suppress loopbacks, however, one possible software heuristic to filter out unwanted loopbacks is to discard any received packet whose link-layer source address is the same as the receiving interface's. Unfortunately, use of that criteria also results in the discarding of all packets sent by another node using the same link-layer address. Duplicate Address Detection will fail on interfaces that filter received packets in this manner:

- o If a node performing Duplicate Address Detection discards received packets having the same source link-layer address as the receiving interface, it will also discard packets from other nodes also using the same link-layer address, including Neighbor Advertisement and Neighbor Solicitation messages required to make Duplicate Address Detection work correctly. This

particular problem can be avoided by temporarily disabling the software suppression of loopbacks while a node performs Duplicate Address Detection.

- o If a node that is already using a particular IP address discards received packets having the same link-layer source address as the interface, it will also discard Duplicate Address Detection-related Neighbor Solicitation messages sent by another node also using the same link-layer address. Consequently, Duplicate Address Detection will fail, and the other node will configure a non-unique address. Since it is generally impossible to know when another node is performing Duplicate Address Detection, this scenario can be avoided only if software suppression of loopback is permanently disabled.

Thus, to perform Duplicate Address Detection correctly in the case where two interfaces are using the same link-layer address, an implementation must have a good understanding of the interface's multicast loopback semantics, and the interface cannot discard received packets simply because the source link-layer address is the same as the interfaces.

## 10. APPENDIX B: CHANGES SINCE RFC 1971

- o Changed document to use term "interface identifier" rather than "interface token" for consistency with other IPv6 documents.
- o Clarified definition of deprecated address to make clear it is OK to continue sending to or from deprecated addresses.
- o Reworded section 5.4 for clarity (no substantive change).
- o Added rules to Section 5.5.3 Router Advertisement processing to address potential denial-of-service attack when prefixes are advertised with very short Lifetimes.
- o Clarified wording in Section 5.5.4 to make clear that all upper layer protocols must process (i.e., send and receive) packets sent to deprecated addresses.



## 11. Full Copyright Statement

Copyright (C) The Internet Society (1998). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

